Entwurf und Implementierung eines wissensbasierten Ähnlichkeitsmaßes für die fallbasierte Planung am Beispiel der Arbeitsplanung für die Teilefertigung

Bernhard Humm

30. September 1994

Inhaltsverzeichnis

Vorwort **iii**

1 Einleitung **1**

2 Fallbasierte Planung **2**
2.1 Planung – Traditionelle Ansätze 2
2.1.1 Planen als Suche im Zustandsraum 3
2.1.2 Rückwärtssuche mit Goalstack: STRIPS 3
2.1.3 Nichtlineare Planung: NOAH 4
2.1.4 Hierarchische Planung: ABSTRIPS 4
2.1.5 Weiterführende Ansätze 4
2.2 Analoges Schließen und fallbasiertes Schließen 5
2.2.1 Analoges Schließen 5
2.2.2 Die Dynamic Memory Theorie 6
2.2.3 Fallbasiertes Schließen 7
2.3 Fallbasierte Planung 9
2.3.1 Fallbasiertes Schließen im Bereich der Planung 9
2.3.2 Relationen zu traditionellen Planungsansätzen 9
2.3.3 Unterscheidungsmerkmale von CBP-Systemen 10
2.3.4 CHEF: ein System für die fallbasierte Planung 11
2.3.5 Diskussion von CHEF 21

3 Arbeitsplanung **23**
3.1 Aufgaben und Methoden der Arbeitsplanung 23
3.1.1 Einordnung der Arbeitsplanung in die technische Auftragsabwicklung . 23
3.1.2 Ziele der Arbeitsplanerstellung 25
3.1.3 Anlässe und Methoden der Arbeitsplanerstellung 33
3.2 Rechnergestützte Arbeitsplanung 35
3.2.1 Arbeitsplanung als Engpaß der technischen Auftragsabwicklung 35
3.2.2 Entwicklung der CAPP-Systeme 36
3.2.3 Klassifikation von CAPP-Systemen 37
3.2.4 Defizite bisheriger Ansätze für CAPP 38
3.3 Das CAPP-System WAP und seine Integration in ein CIM-Konzept 38
3.3.1 Das integrierte Datenmodell 39
3.3.2 Die formelementbasierte Werkstückrepräsentation 40

3.3.3 Die Bearbeitungsbereichsrepräsentation 47
3.3.4 Die Arbeitsplanrepräsentation 48
3.3.5 Die CIM-Module in ihrer Abfolge 48
3.3.6 Die Implementierungssprache KEE 51

4 Ein System für die fallbasierte Arbeitsplanung 53
4.1 Vorüberlegungen 53
4.2 Ähnlichkeit als Minimierung des Modifikationsaufwandes 56
4.3 Fallbasis und Retrieval 58
4.4 Das Ähnlichkeitsmaß 59
4.4.1 Präzisierung und Diskussion der Definition von Ähnlichkeit 59
4.4.2 Der Prozeß der Ähnlichkeitsbestimmung 60
4.5 Der Modifier 67
4.6 Verbesserung von Ähnlichkeitsmaßen 70
4.7 Diskussion 70
4.7.1 Klassifikation als CBP-System 70
4.7.2 Klassifikation als CAPP-System 73
4.7.3 Kompensierte Defizite bisheriger Ansätze für CAPP 74

5 Implementierung 75
5.1 Generierung technologischer Constraints 75
5.2 Vergleich von aktuellem und altem Fall 78
5.3 Generierung von Hypothesen für durchzuführende Modifikationen 80
5.4 Aufruf des Systems 82

Abbildungsverzeichnis

2.1 Analoges Schließen . . . 5
2.2 Beispiel für das Retrieval in *CHEF* . . . 12
2.3 Beispiel für die Modification in *CHEF* . . . 12
2.4 Beispiel für den Plan Repair in *CHEF* . . . 14
2.5 Beispiel für die Generierung von Plan Failure Predictors in *CHEF* . . . 15
2.6 *CHEF*s Module . . . 16
2.7 *CHEFs* Plan Memory . . . 17
2.8 Ausschnitt aus *CHEFs* Memory of Failures . . . 18
2.9 Tabelle der Standard Modification Rules . . . 18
2.10 *CHEFs* Critics . . . 19
2.11 Auswahl von *TOP*s . . . 20

3.1 Die Stellung der Arbeitsvorbereitung in der technischen Auftragsabwicklung . . . 24
3.2 Ablauf der Arbeitsplanerstellung . . . 26
3.3 Beispiel für die Ausgangsteilbestimmung . . . 28
3.4 Alternative Bearbeitungsvorgänge für die Bearbeitung einer Welle . . . 29
3.5 Beispiel für die Arbeitsvorgangsfolgeermittlung . . . 31
3.6 Beispiel für die Fertigungsmittelauswahl . . . 32
3.7 Beispiel für die Vorgabezeitbestimmung . . . 34
3.8 Gegenüberstellung von Planungsanlässen und -prinzipien . . . 35
3.9 Beispiel einer Entscheidungstabelle für das Vorbohren . . . 37
3.10 Aufteilung eines Werkstückes in Formelemente . . . 42
3.11 Werkstückrepräsentationsbaum für das Beispielwerkstück . . . 43
3.12 Schema für ein Materialelement . . . 45
3.13 Typen von Materialelementen . . . 45
3.14 Zusammensetzung von Bearbeitungsatomen zu -molekülen . . . 46
3.15 Schema für ein Bearbeitungsatom . . . 47
3.16 Aufteilung in Bearbeitungsbereiche . . . 49
3.17 Der Prozeß von der Konstruktion zur Fertigung . . . 50

4.1 Phasen, die vor dem Retrieval durchgeführt werden . . . 55
4.2 Berechnung der Ähnlichkeit zwischen aktuellem und altem Fall . . . 61
4.3 Fall B . . . 64
4.4 Arbeitsplanrepräsentation für Fall A . . . 66
4.5 Integration von fallbasierter und generativer Planung . . . 69
4.6 Verbesserung von Ähnlichkeitsmaßen . . . 71

Vorwort

Zu Beginn möchte ich kurz den Rahmen vorstellen, in dem ich diese Diplomarbeit durchgeführt habe und dessen Zustandekommen skizzieren. Die Motivation für die Ausschreibung dieser Arbeit war die Notwendigkeit für eine Aufarbeitung der Gebiete *fallbasierte Planung* und *Arbeitsplanung* und die Anwendung von fallbasierter Planung auf die Arbeitsplanung in Hinblick auf das SFB[1]-Projekt X9, das ab 1991 am Lehrstuhl für Künstliche Intelligenz – Expertensysteme (Prof. Dr. M. M. Richter) durchgeführt wird. Schwerpunkt dieses Projekts ist die Entwicklung von Wissensrepräsentationsformalismen für Ähnlichkeitsmaße und Lösungstransformationen für fallbasiertes Schließen sowie von Verfahren, mit deren Hilfe diese vom System selbst gelernt werden können. Als Anwendung dient die Arbeitsplanung in der Teilefertigung [Althoff & al. 90].

Um eine fachlich qualifizierte Unterstützung in dem Anwendungsgebiet aus dem Maschinenbau zu erhalten, wurde eine Zusammenarbeit mit dem CIM-Centrum des Lehrstuhls für Fertigungstechnik und Betriebsorganisation (Prof. Dr. G. Warnecke) durchgeführt. In diesem Rahmen trug ich durch die Implementierung einer formelementbasierten Werkstückrepräsentation zur Entwicklung eines wissensbasierten Arbeitsplanungssystems bei und hatte die Möglichkeit, die Wissenserhebung und eine prototypische Implementierung eines von mir entwickelten Ansatzes für die fallbasierte Arbeitsplanung durchzuführen. Die Zusammenarbeit kann als sehr erfolgreich bezeichnet werden und von einer weiteren Pflege erwarte ich interessante interdisziplinäre Ergebnisse.

Sehr herzlich bedanken möchte ich mich bei meinen beiden Betreuern Reinhard Praeger (AG Prof. Dr. Richter) und Christian Schulz (Leiter des CIM-Centrums, AG Prof. Dr. Warnecke) für ihre ausgezeichnete Betreuung in fachlichen, organisatorischen und persönlichen Fragen.

Für die Durchsicht des Manuskriptes danke ich außerdem Herrn Prof. Richter, Stefan Weß, Jürgen Paulokat, Mirko Radtke und Anke Geißler.

[1] Sonderforschungsbereich 314 für Künstliche Intelligenz – Wissensbasierte Systeme

Kapitel 1

Einleitung

Diese Diplomarbeit hat zwei Schwerpunkte: Zum einen werden die Gebiete *fallbasierte Planung* und *Arbeitsplanung* untersucht, bisherige Methoden und Systeme vorgestellt und deren Fähigkeiten und Schwächen herausgearbeitet. Dieser Teil macht etwa die Hälfte der Arbeit aus. Zum anderen wird, basierend auf den Ergebnissen dieser Untersuchungen, ein Konzept für die fallbasierte Arbeitsplanung vorgestellt, gegenüber alternativen Ansätzen abgegrenzt und die Möglichkeit der Realisierung anhand einer prototypischen Implementierung aufgezeigt. Während dem ersten Teil vorwiegend eine Literaturarbeit zugrundeliegt, beinhaltet der zweite Teil ausschließlich eigene Ansätze.

Der Aufbau der schriftlichen Ausarbeitung spiegelt diese Konzeption wider. In Kapitel 2 wird das Gebiet der fallbasierten Planung und in Kapitel 3 das Gebiet der Arbeitsplanung und Ansätze für deren Rechnerunterstützung aufgearbeitet. Kapitel 4 beschreibt die Konzeption meines Systems für die fallbasierte Arbeitsplanung. Referenzen auf die Kapitel 2 und 3 werden stets explizit gemacht. Somit ist es für einen in fallbasierter Planung bzw. Arbeitsplanung kundigen Leser möglich, diese Kapitel zu überspringen, ohne das Verständnis für Kapitel 4 einzubüßen. Kapitel 5 zeigt die Möglichkeit der Realisierung eines wissensbasierten Ähnlichkeitsmaßes für die fallbasierte Arbeitsplanung nach dem vorgestellten Konzept anhand einer prototypischen Implementierung auf. Sie reflektiert die in Kapitel 4 vorgestellte Sichtweise; daher sollte dieses Kapitel auch nach Kapitel 4 gelesen werden.

Zu Beginn jedes Kapitels stelle ich kurz die einzelnen Abschnitte vor, um sie zu motivieren.

Ich möchte nun einige Anmerkungen zu den Voraussetzungen machen, die ich an den Leser dieser Arbeit stelle: Grundbegriffe der Informatik und speziell der Künstlichen Intelligenz wie z.B. „Frame Problem“ werden als bekannt vorausgesetzt und nicht weiter erläutet. Da dies eine Diplomarbeit der Informatik ist, werden jedoch im Gegensatz dazu Begriffe aus dem Bereich des Maschinenbaus wie „Sachmerkmalsleiste“ stets bei ihrem ersten Auftreten kurz erklärt.

Kapitel 2

Fallbasierte Planung

In Kapitel 2 wird das Gebiet der fallbasierten Planung vorgestellt und gegen andere Planungsmethoden abgegrenzt. In Abschnitt 2.1 werden traditionelle KI-Ansätze für Planung erwähnt, um Planungsmethoden einzuführen, die sich teilweise auf die fallbasierte Planung übertragen lassen (wie z.B. hierarchische oder nichtlineare Planung), Schwächen hervorzuheben, die in der fallbasierten Planung behoben werden können und Anwendungen vorzustellen, in denen fallbasierte Planung nicht sinnvoll einsetzbar ist. Es findet in diesem Abschnitt keine ausführliche Vorstellung der Methoden selbst statt; dafür wird jeweils auf die zugrundeliegende Literatur verwiesen.

In Abschnitt 2.2.1 wird fallbasiertes Schließen und fallbasiertes Planen als spezielle Form des analogen Schließens eingeführt. Damit möchte ich den Kontext deutlich machen, aus dem sich das Gebiet des fallbasierten Schließens entwickelt hat und den Erinnerungs- und Transformationsprozeß aus diesem Kontext heraus motivieren.

In Abschnitt 2.3 schließlich wird unter anderem mit *CHEF* ein System für die fallbasierte Planung vollständig vorgestellt. Dies geschieht aus mehreren Gründen: Zum einen ist es eine der wenigen Implementierungen eines kompletten fallbasierten Planungssystems und seine Untersuchung macht Restriktionen und Probleme, aber auch Möglichkeiten eines solchen Systems deutlich. Zum anderen möchte ich anhand der Schwächen dieses Systems Ansätze motivieren, die für mein in Kapitel 4 vorgestelltes Systemkonzept gewählt wurden.

2.1 Planung – Traditionelle Ansätze

[1] Gegenüber der umgangssprachlichen Bedeutung von Planen, die auch Absichten („Ich plane, nach Australien zu gehen“) oder Entwürfe (z.B. „Stadtplan“) einschließt, betrachten wir hier nur *Handlungspläne*: Folgen von *Aktionen*, die einen definierten *Ausgangszustand* in einen *Zielzustand* transformieren. Aktionsplanung wird spätestens dann unabdinglich, wenn eine fehlerhafte Ausführung von Aktionen irreparable Folgen haben kann („Wenn die Brennelemente falsch eingesetzt werden, dann macht es Bumm und es fallen alle Kühe um (...)“[2]).

Die meisten traditionellen Planungsverfahren sind am Beispiel der *Blockwelt* entwickelt worden, einem einfachen Modell des Arbeitsplatzes eines stationären Roboters. Folgende ver-

[1] in Anlehnung an [Steel 87], [Herzberg 89], [Rich 83]
[2] aus Loriots Weihnachtssketch

einfachende Annahmen werden für die Blockwelt gemacht und kennzeichnen daher auch Beschränkungen vieler traditioneller Verfahren:

- Die Beschreibung der Modellwelt ist vollständig: Alle möglichen Aktionen sind bekannt und ihr Verhalten steht fest; jeder Weltzustand kann eindeutig und vollständig beschrieben werden (*STRIPS-Assumption* zur Lösung des Frame-Problems).
- Während der Planausführung treten keine unvorhergesehenen Ereignisse (z.B. Erdbeben) auf; insbesondere gibt es nur einen Akteur, der die Welt verändern kann.
- Alle Aktionen laufen streng sequentiell ab (im Gegensatz zur Welt des Backens, wo z.B. der Hefeteig vorbereitet wird und dann aufgeht, während man schon Füllung, Backform und Ofen vorbereitet).

2.1.1 Planen als Suche im Zustandsraum

Ereignisse in einer Modellwelt können als *Sequenz von Zuständen* dargestellt werden, wobei Aktionen den Übergang von einem Weltzustand in einen anderen beschreiben. So kann man hypothetische Ereignisse (in der Zukunft oder in der Vergangenheit) als Baum darstellen, wobei die Knoten Weltzustände repräsentieren (z. B. die Wurzel einen Ausgangszustand und die Blätter mögliche Zielzustände) und die Kanten Aktionen, die einen Zustand in den des adjazenten Knotens überführen.

Planung gestaltet sich dann als Suche im Zustandsraum, wobei die breite Palette der Suchverfahren Anwendung finden kann: uninformierte Suchverfahren wie Tiefen- oder Breitensuche und informierte Verfahren wie Bestensuche oder Dependency Directed Backtracking[3].

Um alle Zustände vollständig zu beschreiben, ohne jedoch unveränderte Teile in jedem Zustand explizit repräsentieren zu müssen, ist es notwendig, eine effiziente Lösung für das Frame Problem zu finden.

2.1.2 Rückwärtssuche mit Goalstack: STRIPS

Im *STRIPS*-System [Fikes & Nilsson 71] wird eine Rückwärtssuche durchgeführt, bei der ein Stack als Datenstruktur für die abzuarbeitenden Ziele verwendet wird, also eine Tiefensuche über den Zielen durchgeführt wird.

Neben den Performanznachteilen eines uninformierten Suchverfahrens besteht das Hauptproblem dieses Ansatzes in seiner Unfähigkeit, Interaktionen zwischen Teilzielen zu erkennen und aufzulösen. Teilziele werden in einer zufälligen Reihenfolge auf dem Keller abgelegt und in dieser Reihenfolge abgearbeitet. Das Zwischenprodukt während der Planung ist ein einstufiger, linearer Plan, eine streng festgelegte Abfolge von primitiven Operationen, die beim Auftreten eines Konfliktes in dieser Folge so weit zurückgenommen werden muß, bis der Konflikt aufgelöst ist. Das Ergebnis ist meist ein „Spaghettiplan“, der anschließend nachoptimiert wird.

[3] Zurücknahme von Entscheidungen unter dem Wissen über Abhängigkeiten zwischen Teilzielen

2.1.3 Nichtlineare Planung: NOAH

Nichtlineare Planung ist das Standardverfahren heutiger Planungssysteme [Steel 87]. Die Idee dabei ist, verschiedene Teilziele getrennt voneinander zu bearbeiten und nichts über ihre Reihenfolge auszusagen, solange keine Interaktionen auftreten; erst, wenn Interaktionen zwischen Teilzielen die Reihenfolge beschränken, wird diese festgelegt *(Least Commitment Strategy)*. Das Zwischenprodukt bei dieser Planung (und wenn keine abschließende Linearisierung durchgeführt wird u.U. auch das Endprodukt) ist ein nichtlinearer Plan, ein Netzwerk von Aktionen, Teilplänen und Teilzielen, deren Abfolge mehr oder weniger festgelegt sein kann.

Das *NOAH*-System [Sacerdoti 75] basiert auf der Least Commitment Strategy. Ein nichtlinearer Plan wird als Graph repräsentiert; für das Erkennen und Auflösen von Interaktionen von Teilzielen wird eine Menge von Hintergrundprozeduren, sogenannten *Critics*, verwendet.

2.1.4 Hierarchische Planung: ABSTRIPS

Umfangreiche Pläne können nicht in einem Schritt auf der Basis primitiver Aktionen generiert werden. Wenn man z.B. eine Reise plant, so beginnt man nicht damit, zu planen, wie man am elegantesten von der Wohnungs- zur Haustüre gelangt, sondern eher, welches Transportmittel man wählt (z.B. den Zug), über welche Städte man am sinnvollsten fährt (oder zumindest, wie man Information darüber erlangt) und dann erst, wie man am betreffenden Tag von seinem Bett bis zum Bahnhof gelangt. Die Idee dabei ist, erst einen Plan auf einer abstrakten Ebene zu generieren (z.B. zum Bahnhof gelangen, Zug fahren, zum Zielpunkt gelangen) und ihn anschließend zu verfeinern. Dafür muß definiert werden, was abstrakte Aktionen sind, ob man eine feste oder variable Anzahl von Hierarchieebenen vorgeben soll etc.

Im *ABSTRIPS*-System [Sacerdoti 74] werden solche Teilziele auf einer abstrakten Ebene geplant, deren Erfüllung als kritisch eingeschätzt wird.

2.1.5 Weiterführende Ansätze

Ich möchte nun noch einige z.T. neuere Planungsverfahren erwähnen (chronologisch angeordnet), die zum einen in ihren Zielen und zum anderen in ihrer Vorgehensweise von den bisher genannten abweichen:

Intermediate Execution z.B. PLANEX [Fikes & al. 72]: Hier geht es um die Verquickung von Planung und Ausführung. Unvorhergesehene Ereignisse während der Ausführung haben Einflüsse auf den Plan.

Opportunistisches Planen z.B. *HEARSAY-II* [Ermann & al. 80]: Es werden einzelne Aufgaben der Planung Spezialisten für bestimmte Zuständigkeitsbereiche übertragen. Als Realisierung wird eine *Blackboard-Architektur* verwendet.

Planen mit Constraints z.B. *MOLGEN* [Stefik 81a] : Methoden der Constraint-Propagierung werden angewendet, um sehr starke Abhängigkeiten zwischen Teilzielen aufzulösen.

Metaplanung z.B. *MOLGEN* [Stefik 81b]: Die Aufgaben, die bei der Planung zu erfüllen sind, werden selbst einem Planungssystem übergeben; es wird eine Planung der Planung durchgeführt.

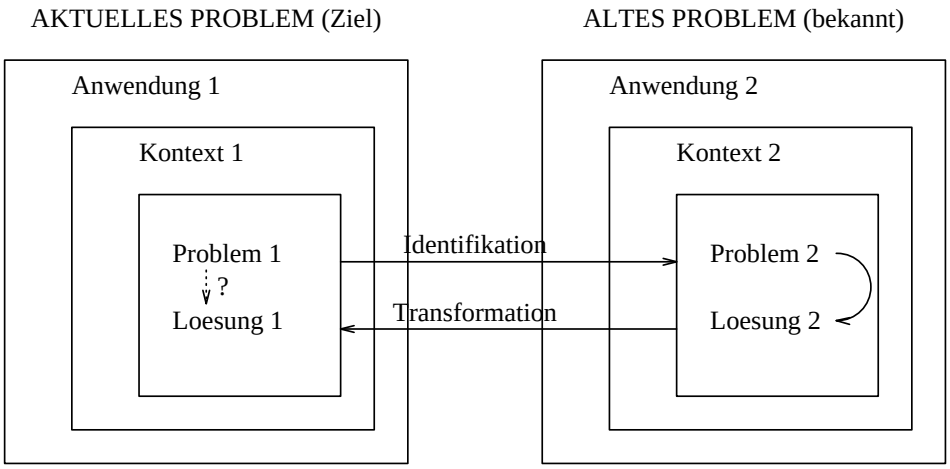

Abbildung 2.1: Analoges Schließen.

Intervallbasierte Planung z.B. TIMELOGIC [Allen 83], *DEVISER* [Vere 83]: Teilpläne werden mit den Zeitintervallen assoziiert, in denen sie ausgeführt werden; Relationen über den Intervallen erlauben es, Aussagen z.B. über die Parallelität von Aktionen oder Reihenfolgeabhängigkeiten zu machen.

Verteiltes Planen z.B. [Lesser & Corkill 87]: Verschiedene Agenten werden durch verschiedene kooperierende Planungssysteme koordiniert.

2.2 Analoges Schließen und fallbasiertes Schließen

2.2.1 Analoges Schließen

[4] *Analoges Schließen (Analogical Reasoning)* spielt eine wesentliche Rolle in menschlichen Denkprozessen. Unter Verwendung von Erfahrungen, die sich im Laufe der Zeit angesammelt haben, werden Probleme auf der Basis von Lösungen *analoger* – d.h. sich in Struktur oder Funktion entsprechender – Probleme gelöst. Dazu wird zu einem Problem in einem Kontext in einer speziellen Domäne ein analoges Problem z.B. eines anderen Kontextes in einer anderen Domäne erinnert und die bekannte Lösung dieses Problems bzw. der Problemlösevorgang auf das ursprüngliche übertragen (siehe Abbildung 2.1). Z.B. kann man in der Domäne „Planen von U-Bahn Fahrten“ im Kontext „U-Bahn Netz von New York“ ein spezielles Problem („Fahrt von A nach B“) lösen, indem man Wissen über Problemlösungen aus einem anderen Kontext („U-Bahn Netz von San Francisco“) derselben Domäne überträgt [Alterman 86].

Der Kreativitätsaspekt von analogem Schließen liegt darin, daß bestehendes Wissen auf neuen Gebieten eingesetzt wird.

[4] in Anlehnung an [Praeger 90] und Gespräche mit R. Praeger.

Die einfachste Form des analogen Schließens ist die wiederholte, eventuell modifizierte Anwendung alter Problemlösungen. Zwei Einschränkungen der allgemeinen Methode machen sie zu einem Spezialfall des analogen Schließens: Zum einen erfolgt kein Domänenwechsel. Zum anderen erfolgt im einfachen Fall weder ein Kontext- noch ein Problemwechsel; die Transformation ist in diesem Fall die Identität. Es wird dabei nicht der Problemlösevorgang, sondern nur die Problemlösung weiterverwendet. Die Vorgehensweise dient dann ausschließlich zur Abkürzung des Problemlöseprozesses. Hervorzuheben ist, daß in der allgemeinen wie in der eingeschränkten Form des analogen Schließens das Erinnern an alte Situationen eine wesentliche Rolle spielt.

Verschiedene Modelle menschlichen analogen Schließens wurden entwickelt und auf von Rechnern ausführbare Modelle übertragen. Als Beispiele sind hier zu nennen: Die *Structure Mapping* Theorie [Gentner 83, Gentner 88], *Purpose Directed Analogy* [Kedar-Cabelli 85, Kedar-Cabelli 88] und die *Dynamic Memory Theory* [Schank 82]. Während sich Gentner und Kedar-Cabelli hauptsächlich mit der Problem- und Lösungstransformation beschäftigten, stellte Schank die Wichtigkeit des Erinnerungsprozesses im analogen Schließen heraus.

Ich möchte im folgenden Abschnitt stellvertretend die Dynamic Memory Theorie vorstellen, da sich aus ihr das *fallbasierte Schließen (Case Based Reasoning)* als Modellierung der oben vorgestellten eingeschränkten Form des analogen Schließens (modifizierte Anwendung alter Problemlösungen) entwickelt hat. Schank untersuchte dabei die Organisation von Erfahrungen und deren Zugriff im menschlichen Gedächtnis. Seine Theorie modelliert den menschlichen Erinnerungsprozeß.

2.2.2 Die Dynamic Memory Theorie

[5] Hintergrund dieser Theorie sind Schanks Arbeiten auf dem Gebiet des Sprachverstehens. Aus der Erfahrung, daß bloße Aneinanderreihung von Inferenzen beim Sprachverstehen sehr schnell zu einer kombinatorischen Explosion führt, entwickelte er die *Scripts*, Wissensstrukturen für die Repräsentation von Standardereignissequenzen [Schank & Abelson 77]. Scripts liefern Hintergrundinformationen für Standardsituationen und erlauben Voraussagen über zukünftige Ereignisse.

Der entscheidende Nachteil von Scripts liegt darin, daß sie unabhängig voneinander definiert werden und keine gemeinsamen Strukturen haben. Veränderungen und Lernprozesse innerhalb eines Scripts haben daher keinen Einfluß auf andere Scripts; Generalisierung von Scripts zu allgemeineren Konzepten ist daher nicht möglich. Die Notwendigkeit für Wissensstrukturen, die Kontextübergreifendes Lernen ermöglichen, führte Schank von der Theorie der statischen Scripts zu der des *Dynamic Memory* [Schank 82], einem selbstorganisierenden Speicher, der seine interne Struktur ändert, sobald es erforderlich wird. Erfahrungen werden durch sogenannte *MOPs (Memory Organization Packets)* dargestellt. Wie Scripts charakterisieren auch *MOPs* Ereignissequenzen; im Unterschied zu Scripts können sie jedoch aus einer Reihe von Komponenten einer niedrigeren Ebene zusammengesetzt sein, die von mehreren *MOPs* geteilt werden können. Mit Hilfe von Indizes kann man auf die einem *MOP* zugehörigen individuellen Erfahrungen zugreifen. Die Indizes können beschriftet werden mit charakteristischen Merkmalen der Erfahrung oder mit Abweichungen von der Norm. Abweichungen werden stets anhand der Merkmale erklärt.

Drei Arten von Lernvorgängen finden auf dieser Struktur statt:

[5] in Anlehnung an [Schank & Leake 89], [Praeger 90]

- Erweiterung der Erfahrung durch Hinzufügen neuer Einzelerfahrungen.
- Lernen von *MOPs* und Indizes durch Generalisierung ähnlicher Erfahrungen *(similarity triggered generalization)*.
- Lernen von Fehlern und deren Behebung aus Erwartungsabweichungen *(failure driven learning)*.

Wenn Menschen auf unvorhergesehene Probleme stoßen, so versuchen sie oft, sie anhand früher aufgetretener Probleme zu erklären. In der Dynamic Memory Theorie werden Fehler und Probleme unter der Situation gespeichert, in der sie aufgetreten sind.

Charakteristisch für Schanks Ansatz ist, daß „sparsam“ gelernt wird, d.h. Lernvorgänge werden nur dann gestartet, wenn sie für das Verständnis erforderlich sind.

2.2.3 Fallbasiertes Schließen

[6] Bei dem *fallbasierten Schließen (Case Based Reasoning)* können Inferenzen direkt auf alten Fällen basieren. Ein *Fall* wird dabei als eine Struktur bezeichnet, die ein Problem mit dazugehöriger Problemlösung beinhaltet. Das Ziel dieser Vorgehensweise besteht in einer Verbesserung des Problemlöseverhaltens in Bezug auf Qualität der Ergebnisse und Effizienz des Problemlösevorgangs. Dies soll erreicht werden durch:

- Vermeidung von wiederholtem Aufwand durch Erinnerung an alte Fälle; kein stets von Grund auf neues Problemlösen.
- Verwendung guter alter Lösungen.
- Vermeidung früher gemachter Fehler.
- Generierung abstrakter Konzepte durch Generalisierung von konkreten Fällen, z.B durch Generalisierung der auf sie verweisenden Indizes.
- Möglichkeit der Erweiterung von Erfahrung durch Akquisition neuer Fälle.

Insgesamt soll fallbasiertes Schließen eine Abkürzung des Problemlöseprozesses leisten. Dazu sind folgende Schritte durchzuführen [Kolodner 87]:

1. Auffinden eines dem aktuellen Problem ähnlichen[7] alten Falles.
2. Konzentration auf die bzgl. der aktuellen Aufgabe geeigneten Teile des alten Falles durch Vergleich von beiden.
3. Verwendung dieser Teile, um eine geeignete Entscheidung für das aktuelle Problem zu treffen.
4. Analyse eines Feedbacks für die generierte Lösung.

[6] in Anlehnung an [Kolodner 87], [Bareiss 89], [Praeger 90]

[7] Ich verwende hier den Begriff „Ähnlichkeit“, wenn bei der Analogiebildung kein Domänen- und Kontextwechsel durchgeführt wird.

Einige Anmerkungen zu diesen Schritten:

Fälle werden in einer sogenannten *Fallbasis (Case Memory)* gespeichert. Ein gängiger Ansatz, der hier betrachtet werden soll, ist es, die Fallbasis unter Verwendung von *Indizes* zu strukturieren. Die Fallbasis kann außer Fällen auch generalisiertes Wissen, z.B. in Form von generalisierten Indizes enthalten.

Das Auffinden eines ähnlichen alten Falles erfolgt durch ein Durchsuchen der Fallbasis. Der Zugriff auf Fälle geschieht über Indizes, die entweder mit Merkmalen des Falles oder mit Abweichungen vom allgemeinen Konzept beschriftet sind. Man kann sich prinzipiell auch vorstellen, daß mehrere alte Fälle assoziiert und gemischt werden. Eine wichtige Anforderung an die Strukturierung der Fallbasis ist, daß das Wiederauffinden alter Fälle bei wachsender Fallbasis nicht (wesentlich) langsamer wird. Zum Vergleich: ein menschlicher Experte wird in seinen Entscheidungen bei wachsender Erfahrung nicht langsamer, sondern in aller Regel schneller.

Die Akquisition der Fälle und die Generierung von *MOPs* und Indizes kann vom Implementierer oder auch vom System selbst erfolgen. Es müssen im zweiten Fall folgende Probleme gelöst werden:

- Generierung neuer *MOPs*.
- Lernen neuer Indizes.
- Auswahl der zu speichernden Fälle; man kann z.B. folgende Strategien anwenden:
  - Alle Fälle werden behalten *(proximity)*.
  - Alle Fälle, die als fehlerhaft klassifiziert wurden, werden behalten *(growth)*.
  - Alle Fälle werden gesammelt und, nach und nach überflüssige Fälle eliminiert *(shrink)*.

Zur Auswahl der geeigneten Teile des gewählten Falles: In der Realität interessieren meist nicht alle Details eines alten Falles, sondern nur Ausschnitte. Je nach Art des alten Falles werden verschiedene Aktionen durchgeführt: Konnte der alte Fall erfolgreich gelöst werden, so konzentriert man sich auf die Merkmale des alten Falles, die mit denen des aktuellen übereinstimmen. Führte der alte Fall jedoch zu einem Fehler, so wird die Aufmerksamkeit auf die Teile des Falles gerichtet, die zu dem Fehler führten.

Zur Lösungstransformation: Falls der alte Fall erfolgreich gelöst werden konnte, sind folgende Vorgehensweisen denkbar:

- Verwende die alte Lösung.
- Verwende die alte Lösung und modifiziere sie auf der Basis der Unterschiede zwischen altem und aktuellem Fall.
- Übertrage die Inferenzmethode, mit der die Lösung gefunden wurde, auf das aktuelle Problem.
- Abstrahiere die beiden Problembeschreibungen zu einem allgemeinen Konzept, in das die abstrahierte Problemlösung für den alten Fall integriert wird, und wende diese auf den aktuellen Fall an.

Welchen Transformationsmechanismus man anwenden kann, hängt von der Komplexität der Anwendung und der Fähigkeit ab, Lösungen zu erklären. Wird eine alte Lösung ohne Modifikation weiterverwendet (wie dies z.B. bei der Diagnose üblich ist), so spricht man von einem *Case Matching System*. Wird jedoch die alte Lösung modifiziert (wie dies bei der Planung notwendig ist), so nennt man dies ein *Case Adaption System*. Dabei wird unterschieden zwischen einer reinen Wiederverwendung der Problemlösung und des Problemlösevorgangs.

2.3 Fallbasierte Planung

2.3.1 Fallbasiertes Schließen im Bereich der Planung

Was im Abschnitt 2.2.1 über menschliche Denkprozesse gesagt wurde, gilt insbesondere für das Planen. Mit folgendem Beispiel aus der Routenplanung möchte ich diese Sichtweise noch einmal verdeutlichen:

Zieht man in eine fremde Stadt, so wird man günstige Routen für häufig benutzte Wege, z.B. von der Wohnung zur Arbeit, anhand eines Stadtplanes (Achtung: kein Plan im hier verwendeten Sinne) generativ erkunden. Nach einigen Wochen wird man keinen Stadtplan mehr benötigen, da man die wichtigsten Straßen und Punkte der Stadt kennt. Dennoch wird man im Laufe der Zeit noch oft Wege planen müssen, die man noch nie vorher zurückgelegt hat, z.B. vom Finanzamt zur Stammkneipe. Dafür werden oft bekannte Pläne (z.B. vom Finanzamt zur Wohnung, von der Wohnung zur Stammkneipe) modifiziert und zu einem ausführbaren Plan für das aktuelle Problem kombiniert. Kontrolliert man einen so generierten Plan anschließend interessehalber anhand des Stadtplans, so stellt man häufig fest, daß er keinesfalls optimal ist; dennoch ist er durchführbar und die Planung selbst sehr effizient. Wege, die besonders oft zurückgelegt werden, z.B. von der Wohnung zur Arbeit (oder zur Stammkneipe), werden vor Antritt nicht mehr generativ geplant; sie werden eigentlich auch nicht aktiv aus dem Erinnerungsschatz geholt, sondern einfach ausgeführt. Man ertappt sich z.B. häufig dabei, in Gedanken eine besonders häufig durchgeführte Route abzufahren, obwohl man eigentlich eine andere Erledigung in dieser Gegend durchführen will.

Bei der *fallbasierten Planung (Case Based Planning)* versucht man, die Fähigkeiten und Performanz eines Planungssystems durch explizite Erinnerung und Wiederverwendung von alten Plänen zu steigern. Für die aktuelle Planungsaufgabe werden alte Pläne aus einer Fallbasis ausgewählt *(Retrieval)*, kombiniert und manipuliert *(Modification)*, so daß man einen Plan erhält, der die aktuelle Aufgabe erfüllt.

2.3.2 Relationen zu traditionellen Planungsansätzen

In Abschnitt 2.1 wurde Planen als Suche im Zustandsraum vorgestellt, wobei eine Folge von Aktionen gesucht wird, die einen Ausgangs- in einen Zielzustand überführen. Betrachtet man alte Pläne selbst als komplexe Planaktionen, die ausgewählt, u.U. kombiniert und angewendet werden, so läßt sich auch die fallbasierte Planung in diese Sichtweise einordnen. Ich werde in Abschnitt 4.5, wo es um die Integration von fallbasierter und generativer Planung geht, auf diesen Aspekt eingehen.

In Abschnitt 2.1 vorgestellte Planungsmethoden wie nichtlineare oder hierarchische Planung sind nicht notwendigerweise nur in der generativen sondern auch in der fallbasierten

Planung anwendbar. Dies ist gerade im Bereich der Arbeitsplanung sinnvoll; ich werde darauf in Abschnitt 4.7.1 kurz eingehen.

Vorteile fallbasierter Planung gegenüber generativer Planung sind die Vermeidung von wiederholter Planung in ähnlichen Situationen, die Möglichkeit des Lernens von neuem Planungswissen durch Akquisition neuer Fälle und Vermeidung alter Planungsfehler. Desweiteren kann man sich in komplizierten Domänen, in denen aus Gründen der Effizienz bei der generativen Planung nur lokal optimiert werden kann, durch die Verwendung von guten (z.B. global optimierten) Fällen eine Steigerung der Qualität der Ergebnisse erhoffen. Eine Abgrenzung von fallbasierter Planung zu traditionellen Ansätzen wird im letzten Teil von Abschnitt 2.3.4 vorgenommen.

2.3.3 Unterscheidungsmerkmale von CBP-Systemen

Ich möchte nun Unterscheidungsmerkmale von CBR[8]- und speziell CBP[9]-Systemen darstellen, anhand derer ich Systeme klassifizieren werde. Sie basieren vorwiegend auf eigenen Überlegungen und Analysen.

- Was enthält ein Fall? Im einfachsten Fall ist dies ein Paar bestehend aus den Planzielen und dem resultierenden Plan. Er kann z.B. auch Beziehungen zwischen Zielen und Planschritten oder Begründungen derselben enthalten. Welche Beschreibungssprache wird für einen Plan verwendet?
- Wie ist die Fallbasis strukturiert? Was wird wie generalisiert? Enthält die Fallbasis nur konkrete Fälle oder werden Klassen von Fällen zu abstrakten Konzepten zusammengefaßt; wie sieht dann ein abstrakter Plan aus? Welche Möglichkeiten eines effizienten Zugriffs auf Fälle sind vorhanden?
- Wie werden die Fälle akquiriert? Geschieht dies ausschließlich durch den Implementierer oder kann auch das System neue Fälle in die Fallbasis aufnehmen? Falls ja: wie wird eine Explosion der Fallbasis vermieden?
- Wird nur ein einziger Fall wiederverwendet oder können mehrere Fälle kombiniert werden?
- Welche Größen gehen in das Retrieval ein?
- Wird ausschließlich fallbasiert geplant oder kann im Falle von nichtsignifikanter Fallinformation auch generativ geplant werden?
- Welche Lernvorgänge finden statt? Welche Art von Hintergrundwissen wird dafür eingesetzt?
- Erfolgt die fallbasierte Planung hierarchisch, d.h. könnnen Teilpläne für die Erfüllung von Teilzielen wiederverwendet werden?
- Kann nichtlinear geplant werden?

[8] Case Based Reasoning.
[9] Case Based Planning.

2.3.4 CHEF: ein System für die fallbasierte Planung

[10] Ich möchte nun das *CHEF*-System [Hammond 89] detailliert vorstellen. Es ist eines der wenigen implementierten CBP-Systeme, das die in diesem Kapitel eingeführten Aspekte fallbasierter Planung, insbesondere Lernaspekte, realisiert.

Kristian J. Hammond arbeitete am *Yale AI Lab* unter Roger C. Schank und übertrug Schanks Konzept des Dynamic Memory auf die Planung. Sein System *CHEF* plant Rezepte der asiatischen Küche.

CHEF als System für die fallbasierte Planung modelliert *Planung durch Erinnerung (Planning as Remembering)*. Für die aktuelle Planungsaufgabe wird ein alter Fall aus der Fallbasis ausgewählt *(Retrieval)* und modifiziert *(Modification)*. Werden durch Tests, die mit einen Simulator durchgeführt werden, Fehler festgestellt, so werden diese erklärt und behoben *(Plan Repair)* und in Zukunft vermieden *(Anticipation & Avoidance)*.

Die wichtigsten Aspekte dieses Ansatzes sind:

- Organisation der Fallbasis: Wie werden Fälle gespeichert, um effizient auf sie zugreifen zu können?
- Lernaspekte. Im *CHEF*-System finden drei Arten von Lernen statt:
  - Lernen neuer Pläne.
  - Lernen von Fehlervorhersagen.
  - Lernen von Fehlerbehebungen.

Ich stelle nun die wichtigsten Module des *CHEF*-Systems vor:

Wiederauffinden eines alten Falles: der RETRIEVER

Der *RETRIEVER* selektiert den alten Fall aus der Fallbasis *(Plan Memory)*, dessen Planziele denen der aktuellen Aufgabe am ähnlichsten sind *(best match)*. Die Fälle im *Plan Memory* sind indiziert unter den Zielen, die sie erfüllen, und unter den Situationen (d.h. den Kontexten), in denen sie anwendbar sind. Für das Maß der Ähnlichkeit werden die Ziele nach einer fest vorgegebenen Gewichtung *(Goal Value Hierarchy)* verglichen. Der Grad der Ähnlichkeit zwischen einzelnen Zielen wird anhand einer festen Abstraktionshierarchie von Zielen *(Goal Abstraction Hierarchy)* bestimmt (siehe Abbildung 2.2)[11].

Modifikation des gewählten alten Falles: MODIFIER

Der *MODIFIER* bildet zusammen mit dem *RETRIEVER* den Kern des CBP-Systems. Er basiert auf einer Menge von Modifikationsregeln *(Library of Modification Rules)*, mit deren Hilfe folgende Operationen durchgeführt werden können (siehe Abbildung 2.3):

- Einfügen von Planschritten, um bisher nicht behandelte Ziele zu erfüllen.
- Löschen von Planschritten, die keine Ziele der aktuellen Aufgabe erfüllen.
- Substitution von Planteilen durch andere.

[10] nach [Hammond 89]; siehe auch [Hammond 86a] und [Hammond 86b].
[11] Die Abbildungen bezüglich des CHEF-Systems sind [Hammond 89] entnommen.

Abbildung 2.2: Beispiel für das Retrieval in *CHEF*.

Abbildung 2.3: Beispiel für die Modification in *CHEF*.

Speicherung neuer Fälle in der Fallbasis: der STORER

Um neue Pläne in der Fallbasis abzulegen, muß der *STORER* die Merkmale bestimmen, die als Indizes dem *RETRIEVER* für einen Zugriff zur Verfügung stehen sollen: die Ziele, die sie erfüllen und die Situationen, in denen sie anwendbar sind. Um das Retrieval von ähnlichen Fällen zu unterstützen, werden die Indizes unter Verwendung einer festen Taxonomie generalisiert.

Der *STORER* trägt nicht zur Lösung des aktuellen Falles bei, sondern unterstützt ausschließlich die Verbesserung des Systems über die Zeit durch das Lernen von neuem Fallwissen.

Behebung von Planungsfehlern: der REPAIRER

Ein Planungsfehler wird in *CHEF* als Fehler des Weltverständnisses des Systems angesehen. Er wird erklärt und begründet (und damit das Verständnis der Welt geändert) und anschließend behoben.

Die Fehlerbeschreibung und -erklärung wird anhand von Simulationsprotokollen generiert. Sie indiziert Fehlerbehebungsstrategien *(Repair Strategies)*, die auf den Plan angewendet werden.

Nach der Fehlerbehebung erhält man einen Plan, der nicht nur die geforderten Ziele erfüllt, sondern auch Probleme vermeidet. Da in der Fallbasis Pläne nicht nur durch die Ziele, die sie erfüllen, indiziert werden, sondern auch durch die Probleme, die sie vermeiden, und diese Indizes generalisiert werden, liegt hierin ein Lerneffekt, der es ermöglicht, in Zukunft ähnlich geartete Fehler zu vermeiden (siehe Abbildung 2.4).

Credit Assignment für Planungsfehler: der ASSIGNER

Tritt ein Planungsfehler auf, der repariert werden kann, so versucht *CHEF*, Fehler dieser Art in Zukunft zu vermeiden. Der *ASSIGNER* generiert aus der Fehlererklärung des *REPAIRERS* eine Menge von Inferenzregeln *(Plan Failure Predictors)*, die für einen neuen Fall das Vorliegen eines bekannten und lösbaren Problems anzeigen. Dieses Modul dient also zum Lernen von Problemvorhersagen zur Vermeidung von Problemen (siehe Abbildung 2.5).

Vorhersage von Problemen: der ANTICIPATOR

Der *ANTICIPATOR* prüft vor dem Retrieval, ob bei der aktuellen Planungsaufgabe ein bekanntes Problem auftritt. Dabei wendet er die vom *ASSIGNER* generierten Plan Failure Predictors an. Das Retrieval kann dann sowohl anhand der zu erfüllenden Ziele als auch der so generierten Problemvorhersagen *(Goals and Predictions)* erfolgen.

Das Zusammenwirken der einzelnen Module bei der Planung

Die aktuelle Planungsaufgabe wird *CHEF* anhand von Zielen *(Planner's Goals)* übergeben. Der *ANTICIPATOR* macht auf der Basis der vom *ASSIGNER* in früheren Planungsläufen generierten Plan Failure Predictors Problemvorhersagen für die Planung. Die Ziele und vorhergesagten Probleme werden dem *RETRIEVER* übergeben, der den best matching Plan aus dem Plan Memory selektiert. Der *MODIFIER* modifiziert den alten Plan unter Verwendung von statischen Modification Rules und dynamischen Critics, so daß er genau die Planziele erfüllt. In einem Simulator wird der Plan getestet und, falls Fehler (z.B. unerfüllte Ziele oder

Abbildung 2.4: Beispiel für den Plan Repair in *CHEF*.

Abbildung 2.5: Beispiel für die Generierung von Plan Failure Predictors in *CHEF*.

Abbildung 2.6: *CHEF*s Module

unerwünschte Seiteneffekte) auftreten, anhand der Simulationsprotokolle eine Fehlerbeschreibung und -erklärung generiert. Der *REPAIRER* wendet Repair Strategies zur Behebung des Fehlers an. Der reparierte Plan wird dann vom *STORER* unter den Zielen, die er erfüllt und den Problemen, die er vermeidet, im Plan Memory abgelegt. Desweiteren generiert der *ASSIGNER* aus der Fehlerbeschreibung Plan Failure Predictors für die zukünfige Vermeidung so gearteter Fehler (siehe Abbildung 2.6).

CHEFs Speicher

Ich möchte im folgenden auf die einzelnen Memories des *CHEF*-Systems eingehen. Sie werden charakterisiert durch die zu speichernden Objekte, das Vokabular zu deren Beschreibung und Retrieval-Mechanismen zum Zugriff auf diese.

Speicherung alter Fälle: Plan Memory

Pläne werden indiziert unter den Zielen, die sie erfüllen und den Problemen, die sie vermeiden. Die Pläne selbst werden nicht generalisiert, damit keine Detailinformation, die bei großer Ähnlichkeit zwischen aktuellem und altem Fall von Nutzen sein kann, verloren geht. Die

Abbildung 2.7: *CHEFs* Plan Memory

Indizes, die auf die Pläne verweisen, sind so weit wie möglich generalisiert. Das Maß für die Generalisierung von Fehlerindizes liefert die Erklärung des Fehlers; Zielindizes werden nach einer festen Taxonomie generalisiert (z.B. Broccoli zu Gemüse; siehe Abbildung 2.7).

Speicherung bislang aufgetretener Fehler: Memory of Failures

Ein Fehler wird beschrieben durch folgende Merkmale:

- den Fehlerzustand
- den Plan, in dem er aufgetreten ist
- die kausale Situation, die zu dem Fehler geführt hat und die Merkmale, die für die Vorhersage herangezogen werden können.

Ein für einen Fehler verantwortliches Merkmal wird generalisiert, um eine allgemeinere Beschreibung für die Fehlerursachen zu erhalten. Das Maß für die Generalisierung wird aus der kausalen Beschreibung abgeleitet (siehe Abbildung 2.8).

Regeln für die Modifikation: Memory of Modifiers

CHEF kennt zwei Arten von Modifikatoren:

Modification Rules werden in einer statischen Tabelle durch den zu modifizierenden Plan und den allgemeinen Typ der Ziele indiziert (siehe Abbildung 2.9).

Critics werden beim Auftreten von Fehlern gelernt; sie generieren feste Planschritte für spezielle Rezeptzutaten (siehe Abbildung 2.10).

Abbildung 2.8: Ausschnitt aus *CHEFs* Memory of Failures

Abbildung 2.9: Tabelle der Standard Modification Rules

Abbildung 2.10: *CHEFs* Critics

Speicherung von Fehlerbehebungsstrategien: Memory of Repairs

*CHEF*s Fehlerbehebungsstrategien *(Repair Strategies)* sind statisch und können nicht gelernt werden. Jede Strategie behandelt eine spezielle Problemart, die anwendungsunabhängig sein kann. Die Strategien sind in sogenannten *TOPs (Thematic Organization Packets)* gespeichert und werden indiziert durch eine kausale Beschreibung des Problemtyps (siehe Abbildung 2.11).

Hammonds Bewertung von CHEF

Nach Hammond ist ein *CBP*-System ist per Definition ein Planungssystem, das lernt. Daher grenzt er sein System ab gegen Planungssysteme auf der einen und lernende Systeme auf der anderen Seite.

Herkömmliche Planungssysteme ordnet er der *Create and Debug* Methode zu, wohingegen *CHEF* nach dem *Anticipate and Avoid* Ansatz arbeitet. Drei Hauptunterschiede stellt er zu allen bisherigen Planungssystemen heraus, die er am Beispiel der Systeme *STIRPS* [Fikes & Nilsson 71], *HACKER* [Sussmann 75], *NOAH* [Sacerdoti 75] und *PANDORA* festmacht [Wilenski 80, Wilenski 83]:

1. Generierung eines initialen Plans.
2. Reaktion auf Planungsfehler.
3. Vokabular zur Beschreibung und Speicherung von Plänen und Indizes.

Abbildung 2.11: Auswahl von *TOP*s

ad (1): In *CHEF* wird der initiale Plan durch Erinnerung an alte Fälle generiert, wobei neben den Zielen auch früher aufgetretene Fehler in die Betrachtung eingehen (Anticipate and Avoid). Auch in *STRIPS* können alte Pläne und Planteile wiederverwendet werden, allerdings wird nur ein exakter Abgleich der Ziele durchgeführt; Kontextinformationen bleiben außer acht. Auch bei *HACKER* und *NOAH* gehen Interaktionen zwischen Teilzielen nicht in die Planauswahl ein.

ad (2): Bei *CHEF* werden Fehler sowohl als Planungs- als auch als Erwartungsfehler behandelt, die ein fehlerhaftes Weltverständnis des Planers deutlich machen. Es wird im Gegensatz zu herkömmlichen Planungssystemen sowohl der Plan als auch das Planungssystem auf der Basis kausaler Fehlerbeschreibungen geändert.

ad (3): Dieser Punkt stellt die Grundlage für (1) dar: Pläne werden außer unter den Zielen, die sie erfüllen auch unter den Problemen, die sie vermeiden, gespeichert.

Hammond stellt drei Hauptunterschiede seines Ansatzes für Lernen gegenüber den Ansätzen des *Inductive Learning* [Lebowitz 80, Michalski & Larson 78, Winston 70] und des *Explanation Based Learning* [DeJong 83, Mitchell 83] heraus:

1. Was wird gelernt?
2. Wie wird es gelernt?
3. Wann wird gelernt?

ad (1): *CHEF* lernt das Erkennen von speziellen Situationen und Verfahren, darauf zu reagieren, d.h. das Erlernte hat einen funktionalen Charakter *(Learning of Functional Categories)*. Inductive Learning hingegen basiert ausschließlich auf semantik- und funktionslosen Merkmalen; es wird keine Unterscheidung zwischen definierenden (predicting) und resultierenden (resulting) Merkmalen eines Konzeptes gemacht.

ad (2): Bei *CHEF* wird das Credit Assignment auf der Basis von kausalen Erklärungen durchgeführt. Das einzige Kriterium für Signifikanz ist bei Verfahren des Inductive Learning die Häufigkeit, mit der Merkmale auftreten. Im Vergleich zu Methoden des Explanation Based Learning generiert *CHEF* nur dann Erklärungen, wenn dies den Lernprozeß unterstützt.

ad (3): Das Lernen in *CHEF* findet sparsam statt; d.h. Lernprozesse werden nur dann aktiviert, wenn dadurch seine Funktionalität als Planer verbessert wird.

2.3.5 Diskussion von CHEF

Eine Abgrenzung von *CHEF* und fallbasierter Planung allgemein gegenüber anderen planenden und lernenden Systemen hat Hammond selbst gegeben; eine kritische Betrachtung seines Systemes erfolgte jedoch nicht. Ich möchte nun das *CHEF*-System anhand der in Abschnitt 2.3.3 vorgestellten Kriterien analysieren und dabei einige kritische Anmerkungen machen.

Was enthält ein Fall?

Ein Fall in *CHEF* enthält einen linearen Plan und die Summe der Indizes, die zu diesem Plan führen. Ein Plan repräsentiert ein Kochrezept und enthält dessen Zutaten, eine lineare Liste der durchzuführenden Aktionen mit den benötigten Zeiten und einen allgemeinen Rezepttyp. Er enthält keine kausalen Abhängigkeiten zwischen Aktionen oder Begründungen derselben. Die Indizes beschreiben erfüllte Ziele und vermiedene Probleme des Plans. Es findet keine Assoziation zwischen einzelnen Indizes und Teilen des Plans statt.

Wie ist die Fallbasis strukturiert?

Die Fallbasis ist als Graph organisiert, in dem ausschließlich die Blätter mehrere Vorgänger haben können; ansonsten hat er Baumstruktur. Die Blätter enthalten konkrete Pläne; die Kanten sind mit Indizes beschriftet. Die Indizes unterliegen einer festen Hierarchie, die die Stufen des Graphen definiert. Somit ist die Dynamik des Speichers eingeschränkt. Es gibt keine Zusammenfassung von mehreren Fällen zu abstrakten Konzepten. Planung durch Verfeinerung von Skelettplänen ist nicht möglich.

Wie werden die Fälle akquiriert?

Das System ist in der Lage, neue Fälle in die Fallbasis aufzunehmen. Dazu werden Indizes unter Verwendung einer festen Taxonomie generalisiert. Indizes, die Fehler beschreiben, werden generiert. Alle neu auftretenden Fälle werden übernommen; es erfolgt weder eine Auswahl von charakteristischen Fällen noch sind Vergessensprozesse vorgesehen, die eine Explosion der Fallbasis verhindern.

Wieviele alte Fälle werden für die aktuelle Planungsaufgabe herangezogen?

CHEF verwendet stets genau einen alten Fall für die Planung. Das System ist nicht in der Lage, zwei Kochrezepte zu kombinieren.

Welche Größen gehen in das Retrieval ein?

In das Retrieval gehen nur die erfüllten Ziele und die vermiedenen Probleme ein, nicht jedoch der Plan selbst. Es wird der Überlegung keine Rechnung getragen, daß verschiedene Pläne für dieselbe Aufgabenstellung unterschiedliche Relevanz für die Wiederverwendung in anderen Situationen haben können.

Wird ausschließlich fallbasiert geplant?

Ja. *CHEF* kann nicht entscheiden, ob seine Fallinformation ausreichend für die aktuelle Aufgabe ist oder nicht. *CHEF* ist nicht in der Lage, generativ zu planen.

Welche Lernvorgänge finden statt?

CHEF lernt Pläne, Fehlervorhersagen und deren Behebungen. Das Retrieval wird nur insofern verbessert, daß neue Fehler vorhergesagt werden können.

Erfolgt die Planung hierarchisch?

Die Modification Rules können als eine Art Skelettpläne für Teilziele angesehen werden. Es wird jedoch nicht erst ein Plan auf einem abstrakten Level generiert und anschließend verfeinert.

Erfolgt die Planung nichtlinear?

Nein. Die Planung ist linear; es besteht keine Möglichkeit, Reihenfolgen zwischen verschiedenen Planteilen offen zu lassen, wie das eigentlich bei Rezepten üblich ist.

Einige der hier vorgestellten Kritikpunkte am *CHEF*-System werden in meinem in Kapitel 4 vorgestellten Ansatz für die fallbasierte Planung anders behandelt werden.

Kapitel 3

Arbeitsplanung

In diesem Kapitel möchte ich das Gebiet der *Arbeitsplanung in der Teilefertigung* vorstellen und dabei besonders die Probleme herausarbeiten, die bei deren Rechnerunterstützung auftreten. Das Kapitel unterteilt sich in drei Abschnitte.

In Abschnitt 3.1 wird der Begriff der Arbeitsplanung eingeführt und seine Stellung in der technischen Auftragsabwicklung beschrieben. Die Beschreibung von Aufgaben und Methoden der Arbeitsplanung erfolgt unabhängig davon, ob sie manuell oder rechnerunterstützt durchgeführt wird.

In Abschnitt 3.2 werden bisher realisierte Ansätze für eine Rechnerunterstützung der Arbeitsplanung vorgestellt. Insbesondere werden Mängel dieser Ansätze aufgezeigt; einige dieser Defizite können in meinem in Kapitel 4 vorgestellten Ansatz für die fallbasierte Arbeitsplanung kompensiert werden.

Im letzten Abschnitt wird beispielhaft die Konzeption eines Systems für die Arbeitsplanung und deren Integration in ein CIM-Konzept vorgestellt. Dies geschieht zum einen, um Grundlagen für das Verständnis der in Kapitel 5 vorgestellten prototypischen Implementierung meines Ansatzes für die fallbasierte Arbeitsplanung zu legen und zum anderen, um die Anforderungen anderer CIM-Module an ein Arbeitsplanungssystem darzustellen.

3.1 Aufgaben und Methoden der Arbeitsplanung

3.1.1 Einordnung der Arbeitsplanung in die technische Auftragsabwicklung

[1] Unter dem Begriff *technische Auftragsabwicklung* versteht man den Prozeß vom Konstruktionsauftrag für ein Werkstück (z.B. Welle, Zahnrad) bis zum fertigen Erzeugnis. Dabei werden die Phasen *Konstruktion, Arbeitsvorbereitung, Fertigung* und *Montage* durchlaufen, wobei Konstruktion und Arbeitsvorbereitung in der Regel auftragsneutral vor der Fertigung durchgeführt werden (siehe Abbildung 3.1)[2].

Es soll hier auf den Bereich der Arbeitsvorbereitung eingegangen werden, wobei besonders die Schnittstellen zur Konstruktion und Fertigung interessieren.

[1] In Anlehnung an [Eversheim 89].

[2] Die Abbildungen in diesem Abschnitt stammen aus [Eversheim 89].

Abbildung 3.1: Die Stellung der Arbeitsvorbereitung in der technischen Auftragsabwicklung.

Der Bereich der Arbeitsvorbereitung kann untergliedert werden in die Bereiche *Arbeitsplanung* und *Arbeitssteuerung* .

Im Rahmen der Arbeitsplanung wird festgelegt, *was wie womit* (z.B. mit welcher Gruppe von Maschinen – eine konkrete Maschine wird nicht festgelegt) hergestellt werden soll; zu den Aufgaben der Arbeitssteuerung gehört es, vorzugeben, *wieviel wann wo* (z.B. auf welcher konkreten Maschine) und *durch wen* zu fertigen ist.

Die Tätigkeiten der Arbeitsplanung werden in *kurzfristige* und *langfristige* Aufgaben gegliedert. Während kurzfristig die Fertigung und Montage für einzelne Aufträge geplant wird, ist es Ziel der langfristigen Planungsaufgaben, geeignete Maßnahmen für die wirtschaftliche Auslegung dieser Bereiche zu entwickeln.

Zu den *kurzfristigen* Aufgaben der Arbeitsplanung zählt man die Erstellung von:

Arbeitsplänen: sie beschreiben detailliert die durchzuführenden Fertigungsschritte.

NC-Programmen: das sind Anweisungen für NC[3]-Maschinen. Sie enthalten Steuerinformationen wie z.B. Vorschub oder Drehzahl von Werkzeugen.

Montageplänen: sie beschreiben, wie einzelne Teile zu Komplexteilen zusammenbaut werden.

Zu den *langfristigen* Aufgaben gehören:

[3]Numeric Control

Investitionsplanung: z. B. Entscheidungen über die Anschaffung neuer Maschinen.

Methodenplanung: z.B. Umstellung auf bessere Planungs- und Fertigungsmethoden.

Lagersortenplanung .

Kurzfristige und langfristige Planungsaufgaben sollen nach bestimmten Kriterien optimiert werden: Kosten (Lohn-, Material- und Maschinenkosten) werden minimiert, die Qualität der gefertigten Produkte maximiert.

Aufgabe der *Arbeitssteuerung* ist die termin-, kapazitäts- und mengenbezogene Planung und Steuerung der Fertigungs- und Montageprozesse. Dabei sollen Zeiten und Kosten minimiert und Kapazitätsauslastungen optimiert werden.

Zu ihren Aufgaben gehören:

Materialdisposition: Bereitstellung und notfalls Beschaffung der für einen Auftrag benötigten Teile.

Termin- und Kapazitätsplanung: Zeitliche Zuteilung der Aufträge zu Mitarbeitern und Fertigungsmitteln.

Werkstattsteuerung: Organisation des Fertigungsprozesses selbst, z.B. Zustellung der benötigten Unterlagen für die einzelnen Mitarbeiter; insbesondere: flexible Reaktion auf nicht vorhergesehene Zwischenfälle.

3.1.2 Ziele der Arbeitsplanerstellung

[4] Die „Eingaben“ der Arbeitsplanerstellung sind die im Bereich Konstruktion erstellten Zeichnungen und Stücklisten für ein zu fertigendes Teil und ein Werkstattmodell, in dem die zur Verfügung stehenden Werkzeugmaschinen beschrieben werden. Ausgegeben wird ein auftragsneutraler Arbeitsplan, eine detaillierte Beschreibung der einzelnen Arbeitsgänge mit Angaben über die verwendeten Maschinen, die benötigten Zeiten und Kosten, jedoch ohne Angabe von Stückzahl, Produktions- und Liefertermin.

Im folgenden werden die bei der (z.B. manuellen) Arbeitsplanerstellung zu erfüllenden Ziele vorgestellt (siehe Abbildung 3.2).

Bestimmung eines Ausgangsteils

Im allgemeinen können für die Fertigung eines Werkstückes verschiedene Rohteile als Ausgangsteile Verwendung finden. Dies können zum einen Rohlinge (z.B. geschmiedete Zylinder), zum anderen sogenannte Halbzeuge (z.B. Profileisen) sein. Die Entscheidung, welches Rohteil verwendet werden soll, hängt von technologischen, wirtschaftlichen und zeitlichen Kriterien ab:

Die technologischen Kriterien resultieren aus den verfahrensspezifischen Eigenschaften der Rohteile wie Festigkeit, Korrosionsbeständigkeit etc. So muß man beispielsweise je nach Beanspruchung eines Zahnrades entscheiden, ob es aus weichem oder harten Metall gefertigt werden muß.

Es wird die Auswahl eines kostengünstigen Rohteils angestrebt. In die Kosten geht jedoch nicht nur der Preis für das Rohteil selbst ein, sondern auch der Aufwand für die Fertigung

[4] In Anlehnung an [Eversheim 89].

Abbildung 3.2: Ablauf der Arbeitsplanerstellung.

ausgehend aus diesem. So ist z.B. ein Gußteil in der Regel teurer als Stangenmaterial; jedoch gestaltet sich die Fertigung als weniger aufwendig, da der Konturverlauf des Gußteils dem Fertigteil angepaßt ist. In der betrieblichen Praxis werden für Abschätzungen der Wirtschaftlichkeit eines Rohteils meist Tabellen mit Erfahrungswerten, sogenannte *Relativkostenkataloge*, verwendet.

Die Bestimmung eines Ausgangsteils ist eine komplexe Selektionsaufgabe, in die sehr viele Optimierungskriterien eingehen und deren Güte einen bedeutenden Einfluß auf die Güte des gesamten Arbeitsplans hat.

Ein zeitkritischer Faktor bei der Fertigung ist die Beschaffung der Rohteile. Wenn es besonders lang dauert, ein spezielles Rohteil zu besorgen, so kann es im Sinne einer zügigen Auftragsabwicklung sinnvoll sein, sich für ein Teil zu entscheiden, das etwas teuerer, aber dafür schneller zu beschaffen ist.

Gerade dieser Aspekt zeigt, daß es sinnvoll ist, die Ausgangsteilbestimmung zu einem möglichst frühen Zeitpunkt durchzuführen. Sie sollte, wenn möglich (z.B. wenn nur ein Rohteil in Frage kommt), schon während des Konstruktionsprozesses stattfinden (siehe Abbildung 3.3).

Bestimmung von Arbeitsvorgängen und Festlegung deren Reihenfolge

Dies ist die eigentliche Planungsaufgabe im Sinne meiner Definition aus Abschnitt 2.1: Ermittlung einer Folge von Aktionen (sog. *Arbeitsvorgänge*), durch die ein Rohteil durch Änderung von Form (z.B. Fräsen) und/oder Stoffeigenschaft (z.B. Härten) schrittweise in das gewünschte Fertigteil überführt werden kann.

Unter einem Arbeitsvorgang versteht man hier eine Einheit, die von einem Arbeiter oder einer Arbeitsgruppe an einem bestimmten Arbeitsplatz zusammenhängend ausgeführt wird. Beispiele für Arbeitsvorgänge sind: Bohren, Sägen, Schleifen, Drehen, Fräsen. Wie bei allen Planungsvorgängen (zumindest solchen, die auch diesen Namen verdienen), gibt es hier im allgemeinen viele Möglichkeiten, ans Ziel zu kommen. Sie unterscheiden sich in der Auswahl und der Reihenfolge der Aktionen. Als Beispiel sollen die drei möglichen Arbeitsvorgangsfolgen für die Fertigung einer Welle dienen (siehe Abbildung 3.4).

Als erstes muß ermittelt werden, welche Verfahren für das zu fertigende Teil technologisch zulässig sind. Dazu gehört z.B., daß das Verfahren die geforderten Qualitätsanforderungen erfüllen kann.

Aus verschiedenen technologisch zulässigen Alternativen soll die beste ausgewählt werden, wobei folgende Kriterien die Güte eines Arbeitsplans bestimmen:

Rohteilkosten werden minimiert. In eine sinnvolle Kostenrechnung sollte jedoch nicht der Anschaffungspreis, sondern der Realwert eines Rohteils eingehen. Liegt z.B. ein Teil schon mehrere Jahre im Lager und ist abzusehen, daß es keine weitere Verwendung mehr haben wird, so hat es keinen Wert mehr, obwohl es u.U. in der Anschaffung sehr teuer war. In der Realität der betrieblichen Praxis wird dies jedoch nicht berücksichtigt.

Maschinenkosten werden minimiert. Hier können sehr viele verschiedene Kriterien wie Anschaffung, Wartung, Auslastung der Maschine etc. eingehen.

Lohnkosten werden minimiert. Da die Arbeitszeit bei der Fertigung an den Maschinen verbracht wird, rechnet man die Lohnkosten üblicherweise in die Maschinenkosten ein.

Abbildung 3.3: Beispiel für die Ausgangsteilbestimmung.

Abbildung 3.4: Alternative Bearbeitungsvorgänge für die Bearbeitung einer Welle.

Fertigungszeit wird minimiert. Dies ist besonders wichtig bei zeitkritischen Aufträgen.

Qualität der gefertigten Teile soll optimiert werden.

Um sich die Komplexität des Zusammenspiels dieser Kriterien bewußt zu machen, betrachte man folgende Situation: Ein guter Kunde der Firma vergibt einen kleinen Auftrag, von dessen Ausführung der Abschluß eines größeren Geschäfts abhängt. In diesem Falle spielen alle Kostenminimierungen (Punkte 1-3) keine Rolle mehr; es geht nur noch darum, den Auftrag schnell (4) und in bester Qualität (5) durchzuführen. Ein guter Arbeitsplan für diese spezielle Situation sieht anders aus als ein guter Plan für die Fertigung einer großen Serie.

Desweiteren wird sich ein guter Arbeitsplan für die Fertigung bei guter Auftragslage (hohe Maschinen- und Personalauslastung) von einem guten Arbeitsplan während einer Auftragsflaute unterscheiden.

Da jedoch in der Regel die Fertigungssituation zur Zeit der Arbeitsplanerstellung nicht bekannt ist, werden auftragsneutrale Arbeitspläne erstellt. Das bedeutet, daß nur nach den ersten drei oben erwähnten Kriterien optimiert werden kann; es wird angestrebt, *kostengünstige* Pläne zu generieren. Als Hilfsmittel für die manuelle Arbeitsvorgangsfolgeermittlung dienen oft verfahrenstechnische Relativkostenkataloge (siehe auch Abbildung 3.5).

Die Optimierung von Arbeitsvorgängen und die Feststellung von Kriterien dafür stellt eines der großen Probleme der Arbeitsplanung dar.

Ein weiteres großes Problem bildet die Festlegung der Reihenfolge von Arbeitsvorgängen, da in der Regel zwischen diesen sehr große Abhängigkeiten bestehen.

Zuordnung von Fertigungsmitteln zu Arbeitsvorgängen

Den einzelnen Arbeitsvorgängen müssen Fertigungsmittel zugeordnet werden. Unter *Fertigungsmitteln* versteht man (Hand-)Werkzeuge und Werkzeugmaschinen mit den dazugehörigen Vorrichtungen (z.B. Spannfutter), die für den (manuellen) Arbeitsplaner in der Regel in Form einer firmeneigenen Kartei zur Verfügung stehen. Während die Arbeitsvorgangsfolgeermittlung relativ betriebsunabhängig durchgeführt werden kann, bezieht sich die Fertigungsmittelauswahl stets auf die im speziellen Betrieb vorhandenen Ressourcen.

Die Auswahl wird in der manuellen Arbeitsplanerstellung typischerweise in zwei Schritten durchgeführt: zuerst wird zu jedem Arbeitsvorgang eine Maschine und dann zu jeder Maschine die entsprechenden Werkzeuge ausgewählt (siehe Abbildung 3.6).

Die Zuordnung von Fertigungsmitteln zu einer festen Folge von Arbeitsvorgängen ist relativ einfach, da häufig Anforderungen der Arbeitsvorgänge an die Fertigungsmittel deren Auswahl stark einschränken. Allerdings bedingen häufig die Fertigungsmittel selbst (z.B. Spezialmaschinen und -werkzeuge) spezielle Arbeitsvorgänge, so daß die Festlegung der Arbeitsvorgänge und deren Reihenfolge in der Regel ohne Kenntnisse der Fertigungsmittel unmöglich ist.

Bestimmung von Vorgabezeiten

Für jeden Arbeitsvorgang muß eine Feinplanung durchgeführt und darauf basierend die dafür benötigte Sollzeit ermittelt werden. Sie liefert Informationen für Termin- und Kosten- und Investitionsplanung und dient als Referenzzeit beim Akkordlohnverfahren.

Abbildung 3.5: Beispiel für die Arbeitsvorgangsfolgeermittlung.

Abbildung 3.6: Beispiel für die Fertigungsmittelauswahl.

Die Feinplanung wird meist in zwei Schritten durchgeführt: Zuerst wird die Werkzeugeinsatzreihenfolge und dann werden die Maschineneinstelldaten wie z.B. Werkzeugvorschub oder Spindeldrehzahl bestimmt.

Die für die einzelnen Tätigkeiten erforderlichen Zeiten werden meist anhand von Tabellen und Diagrammen ermittelt und aufaddiert (siehe Abbildung 3.7).

Somit ist es für eine feste Folge von Arbeitsvorgängen und ihrer Zuordnung zu Fertigungsmitteln relativ einfach, Vorgabezeiten zu ermitteln. Da jedoch die Fertigungszeit ein wesentlicher Kostenfaktor für die Fertigung und somit Optimierungsfaktor für die Arbeitsplanung ist, kann eine gute Arbeitsvorgangsfolgeermittlung und Fertigungsmittelauswahl nicht ohne Wissen über Fertigungszeiten durchgeführt werden.

Wurde bei der Maschinenauswahl eine NC-Maschine für einen Arbeitsvorgang bestimmt, so muß ein NC-Steuerprogramm erstellt werden. Es enthält die geometrischen Angaben zu allen Werkzeugbewegungen, die oben erwähnten Maschineneinstelldaten und Zusatzinformationen bezüglich Werkzeugwechsel, Kühlmittel etc.

Es ergibt sich als grundsätzliche Schwierigkeit bei der Arbeitsplanerstellung, daß die einzelnen vorgestellten Ziele nur unter enormer Einbuße der Planqualität unabhängig voneinander sequentiell durchgeführt werden können, da sie untereinander starke Abhängigkeiten aufweisen: das Ausgangsteil soll so gewählt werden, daß die gesamte Fertigung wenig aufwendig wird, die Arbeitsvorgänge bedingen Fertigungsmittel und umgekehrt und Vorgabezeiten bedingen deren Auswahl.

3.1.3 Anlässe und Methoden der Arbeitsplanerstellung

Unterschiedliche Situationen für die Arbeitsplanerstellung bedingen verschiedene Planungsmethoden, die in der betrieblichen Praxis entwickelt wurden. Ich möchte sie einander gegenüberstellen. Folgende Planungsanlässe können unterschieden werden je nach vorliegender Situation:

Neuerstellen eines Arbeitsplans wird notwendig, wenn für ein bestimmtes Werkstück bislang noch kein Arbeitsplan generiert worden ist.

Ändern eines schon früher generierten Arbeitsplans für ein Werkstück kann durch eine Änderung der Betriebsmittel, z.B. Anschaffung einer Spezialmaschine, notwendig werden.

Wiederholen eines früher generierten Arbeitsplans für ein Werkstück ist dann möglich, wenn keine Änderung von Betriebsmitteln eine Änderung des Arbeitsplans bedingt.

Die folgenden Planungsprinzipien sowohl für die manuelle als auch für die rechnergestützte Arbeitsplanerstellung wurden in der betrieblichen Praxis entwickelt:

Generierungsprinzip: Eine generative Planung wird in der Regel für Teile durchgeführt, die in dieser Art noch nie geplant oder gefertigt wurden. Dabei stützt sich der Arbeitsplaner auf Tabellen, Handbücher, Zeitenberechnungsformulare, unternehmensspezifische Konventionen und im wesentlichen auf seine Erfahrung. Auf dieser Basis werden in der Regel die Phasen Ausgangsteilbestimmung, Arbeitsvorgangsfolgeermittlung, Fertigungsmittelauswahl und Vorgabezeitbestimmung in dieser Reihenfolge durchgeführt.

Abbildung 3.7: Beispiel für die Vorgabezeitbestimmung.

	Planungsanlaß		
Planungsprinzip	Neuerstellen	Ändern	Wiederholen
Generierungsprinzip	Neuplanung		
Ähnlichkeitsprinzip	Ähnlichkeitsplanung	Änderungsplanung	Wiederholplanung
Variantenprinzip	Variantenplanung	Variantenplanung	

Abbildung 3.8: Gegenüberstellung von Planungsanlässen und -prinzipien.

Ähnlichkeitsprinzip: Neben den oben angeführten Unterlagen verwendet der Arbeitsplaner bei der Planung nach dem Ähnlichkeitsprinzip eine unternehmenseigene Arbeitsplankartei, aus der der Plan für ein ähnliches oder das identische Werkstück ausgewählt und evtl. modifiziert wird. Die Suche nach einem ähnlichen Teil richtet sich nach der Organisation der Kartei und spiegelt in hohem Maße die Erfahrung des Arbeitsplaners wider.

Variantenprinzip: Unter Varianten versteht man Werkstücke, die einer sogenannten Teilefamilie angehören, die sich durch die Existenz eines Standardarbeitsplans für alle Teile der Familie auszeichnet. Für die Generierung eines speziellen Arbeitsplans müssen lediglich teilespezifische Parameter in den Standardarbeitsplan eingetragen werden. Da sich Standardarbeitspläne jedoch nur für eingeschränkte Teilefamilien aufstellen lassen, kann diese Planungsmethode nur in einem begrenzten Produktspektrum Anwendung finden.

Bei dieser Einteilung geht es nur um die Planungsprinzipien, die der (menschliche) Arbeitsplaner systematisch anwendet. Natürlich wird der Experte bei der generativen Planung neben den Planungsunterlagen wie Zeichnung und Tabellen auch die Erfahrungen aus alten Plänen einsetzen; jedoch verwendet er in diesem Fall nicht explizit einen alten Arbeitsplan aus der Kartei.

Verschiedene Planungsanlässe machen die Anwendung unterschiedlicher Planungsprinzipien sinnvoll; Abbildung 3.8 verdeutlicht dies.

3.2 Rechnergestützte Arbeitsplanung

3.2.1 Arbeitsplanung als Engpaß der technischen Auftragsabwicklung

[5] Arbeitsplanung als Teil der Arbeitsvorbereitung steht im Prozeß der technischen Auftragsabwicklung am Übergang von der Konstruktion zur Fertigung. Durch zunehmende Rechnerunterstützung respektive (Teil-)Automatisierung traten Rationalisierungseffekte in diesen der Arbeitsplanung vor- und nachgelagerten Bereichen auf. Da es jedoch bislang für eine Automatisierung der Arbeitsplanung nur unzureichende Konzepte gibt, wird sie noch heute in vielen Betrieben manuell durchgeführt und wird daher als *Engpaß der technischen Auftragsabwicklung* [Heiob 88] bezeichnet.

[5] In Anlehnung an [Heiob 88, Schulz 88].

Folgende Vorteile gegenüber der manuellen Arbeitsplanung erwartet man von CAPP[6]-Systemen:

- Entlastung qualifizierter Mitarbeiter von Routinetätigkeiten.
- Verkürzung von Durchlaufzeiten.
- Verbesserung der Planungsqualität, d.h. Generierung kostengünstiger Arbeitspläne und objektiver Vorgabezeiten.
- Vereinheitlichung der Pläne und des Planungsvorgangs; dadurch bessere Nachvollziehbarkeit der Planung und Wiederverwendung der Pläne.

Folgende Eigenschaften der Domäne stellen Anforderungen an CAPP-Systeme:

- Viele Unternehmen haben interne Konventionen und Beschränkungen. CAPP-Systeme müssen einfach an solche unternehmensspezifische Eigenheiten anpaßbar sein.
- Richtlinien und Ressourcen eines Unternehmens und der Stand der Produktionstechnik unterliegen einer hohen Dynamik. Daher muß das Planungsverhalten eines CAPP-Systems einfach modifizierbar sein.
- Es müssen Interaktionsmöglichkeiten für eine Verifikation und u.U. Modifikation der Planungsergebnisse durch den Benutzer zur Verfügung gestellt werer.

3.2.2 Entwicklung der CAPP-Systeme

[7] Schon Anfang der siebziger Jahre wurden CAPP-Systeme erstellt. Da in dieser Zeit jedoch Dialogsysteme und Arbeitsplatzrechner noch nicht verbreitet waren, liefen diese Programme meist im Stapelbetrieb ab. Infolgedessen war keine Interaktion zwischen Experten und System möglich. Die Planungsergebnisse waren meist weder verständlich noch befriedigend und mußten oft anschließend vom Experten in großen Teilen revidiert werden.

Durch vermehrten Einsatz von Kleinrechnern kamen schließlich Ende der siebziger und im Laufe der achtziger Jahre vermehrt Programme zum Einsatz, die relativ gut algorithmisierbare Routinetätigkeiten wie z.B. Auswahl technologisch möglicher Rohteile oder Berechnung von Vorgabezeiten durchführten, die komplexe Planungstätigkeiten aber menschlichen Experten überließen. Beispiele für algorithmische Arbeitsplanungs-Systeme sind: CAPSY [Spur & al. 78], AUTAP [Eversheim & Fuchs 79], DREKAL [Tönshoff & Meyer 79], AVOPLAN.

Eine enorme Verbesserung früher üblicher Systeme, die auf reinem Programmiersprachen-Code basierten, stellen Systeme dar, die die sogenannte *Entscheidungstabellentechnik* verwenden. Sie können als einfache Form von Produktionsregelsystemen bezeichnet werden, wobei die Bedingungsteile der Regeln Konjunktionen von Attribut-Wert-Paaren fester Länge und die Aktionsteile im wesentlichen `goto`-Kommandos darstellen (siehe Abbildung 3.9[8]). Für eine Diskussion der Entscheidungstabellentechnik siehe [Heiob 81, Heiob 88, Bartenschlager 87].

[6] Computer Aided Process Planning.
[7] In Anlehnung an [Schwetz 83].
[8] aus [Bartenschlager 87]

Durchmesser Werkstoff	≤ 10	$> 10 \wedge \leq 50$ Aluminium	$> 10 \wedge \leq 50$ Stahl	> 50
Vorbohren? Bohrerdurchmesser	nein	ja $0.5 *$ Durchmesser	ja $0.8 *$ Durchmesser	ja 30

Abbildung 3.9: Beispiel einer Entscheidungstabelle für das Vorbohren.

Die meisten momentan auf dem Markt befindlichen CAPP-Systeme basieren auf dieser Technik. Ihr Vorteil gegenüber algorithmischen Systemen ist die Transparenz und leichtere Anpaßbarkeit und Austauschbarkeit von Wissen. Sie bieten insofern eher die Möglichkeit, die beiden oben erwähnten Anforderungen an CAPP-Systeme zu erfüllen. Bei Anpassungen und Änderungen müssen jedoch bei heutigen kommerziellen Systemen die Wissensbasen, d.h. die Entscheidungstabellen, manuell abgeändert werden. Akquisitionswerkzeuge für die Generierung und Techniken des maschinellen Lernens zur dynamischen Änderung von Entscheidungstabellen finden bislang keinen Einsatz. Beispiele für Arbeitsplanungssysteme, die auf der Entscheidungstabellentechnik basieren sind: ET-CAP [Heiob 88], ENGINE.

Viele CAPP-Systeme gehen nach dem sogenannten *Ähnlichkeitsplanungsprinzip* vor. Dabei wird keine Planung im eigentlichen Sinne durchgeführt, sondern nur das Retrieval eines alten Plans aus der betriebseigenen Arbeitsplankartei unterstützt – die Anpassung auf die aktuelle Planungsaufgabe muß manuell vom Arbeitsplaner selbst durchgeführt werden. Dafür werden sogenannte *Sachmerkmalsleisten* – Listen fester Länge kodierter charakteristischer Aussagen über Werkstücke – des aktuellen und eines alten Werkstücks verglichen.

In den achtziger Jahren wurden auch Expertensysteme für die Arbeitsplanung entwickelt, da sie aufgrund der Trennung von Inferenzmechanismen und Wissensbasen die Forderungen nach Austauschbarkeit und flexibler Anpaßbarkeit von Wissen eher erfüllen als konventionelle Programme. Sie fanden bislang jedoch noch keine kommerzielle Anwendung. Beispiele von Expertensystemen für die Arbeitsplanung sind: IXPRESS [Bullinger 89] GARI [Descotte & Latombe 81], TOM [Matsushima & al. 82].

3.2.3 Klassifikation von CAPP-Systemen

In [Eversheim 89] werden die folgenden Kriterien für die Klassifikation von CAPP-Systemen eingeführt. Ich stelle sie hier vor, um in Abschnitt 4.7.2 meine Konzeption für fallbasierte Arbeitsplanung auch aus der Sicht des Maschinenbaus als CAPP-System einordnen zu können.

Planungsmethoden: Für welche Planungsanlässe werden welche Planungsmethoden eingesetzt? (Siehe Abschnitt 3.1.3.)

Teilespektrum: Für welche Teileklassen (z.B. Drehteile) können Arbeitspläne generiert werden?

Funktionsumfang des Systems: Z.B. werden NC-Programme erstellt? Wenn ja, in welchem Format? Existieren Schnittstellen zu PPS[9]-Systemen? Werden Informationen berechnet wie Sollzeiten für Arbeitsvorgänge, Kosten und Gewicht von Fertigteilen etc.?

[9] Produktionsplanung und -steuerung

Bearbeitungsverfahren: Die Standardbearbeitungsverfahren sind Bohren, Drehen, Schleifen und Fräsen; daneben gibt es unzählige Spezialverfahren. Welche Bearbeitungsverfahren können geplant werden?

Automatisierungsgrad: Welche Tätigkeiten können vollautomatisch, welche müssen interaktiv durchgeführt werden?

Werkstückrepräsentation: Die Repräsentation des zu fertigenden Werkstücks als Aufgabenbeschreibung für die Arbeitsplanung hat einen großen Einfluß auf das gesamte CAPP-System (siehe Abschnitt 3.3.2). Während bei älteren CAPP-Systemen ausschließlich Varianten durch eine Liste von Parametern repräsentiert werden konnten, entstand mit dem Wunsch nach einer Bearbeitung eines größeren Teilespektrums der Bedarf nach Repräsentationsformalismen, die ein Werkstück vollständig beschreiben und gleichzeitig für Planungssysteme sinnvolle Informationen beinhalten. Dafür muß eine solche Repräsentation nicht nur geometrische, sondern auch technologische Informationen auf verschiedenen Abstraktionsebenen beinhalten [Warnecke & al. 89]. Dabei soll auch die Möglichkeit gewährleistet werden, schon während der Konstruktion technologische Angaben zu machen. In vielen der heutigen CAPP-Systeme finden Repräsentationsformalismen der KI Anwendung.

3.2.4 Defizite bisheriger Ansätze für CAPP

Ich möchte nun Defizite bisheriger Ansätze für CAPP vorstellen. Einige dieser Mängel sollen in meinem in Kapitel 4 vorgestellen Systemansatz kompensiert werden.

- Die meisten bisher entwickelten CAPP-Systeme bieten einen sehr eingeschränkten **Planungsumfang** an. Basieren sie auf dem Variantenprinzip, so können nur für sehr geringe Ausschnitte einer großen Produktpalette Arbeitspläne generiert werden. Basieren sie auf dem Ähnlichkeitsplanungsprinzip, so wird nur ein Retrieval durchgeführt und die eigentliche Planung dem menschlichen Experten überlasssen.

- Die Systeme, die auf dem Neuplanungsprinzip basieren, liefern meist unzureichende **Planungsergebnisse**, die meist von menschlichen Experten nachoptimiert werden müssen. Ein Grund dafür ist die mangelnde Planungsflexibilität; häufig wird nach einer starren Strategie eine lokale Optimierung durchgeführt.

- **Effizienz** ist eines der größten Probleme bei CAPP-Systemen. Oft wird diese durch mangelnde Qualität der Ergebnisse erkauft. Der Grund ist, daß häufig wiederkehrende Planungsaufgaben stets vollständig neu gelöst werden.

- Die Domäne der Arbeitsplanung ist durch sich ändernde Ressourcen gekennzeichnet. Die meisten Systeme sind nicht in der Lage, eine ständige **Aktualisierung des Planungswissens** zu ermöglichen. Dies liegt meist an der fehlenden Trennung zwischen betriebsspezifischem, domänenspezifischem und allgemeinem (Planungs-) Wissen.

Ich werde am Ende von Kapitel 4 vorstellen, welche der hier aufgeführten Probleme innerhalb meines Ansatzes für CAPP kompensiert werden können.

3.3 Das CAPP-System WAP und seine Integration in ein CIM-Konzept

Ich möchte nun die Konzeption des CAPP-Systems *WAP*[10] und dessen Integration in ein CIM-Konzept vorstellen, das zur Zeit im CIM-Centrum des Lehrstuhls für Fertigungstechnik und Betriebsorganisation (FBK) der Universität Kaiserslautern (Prof. Dr. G. Warnecke) entwickelt und implementiert wird. Eine prototypische Implementierung des von mir entwickelten Systems für die fallbasierte Arbeitsplanung wurde in dieses Projekt integriert.

3.3.1 Das integrierte Datenmodell

Dem CIM-Konzept liegt ein integriertes Modell zugrunde, das bauteilbezogene Daten für alle CIM-Module auf verschiedenen Abstraktionsebenen und in verschiedenen Sichtweisen enthält [Radtke 91]. Alle Module greifen auf diese Daten zu; Änderungen in einzelnen Submodellen müssen u.U. auf andere übertragen werden. Dieser Ansatz vermeidet die Nachteile möglicher Informationsverluste, übermäßige Datenredundanz und Ineffizienz beim Übergang von einem Bereich in einen anderen, die auftreten, wenn alle Module lokale Datenbereiche haben.

Es werden Werkstücke und deren Fertigung beschrieben; die Bereiche Konstruktion, Arbeitsplanung, NC-Programmierung, Prüfplanung und Produktionssteuerung verwenden das integrierte Datenmodell. Es unterteilt sich in die Submodelle *Gestalt-, Technologie-, Formelement (Feature)-* und *Applikationsmodelle*, die im folgenden kurz vorgestellt werden.

Gestaltsmodell

Hier wird die Geometrie und Topologie eines Werkstücks mit STEP[11] exakt beschrieben. Mehrere Beschreibungsansätze werden parallel gewählt:

linienorientiert: eine linienorientierte Werkstückrepräsentation liegt nahe an der Werkstückzeichnung und erlaubt eine einfache Generierung von Konturverläufen für die NC-Programmierung.

flächenorientiert: Die flächenorientierte Werkstückbeschreibung soll die natürliche Zuordnung von technologischen Angaben (wie z.B. Oberflächengüten) zu Elementen sowie die Erkennung von Formelementen unterstützen. Man beachte, daß in der Fertigungstechnik Prozesse stets auf Flächen referenzieren (Bearbeitungsflächen, Funktionsflächen etc.). Die Flächendarstellungen enthalten ihrerseits wieder Referenzen auf die Linienrepräsentation.

Form-Features: STEP bietet die Möglichkeit zur Darstellung von volumenorientierten Elementen (sogenannten *Form-Features*), die geometrische Körper wie z.B. Zylinder beschreiben.

[10] Wissensbasierte Arbeitsplanung
[11] Standard for the Exchange of Product Model Data [Scholz 88]

Technologiemodell

Im Technologiemodell werden technologische Angaben zu einem Werkstück gemacht wie Toleranzen, Materialeigenschaften und Oberflächenbeschaffenheiten. Zusammen mit dem Gestaltsmodell bildet es eine vollständige Werkstückbeschreibung, d.h. beinhaltet alle Angaben der Werkstückzeichnung.

Formelementmodell (Features)

Es wird versucht, die Geometrie eines Werkstücks aus der Sichtweise von Experten des Maschinenbaus, wie Konstrukteure oder Arbeitsplaner, darzustellen. Dazu wird ein Vokabular zur Beschreibung von sogenannten *Formelementen (Features)*, wie z.B. „Nut“ verwendet. Die Struktur des Formelementmodells hat einen wesentlichen Einfluß auf die CIM-Bereiche (formelementbasierte) Konstruktion, Arbeits-, Prüf- und Produktionsplanung und insbesondere auf die fallbasierte Arbeitsplanung. Sie werden daher in Abschnitt 3.3.2 detailliert vorgestellt.

Applikationsmodelle

Für folgenden Applikationen werden Datenmodelle vorgesehen:

Konstruktion: Der Konstrukteur denkt (unter anderem) in funktionalen Termini; für deren Repräsentation wird ein Funktionselementemodell vorgesehen.

Arbeitsplanung: hier sind zwei Applikationsmodelle zu nennen: Die Repräsentation von Bearbeitungsbereichen als Ziele der Arbeitsplanung und die Repräsentation von Arbeitsplänen. Sie werden in den Abschnitten 3.3.3 und 3.3.4 kurz vorgestellt, da sie für das Verständnis der Kapitel 4 und 5 notwendig sind.

NC-Programmierung: Repräsentation von Konturverläufen.

Prüfplanung: Repräsentation des Prüfplans.

3.3.2 Die formelementbasierte Werkstückrepräsentation

In diesem Abschnit wird detailliert die der Arbeitsplanung und anderen CIM-Bereichen zugrundegelegte formelementbasierte Werkstückrepräsentation vorgestellt. In der manuellen Arbeitsvorbereitung dient neben Stücklisten ausschließlich die Werkstückzeichnung als Repräsentation; die Experten der einzelnen Bereiche generieren beim Betrachten eine interne Repräsentation, die hier nachgebildet werden soll. Sie wird im Gegensatz zu den anderen Datenmodellen detailliert beschrieben, da ihr Informationsgehalt und ihre Strukturierung stark die wissensbasierte Fallauswahl prägen und somit wichtig für das Verständnis der Kapitel 4 und 5 ist.

Die Werkstückrepräsentation als Schnittstelle zwischen Konstruktion und Arbeitsplanung

Der Konstrukteur hat eine teilweise andere Sichtweise auf ein Werkstück als der Arbeitsplaner. Während er u.a. in funktionalen Kategorien denkt (denn seine Aufgabe ist es, ein Teil zu konstruieren, das eine bestimmte Funktion erfüllt), hat der Arbeitsplaner im wesentlichen die Fertigung des Teils im Sinn; z.B. kann ein und dasselbe Formelement vom Konstrukteur

als Führung (Funktion) und vom Arbeitsplaner als Einstich (Fertigung) bezeichnet werden. Ganz sicher haben jedoch beide Experten eine große Einsicht in die Sichtweise des jeweils anderen und ihre interne Werkstückrepräsentation stimmt in großen Teilen überein, was es ihnen ermöglicht, miteinander zu kommunizieren.

Die formelementbasierte Werkstückrepräsentation versucht, die Sichtweise eines Experten des Maschinenbaus zu modellieren; sie ist aber bewußt nicht als Applikationsmodell ausgelegt, so daß sie von den Bereichen Konstruktion, Arbeitsplanung, aber auch NC-Programmierung, Prüfplanung und Produktionssteuerung gleichermaßen verwendet werden kann. Daher enthält sie als Vokabular nur geometrische Formelemente, die von Experten aller dieser Bereiche gleichermaßen verstanden und verwendet werden und aus denen diese die für ihre Applikation relevanten Schlüsse ziehen können. Sie enthält keine applikationsspezifischen Termini; sie werden in die einzelnen Applikationsmodelle verlegt.

Ich möchte dies an drei Beispielen verdeutlichen:

Nut faßt eine Gruppe geometrisch klassifizierbarer Formelemente zusammen, wenngleich Experten eines speziellen Bereichs damit applikationsspezifisches Wissen assoziieren mögen (z.B. der Arbeitsplaner das Verfahren Fräsen). Der Begriff wird von den Experten aller oben angesprochenen Bereiche verwendet; er geht in das Vokabular der formelementbasierten Werkstückrepräsentation ein.

Stufenbohrung scheint vom Begriff her eine fertigungstechnische Aussage zu machen (Verfahren Bohren); tatsächlich assoziiert allerdings der Maschinenbauer damit nur eine geometrische Form, die durch Bohren, allerdings z.B. auch durch Drehen gefertigt werden kann. Daher gehört auch „Stufenbohrung“ zum Vokabular der Werkstückrepräsentation.

Längsdrehbereich macht eine Aussage, wie ein bestimmter Bereich gefertigt werden soll (nämlich durch Drehbearbeitung in axialer Richtung); es impliziert also eine fertigungsspezifische Aussage und erscheint daher nicht im Vokabular der Werkstückrepräsentation.

Realisierung der Werkstückrepräsentation

Die Werkstückrepräsentation wurde, basierend auf Erfahrungen aus den Projekten am CIM-Centrum entwickelt und von mir implementiert. Sie ist framebasiert und und liefert eine volumenorientierte Beschreibung von rotationssymmetrischen Werkstücken[12] auf einer abstrakten Ebene; sie verweist zur exakten Beschreibung z. B. von Flächen oder Konturverläufen auf das Gestalts- und Technologiemodell. Sie enthält keine Fertigungsinformation, wohl jedoch Information, aus der man möglichst direkt solche erschließen kann. Dies soll bei der Besprechung der einzelnen Slots deutlich gemacht werden.

Die Repräsentation für ein konkretes Werkstück stellt im wesentlichen[13] einen Baum verschiedener Knotentypen dar, die jeweils Instanzen von Formelementklassen sind. Von allen Formelementen wird außerdem auf das Gestalts- und Technologiemodell verwiesen. Die Slots der einzelnen Knoten enthalten zum einen Informationen über das Formelement und zum anderen Verweise auf die Söhne im Werkstückrepräsentationsbaum (siehe Abbildungen 3.10 und 3.11).

[12] eine Erweiterung auf allgemeine Drehteile, also solche, die auch prismatische Formelemente enthalten, ist vorgesehen.

[13] Ausschließlich Instanzen der Klasse Übergangselemente können verschiedene Vorgänger haben.

Abbildung 3.10: Aufteilung eines Werkstückes in Formelemente.

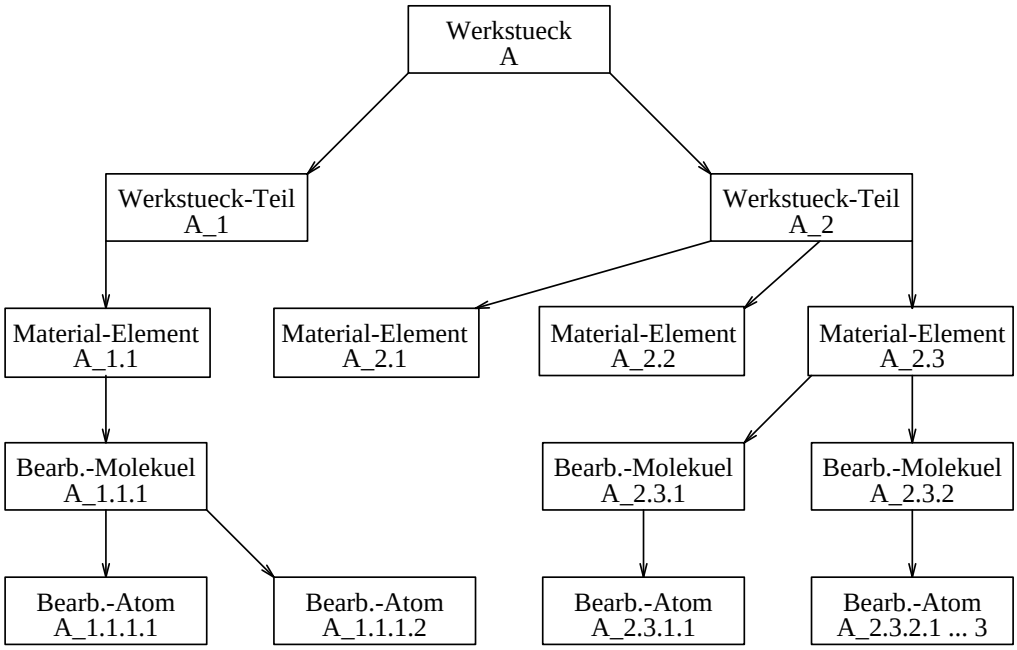

Abbildung 3.11: Werkstückrepräsentationsbaum für das Beispielwerkstück.

Ich werde nun die einzelnen Frames für die Werkstückrepräsentation und ihre Slots vorstellen.

Werkstück

Der Frame `Werkstück` bildet die Wurzel eines Repräsentationsbaums für ein konkretes Werkstück. Seine Slots – in vier Gruppen eingeteilt und durch Schreibmaschinenschrift gekennzeichnet – sind:

Geometrische Angaben: `Gesamtlänge, größter Durchmesser, Volumen, Gewicht, Schwerpunkt`. Sie definieren u.a. globale Beschränkungen an die Maschinen, z.B. die Größe von Spannfuttern.

technologische Angaben: `Werkstoff, Werkstoffeigenschaften, Form-Lage-Toleranz, höchste Oberflächengüte, schärfste Maßtoleranz.` Es werden nur die am schwierigsten zu erfüllenden technologischen Anforderungen des gesamten Werkstücks erwähnt, da sie maßgeblichen Einfluß u.a. auf Maschinen- und Werkzeugauswahl haben.

Fertigungstechnische Besonderheiten: `Axiale` und `radiale Einschnitte, Innenbearbeitungen, prismatische Elemente` und `andere Besonderheiten.` Sie stellen Anforderungen an alle Bereiche der Arbeitsplanung; z.B. kann eine Innenbearbeitung ein Spannen mit einer Spitze unmöglich machen.

Verweis auf Söhne: `Werkstückteile.`

Werkstückteil

Für Drehteile kann man stets bis zu drei charakteristische Bereiche, eine aufsteigende, horizontale und fallende Kontur, unterscheiden, die häufig auch getrennt voneinander gefertigt werden. Diese Einteilung wird auch in der Werkstückrepräsentation vorgenommen, um Aussagen über die einzelnen charakteristischen Bereiche explizit vornehmen zu können.

Im wesentlichen enthält dieser Frame dieselben Slots für geometrische, technologische und fertigungstechnische Angaben wie der Frame Werkstück. Zusätzlich wird ein Slot `Typ` zur Unterscheidung zwischen den drei oben vorgestellten Konturarten eingeführt. Die Söhne von `Werkstückteil` sind Instanzen der Klasse `Materialelement.`

Materialelement

Materialelemente sind additive Formelemente, die in ihrer Summe eine Umhüllende des Fertigteils bilden. Es wird ein Grundtyp für ein Materialelement zugrundegelegt, mit dem man technologisch charakteristische Formen, wie z.B. eine Schulter, modellieren kann. In Materialelementen können Einschnitte beliebiger Form vorkommen, die dann durch sog. Bearbeitungselemente dargestellt werden. Es ist wichtig zu erwähnen, daß Materialelemente in der Regel keine schmalen Scheibchen sind; sie sind Komplexe oft einheitlicher Fertigung, aus dessen charakteristischer Kontur der Arbeitsplaner fertigungstechnische Information ziehen kann. Für die Beschreibung eines Materialelementes wird ein festes Schema zugrundegelegt (siehe Abbildungen 3.12 und 3.13).

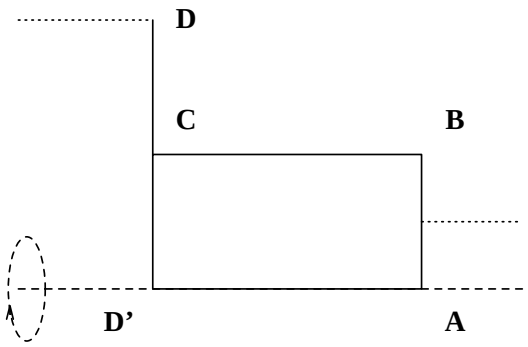

Abbildung 3.12: Schema für ein Materialelement.

Abbildung 3.13: Typen von Materialelementen.

Abbildung 3.14: Zusammensetzung von Bearbeitungsatomen zu -molekülen.

Klassifikation: `Typ` (mit den Ausprägungen `Kegel, Zylinder, Konvex und Konkav`; siehe Abbildung 3.13), `Spiegelung und Planbereich` (legen die Lage der Punkte A, B, C und D zueinander fest) und `Randelement` (gibt an, ob sich das Materialelement an einem der beiden Ränder der Werkstücks befindet).

Geometrische Angaben: `Länge, Abstand AB, BC` und `CD, Durchmesser B, C` und `D, Steigung` und `Krümmungsradius BC, Eckenwinkel B` und `C.`

Technologische Angaben: höchste Qualitätsanforderungen analog zum Frame `Werkstück.`

Fertigungstechnische Besonderheiten: analog zum Frame `Werkstück.`

Söhne: `Bearbeitungsmoleküle` in den Bereichen `AB, BC` und `CD, Übergangselemente B, C` und `D.`

Bearbeitungsmolekül

Bearbeitungselemente sind subtraktive Elemente, die durch die Bearbeitung aus den Materialelementen herausgenommen werden sollen. Sie teilen sich auf in Moleküle und Atome, wobei Moleküle aus Atomen zusammengesetzt sind. Während Atome rein geometrische Informationen beinhalten, können und sollen Moleküle mit Termini des Maschinenbaus wie z.B. Stufenbohrung benannt werden (siehe Abbildung 3.14).

Klassifikation: `Typ, Untertyp, DIN-Norm, Lage` (mit den Ausprägungen `axial, radial`) und `Spiegelung.`

Technologische Angaben: höchste Qualitätsanforderungen analog zum Frame `Werkstück.`

Söhne: `Bearbeitungsatome` und ein Graph `Relationen`, der die Lage der Atome zueinander definiert.

Bearbeitungsatom

Wie bei den Materialelementen wird auch bei den Bearbeitungsatomen ein Schema für ein allgemeines Element zugrundegelegt (siehe Abbildung 3.15).

Abbildung 3.15: Schema für ein Bearbeitungsatom.

Klassifikation: `Typ` (z.B. `2 Planseiten + Boden`), `Untertyp` (z.B. `konvex - konkav`), `Drehung` und `Spiegelung`.

Geometrische Angaben: `Abstand AB` und `BC`, `Tiefe`, `Eckenwinkel B` und `C`, `Steigung` und `Krümmungsradien AB` und `CD`.

Technologische Angaben: höchste Qualitätsanforderungen analog zum Frame `Werkstück`.

Söhne: `Übergangselemente A`, `B` und `C`.

Übergangselement

Übergangselemente beschreiben Übergänge von einem Formelement zu einem anderen oder auch innerhalb von Formelementen. Es werden zwei Arten von Übergangselementen eingeführt: `Fasen` (Anschrägungen) und `Rundungen`; sie können von Materialelementen und Bearbeitungsatomen referenziert werden.

Klassifikation `Typ` (Ausprägungen `Fase`, `Rundung`), `Spiegelung`.

Geometrische Angaben: `Länge`, `Winkel` (für Fasen), `Krümmungsradius` (für Rundungen).

Technologische Angaben: höchste Qualitätsanforderungen analog zum Frame `Werkstück`.

Die Werkstückrepräsentation erlaubt durch ihre Hierarchisierung die Betrachtung von Formelementen unterschiedlicher Komplexität und das Gewinnen von fertigungstechnisch relevanter Information direkt aus den Slots des entsprechenden Frames. Reicht diese Information nicht aus, so kann jederzeit auf das durch Slots referenzierte Gestalts- und Technologiemodell zugegriffen werden.

3.3.3 Die Bearbeitungsbereichsrepräsentation

Es wird zwischen *primitiven* und *komplexen Bearbeitungsbereichen* unterschieden. Primitive Bearbeitungsbereiche stellen geometrische Elemente mit rechteckigem Querschnitt dar, die durch ihre Dimensionen beschrieben werden. Komplexe können aus Komplexen und Primitiven zusammengesetzt sein. Sie enthalten neben Verweisen auf ihre Komponenten Informationen über (Teil-)Bearbeitungen, die im Laufe des Arbeitsplanungsprozesses eingetragen werden. Die Zusammenfassung zu Komplexen spiegelt Gruppen gemeinsamer Fertigung wider.

Da dies endgültig erst durch die Arbeitsplanung festgelegt werden kann, startet man für das Retrieval in CBP, in das die Bearbeitungsbereiche als wichtiges Vergleichskriterium eingehen, mit einer initialen hypothetischen Komplexbildung. Dabei werden bis zu fünf komplexe Bearbeitungsbereiche generiert: Plandrehbereich links, aufsteigende Kontur, horizontale Kontur, abfallende Kontur und Plandrehbereich rechts. Die Einteilung in diese Bereiche kann rein algorithmisch erfolgen. Während des Planungprozesses wird diese Komplexbildung detailliert und u.U. modifiziert (siehe Abbildung 3.16).

3.3.4 Die Arbeitsplanrepräsentation

Ein konkreter Arbeitsplan wird durch einen fünfstufigen Baum repräsentiert, wobei die Knoten jeder Stufe Instanzen einer Arbeitsplanrepräsentationsklasse sind; sie werden kurz vorgestellt:

Arbeitsplankopf bildet die Wurzel jedes Repräsentationsbaums eines konkreten Arbeitsplans. Er enthält organisatorische Angaben zum Arbeitsplan wie Arbeitsplannummer, Bezeichnung, Bearbeiter, Auftragsdaten oder Losgröße, technologische Angaben wie Werkstoff oder Gewicht und Verweise auf Arbeitsvorgänge, die Söhne im Repräsentationsbaum.

Arbeitsvorgang beschreibt alle Fertigungsschritte, die auf einer Maschine durchgeführt werden können. Damit wird gleichzeitig eine Festlegung auf ein Verfahren wie Drehen, Fräsen, Bohren etc. gemacht. Maschinenart und -typ werden genau beschrieben und ein Verweis auf Unterarbeitsvorgänge gemacht.

Unterarbeitsvorgang beschreibt alle Fertigungsschritte, die mit einer Aufspannung durchgeführt werden. Es werden die Spannmittel und -flächen genau beschrieben und auf Teilarbeitsvorgänge verwiesen.

Teilarbeitsvorgang beschreibt alle Fertigungsschritte, die mit einem Werkzeug durchgeführt werden. Werkzeugart und -typ werden genau beschrieben und auf die Blätter des Arbeitsplanrepräsentationsbaumes, NC-Arbeitsvorgänge, verwiesen.

NC-Arbeitsvorgang beschreibt einen Steuervorgang für einen NC-Arbeitsgang. Er enthält z.B. für einen Drehvorgang eine Konturbeschreibung, Umdrehungszahl, Vorschubgeschwindigkeit, Schnittiefe etc.

(Siehe auch Abbildung 4.4.)

3.3.5 Die CIM-Module in ihrer Abfolge

Es werden die CIM-Module der Bereiche Konstruktion, Planung und Fertigung vorgestellt (siehe Abbildung 3.17).

Konstruktion

Die Konstruktion kann zum einen über konventionelle CAD-Systeme und zum anderen über einen formelementbasierten Werkstückeditor erfolgen.

Sinn eines formelementbasierten Editors ist es, den qualifizierten (und damit teuren) Konstrukteur in seiner Arbeit zu unterstützen und dabei nachgelagerte Bereiche möglichst zu

Abbildung 3.16: Aufteilung in Bearbeitungsbereiche.

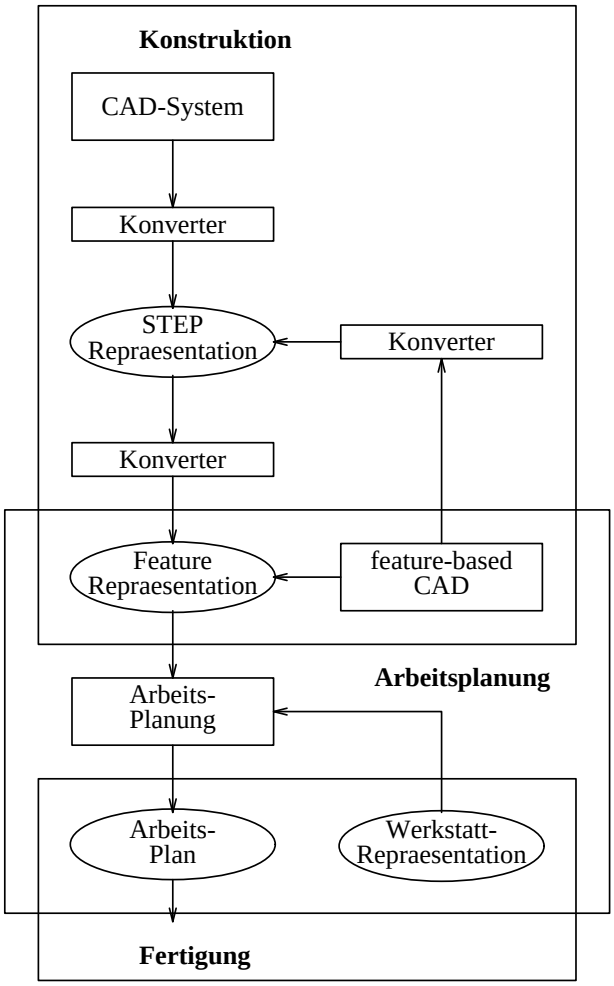

Abbildung 3.17: Der Prozeß von der Konstruktion zur Fertigung.

vereinfachen. Der Konstrukteur denkt u.a. in funktionalen Kategorien und erhält daher eine Unterstützung duch Anbieten funktionaler Elemente. Auf der anderen Seite muß er schon während der Konstruktion Aspekte der Fertigung berücksichtigen, um technologisch fertigbare und möglichst kostengünstige Werkstücke zu konstruieren. Daher sollte er auch die Möglichkeit haben, schon während der Konstruktion Fertigungshinweise zu geben. Die Ergebnisse der formelementbasierten Konstruktion können direkt in das Formelementemodell übertragen werden; allerdings muß auch eine Transformation in das Gestalts- und Technologiemodell erfolgen.

Erfolgt die Konstruktion mittels eines konventionellen CAD-Systems, so wird die Zeichnungsinformation mittels Konvertern über das Gestaltsmodell in das Formelementemodell übertragen. Bei der Konvertierung treten Probleme auf, die denen beim Bildverstehen ähnlich sind: Erkennen komplexer Elemente, Zuordnung von Informationen wie Bemaßung oder technologischen Angaben zu Elementen etc. In der Regel ist eine algorithmische Lösung nicht möglich; es muß Hintergrundwissen eingesetzt werden, um Mehrdeutigkeiten oder kleine Fehler in der Zeichnung richtig zu erkennen.

Arbeitsplanung

Die Arbeitsplanung verwendet aus dem Bereich Konstruktion im wesentlichen die formelementbasierte Werkstückrepräsentation und aus dem Bereich Fertigung die Werkstattrepräsentation. Fallbasierte und generative Planung werden integriert: die fallbasierte Planung liefert eine Abkürzung des generativen Planungsprozesses bzw. das Wissen der generativen Planung trägt zur Modifikation bei. Die fallbasierte Planung wird in den Kapiteln 4 und 5 ausführlich vorgestellt; bei der generativen Planung werden die Teilziele der Arbeitsplanung Rohteil-, Verfahrens-, Maschinen-, Spannmittel- und Werkzeugauswahl in dieser Reihenfolge durchgeführt. Dabei können alle Bereiche Constraints für andere festlegen, z.B. stellt die Auswahl eines Keramikdrehmeißels Anforderungen an die Leistung und Umdrehungszahl der Werkzeugmaschine.

Fertigung

Arbeitspläne werden in der Regel auftragsneutral erstellt (siehe Abschnitt 3.1); d.h. sie beschreiben z.B. nur Maschinentypen, legen sich jedoch nicht auf eine konkrete Maschine fest. Für die Fertigung müssen dann in Abhängigkeit von der aktuellen Werkstattauslastung noch die Produktionsabfolgen geplant und organisiert werden.

3.3.6 Die Implementierungssprache KEE

Als Implementierungssprache wurde die *LISP*-basierte Expertensystementwicklungsumgebung *KEE*[14] der Firma *IntelliCorp* gewählt. *KEE* legt einen framebasierten Wissensrepräsentationsmechanismus zugrunde. Zur Verarbeitung von Wissen wird objektorientierte und regelorientierte Programmierung (Vorwärts- und Rückwärtsinterpreter mit diversen Konfliktlösungsstrategien) zur Verfügung gestellt. Ein Weltenmodell und ATMS[15] unterstützen nichtmonotones und hypothetisches Schließen.

[14] Knowledge Engineering Environment
[15] Assumption based Truth Maintenance System

Aus folgenden Gründen wurde KEE für die Implementierung des integrierten Fertigungssystems verwendet:

- Das Framesystem unterstützt die Implementierung des integrierten Datenmodells für alle CIM-Bereiche.
- Aufgrund der Entwicklungsunterstützung für Wissensbasen und Oberflächen ist eine schnelle Implementierung lauffähiger und vorzeigbarer Prototypen möglich.
- Vergleiche der Effizienz verschiedener Inferenzstrategien für eine spezielle Anwendung können mit relativ geringem Aufwand durchgeführt werden.
- Durch die direkte Verwendung von LISP, die Einbettbarkeit von C-Programmen und die Möglichkeit der Ankopplung von Datenbanksystemen wie z.B. Oracle besteht die Möglichkeit zur Ankopplung an andere Systeme, z.B. der integrierten Fertigung.

Kapitel 4

Ein System für die fallbasierte Arbeitsplanung

In diesem Kapitel stelle ich eine Konzeption für fallbasierte Arbeitsplanung und deren Integration in ein generatives CAPP-System vor, die es gestattet, die in Abschnitt 3.2.4 vorgestellten Defizite bisheriger Ansätze für CAPP zu kompensieren. Desweiteren erlaubt sie es, Ähnlichkeitsmaße automatisch zu verbessern (dies wird in Abschnitt 4.6 erläutert). Dabei gehe ich in besonderem Maße auf das Ähnlichkeitsmaß ein, das auf einer neuen[1] Sichtweise von Ähnlichkeit für die fallbasierte Planung basiert.

In Abschnitt 4.2 werde ich diese Sichtweise motivieren und skizzieren. Die Abschnitte 4.3, 4.4 und 4.5 beschreiben die Konzeption des fallbasierten CAPP-Systems, wobei in Abschnitt 4.4 das Ähnlichkeitsmaß vorgestellt und diskutiert wird.

Abschließend klassifiziere ich das vorgestellte Konzept in Bezug auf CBP- und CAPP-Systeme, vergleiche es mit anderen, in den Kapiteln 2 und 3 vorgestellten Ansätzen und Systemen und stelle dar, welche Defizite traditioneller Ansätze kompensiert werden können.

4.1 Vorüberlegungen

In Abschnitt 2.2.1 wurde das Planen auf der Basis alter Fälle als häufig angewendete Technik menschlichen Schließens vorgestellt. Auch bei der manuellen Erstellung von Arbeitsplänen wird sehr häufig von dieser Technik in systematischer Weise Gebrauch gemacht, indem der Arbeitsplaner auf eine unternehmenseigene Kartei alter Fälle zugreift und einen Arbeitsplan auf das aktuelle Planungsproblem anpaßt (siehe Abschnitt 3.1.3). Häufig wird er dabei von Retrieval-Systemen unterstützt. Diese Tatsache legt die Verwendung von fallbasierter Planung in CAPP-Systemen nahe. Aus folgenden weiteren Gründen ist die fallbasierte Planung in der Arbeitsplanung – falls möglich – der rein generativen vorzuziehen:

- Die Festlegung der Arbeitsvorgangsfolge kann eine höchst komplexe Aufgabe sein, falls die einzelnen Arbeitsvorgänge untereinander starke Abhängigkeiten haben. Die Optimierung ist so kompliziert, daß meist in einem vertretbaren Aufwand nur eine lokale Opti-

[1] Mir ist keine Publikation bekannt, in der ein ähnlicher Ansatz gewählt wird.

mierung durchgeführt werden kann. Eine einmal gemachte sehr gute Entscheidungsfolge sollte daher nicht vergessen, sondern in geeigneter Form wiederverwendet werden.

- Pläne für Varianten haben feste Schemata; ihre Generierung ist eine Routinetätigkeit und sollte daher nicht generativ durchgeführt werden.
- Auch für einzelne Arbeitsvorgänge wie das Fräsen einer Nut gibt es oft Planskelette, die verwendet werden sollten.

Es stellt sich die Frage, ob es sinnvoll ist, CAPP ausschließlich fallbasiert durchzuführen (wie dies z.B. im *CHEF*-System der Fall ist). Das bedeutet, daß das System bei fehlender signifikanter Fallinformation kein Ergebnis liefern würde, was die Einsetzbarkeit des Systems in der Praxis in Frage stellen würde. Der hier gewählte Ansatz ist anders. Das wesentliche daran ist eine Integration von fallbasierter und generativer Planung in folgendem Sinne: die fallbasierte Komponente erlaubt dann und nur dann, wenn die Fallbasis signifikantes Wissen enthält, eine Abkürzung des generativen Planungsvorgangs durch Erinnerung an einen alten Fall und Extraktion eines u.U. partiellen Plans. Bei nicht-signifikantem Fallwissen wird rein generativ geplant. Man könnte also sagen, die fallbasierte Planung ist eine Komponente der generativen, da die Zusammenfassung und gleichzeitige Durchführung mehrerer Planungsschritte eine Abstraktion von Operatoren darstellt (siehe Abschnitte 2.3.2 und 4.7.1), oder die generative Planung ist Teil des Modifiers (und damit des fallbasierten Planungssystems), der u.U. einen leeren Plan auf das aktuelle Problem anpassen kann.

Von großer Wichtigkeit ist die Frage, ob das Retrieval auch die Rohteilauswahl durchführen sollte, indem das Rohteil des selektierten alten Falles wiederverwendet (und evtl. modifiziert) wird, oder ob die Rohteilauswahl vor dem Retrieval unabhängig von diesem durchgeführt wird, so daß somit die klassische Planungssituation vorliegt: gegeben ein Ausgangszustand (das Rohteil) und ein Zielzustand (das Fertigteil); gesucht: ein Plan, der den Ausgangszustand in den Zielzustand überführt.

Ich möchte die Argumente für beide Vorgehensweisen gegenüberstellen.

Rohteilauswahl durch das Retrieval:

- Bei der Rohteilauswahl geht es darum, ein technologisch mögliches Ausgangsteil für die Fertigung zu wählen. Dabei wird eine kostengünstige Lösung angestrebt. In die Kosten geht jedoch nicht nur der Preis für das Rohteil selbst ein, sondern auch der Aufwand für die Fertigung ausgehend aus diesem. So ist z.B. ein Gußteil in der Regel teurer als Stangenmaterial; jedoch gestaltet sich die Fertigung als weniger aufwendig, da der Konturverlauf des Gußteils dem Fertigteil angepaßt ist. Um die komplizierten Abhängigkeiten zwischen Rohteil und Fertigungsaufwand nicht immer neu auflösen zu müssen, ist es sinnvoll, auf die Erfahrung in alten Fällen zurückzugreifen.

Rohteilauswahl unabhängig von dem Retrieval:

- Es ist so leichter möglich, dem Konstrukteur oder dem Benutzer die Möglichkeit zu geben, interaktiv auf den Prozeß der Rohteilauswahl Einfluß zu nehmen.
- Wenn die Rohteilauswahl durchgeführt worden ist, können Bearbeitungsbereiche als Repräsentation der Planungsziele bestimmt werden, die Rückschlüsse auf Constraints und Probleme der Fertigung zulassen (wie eine initiale Bearbeitungsbereichsrepräsentation

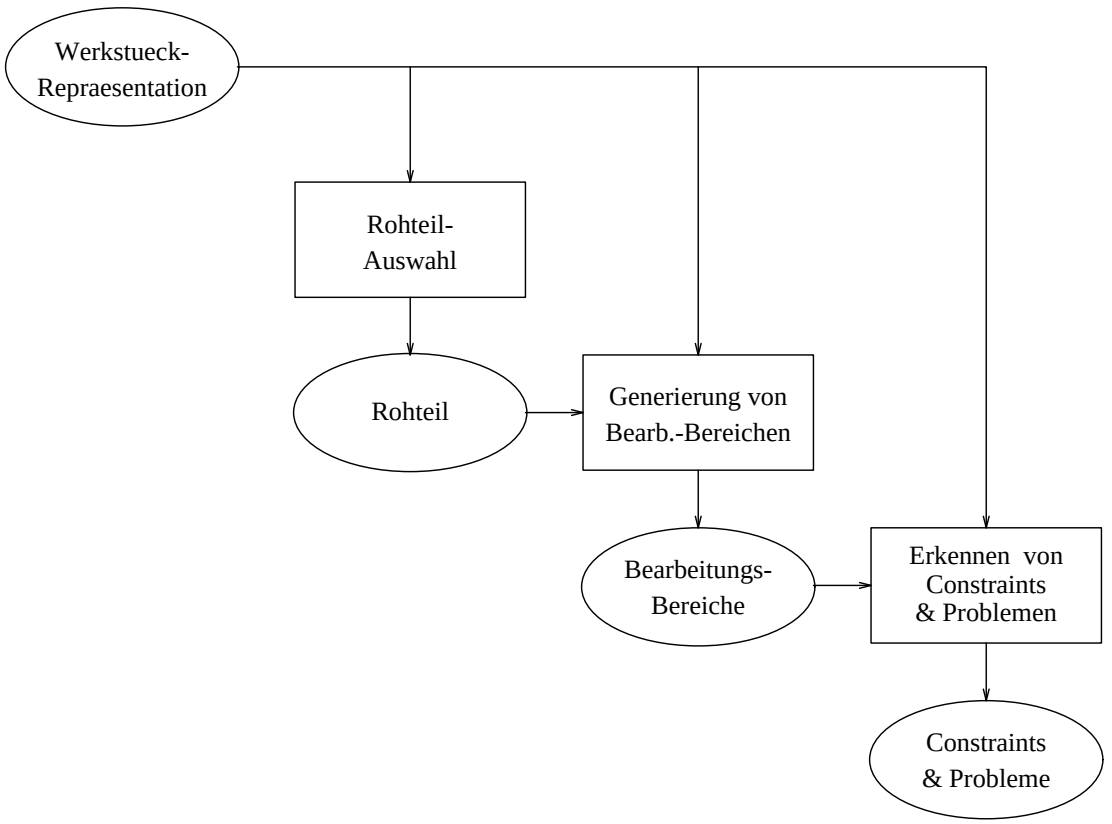

Abbildung 4.1: Phasen, die vor dem Retrieval durchgeführt werden.

generiert werden kann, wurde am Ende von Abschnitt 3.3.2 vorgestellt). Ein Retrieval, das auf dieser Information basiert, ist wesentlich fundierter als ein Retrieval, das nur Werkstückinformation zur Verfügung hat.

Den Ausschlag für die Entscheidung, die Rohteilauswahl vor dem Retrieval, also unabhängig von der fallbasierten Planung duchzuführen, gab der letztgenannte Punkt. Nach der Rohteilauswahl können Informationen über die Entsprechung der Bearbeitungsbereiche des aktuellen und alten Werkstücks in das wissensbasierte Ähnlichkeitsmaß eingehen.

Abbildung 4.1 zeigt, daß mit Hilfe der formelementbasierten Werkstückrepräsentation das Rohteil bestimmt und dafür initiale Bearbeitungsbereiche festgelegt werden können; aus diesen Informationen kann man auf technologische Constraints, Möglichkeiten und Probleme schließen.

Weiterhin ist zu überlegen, ob verschiedene alte Fälle kombiniert, oder ausschließlich ein alter Fall ausgewählt und modifiziert werden soll. Dies wäre dann sinnvoll möglich, wenn

die einzelnen Teilziele relativ unabhängig voneinander bearbeitet werden können. Da jedoch Arbeitsvorgangsfolgen der Teilefertigung sehr starke und schwer aufzulösende Abhängigkeiten untereinander aufweisen, wird in dem hier vorgestellten System auch stets nur ein alter Fall wiederverwendet.

Um die Betonung der wissensbasierten und fertigungsorientierten Fallauswahl in dieser Arbeit zu motivieren, möchte ich einen Vergleich zum Bereich der informierten Suchverfahren ziehen. Dort hat die Qualität der heuristischen Funktion einen entscheidenden Einfluß auf den Aufwand der Suche, wobei das Spektrum von vollkommen uniformierten heuristischen Funktionen (impliziert systematische Suche) bis hin zu voll informierten Funktionen (z.B. durch interne Vorwegnahme der Suche; dabei entartet die Suche selbst zu einem linearen Weg durch den Suchraum) reicht. Heuristische Funktionen bewirken durch eine Steuerung der nachgeschalteten Suche eine Verringerung des Aufwandes; dabei kann z.B. eine Relaxierung des Problems durchgeführt werden. Grundsätzlich kann man vereinfachend sagen: je „intelligenter“ die heuristische Funktion, desto geringer der Suchaufwand. Da es jedoch das Ziel ist, den gesamten Problemlöseprozeß (d.h. Berechnung der heuristischen Funktion plus Suche) zu optimieren, ist es sinnvoll, zwar möglichst intelligente, aber dennoch vom Aufwand vertretbare heuristische Funktionen zu verwenden.

Analog dazu hat die Güte der Ergebnisse des Retrievals bei CBP einen entscheidenden Einfluß auf die Performanz der nachgelagerten Modifikation und damit des gesamten Systems. Daher wird in dieser Arbeit ein besonderer Schwerpunkt auf ein wissenbasiertes „intelligentes“ Ähnlichkeitsmaß gelegt, das auf einer Abschätzung der nachgeschalteten Modifikation basiert. In das Ähnlichkeitsmaß gehen zum einen geometrische Informationen und zum anderen fertigungsrelevante Aussagen ein. Es ist in seiner Struktur so ausgelegt, daß es selbst beurteilt und u.U. automatisch verbessert werden kann.

Es muß dabei jedoch (in Analogie zu den informierten Suchverfahren) vermieden werden, zuviel Berechnungsaufwand in das Retrieval zu stecken.

4.2 Ähnlichkeit als Minimierung des Modifikationsaufwandes

Schank hat in seiner Dynamic Memory Theorie als erster auf die Wichtigkeit des Erinnerns im Prozeß des analogen Schließens hingewiesen und damit das zugrundegelegte Gedächtnis und dessen Organisation in den Vordergrund gestellt (siehe Abschnitt 2.2.2). Das ist sehr wichtig; man darf jedoch daraus nicht ableiten, daß die Organisation der Fallbasis – d.h. das Vokabular zur Beschreibung von Fällen und Indizes und die Strukturierung der Indizes – unabhängig von anderen Moduln des Systems, z.B. des Retrievers festgelegt werden kann. Die Organisation eines Speichers sollte stets die Anforderungen wiederspiegeln, die an ihn gestellt werden (siehe dazu auch [Hammond 89]). Die Anforderung an die Fallbasis ist es, Fälle zu speichern, die, falls sie einer aktuellen Aufgabe ähnlich sind (was auch immer das heißen mag), für eine Wiederverwendung mit Hilfe des Retrievers zur Verfügung gestellt werden können. Da jedoch die Organisation der Fallbasis und speziell die Strukturierung der Indizes in großem Maße die Möglichkeiten des Retrievals bestimmen, sollte sie so ausgelegt sein, daß man möglichst effizient einen zur aktuellen Aufgabe ähnlichen Fall für eine Modifikation zur Verfügung stellen kann (zum Vergleich: eine logarithmische Zeitkomplexität für die Suche eines Eintrags in einer Liste kann z.B durch die Verwendung eines balancierten Suchbaums ermöglicht werden). Das

bedeutet insbesondere, daß vor der Festlegung einer Fallbasisorganisation eine Diskussion des Begriffs Ähnlichkeit erfolgen muß.

Eine einfache Herangehensweise, die Ähnlichkeit zwischen Fällen der Arbeitsplanung festzustellen, ist, die Geometrie des aktuellen Werkstücks mit der eines alten zu vergleichen, d.h., ausschließlich die Ziele der Planung zu vergleichen. Dies ist die Vorgehensweise der Sachmerkmalsleistensysteme (siehe Abschnitt 3.2.2).

Um die Nachteile dieses Ansatzes herauszustellen und somit meine Definition der Ähnlichkeit zu motivieren, möchte ich noch einmal das Ziel fallbasierter Planung vorstellen, so wie ich es sehe:

Erinnerung an alte Fälle bewirkt eine Reduzierung des Planungsaufwandes durch Vermeidung von wiederholter (generativer) Planung.

Der Retriever muß einen alten Fall selektieren, der dieser Aufgabe möglichst dienlich ist, d.h. den Planungsaufwand (für Retrieval und Modifikation) minimiert. Vor diesem Hintergrund definiere ich Ähnlichkeit wie folgt:

Ein alter Fall ist dem aktuellen ähnlich, falls die Modifikation des alten Plans (d.h. die Anpassung auf das aktuelle Problem) einfach[2] *ist.*
oder kurz:
„ähnlich“ heißt „einfach zu modifizieren“.

Das impliziert, daß die Fähigkeiten des Modifiers das Retrieval beeinflussen.

Ein Vergleich lediglich der Geometrie von Werkstücken liefert oft Ergebnisse, die der Zielsetzung, Planungsaufwand zu vermeiden, nicht gerecht werden. Oft werden zwei Werkstücke, die eine sehr große geometrische Ähnlichkeit aufweisen, aufgrund bestimmter, geometrisch geringer Abweichungen vollkommen unterschiedlich gefertigt. Es wird dies an Beispielen deutlich gemacht werden.

Wichtig für die Realisierung eines so definierten Ähnlichkeitsmaßes ist es auch, den alten Plan in die Betrachtung der Ähnlichkeit eingehen zu lassen (wie es weder bei den Sachmerkmalsleistensystemen noch bei *CHEF* der Fall ist). Sehr häufig existieren nämlich für eine Fertigungsaufgabe verschiedene technologisch mögliche Pläne, von denen die generative Planung einen liefert. Es kann passieren, daß der ausgewählte Plan für die Wiederverwendung in einer neuen Planungsaufgabe ungeeignet ist, wohingegen ein dazu alternativer technologisch möglicher Plan sehr wohl in der Lage gewesen wäre.

Wichtig an dieser Modellierung der Ähnlichkeit für fallbasierte Planung ist, daß sie es erlaubt, die Güte des Retrievals zu messen (denn ob zwei Fälle wirklich ähnlich sind, erkennt man daran, daß sich die Modifikation im nachhinein tatsächlich als einfach herausstellte) und darauf basierend das Ähnlichkeitsmaß (automatisch) zu verbessern.

Desweiteren erlaubt diese Sicht, die Signifikanz von Fallwissen in der Fallbasis abzuschätzen: fallbasierte Planung ist genau dann generativer Planung vorzuziehen, wenn der (geschätzte) Modifikationsaufwand den (geschätzten) Aufwand für die generative Planung unterschreitet.

Die hier vorgestellte Sichtweise auf Ähnlichkeit ist eine rein pragmatische, die es zum Ziel hat, das Problemlöseverhalten des Systems zu verbessern. Eine Diskussion der kognitiven Adäquatheit dieser Vorgehensweise und ob sie in der Lage ist, gute (im Sinne von Abschnitt 3.1.2) Pläne zu generieren, wird in Abschnitt 4.4 ausführlicher diskutiert.

[2] Eine Diskussion des Begriffs „einfach“ erfolgt in Abschnitt 4.4

Ich werden nun die Systemkomponenten Fallbasis, Retriever (und speziell Ähnlichkeitsmaß) und Modifier vorstellen.

4.3 Fallbasis und Retrieval

Ein Fall enthält folgende Komponenten:

Werkstückrepräsentation wie in Abschnitt 3.3.2 vorgestellt: aus der formelementbasierten Werkstückrepräsentation wird auf Gestalts- und Technologiemodelle verwiesen.

Rohteil wird wie ein Fertigteil repräsentiert.

Bearbeitungsbereiche werden hierarchisch repräsentiert und stellen die Ziele der Arbeitsplanung dar (siehe Abschnitt 3.3.3).

Arbeitsplan wird hierarchisch als Baum repräsentiert (siehe Abschnitt 3.3.4) mit Verweisen auf Bearbeitungsbereiche und Betriebsmittel.

Folgende grundlegende Forderungen an die Organisation der Fallbasis sind zu stellen [Kolodner 83]:

- Die Speicherung muß **effizient** sein; d.h., daß bei zunehmendem Fallwissen die Zeit zum Auffinden brauchbarer Fälle nicht (wesentlich) zunehmen darf. Zum Vergleich: Der menschliche Experte erinnert sich in der Regel schneller, je mehr Erfahrung er hat.
- Sie muß **dynamisch**, d.h. offen für neue Strukturierungen – insbesondere sich ändernde Ähnlichkeitsmaße – sein.

Anders gesagt: man muß „einfach“ ein „gutes“ Retrieval durchführen können.

Effizienz und Dynamik sind zwei widerstrebende Forderungen. Die dynamischste Speicherorganisation ist ohne Zweifel eine Liste aller Fälle, bei der keine explizite Strukturierung in Form von Indize vorgenommen wird, sondern die gesamte Fallinformation direkt für das Retrieval verwendet werden kann. Dieses ist höchst ineffizient, erlaubt aber alle erdenklichen Formen der Ähnlichkeit über diesen Fällen.

Die effizienteste Organisation spiegelt in ihrer Indexierung exakt das Ähnlichkeitsmaß wider: Retrieval gestaltet sich dann fast als direkter Zugriff. Eine Änderung des Ähnlichkeitsmaßes ist aber dann auch stets mit einer Umstrukturierung der Fallbasis verbunden.

Wie in Abschnitt 4.2 betont, sollte sich die Strukturierung der Fallbasis nach der Definition der Ähnlichkeit richten. Da dies der zentrale Punkt des praktischen Teils dieser Arbeit ist, wurde für die Fallbasis vorerst die einfachste und flexibelste (und damit auch ineffizienteste) Form der Fallbasis gewählt: Eine lineare Liste der Fälle, die nicht in eine Indexstruktur integriert sind. Das Retrieval gestaltet sich dann als lineare Suche durch diese Liste, wobei jeweils die Ähnlichkeit eines alten Falles mit dem aktuellen gemessen wird und anschließend der am ähnlichsten klassifizierte ausgewählt wird.

Die Organisation der Fallbasis widerspricht massiv der Effizienzanforderung. Es sei jedoch hervorgehoben, daß dies eine vorläufige und keine endgültige Lösung ist; Arbeiten auf diesem Gebiet müssen auf der Basis z.B. der hier vorgestellten Ergebnisse durchgeführt werden; vor der Festlegung und Validierung eines Ähnlichkeitsmaßes ist dies jedoch meiner Meinung nach nicht sinnvoll möglich.

4.4 Das Ähnlichkeitsmaß

4.4.1 Präzisierung und Diskussion der Definition von Ähnlichkeit

In Abschnitt 4.2 wurde definiert: „Ein alter Fall ist ähnlich, falls er einfach zu modifizieren ist“. Ich möchte nun diese komprimierte Aussage diskutieren und verdeutlichen.

Was bedeutet „einfach“?

„Einfach“ bedeutet hier effizient, d.h. von geringem Berechnungsaufwand. Als Maß dafür könnte z. B. die Anzahl der Regelanwendungen dienen oder ein Ausdruck der Komplexität von Ableitungsketten. Die einfachste Methode – und durchaus die Intention treffend – ist es, die absolute Rechenzeit für die Modifikation zu minimieren; „einfach“ bedeutet dann: die Modifikation benötigt eine geringe Rechnerzeit.

Wie wirkt sich diese Vorgehensweise auf die Qualität der Pläne aus?

In Abschnitt 3.1.2 wurde einiges über Gütekriterien von Arbeitsplänen gesagt. Es muß Ziel der fallbasierten Planung sein, gute Pläne in bezug auf diese Kriterien zu generieren.

Grundsätzlich ist zu der Planqualität bei der fallbasierten Planung folgendes zu sagen: durch die Zweiteilung in Retrieval und Modifikation („mache erst ein gutes Retrieval und anschließend eine gute Modifikation“) wird eine lokale Optimierung durchgeführt. Prinzipiell besteht jedoch die Möglichkeit, daß der Modifier u.U. durch Rücknahme beliebig vieler Planschritte den alten Plan so lange nachoptimiert, bis in jedem Fall ein guter Plan generiert wird.

Nun speziell zu meiner Sichtweise von fallbasierter Planung: Widerspricht das oben vorgestellte Ähnlichkeitsmaß dem Ziel, gute Pläne zu generieren? Ziel ist es, einen alten Fall zu finden, dessen Modifikation wenig rechenaufwendig ist. Das bedeutet von vorneherein noch nicht, daß der modifizierte Plan günstig in bezug auf Fertigungskosten und -qualität ist. Dies kann allerdings dann gewährleistet werden, wenn der Modifier in der Lage ist, technologisch mögliche, aber schlechte Pläne nachzuoptimieren. Ein Ähnlichkeitsmaß, das den Modifikationsaufwand berücksichtigt, wird dann einen Plan, der in großen Teilen optimiert werden muß, als wenig ähnlich einschätzen. Um dies zu leisten, muß das Ähnlichkeitsmaß zu optimierende Planteile erkennen und den Aufwand für die Optimierung abschätzen.

Ist dieser pragmatische Ansatz der Vorgehensweise des Menschen entsprechend?

Es gibt verschiedene Gründe, die Methoden von KI-Systemen an das menschliche Problemlöseverhalten anzulehnen:

- Es besteht die Hoffnung, die Qualität und Performanz menschlichen Problemlösens so auf das maschinelle Problemlösen übertragen zu können.
- Entwickler und Anwender haben bessere Möglichkeiten der Interaktion und des Debugging, da der Mensch die Schlußfolgerungen des Systems nachvollziehen kann.
- Implementierungen von kognitiv adäquaten Schlußfolgerungsmechanismen lassen interessante Rückschlüsse auf das menschliche Problemlöseverhalten zu.

Der vorgestellte Ansatz ist nicht kognitiv adäquat in dem Sinne, daß das System in bestimmten Situationen andere alte Fälle zur Wiederverwendung auswählen wird als der menschliche Experte und insofern die Schlußweisen des Systems nicht immer vom Arbeitsplaner nachvollziehbar sind. Der Grund dafür ist nicht im Konzept für die Ähnlichkeit zu suchen. Auch der menschliche Arbeitsplaner wählt, wenn er z.B. eine Ähnlichkeitsplanung durchführt, den Plan aus, bei dem er einen möglichst geringen Modifikationsaufwand hat. Der Unterschied liegt darin, daß bestimmte Modifikationen für Mensch und System verschieden aufwendig sind. Typischerweise sind z.B. numerische Berechungen für ein Computerprogramm einfacher durchzuführen als für den Menschen. Man könnte sich nun auf den Standpunkt stellen, daß aus Gründen der Nachvollziehbarkeit das menschliche Problemlöseverhalten oder speziell das menschliche Verständnis von Ähnlichkeit zwischen Arbeitsplänen exakt nachgebildet werden sollte. Ich halte dies nicht für sinnvoll und möchte es anhand eines Beispiels belegen:

Für einen gehfaulen menschlichen Arbeitsplaner mag es sehr aufwendig sein, zu einem Aktenschrank zu gehen um spezielle Unterlagen einzusehen; er wird sich daher diese Information statt dessen lieber aus verschiedenen Unterlagen auf seinem Schreibtisch erschließen, falls dies möglich ist. Stehen beide Informationsquellen dem System z.B. in Form von Dateien zur Verfügung, so wird man sinnvollerweise zur Vermeidung unnötiger Schlußfolgerungen direkt auf die Informationen „im Aktenschrank“ zugreifen; die Gehfaulheit des Menschen zu simulieren wäre unsinnig.

In einem weiter gefaßten Sinne ist die vorgestellte Vorgehensweise jedoch sehr wohl kognitv adäquat. Sie versucht nämlich, analog zur Vorgehensweise des Menschen, den Aufwand der Modifikation so gering wie möglich zu halten und diese Tatsache läßt die Hoffnung zu, die Performanz menschlichen Problemlösens so auf das System zu übertragen.

4.4.2 Der Prozeß der Ähnlichkeitsbestimmung

Folgende Aufgabe ist zu lösen: gegeben ist ein alter Fall (bestehend aus Werkstückrepräsentation, Rohteil, Bearbeitungsbereichen, Arbeitsplan und Verweisen von Teilen des Arbeitsplans auf Bearbeitungsbereiche) und eine aktuelle Planungsaufgabe (bestehend aus Werkstückrepräsentation, Rohteil und Bearbeitungsbereichen). Gesucht ist eine Hypothese für die durchzuführenden Modifikationen und eine Abschätzung für den Berechnungsaufwand für diese. Der Prozeß der Ähnlichkeitsbestimmung besteht aus vier Phasen, die sequentiell durchgeführt werden.

1. Analyse der aktuellen Planungsaufgabe und Generierung technologischer Constraints.
2. Geometrischer und technologischer Vergleich von aktuellem und altem Fall.
3. Generierung von Hypothesen für durchzuführende Modifikationen.
4. Abschätzung des Modifikationsaufwandes.

Die einzelnen Phasen werden nun beschrieben und jeweils anhand eines Beispiels erläutert (siehe Abbildung 4.2).

Generierung technologischer Constraints

In der ersten Phase wird das zu fertigende Teil technologisch untersucht und unter Verwendung von Regeln Aussagen über technologische Constraints, Fertigungsmöglichkeiten und

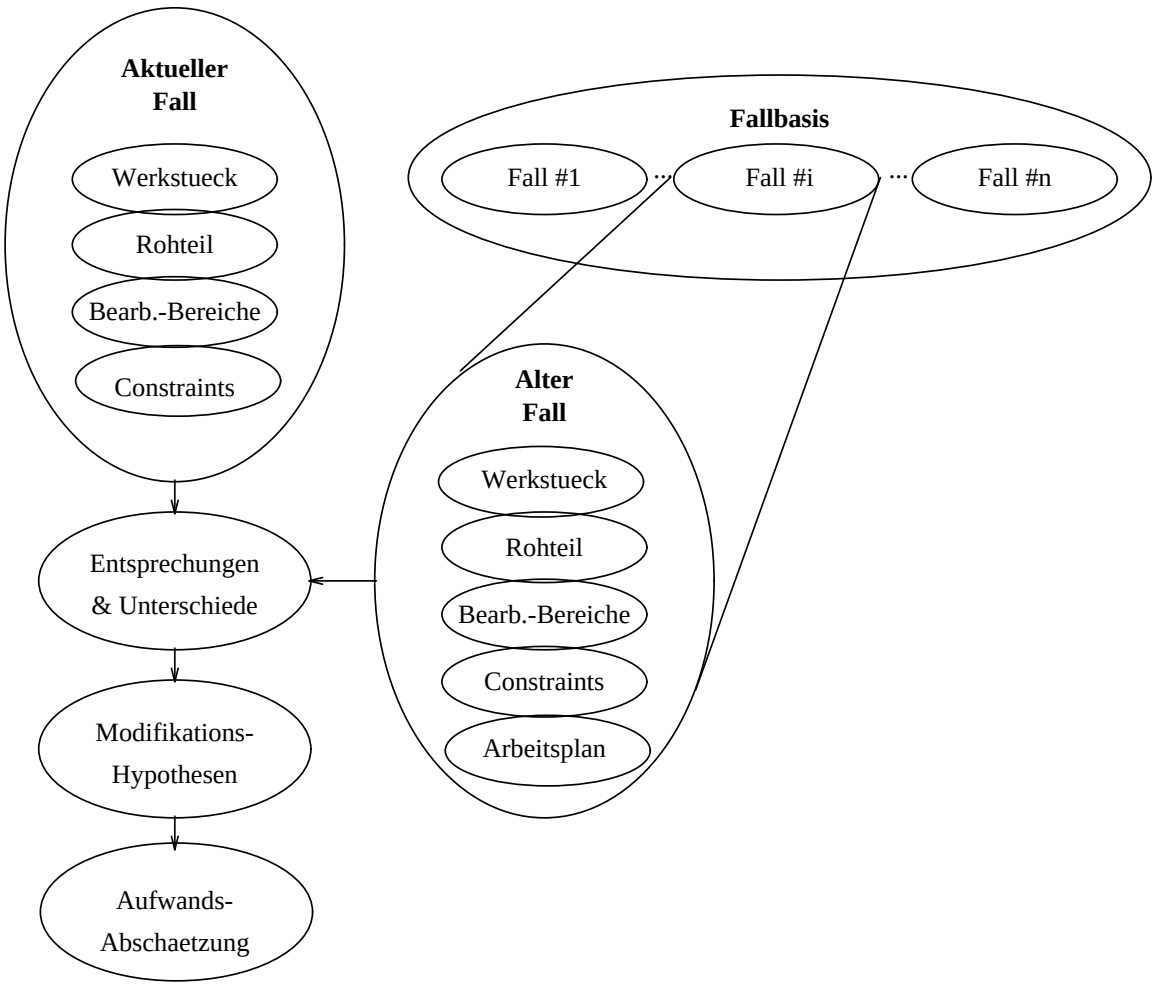

Abbildung 4.2: Berechnung der Ähnlichkeit zwischen aktuellem und altem Fall.

-schwierigkeiten generiert. Das Ergebnis kann auch für eine Einschränkung des Planungssuchraums verwendet werden, falls rein generativ geplant wird. Es ist wichtig zu erwähnen, daß es sich hier um eine reine Analysephase handelt, d.h. daß hier keinerlei Fertigungsentscheidungen getroffen werden, die prinzipiell auch anders hätten getroffen werden können. Liegt jedoch eine Entscheidung, z.B. für das Fertigungsverfahren eindeutig fest, so wird dies erkannt und festgehalten. Wenn viele Randbedingungen Planungsentscheidungen eindeutig machen, kann sich schon in dieser Phase ein relativ konkretes Planskelett ergeben. Die Aussagen können sich auf alle Planbereiche beziehen, also z.B. sowohl auf die Verfahrens- als auch auf die Werkzeugauswahl. Die generierten Constraints sind Aussagen, die für alle technologisch möglichen Arbeitspläne für das Werkstück gültig sind und können somit als „abstrakter Arbeitsplan" angesehen werden.

Ich werde nun am Beispiel des Werkstücks aus Abbildung 3.10 einen Ausschnitt der generierten Constraints und die dazugehörigen Regeln, die zu diesen geführt haben, vorstellen:

- Das Werkstück kann vollständig durch Drehbearbeitung gefertigt werden. Dies kann daraus geschlossen werden, daß der Slot `prismatische Elemente` im Frame `Werkstück A` leer ist.

- Für Drehteile werden die Typen `Scheibe, Normal` und `Welle` unterschieden, die nach dem Längen-Durchmesser-Verhältnis unterschieden werden. Das Beispielwerkstück fällt mit dem Verhältnis 182/115 in die Klasse `Normal.`

- Für Werkstücke des Typs Normal wird aufgrund des Längen-Durchmesser-Verhältnisses vorgeschrieben, daß sie stets mit mindestens zwei Einspannstellen gespannt werden müssen, also z.B. mit einem Futter auf der linken und einer Spitze auf der rechten Seite.

- Auf der Basis dieses Ergebnisses werden Stellen auf dem Werkstück gesucht, auf denen man mit diesen Spannmitteln spannen kann. Für unser Beispielwerkstück wird gefunden: Radiale Einspannstellen (z.B. für Spannen mit Futter) für Außenspannung mit Anschlag auf den `Materialelementen A_1.1` und `A_2.3`, eine radiale Einspannstelle für Innenspannung mit Anschlag auf dem `Bearbeitungsmolekül A_1.1.1` und eine axiale Einspannstelle (z.B. für Spitze) auf dem `Materialelement A_2.3.`

- Sehr entscheidend für die weitere Planung und insbesondere für die Fallauswahl ist das Erkennen von Fertigungsproblemen, da sie häufig in großem Maße die Auswahl und Reihenfolge von Fertigungsschritten mitbestimmen. Für das Beispielwerkstück wird an Problemen erkannt:

  - Planbereiche (am rechten und linken Werkstückrand) zu drehen bei zusätzlicher Forderung nach zwei Einspannstellen für das Fertigteil.
  - Innenbearbeitung bei zusätzlicher Forderung nach zwei Einspannstellen für das Fertigteil.
  - verschiedene Oberflächengüten innerhalb eines `Bearbeitungsatoms (A_1.1.1)`
  - Hinterschneidung entgegengesetzt der Hauptbearbeitungsrichtung im `Bearbeitungsatom A_1.1.1.`

- Für die durchzuführenden Bearbeitungen (aus der Bearbeitungsbereichsrepräsentation) wird untersucht, ob es Abhängigkeiten bezüglich deren Reihenfolge gibt. Für das Beispielwerkstück wird erkannt, daß die Innenbearbeitung (`Bearbeitungsatom A_1.1.1`) vor der rechten Kontur (`komplexer Bearbeitungsbereich A_BB_II`) gefertigt werden muß, da aus der Forderung nach zwei Einspannstellen für das Fertigteil folgt, daß für die Innenbearbeitung auf dem Rohteil, also vor Bearbeitung der rechten Kontur, gespannt werden muß.

In diesem Fall bestimmt die eindeutige Festlegung auf das Verfahren Drehen und Besonderheiten – im wesentlichen die Innenbearbeitung – die Arbeits- und Unterarbeitsvorgänge aller möglichen Arbeitspläne und die Reihenfolge derselben. Diese Information schränkt natürlich den Suchraum für das Retrieval enorm ein und ermöglicht es, alte Fälle zu finden, deren Arbeitspläne eine dem gesuchten Plan ähnliche Struktur haben.

Vergleich von aktuellem und altem Fall

In dieser Phase wird der aktuelle Fall sowohl geometrisch als auch fertigungstechnisch mit dem alten Fall verglichen. Dazu stehen für beide Fälle die formelementbasierte Werkstückrepräsentation, das Rohteil, die Bearbeitungsbereiche und prinzipiell auch der alte Arbeitsplan zur Verfügung.

Für einen geometrischen Vergleich wird versucht, Entsprechungen zwischen Formelementen und Bearbeitungsbereichen des alten Werkstücks mit denen des aktuellen zu finden. Dies geschieht für die Werkstücke im wesentlichen durch gewichtete Vergleiche der Slotwerte der entsprechenden Frame-Instanzen der formelementbasierten Werkstückrepräsentation. Das Ergebnis ist eine Liste sich entsprechender Formelemente von aktuellem und altem Fall und je einem numerischen Maß für deren Entsprechung (in %). Bei den komplexen Bearbeitungsbereichen werden die Entsprechungen danach beurteilt, in welchen Nachbarschaftsbeziehungen (links, rechts, außerhalb, innerhalb) sie zu sich entsprechenden Formelementen stehen.

Der technologische Vergleich beider Fälle generiert qualitative Aussagen über signifikante Fertigungsgemeinsamkeiten und -unterschiede. Während in den geometrischen Vergleich hauptsächlich die formelementbasierte Werkstückrepräsentation eingeht, werden hier hauptsächlich die Fertigungs-Constraints und der alte Arbeitsplan einbezogen.

Im Beispiel werde ich das Werkstück aus Abbildung 3.10 als alten Fall verwenden und als aktuellen Fall das identische Werkstück mit dem einzigen Unterschied, daß die Innenbearbeitung (`Bearbeitungsmolekül A_1.1.1`) durch eine Zentrierbohrung ersetzt wird(siehe Abbildung 4.3). Das bedeutet, daß beide Werkstücke sehr große geometrische Gemeinsamkeiten aufweisen; es wird sich jedoch zeigen (und aus diesem Grunde habe ich die beiden Fälle auch für das Beispiel ausgewählt), daß das Wegfallen der Innenbearbeitung fertigungstechnologische Änderungen bewirkt. Der alte Fall wird ab hier mit `A` und der neue Fall mit `B` (mit den Formelementen `B, B_1, B_1.1` etc.) bezeichnet.

Die fertigungstechnischen Constraints, die für Fall B generiert werden, sind im wesentlichen eine Untermenge der von Fall A, wobei alle Aussagen, die sich direkt oder indirekt auf die Innenbearbeitung beziehen, wegfallen. Folgende Unterschiede bestehen:

- Es wird eine zusätzliche axiale Spannstelle auf dem Materialelement A_1.1 erkannt.

- Die drei Fertigungsprobleme Innenbearbeitung bei zusätzlicher Forderung nach zwei

Abbildung 4.3: Fall B.

Einspannstellen, verschiedene Oberflächengüten innerhalb eines Bearbeitungsatoms und Hinterschneidung entgegengesetzt der Hauptbearbeitungsrichtung fallen weg.

- Kein Erkennen einer zwingenden Fertigungsreihenfolge.

Da der technologische Vergleich beider Fälle Informationen über Entsprechungen von Formelementen mitbenutzt, wird zuerst der geometrische Vergleich durchgeführt. Er liefert folgende Ergebnisse:

- Relativ hohe geometrische Entsprechung der Frames `Werkstück A` und `B` und der `Werkstückteile A_1` und `B_1` (95 %).
- Absolute Entsprechung (100 %) der `Werkstückteile A_2` und `B_2`, der `Materialelemente A_2.1` und `B_2.1`, `A_2.2` und `B_2.2`, `A_2.3` und `B_2.3`, und der `Bearbeitungsmoleküle A_2.3.1` und `B_2.3.1`.
- Keine Entsprechung für das Bearbeitungsmolekül A_1.1.1 (diese Aussage wird dann gemacht, wenn das maximale Maß für eine Entsprechung einen gewissen Schwellenwert – z.B. 75 % – unterschreitet).
- Entsprechung der komplexen Bearbeitungsbereiche: `A_BB_I` und `B_BB_I`, `A_BB_II` und `B_BB_II`, `A_BB_III` und `B_BB_III`, `A_BB_IV` und `B_BB_IV`.

Für den fertigungstechnologischen Vergleich ergeben sich die folgenden Aussagen:

- Beide Werkstücke können vollständig durch Drehbearbeitung gefertigt werden.
- Sie werden beide als Typ `Normal` klassifiziert.
- Daraus ergibt sich für beide, daß sie mit mindestens zwei Einspannstellen gefertigt werden müssen.
- Im Unterschied zu Fall A werden in Fall b keine offensichtlichen Reihenfolgeabhängigkeiten für die Bearbeitung erkannt.

Generierung von Hypothesen für durchzuführende Modifikationen

In dieser Phase gehen sehr stark der alte Arbeitsplan (siehe Abbildung 4.4), die Verweise von Teilen des Arbeitsplans auf Bearbeitungsbereiche und die Entsprechungen zwischen alten und aktuellen Bearbeitungsbereichen ein. Es wird untersucht, welche Teile des alten Arbeitsplans prinzipiell für den neuen Fall übernommen werden können und welche Teile modifiziert oder gelöscht werden müssen. Diese Untersuchung wird nur für die Arbeits- und Unterarbeitsvorgänge durchgeführt, da bei der Prozeßplanung (Maschinen- und Spannmittelauswahl und Reihenfolgeplanung) sehr große Interaktionen zwischen den Teilzielen auftreten, während die Operationsplanung (Werkzeugauswahl und Ermittlung von Steuervorgängen) eher separat für jedes Teilziel durchgeführt werden kann.

Folgende Vorgehensweise wird gewählt: Für die Arbeits- und Unterarbeitsvorgänge wird über die Verweise auf die alten Bearbeitungsbereiche und den Entsprechungen zwischen alten und aktuellen Bearbeitungsbereichen eine Verbindung zum aktuellen Fall hergestellt und so untersucht, ob ein Arbeitsschritt für den aktuellen Fall benötigt wird. Ist dies nicht der

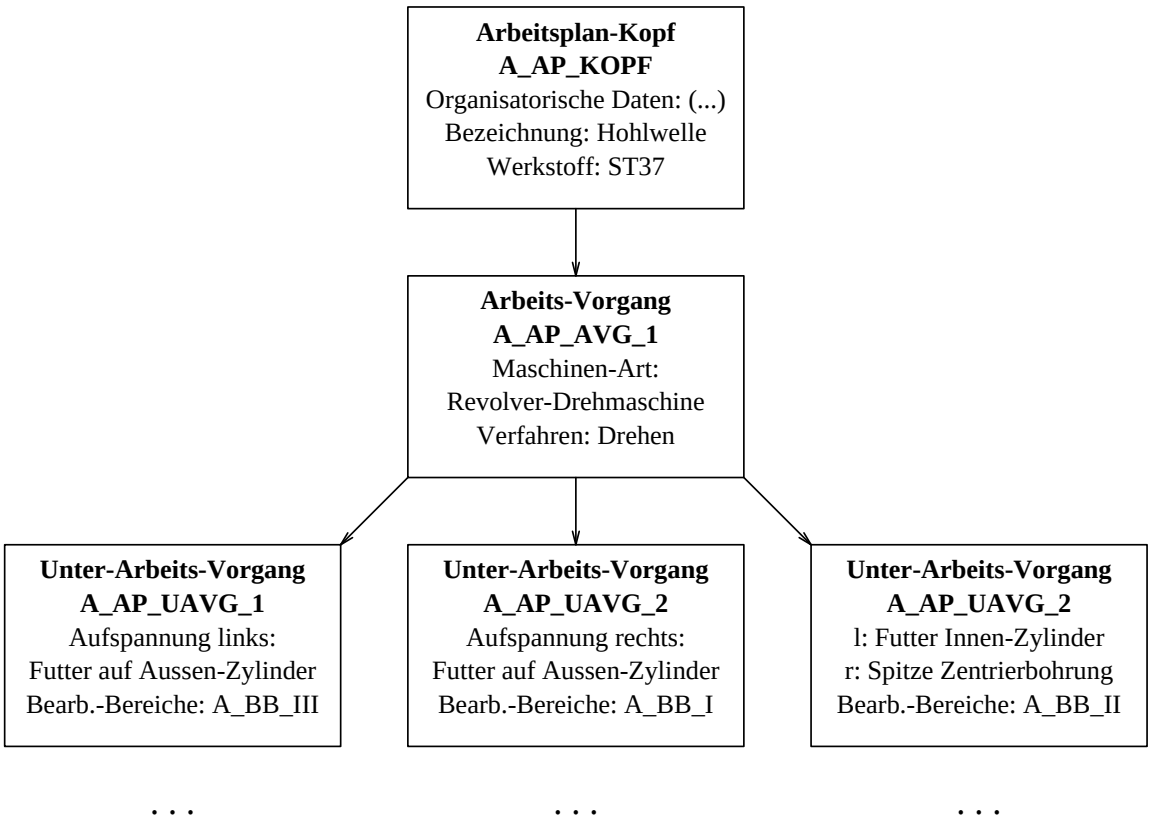

Abbildung 4.4: Arbeitsplanrepräsentation für Fall A.

Fall, so ist das Löschen des Arbeitsschrittes eine notwendige Modifikation. Ansonsten wird untersucht, ob fertigungstechnologische Constraints seine Anwendung (in diesem Kontext) verbieten. Ist dies der Fall, so muß er modifiziert werden; andernfalls wird untersucht, ob er in diesem Kontext zu einem guten Arbeitsplan führt. Beispielsweise kann die Festlegung eines Arbeitsschrittes im alten Plan aus einer speziellen Bedingung (z.B. einer Qualitätsanforderung) herrühren, die im aktuellen Fall nicht vorhanden ist. Dann kann der Arbeitsschritt, der zwar technologisch möglich aber zu aufwendig ist, vereinfacht werden. Es muß eine optimierende Modifikation durchgeführt werden.

Auf diese Art und Weise wird vom alten Arbeitsplan auf die Bearbeitung der neuen Bereiche geschlossen. Werden bei dieser Betrachtung aktuelle Bearbeitungsbereiche nicht erfaßt, so muß für diese im Rahmen der Modifikation eine generative Planung durchgeführt werden.

Der Arbeitsplan des Beispiels (der alte Fall ist Fall A) sieht wie folgt aus: Das komplette Werkstück kann auf einer einzigen Maschine gefertigt werden, einer sogenannten „Revolver-

Drehmaschine“ (`Arbeits-Vorgang A_AP_AVG_1`). Das Verfahren ist Drehen. Drei Aufspannungen werden benötigt: In der ersten Aufspannung (`Unter-Arbeits-Vorgang A_AP_UAVG_1`) wird mit einem Futter links auf dem Rohteil gespannt und auf der rechten Seite der Plandrehbereich und die Zentrierbohrung (`Bearbeitungs-Bereich A_BB_III`) gefertigt. In der zweiten Aufspannung (`Unter-Arbeits-Vorgang A_AP_UAVG_2`) wird rechts mit Futter auf dem Rohteil gespannt und links der Plandrehbereich und die Innenbearbeitung (`Bearbeitungs-Bereich A_BB_I`) gefertigt. Schließlich wird in der letzten Aufspannung (`Unter-Arbeits-Vorgang A_AP_UAVG_3`) links mit einem Innenfutter auf der Innenbearbeitung (`Bearbeitungs-Atom A_1.1.1`) und rechts mit einer Spitze auf der Zentrierbohrung (`Bearbeitungs-Atom A_2.3.2`) gespannt und die gesamte abfallende Kontur (`Bearbeitungs-Bereich A_BB_II`) gefertigt (siehe Abbildung 4.4).

Folgende Modifikationshypothesen werden generiert:

- Löschen der Bearbeitungsschritte für die Innenbearbeitung `A_1.1.1`, da sie im aktuellen Werkstück keine Entsprechung hat.
- Modifizieren der Aufspannung mit Innenfutter (verlangt Innenbearbeitung!), da sie im aktuellen Werkstück nicht möglich ist (z.B. Ersetzen der Innenaufspannung durch eine Außenaufspannung; dabei darauf achten, daß die Außenspannfläche zu einem früheren Zeitpunkt, z.B. in der zweiten Aufspannung, gefertigt wird).
- generative Planung für die Fertigung der Zentrierbohrung `B_1.1.1`, die keine Entsprechung im alten Werkstück hat.

Abschätzung des Modifikationsaufwandes

In der letzten Phase der Ähnlichkeitsbestimmung wird für jede Modifikationshypothese der absolute Zeitaufwand abgeschätzt und auf einen von der Größe des Arbeitsplans abhängigen Standardmodifikationsaufwand addiert. Anhand dieses numerischen Wertes wird durch das Retrieval ein alter Fall aus der Fallbasis ausgewählt und dessen Arbeitsplan unter Verwendung der Modifikationshypothesen angepaßt (falls der geschätzte Modifikationsaufwand den geschätzten Aufwand für eine generative Planung unterschreitet): je geringer der geschätzte Änderungsaufwand um so ähnlicher sind die Fälle.

Die letzte Phase der Ähnlichkeitsbestimmung wurde bislang nicht implementiert, da darin die Performanz des Modifiers eingehen muß (dessen Implementierung noch aussteht). Sicherlich werden jedoch die Modifikationen in folgener Reihenfolge (nach fallendem Aufwand) eingeschätzt:

- Modifikation der unmöglichen Aufspannung.
- generative Planung für die Zentrierbohrung.
- Löschen der Fertigungsschritte für die Innenbearbeitung.

4.5 Der Modifier

Folgende Anforderungen werden an den Modifier gestellt:

- Er muß „gute“ Pläne (im Qualitäts- und Kostensinne nach Abschnitt 3.1.2) generieren, u.U. durch Nachoptimieren von Teillösungen (man beachte, daß der Modifier auch durchzuführende Optimierungen erkennt).
- Er muß mit beliebig wenig Fallinformation zurechtkommen; d.h. er sollte in der Lage sein, vollständig generativ zu planen.

Die vom Modifier generierten Hypothesen für durchzuführende Modifikationen gehen als Aufgaben z.B. in eine Agenda des Modifiers ein. Folgende Arten von Fehlern im alten Plan müssen dabei behoben werden:

- Es werden zu viele Bearbeitungen durchgeführt; z.B. im alten Plan wird eine Nut gefräst, die im aktuellen Fall nicht vorhanden ist. Dann müssen die entsprechenden Planungsschritte aus dem Plan entfernt werden, wobei auf evtl. Zusammenhänge zu anderen Planteilen geachtet werden muß.
- Es werden zu wenige Bearbeitungen durchgeführt, z.B. muß eine zusätzliche Nut gefräst werden. Die fehlenden Schritte müssen generativ geplant und unter Berücksichtigung von Abhängigkeiten eingeführt werden.
- Bestimmte Parameter müssen modifiziert werden, z.B. absolute Maßangaben.
- Getroffene Entscheidungen müssen revidiert werden, da sie falsch (oder zu schlecht) sind; z.B. nach dem Drehen eines Gewindes wird das Spannen auf diesem Abschnitt kaum möglich sein.

Die folgende Vorgehensweise wird gewählt:

Nach dem Retrieval eines alten Falles aus der Fallbasis wird untersucht, ob der alte Arbeitsplan für eine Modifikation weiterverwendet werden soll, oder ob es effizienter ist, eine generative Neuplanung durchzuführen. Dazu wird der geschätzte Modifikationsaufwand mit einer Abschätzung des Aufwandes für eine generative Planung verglichen. Der Ansatz mit geringstem geschätzten Aufwand wird gewählt.

Wenn der alte Arbeitsplan weiterverwendet werden soll, so muß er zunächst dafür angepaßt werden. Dazu werden Teile des alten Arbeitsplans mit aktuellen Bearbeitungsbereichen assoziiert, indem Verweise auf alte Bearbeitungsbereiche durch solche auf die entsprechenden aktuellen ersetzt werden. Planteile, die dabei unberücksichtigt bleiben, werden gelöscht. Es entsteht so ein partieller Arbeitsplan, der u.U. nur einige Teilziele der Planungsaufgabe erfüllt und teilweise inkorrekt in Bezug auf die aktuelle Planungsaufgabe ist. Dieser partielle Plan wird der generativen Planungskomponente übergeben, die als Modifier fungiert.

Für die Modifikation wird folgende Vorgehensweise gewählt: zuerst werden bestehende Planteile auf ihre Korrektheit bezüglich der aktuellen Aufgabe getestet; dabei können falsche Entscheidungen zurückgenommen werden. Anschließend werden durch generative Planung fehlende Teilziele erfüllt.

Eine genaue Beschreibung der Integration von fallbasierter und generativer Planung unter Verwendung einer *Blackboard Architektur* erfolgt in [Humm & al. 91] (siehe auch Abbildung 4.5).

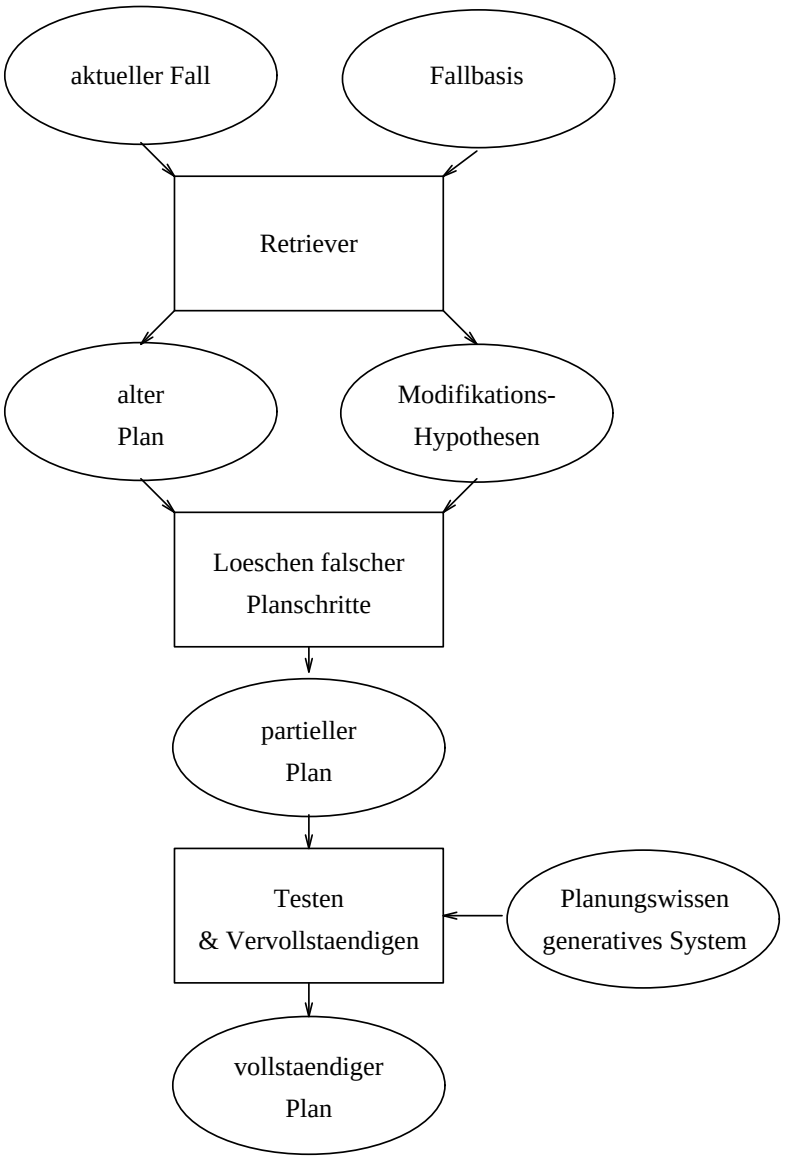

Abbildung 4.5: Integration von fallbasierter und generativer Planung.

4.6 Verbesserung von Ähnlichkeitsmaßen

Die hier vorgestellte Methode für die fallbasierte Planung liefert die Möglichkeit, das dem Retrieval zugrundeliegende Ähnlichkeitsmaß im Laufe des Einsatzes automatisch anzupassen bzw. zu verbessern. Dabei wird folgende Vorgehensweise gewählt (siehe Abbildung 4.6).

Der Retriever sucht in der Fallbasis nach dem der aktuellen Planungsaufgabe ähnlichsten Fall, generiert Hypothesen für durchzuführende Modifikationen und Abschätzungen für deren Aufwand. Anschließend wird der alte Arbeitsplan so lange modifiziert, bis man einen guten Arbeitsplan für das aktuelle Problem erhält und somit die Planungsaufgabe gelöst hat. Anhand der Modifikationsprotokolle kann anschließend getestet werden, ob das Ähnlichkeitsmaß die richtigen Modifikationen vorausgesagt hat und für diese die richtigen Zeitabschätzungen durchgeführt hat. Ist dies nicht der Fall, so muß das Ähnlichkeitsmaß geändert, z.B. spezialisiert werden. Dazu müssen aus dem Protokoll für die Hypothesengenerierung die entscheidenden Regeln aufgefunden und entsprechend abgeändert, z.B. spezialisiert werden.

4.7 Diskussion

4.7.1 Klassifikation als CBP-System

Ich werde meine Konzeption anhand der Kriterien aus Abschnitt 2.3.3 vorgestellten Merkmale klassifizieren und so mit dem im Abschnitt 2.3.4 vorgestellen *CHEF*-System vergleichen.

Was enthält ein Fall?

Ein Fall enthält weit mehr als nur die Planziele und den Plan selbst, wie dies bei *CHEF* der Fall ist. Ein Fall enthält die formelementbasierte Werkstückrepräsentation, aus der Fertigungsinformation gezogen werden kann, das Rohteil und die Bearbeitungsbereiche, die als Ziele der Planung bezeichnet werden können, da sie die Bereiche repräsentieren, die durch die Fertigung entfernt werden sollen. Desweiteren werden Fertigungs-Constraints gespeichert, die als Dokumentation des Plans dienen können; z.B. kann es wichtig sein, zu wissen, ob die Entscheidung für ein Spannmittel im Arbeitsplan zwingend war, oder ob aus einer Menge technologisch möglicher Spannmittel eines nach Optimalitätskriterien ausgewählt worden ist. Außerdem werden Verweise von Teilen des Arbeitsplans auf die entsprechenden Bearbeitungsbereiche gespeichert.

Wie ist die Fallbasis strukturiert?

Die Fallbasis ist als lineare Liste organisiert. Dies hat im Gegensatz zu *CHEF*, wo eine starre Baumstruktur verwendet wird, den Vorteil der größeren Flexibilität, jedoch den Nachteil der Ineffizienz. Eine flexible aber effiziente Speicherorganisationsform muß das Thema nachfolgender Arbeiten sein.

Wie werden Fälle akquiriert?

Die Fälle sollen wie bei *CHEF* durch das System akquiriert werden können; eine Konzeption für die Speicherungskomponente wurde jedoch nicht durchgeführt.

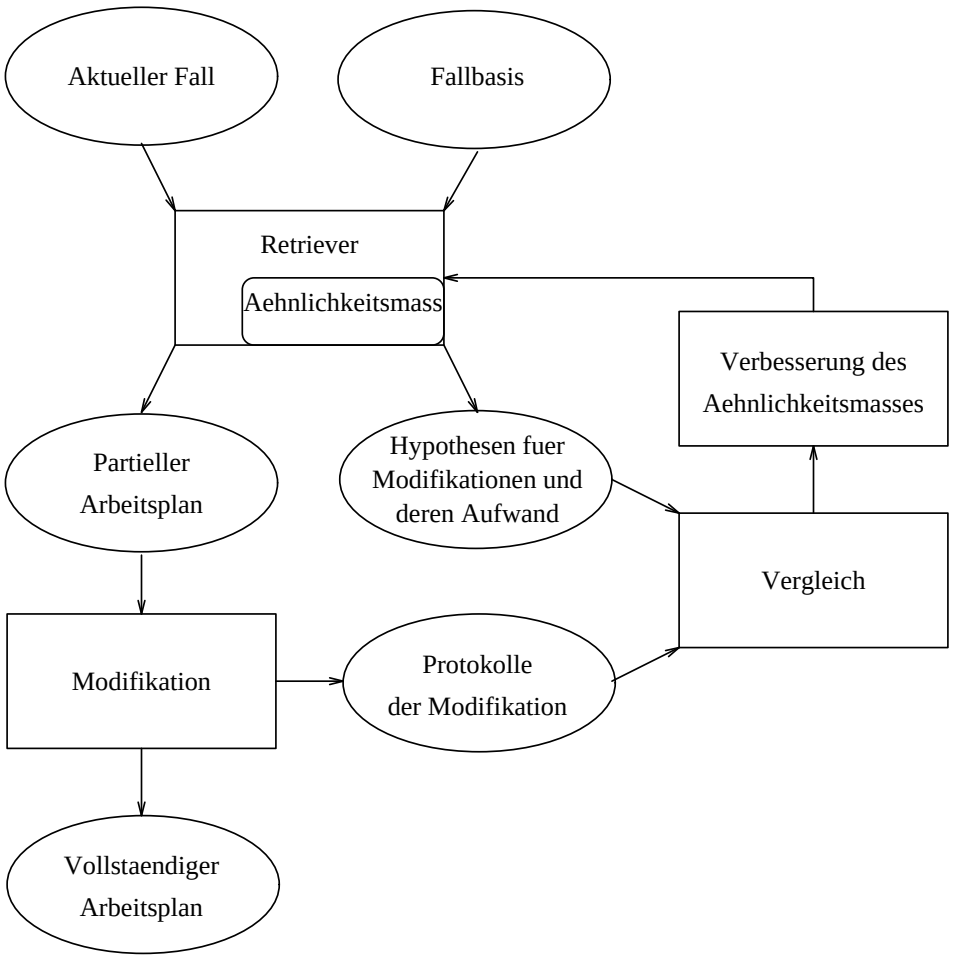

Abbildung 4.6: Verbesserung von Ähnlichkeitsmaßen.

Wieviele alte Fälle werden für die aktuelle Planungsaufgabe herangezogen?

Es wird wie bei *CHEF* ein einziger Fall wiederverwendet. Die Begründung für diese Vorgehensweise liegt darin, daß die Interaktionen zwischen Teilzielen der Arbeitsplanung (besonders in der Prozeßplanung) zu groß sind, um Fälle aufzutrennen.

Welche Größen gehen in das Retrieval ein?

Es können im Gegensatz zu *CHEF* alle Informationen, die ein Fall enthält, in das Retrieval eingehen, insbesondere der alte Plan. Dies ist aus dem folgenden Grund besonders wichtig: Es ist in der Regel möglich, für ein bestimmtes Werkstück verschiedene unterschiedliche Arbeitspläne zu generieren, die alle technologisch korrekt aber z.B. unterschiedlich kostengünstig sind. Für eine Wiederverwendung in einer neuen Situation können jedoch die verschiedenen Arbeitspläne unterschiedlich brauchbar sein; in der Fallbasis wird zu jedem Werkstück aber nur je ein Arbeitsplan abgespeichert. Daher muß in eine Ähnlichkeitsbetrachtung auch der mit dem Werkstück assoziierte Arbeitsplan eingehen.

Wird ausschließlich fallbasiert geplant?

Nein, es kann im Gegensatz zu *CHEF* abgeschätzt werden, ob der Aufwand für die fallbasierte Planung den der generativen übersteigt (dies geschieht auf der Basis der Modifikationshypothesen und der Abschätzung für deren Aufwand; für die Abschätzung des Aufwandes der generativen Planung wurde bislang kein Konzept entwickelt). Ist dies der Fall, so wird ausschließlich generativ geplant.

Welche Lernvorgänge finden statt?

Neben dem Lernen neuen Fallwissens wird hier besonders die Möglichkeit betont, Lernprozesse auf das Ähnlichkeitsmaß anzuwenden. Dies geschieht auf der Basis eines Vergleichs der Modifikationshypothesen und der Modifikationsprotokolle. Mir ist keine Systemkonzeption aus der Literatur bekannt, die das Verbessern von Ähnlichkeitsmaßen in dieser Form unterstützt. Bei *CHEF* wird durch das Erlernen von Failure Predictors eine Verbesserung des Retrievals erzielt. Allerdings bezieht sich diese ausschließlich auf zu vermeidende und nicht auf zu erfüllende Ziele.

Erfolgt die Planung hierarchisch?

Ja. In meiner Konzeption wird die Prozeßplanung fallbasiert durchgeführt; die Operationsplanung kann anschließend generativ oder auch fallbasiert (bzw. auf der Basis von Skelettplänen[3]) durchgeführt werden.

Erfolgt die Planung nichtlinear?

Für die Arbeitsplanung ist es sinnvoll, Reihenfolgen zwischen Teilzielen so lange offen zu lassen, bis sie durch Constraints festgelegt werden. Es macht daher auch Sinn, in die Fälle Informationen über Reihenfolgeabhängigkeiten mit aufzunehmen. Dies ist momentan noch nicht

[3] Für eine Diskussion der Generierung von Skelettplänen im Bereich Arbeitsplanung siehe [Bergmann 90].

realisiert; allerdings besteht die Möglichkeit, über die Fertigungs-Constraints u.a. Information über Reihenfolgeabhängigkeiten zu erhalten.

4.7.2 Klassifikation als CAPP-System

Ich werde nun die Konzeption des Systems nach den Kriterien aus Abschnitt 3.2.3 klassifizieren.

Planungsmethoden

Das System ist prinzipiell in der Lage, für die drei in Abschnitt 3.1.3 vorgestellten Planungsanlässe adäquate Methoden zur Verfügung zu stellen:

Ist ein Arbeitsplan neu zu erstellen, so kann, falls keine signifikante Fallinformation zur Verfügung steht, generativ geplant werden. Andernfalls wird fallbasiert geplant. Auch Arbeitspläne für Varianten können in die Fallbasis aufgenommen werden.

Ist ein Arbeitsplan zu ändern, selektiert der Retriever ihn aus der Fallbasis und der Modifier ändert u.U. einzelne Planteile aufgrund von Betriebsmitteländerungen.

Ist ein Plan einfach zu wiederholen, so selektiert der Retriever ihn aus der Fallbasis.

Teilespektrum

Die formelementorientierte Werkstückrepräsentation ist bislang nur für rotationssymmetrische Teile realisiert (und damit ist das CAPP-System auch nur für dieses Spektrum ausgelegt), kann und soll aber auf allgemeine Drehteile ausgedehnt werden.

Funktionsumfang

Eine Anbindung an alle CIM-Bereiche ist geplant.

Bearbeitungsverfahren

Alle Standard-Bearbeitungsverfahren sollen unterstützt werden.

Automatisierungsgrad

Das Retrieval läuft prinzipiell vollautomatisch ab. Benutzerinteraktionsmöglichkeiten während der Planung müssen vorgesehen werden, damit der Arbeitsplaner z.B. die Möglichkeit hat, in bestimmten Situationen andere Optimalitätskriterien anzuwenden (z.B. „Geld spielt keine Rolle; wichtig ist nur die Fertigungszeit“).

Bisher im Einsatz befindliche CAPP-Systeme, die der Methode der Ähnlichkeitsplanung zugeordnet werden, sind reine Retrieval-Systeme; sie suchen (ausschließlich basierend auf den geometrischen Informationen in den Sachmerkmalsleisten) einen alten Arbeitsplan aus, der auf dem Bildschirm zur manuellen Modifikation zur Verfügung gestellt wird. Im Gegensatz zum hier vorgestellten Ansatz wird keine automatische Modifikation bzw. Planung durchgeführt.

Werkstückrepräsentation

Die formelementbasierte Werkstückrepräsentation versucht die interne Werkstückrepräsentation von Experten des Maschinenbaus widerzuspiegeln. Das Vokabular ist so ausgelegt, daß z.B. Arbeitsplaner mit Namen von Bearbeitungsmolekülen spezielle Fertigungsverfahren assoziieren.

4.7.3 Kompensierte Defizite bisheriger Ansätze für CAPP

In Abschnitt 3.2.4 wurden einige Defizite bisheriger Ansätze für CAPP vorgestellt. Es soll nun untersucht werden, welche dieser Probleme durch den hier vorgestellten Ansatz kompensiert werden können:

Planungsumfang: Der vorgestellte Ansatz ermöglicht es prinzipiell, ein beliebiges Teilespektrum zu bearbeiten; insbesondere ist er nicht auf Varianten beschränkt. Grundsätzlich kann auch die gesamte Planung vom System durchgeführt werden; man sollte jedoch darauf achten, dem Benutzer stets die Möglichkeit der Interaktion offen zu lassen.

Planungsergebnisse: Verwendet man in der Fallbasis gute (z.B. global optimierte) Fälle, so besteht die Hoffnung, daß man deutlich bessere Ergebnisse erzielt, als bei einer generativen Planung mit lokaler Optimierung.

Effizienz: Der vorgestellte Ansatz ist in besonderem Maße auf die Effizienz des Planungsverfahrens ausgerichtet; die der Konzeption zugrundeliegende Definition für Ähnlichkeit verdeutlicht dies. Durch Vermeidung wiederholter (generativer) Planung läßt sich der gesamte Planungsvorgang verkürzen.

Aktualisierung des Planungswissens: Die Akquisition von Wissen über neue Ressourcen läßt sich elegant dadurch unterstützen, daß man Fälle in die Fallbasis aufnimmt, die diese Ressourcen verwenden. Eine Aktualisierung des Planungswissens der generativen Komponente (die als Modifier dient) wird damit jedoch nicht überflüssig.

Kapitel 5

Implementierung

Von der Konzeption für das fallbasierte CAPP-System wurden die drei ersten Phasen der Ähnlichkeitsbestimmung prototypisch implementiert und in das in Kapitel 3.3 vorgestellte generative CAPP-System integriert; das bedeutet, daß das fallbasierte System vom integrierten Datenmodell die Repräsentationen für Werkstücke, Bearbeitungsbereiche und Arbeitspläne verwendet. Als Implementierungssprache dient *KEE*. Die letzte Phase der Ähnlichkeitsbestimmung wurde nicht implementiert, da sie von der Implementierung des Modifiers abhängt, die noch aussteht. Ziel der prototypischen Implementierung ist es, Möglichkeiten und Probleme zu skizzieren, die bei der Realisierung des vorgestellten Konzeptes auftreten. Es besteht jedoch kein Anspruch auf ein umfangreiches in der Praxis einsetzbares System.

Ich werde nun die Implementierungen der drei Phasen im einzelnen erläutern und durch Beispiele einiger Regeln und Funktionen illustrieren.

5.1 Generierung technologischer Constraints

Die *KEE-Knowledge Base* `Constraints` auf der Datei `constraints.u` enthält Regeln für die Generierung sogenannter *Unstructured Facts*, *KEE*-Objekten, die beliebig strukturierte Aussagen über Constraints, Möglichkeiten und Probleme der Fertigung eines Werkstücks beinhalten. Die Datei `constraints.fcts.lisp` enthält die Funktion `generiere.constraints`, die im wesentlichen eine Vorwärtsregelaktivierung durchführt, und Hilfsfunktionen für die Regeln.

An einigen Beispielen sollen Regeln und Funktionen erläutert werden. Dabei werde ich erst die Regel in verbaler Form beschreiben, dann den Regeltext vorstellen und dokumentieren.

Die Regel „BEARBEITUNG_NOTWENDIG_(...)“

Beschreibung: Falls ein Materialelement, das sich am (rechten oder linken) Rand eines Werkstückes befindet, eine Innenbearbeitung enthält, so ist eine Bearbeitung des Werkstücks von dieser Seite aus notwendig.

Regeltext:

```
(BEARBEITUNG_NOTWENDIG_VON_?SEITE_?INNEN-BEARBEITUNG
```

```
(IF
    (?MATERIAL-ELEMENT IS IN CLASS MATERIAL-ELEMENT)
    (THE RAND-ELEMENT OF ?MATERIAL-ELEMENT IS ?SEITE)
    (AN INNEN-BEARBEITUNGEN
        OF
        ?MATERIAL-ELEMENT
        IS
        ?INNEN-BEARBEITUNG)
    THEN
    (TEXT (BEARBEITUNG NOTWENDIG VON ?SEITE
                              ?INNEN-BEARBEITUNG))))
```

Dokumentation der Regel: Der Name der Regel ist nach dem Fakt gewählt, der durch die Regel generiert wird. Es wird eine Instanz der Klasse `Material-Element` gesucht und an die Variable `?Material-Element` gebunden. Anschließend wird getestet, ob dieses Materialelement ein Randelement ist (d.h. der Wert des Slots `Rand-Element` ist nicht `nil`); wenn ja, wird der Slotwert (Ausprägung `links` oder `rechts`) an die Variable `?Seite` gebunden. Es wird nun überprüft, ob das Materialelement eine Innenbearbeitung aufweist; wenn ja, so wird der Name der Innenbearbeitung an die Variable `?Innen-Bearbeitung` gebunden. Sind alle drei Prämissen erfüllt (konjunktive Verknüpfung), so feuert die Regel und die Aktionen der Konklusion werden ausgeführt. In diesem Fall wird aufgrund des `TEXT`-Operators der Unstructured Fact `(BEARBEITUNG NOTWENDIG VON ?SEITE ?INNEN-BEARBEITUNG)` in die Faktenbasis übernommen, wobei die beiden Variablen durch ihre Bindungen ersetzt werden.

Die Regel „BEARBEITUNG_PROBLEM_(...)“

Beschreibung: Ein besonderes Problem stellt die Bearbeitung von Einstichen dar, wenn keine einheitliche Bearbeitungsrichtung vorliegt, sondern in verschiedene Richtungen gespant werden muß. Dann kann z.B. ein Werkzeugwechsel notwendig werden, der möglichst vermieden werden sollte.

Regeltext:

```
(BEARBEITUNG_PROBLEM_HINTERSCHNEIDUNG-(...)
  (IF (?BEARBEITUNGS-MOLEKUEL IS
                              IN
                              CLASS
                              BEARBEITUNGS-MOLEKUEL)
      (LISP (WECHSEL-BEARBEITUNGS-RICHTUNG?
             ?BEARBEITUNGS-MOLEKUEL))
      THEN
      (TEXT
        (BEARBEITUNG PROBLEM
                     HINTERSCHNEIDUNG_(...)
                     ?BEARBEITUNGS-MOLEKUEL))))
```

Dokumentation der Regel: Es wird eine Instanz der Klasse `Bearbeitungs-Molekuel` gesucht und an die Variable `?Bearbeitungs-Molekuel` gebunden, für das das Prädikat `Wechsel-Bearbeitungs-Richtung?` gültig ist. Das Prädikat ist als *LISP*-Funktion definiert und wird durch den `LISP`-Operator aufgerufen. Sie soll anschließend vorgestellt werden.

Die Funktion „wechsel-bearbeitungs-richtung?"

Beschreibung: Ein Wechsel in der Bearbeitungsrichtung wird dann notwendig, wenn Bearbeitungsatome eines Moleküls gedreht zueinander stehen.

Funktions-Code:

```
(defun wechsel-bearbeitungs-richtung?
        (bearbeitungs-molekuel)
  (NOT
   (APPLY #'=
     (MAPCAR
       #'(LAMBDA (BEARBEITUNGS-ATOM)
         (GET.VALUE BEARBEITUNGS-ATOM 'DREHUNG))
       (GET.VALUES BEARBEITUNGS-MOLEKUEL 'BEARBEITUNGS-ATOME)))))
```

Dokumentation der Funktion: Der Slot `Bearbeitungsatome` des Frames `Bearbeitungsmolekül` enthält alle Atome, aus denen das Molekül zusammengesetzt ist. Die *KEE*-Funktion `get.values` greift auf Slotwerte zu. Es wird getestet, ob alle Atome denselben Drehsinn aufweisen. Der Slot `Drehung` eines Bearbeitungs-Atoms drückt den Drehwinkel des Atoms von einer definierten Normalstellung aus. Haben nicht alle Atome denselben Drehsinn, so liefert das Prädikat `T`, ansonsen `NIL`.

Generierte Fakten

Auf diese Art werden Unstructured Facts für Constraints, Möglichkeiten und Probleme der Bearbeitung eines Werkstücks generiert. Für `Werkstück A` ergeben sich z.B. folgende Fakten:

```
(VERFAHREN DREHEN)
(KONTUR HORIZONTAL_ABFALLEND)
(TYP NORMAL)
(EINSPANNUNG FERTIGTEIL ANZAHL 2)
(EINSPANNUNG ROHTEIL RECHTS STELLE RADIAL AUSSEN MIT_ANSCHLAG A_2.3)
(EINSPANNUNG ROHTEIL LINKS STELLE RADIAL AUSSEN MIT_ANSCHLAG A_1.1)
(EINSPANNUNG FERTIGTEIL RECHTS STELLE AXIAL ZENTRIERBOHRUNG A_2.3)
(EINSPANNUNG FERTIGTEIL LINKS STELLE RADIAL INNEN MIT_ANSCHLAG A_1.1.1)
(BEARBEITUNG PROBLEM INNEN-BEARBEITUNG_BEI_2_EINSPANN-STELLEN A_1.1.1)
(BEARBEITUNG PROBLEM HINTERSCHNEIDUNG_ENTGEGENGESETZT_BEARBEITUNG A_1.1.1)
(BEARBEITUNG PROBLEM VERSCHIEDENE_OBERFLAECHEN-GUETEN A_1.1.1)
(BEARBEITUNG REIHENFOLGE A_BB_I A_BB_II)
```

Sie beschreiben i.w. die in Abschnitt 4.4.2 beschriebenen Aussagen.

5.2 Vergleich von aktuellem und altem Fall

Die Datei `aehnlichkeit.fcts.lisp` enthält Funktionen für den geometrischen Vergleich der Formelemente von aktuellem und altem Fall, die Knowledge Base `aehnlichkeit` in der Datei `aehnlichkeit.u` enthält Regeln zum Vergleich der Constraints von aktuellem und altem Fall.

Der geometrische Vergleich wird beispielhaft am Vergleich von Bearbeitungsmolekülen vorgestellt.

Die Funktion „entsprechung_material-elemente“

Beschreibung: Es werden für ein Werkstückteil des aktuellen und alten Falles alle Material-Elemente miteinander verglichen und jeweils die Paare mit der höchsten Übereinstimmung erkannt. Dabei wird nur Information verwendet, die in der formelementbasierten Werkstückrepräsentation enthalten ist. Sie enthält zwar nur geometrische Informationen; diese sind aber so ausgewählt, daß man aus ihnen einfach fertigungstechnologische Informationen ziehen kann.

Funktions-Code:

```
(defun entsprechung_material-elemente (werkstueck-teil1
                                       werkstueck-teil2)
  (...)
 (mapcar
  #'(lambda (material-element1)
      (vergleiche_material-elemente
       material-element1
       material-element2))
  material-elemente1)
  (...)
(if
 (>= mass 75)
 (assert(...)
(entsprechung_bearbeitungs-molekuele
         material-element1
         material-element2))
  (...)
```

Dokumentation der Funktion: Für jedes Materialelement des alten Falles werden alle Materialelemente des aktuellen Falles verglichen und für das mit der höchsten Übereinstimmung der Fakt `(entsprechung_bearbeitungs-molekuele material-element1 material-element2)` generiert. Dabei generiert die Funktion `vergleiche_material-elemente` einen numerischen Wert zwischen 0 und 100, wobei 100 die vollkommene Übereinstimmung der untersuchten Slotwerte darstellt. Werden mehrere Materialelemente des aktuellen Falls als gleich ähnlich eingestuft, so wird dasjenige ausgewählt, dessen Position innerhalb des Werkstücks dem des alten am nächsten ist. Das gewährleistet, daß beim Vergleich zweier identischer Werkstücke jedes Materialelement (entsprechendes gilt für alle Formelemente) mit sich selbst assoziiert wird.

Für alle miteinander assoziierten Materialelemente wird durch Aufruf der Funktion `entsprechung_bearbeitungs-molekuele` der Vergleich der in ihnen enthaltenen Bearbeitungsmoleküle durchgeführt.

Die Funktion „vergleiche_bearbeitungs-molekuele“

Beschreibung: Es werden zwei Bearbeitungsmoleküle miteinander verglichen und ein numerisches Ähnlichkeitsmaß zwischen 0 und 100 generiert.

Funktions-Code:

```
(defun vergleiche_bearbeitungs-molekuele
        (bearbeitungs-molekuel1
         bearbeitungs-molekuel2)
  (...)
   (alist
    'bearbeitungs-atome           '(1 =length)
    'DIN-norm                     '(3 equal)
    'form-lage-toleranz           '(1 equiv)
    'hoechste_oberflaechen-guete  '(1 equiv)
    'lage                         '(3 equal)
    'schaerfste_mass-toleranz     '(1 equiv)
    'typ                          '(3 equal)
    'unter-typ                    '(2 equal)
  (...)
      (if
       (funcall1
                        (third (assoc slot parameter))
                        (get.values bearbeitungs-molekuel1 slot)
                        (get.values bearbeitungs-molekuel2 slot))
       (second (assoc slot parameter))
       0))
  (...)
```

Dokumentation der Funktion: Für jeden Slot des Frames `Materialelement` wird ein Wert für dessen Gewichtung und eine Vergleichsfunktion angegeben. Mit diesen Funktionen werden die entsprechenden Slotwerte miteinander verglichen, mit dem Skalierungsfaktor gewichtet, aufaddiert und normiert.

Nun zum technologischen Vergleich der Werkstücke. Dabei werden die generierten Constraints für beide Werkstücke miteinander verglichen.

Die Regel „GEMEINSAMKEIT_CONSTRAINTS_(...)“

Beschreibung: Wenn zwei Formelemente dasselbe Fertigungsproblem implizieren, so ist das ein positiver Hinweis auf die Ähnlichkeit der Werkstücke.

Regeltext:

```
(GEMEINSAMKEIT_CONSTRAINTS_BEARBEITUNG_PROBLEM_?PROBLEM
  (IF (TEXT
        (CONSTRAINT AKTUELLER_FALL
         BEARBEITUNG PROBLEM ?PROBLEM ?ELEMENT1))
      (TEXT
        (CONSTRAINT ALTER_FALL
         BEARBEITUNG PROBLEM ?PROBLEM ?ELEMENT2))
      THEN
      (TEXT (GEMEINSAMKEIT CONSTRAINTS
             BEARBEITUNG PROBLEM ?PROBLEM))
      (TEXT
        (ENTSPRECHUNG ?ELEMENT1 ?ELEMENT2
         BEARBEITUNG PROBLEM ?PROBLEM))))
```

Dokumentation der Regel: Es werden in der Faktenbasis Fakten für alten und aktuellen Fall gesucht, die dieselbe Problembeschreibung beinhalten.

Generierte Fakten

```
(ENTSPRECHUNG B A 93.75)
(ENTSPRECHUNG B_1 A_1 95.0)
(ENTSPRECHUNG B_2 A_2 100.0)
(ENTSPRECHUNG B_1.1 A_1.1 94.12)
(ENTSPRECHUNG B_2.1 A_2.1 100.0)
(ENTSPRECHUNG B_2.2 A_2.2 100.0)
(ENTSPRECHUNG B_2.3 A_2.3 100.0)
(ENTSPRECHUNG B_2.3.1 A_2.3.1 100.0)
(ENTSPRECHUNG B_2.3.2 A_2.3.2 100.0)
(ENTSPRECHUNG B_BB_I A_BB_I 100.0)
(ENTSPRECHUNG B_BB_II A_BB_II 100.0)
(ENTSPRECHUNG B_BB_III A_BB_III 100.0)

(GEMEINSAMKEIT CONSTRAINTS VERFAHREN DREHEN)
(GEMEINSAMKEIT CONSTRAINTS KONTUR HORIZONTAL_ABFALLEND)
(GEMEINSAMKEIT CONSTRAINTS TYP NORMAL)
(GEMEINSAMKEIT EINSPANNUNG ANZAHL 2)
```

5.3 Generierung von Hypothesen für durchzuführende Modifikationen

Die Datei `modifikationen.fcts.lisp` enthält Funktionen zur Generierung von Fakten, die Aussagen über durchzuführende Modifikationen des alten Arbeitsplans machen. Die Funktion `generiere.modifikationen` ruft diesen Prozeß auf.

Die Funktion „generiere.modifikationen“

Beschreibung: Es wird über Verweise von Teilen des alten Arbeitsplans auf alte Bearbeitungsbereiche und erkannten Entsprechungen von alten und neuen Bearbeitungsbereichen auf die Anwendbarkeit von Teilen des alten Arbeitsplans für das aktuelle Problem geschlossen. Anschließend wird untersucht, welche aktuellen Bearbeitungsbereiche dabei nicht erfaßt werden.

Funktions-Code:

```
(defun generiere.modifikationen (aktueller_fall alter_fall)
  (...)
  (modifikationen.ap ap-kopf)
  (...)
       (if
        (not
         (query `(text
          (modifikation behandelt ,(unit.name bb)))))
        (assert.verbose `(text
         (modifikation generieren ,(unit.name bb))))))
  (...)
```

Dokumentation der Funktion: Durch Aufruf der Funktion `modifikationen.ap` mit dem Arbeitsplankopf des alten Falles als Argument wird die Untersuchung der Anwendbarkeit des alten Arbeitsplans durchgeführt. Dabei wird auch für alle in irgend einer Weise durch den alten Arbeitsplan erfaßten neuen Bearbeitungsbereiche ein Fakt `(modifikation behandelt ?Element)` generiert. Es wird anschließend für alle Bearbeitungsbereiche und -moleküle untersucht, ob sie schon behandelt sind; ansonsten wird ein Fakt `(modifikation generieren ?Element)` generiert um festzuhalten, daß für diese Elemente eine generative Arbeitsplanung durchzuführen ist.

Die Funktion „modifikationen.ap“

Beschreibung: Es wird für alle Arbeitsvorgänge eines Arbeitsplans untersucht, ob die Verfahren auch im aktuellen Fall angewendet werden können.

Funktions-Code:

```
(defun modifikationen.ap (ap-kopf)
  (mapc
   #'(lambda (avg)
       (if
        (query `(text (constraint aktueller_fall
                       verfahren (...)verfahren)))
        (modifikationen.avg avg)
        (assert.verbose `(text (modifikation loeschen
                                ,avg)))))
   (get.values ap-kopf 'arbeits-vorgaenge))
  )
```

Dokumentation der Funktion: Ist ein im alten Arbeitsplan verwendetes Verfahren, z.B. Drehen, in der aktuellen Situation nicht anwendbar, so wird der Fakt `(modifikation loeschen ?Arbeitsvorgang)` generiert.

Die Funktion „modifikationen.avg“

Beschreibung: Es wird für jeden Unterarbeitsvorgang eines Arbeitsvorgangs untersucht, ob er im aktuellen Fall sinnvoll, möglich und gut (im Sinne von Optimalitätskriterien für Arbeitspläne) ist.

Funktions-Code:

```
(defun modifikationen.avg (avg)
  (mapc
   #'(lambda (uavg)
       (and
         (aufspannung.notwendig uavg)
         (aufspannung.moeglich uavg)
         (aufspannung.gut uavg)))
   (get.values avg 'unter-arbeits-vorgaenge)))
```

Dokumentation der Funktion: Es werden die Funktionen `aufspannung.notwendig`, `aufspannung.moeglich` und `aufspannung.gut` aufgerufen, wobei hier die nicht strikte Implementierung von `and` in LISP verwendet wird: wenn eine Aufspannung nicht notwendig ist, so wird weder ihre Möglichkeit noch ihre Güte untersucht, und wenn eine Aufspannung nicht möglich ist, so wird auch nicht ihre Güte untersucht.

Die Funktion `aufspannung.notwendig` untersucht im wesentlichen Entsprechungen von Bearbeitungsbereichen und -molekülen; `aufspannung.moeglich` verwendet Fertigungs-Constraints des aktuellen Falls; `aufspannung.gut` ist noch nicht weiter ausgeführt.

Generierte Fakten

```
(MODIFIKATION MODIFIZIEREN A_AP_UAVG_3 LINKS)
(MODIFIKATION GENERIEREN B_1.1.1_ZENTRIERB.)
(MODIFIKATION LOESCHEN A_AP_UAVG_2 A_1.1.1)
(MODIFIKATIONEN KEINE VERFAHREN A_AP_AVG_1)
```

5.4 Aufruf des Systems

Die Funktion `aehnlichkeits.mass` startet den Vergleichsprozeß eines aktuellen Falls mit einem alten.

Die Funktion „aehnlichkeits.mass“

Beschreibung: Es werden für den aktuellen Fall Constraints generiert, aktueller und alter Fall geometrisch und technologisch verglichen und anschließend Hypothesen für durchzuführende Modifikationen aufgestellt.

Funktions-Code:

```
(defun aehnlichkeits.mass (aktueller_fall alter_fall)
  (...)
(generiere.constraints 'aktueller_fall)
  (...)
(entsprechung_werkstuecke
 (werkstueck.unit 'aktueller_fall)
 (werkstueck.unit 'alter_fall))
(entsprechung_bbs
 (alle_bearbeitungs-bereiche.unit 'aktueller_fall)
 (alle_bearbeitungs-bereiche.unit 'alter_fall))
(forward.chain '(fertigungs-orientiert aehnlichkeit))
(forward.chain '(modifikationen modifikationen))
(generiere.modifikationen 'aktueller_fall 'alter_fall)
  (...)
```

Dokumentation der Funktion: Die Funktion `generiere.constraints` stößt die Generierung von Fertigungs-Constraints an. `entsprechung_werkstuecke` und `entsprechung_bbs` generieren als geometrischer Vergleich der beiden Werkstücke Aussagen über die Entsprechung von Formelementen aller Klassen: Werkstückteile, Materialelemente, Bearbeitungsmoleküle, -atome und -bereiche. Für den technologischen Vergleich wird die Regelmenge `fertigungs-orientiert` der Knowledge Base `modifikationen` durch Aufruf der *KEE*-Funktion `forward.chain` aktiviert. Schließlich bewirken die Aktivierung der Regelmenge `modifikationen` und Aufruf der Funktion `generiere.modifikationen` die Generierung von Hypothesen für durchzuführende Modifikationen.

Literaturverzeichnis

[Allen 83] J. F. Allen. *Maintaining knowledge about temporal intervals.* CACM 26:11. 1983.

[Alterman 86] R. Alterman. *An adaptive planner* Proceedings of AAAI-86, pp. 65-69, American Association for Artificial Intelligence, Morgan Kaufmann Publishers, Inc. Los Altos, CA, 1986.

[Althoff & al. 90] K.-D. Althoff, F. Maurer, M. M. Richter, Ch. Schulz, P. Spieker. *Lernen und Analogie in technischen Expertensystemen.* Projektantrag 1991-1993 Sonderforschungsbereich 314 Künstliche Intelligenz – Wissensbasierte Systeme, Universität Kaiserslautern 1990.

[Bareiss 89] R. Bareiss. *Exemplar-Based Knowledge Acquisition.* Academic Press, Inc. 1989.

[Bartenschlager 87] H. Bartenschlager. *Rechnerunterstützung bei der Arbeitsplanerstellung – gibt es eine universelle Lösung?* CAT 87, C 12, S. 142-144.

[Bergmann 90] R. Bergmann. *Generierung von Skelettplänen als Problem der Wissensakquisition.* Diplomarbeit an der Universität Kaiserslautern, 1990.

[Bullinger 89] H. J. Bullinger. *Künstliche Intelligenz in Konstruktion und Arbeitsplanung.* Verlag Moderne Industrie, 1989.

[DeJong 83] G. DeJong. *An approach to learning from observation.* Proceedings of the International Machine Learning Workshop. pp 171-176. Monticello, IL, June 1983, University of Illinois.

[Descotte & Latombe 81] Y. Desscotte, J. C. Latombe. *GARI: A problem solver that plans how to machine mechanical parts.* Proceedings of IJCAI - 7, Vancouver, Canada, 1981.

[Ermann & al. 80] L. D. Ermann, F. Hayes-Roth. R. Lesser, D. R. Reddy. *The Hearsay-II Speech-Understanding System: Integrating Knowledge to Resolve Uncertainty.* Computing Surveys 12(2), June 1980.

[Eversheim 89] W. Eversheim. *Organisation in der Produktionstechnik, Band 3: Arbeitsvorbereitung.* Verlag des Vereins Deutscher Ingenieure (VDI), Düsseldorf 1989.

[Eversheim & Fuchs 79] W. Eversheim, H. Fuchs. *Automatische Arbeitsplanerstellung – Anwendung des Systems AUTAP für allgemeine Rotationsteile.* Industrie-Anzeiger Bd. 101 (1979) Nr. 7, S. 21-25.

[Fikes & Nilsson 71] R. E. Fikes, N. J. Nilsson. *STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving.* Artificial Intelligence, Vol. 2, 1971.

[Fikes & al. 72] R. E. Fikes, P. Hart, N. J. Nilsson. *Learning and Executing Generalized Robot Plans.* Artificial Intelligence, Vol. 3(251) 1972.

[Gentner 83] D. Gentner. *Structure Mapping: A theoretical framework for analogy.* Cognitve Science 7(2), pp. 155-170. 1983.

[Gentner 88] D. Gentner. *Analogical Inference and Analogical Access.* Analogica, pp. 63-88. Ed. Armand Prieditis. Pitman, London 1988.

[Hammond 86a] K. J. Hammond *CHEF: A Model of Case-based Planning.* Proceedings of the 5th National Conference on Artificial Intelligence – AAAI 86, Menlo Park, CA, USA. Vol 1, pp 267-271.

[Hammond 86b] K. J. Hammond *Learning to Anticipate and Avoid Planning Problems through the Explanation of Failures.* Proceedings of the 5th National Conference on Artificial Intelligence – AAAI 86, Menlo Park, CA, USA. Vol 1, pp 256-260.

[Hammond 89] K. J. Hammond *Case-Based Planning: Viewing Planning as a Memory Task.* Academic Press, Inc. San Diego 1989.

[Hayes-Roth & Hayes-Roth 79] B. Hayes-Roth, F. Hayes Roth. *A Cognitive Model of Planning.* Cognitive Science, 3, 275 (1979).

[Heiob 81] W. Heiob. *Einsatz dialogorientierter Entscheidungstabellentechnik in der Angebots- und Auftragsbearbeitung von Unternehmen mit auftragsgebundener Produktion.* Dissertation Universität Karlsruhe, 1981.

[Heiob 88] W. Heiob. *Rechnergestützte Arbeitsplanung (CAP) mit Entscheidungstabellen.* AV 25, S. 14-18. Carl Hanser Verlag, München 1988.

[Herzberg 89] J. Herzberg. *Planen – Einführung in die Planerstellungsmethoden der Künstlichen Intelligenz.* BI-Wissenschaftsverlag, Reihe Informatik, Band 65, Mannheim, Wien, Zürich 1989.

[Humm & al. 91] B. Humm, Ch. Schulz, M. Radtke, G. Warnecke. *A System for Case-Based Process Planning.* Proceedings of the 1st CIRP Workshop of the Intelligent Manufacturing Systems (IMS), Seminars on Learning in IMS. Budapest, Hungary, 1991.

[Kedar-Cabelli 85] S. Kedar-Cabelli. *Purpose-directed analogy.* Proceedings of the Cognitive Science Conference, pp. 150-159. 1985.

[Kedar-Cabelli 88] S. Kedar-Cabelli. *Toward a Computational Model of Purpose-Directed Analogy.* Analogica, pp. 89-107. Ed. Armand Prieditis. Pitman, London. 1988.

[Kolodner 83] J. L. Kolodner. *Maintaining Organization in a Dynamic Long-Term Memory* Cognitive Science(7), pp. 243-280, 1983.

[Kolodner 87] J. L. Kolodner. *Extending Problem Solver Capabilities Through Case-Based Inference.* Proceedings of the 4th International Workshop on Machine Learning. Los Altos, CA, USA. Morgan Kaufmann 1987. pp. 167-178.

[Lebowitz 80] M. Lebowitz. *Generalization and Memory in an Integrated Understanding System.* Ph.D. thesis, Yale University, October 1980.

[Lesser & Corkill 87] V. R. Lesser, D. D. Corkill. *Distributed Problem Solving.* in: S. C. Shapiro (Ed.): Encyplopedia of Artificial Intelligence. New York et al. 1987.

[Matsushima & al. 82] K. Matsushima, N. Okada, T. Sata. *The integration of CAD and CAM by application of artificial intelligence techniques.* Annals of CIRP, 31, pp. 329-332. 1982.

[McDermott 78] D. McDermott. *Planning and Acting.* Cognitive Science Vol. 2, 1978.

[Michalski & Larson 78] R. Michalski, J. Larson. *Selection of most representative training examples and incremental addition of vli hypothesis: The underlying methodology and the description of programs esel and aq11.* Technical Report 867, Computer Science Department, University of Illinous, Urbana, 1987.

[Mitchell 83] T. Mitchell. *Learning and problem solving.* The Proceedings of the 8th International Joint Conference on Artificial Intelligence, Karlsruhe, Germany, August 1983. Computers and Thought Award Lecture.

[Praeger 90] R. Praeger. *Analogie und ihre Anwendung auf die Diagnoseproblematik.* Diplomarbeit an der Universität Kaiserslautern. 1990.

[Radtke 91] M. Radtke. *Das Integrierte Bauteilmodell BAMOF.* Interner Bericht, CCK, FBK, Universität Kaiserslautern, 1991.

[Rich 83] E. Rich. *Artificial Intelligence.* McGraw-Hill, Inc. 1983.

[Sacerdoti 74] E. D. Sacerdoti. *Planning in a Hierarchy of Abstraction Spaces.* Artificial Intelligence Vol. 5, 1974, pp. 115-135.

[Sacerdoti 75] E. D. Sacerdoti. *The Nonlinear Nature of Plans.* IJCAI 4, 1975.

[Schank 82] R. C. Schank. *Dynamic Memory: A Theory of Learning in Computers and People.* Cambridge University Press. Cambridge 1982.

[Schank & Abelson 77] R. C. Schank, R. Abelson. *Scripts, Plans, Goals and Understanding.* Erlbaum, Hillsdale, NJ, 1977.

[Schank & Leake 89] R. C. Schank, D. B. Leake. *Creativity and Learning in a Case-Based Explainer.* Artificial Intelligence Vol. 40 (1989), No. 1-3.

[Scholz 88] B. Scholz. *CIM-Schnittstellen.* R. Oldenbourg Verlag München Wien 1988.

[Schulz 88] C. Schulz. *Wissensbasiertes Modul zur Maschinen- und Spannmittelauswahl bei der Drehbearbeitung.* Diplomarbeit an der Universiät Kaiserslautern. Mai 1988.

[Schwetz 83] R. Schwetz. *Rechnerunterstützte Arbeitsplanerstellung.* wt – Zeitschrift für industrielle Fertigung. 73 (1983) 91-94. Springer Verlag 1983.

[Spur & al. 78] G. Spur, M. Anger, W. Kunzendorf, G. Stuckmann. *CAPSY – A dialogue system for computer aided manufacturing process planning.* 19. MTDR–Konferenz, Manchester (England), Sept. 1978.

[Steel 87] S. Steel. *The bread and butter of planning.* Artificial Intelligence Review (1987) 1, pp. 159-181.

[Stefik 81a] M. Stefik. *Planning with Constraints (MOLGEN: Part I).* Artificial Intelligence, Vol. 16, No. 2, pp. 111-139, 1981.

[Stefik 81b] M. Stefik. *Planning and Meta-Planning (MOLGEN: Part II).* Artificial Intelligence, Vol. 16, No. 2, pp. 141-169, 1981.

[Sussmann 75] G. J. Sussmann. *A computer model of skill acquisition.* Vol. 1 of Artificial Intelligence Series. American Elsevier, New York 1975.

[Tönshoff & Meyer 79] H. K. Tönshoff, K.-D. Meyer. *Demands and Experiences from the Introduction of a Work Planning System in Practice.* Annals of the CIRP. Band 28/1/1979.

[Vere 83] S. A. Vere. *Planning in Time: Windows and Durations for Activities and Goals.* IEEE Transactions on Pattern Analysis and Machine Intelligence. PAMI-5, 246, 1983.

[Warnecke & al. 89] G. Warnecke, P. Mertens, Ch. Schulz. *Artificial Intelligence in Computer Integrated Manufacturing – A Proposal for the Integration of CAD and CAPP.* Proceedings of the CIRP International Workshop on Computer Aided Process Planning (CAPP), pp. 185-196, Hannover/FGR 1989.

[Wilenski 80] R. Wilenski. *Meta-planning.* Technical Report M80 33, UB College of Engineering, August 1980.

[Wilenski 83] R. Wilenski. *Planning and Understanding: A Computational Approach to Human Reasoning.* Addison-Wesley Publishing Company, Reading Massachusetts, 1983.

[Winston 70] P. Winston. *Learning structural descriptions from examples.* Technical Report 231, AI Labatory, Massachusetts Institute of Technology, 1970. Reprinted in P. Wiston, Ed. *The Psychology of Computer Vision.* 1975, McGraw-Hill.

Index

ABSTRIPS, 4
Ähnlichkeit, 7, 56, 58–60
Ähnlichkeitsplanung, 35
Akquisition von Fällen, 8, 10, 21, 70
Aktion, 2
analoges Schließen, 5, 56
Änderungsplanung, 33, 35
Anticipate and Aviod, 13
Anticipate and Avoid, 10, 18
Applikationsmodelle, 40
Aquisition von Fällen, 7
Arbeitsplan, 24, 48, 58
Arbeitsplanung, 23, 35, 39–41, 51, 53
Arbeitssteuerung, 23
Arbeitsvorbereitung, 23
Arbeitsvorgang, 27, 53
Assigner, 13
Ausgangsteil, 25
Ausgangszustand, 2

Bearbeitungsatom, 46
Bearbeitungsbereich, 47, 58
Bearbeitungsmolekül, 46
Blockwelt, 2

CAPP, 35, 39, 53, 70, 73, 75
CBP, 10, 70
CBR, 10
CHEF, 10, 54, 57, 70
CIM, 39, 48
Constraint, 62, 75
Create and Debug, 18
Credit Assignment, 13

Dynamic Memory, 6, 56
Dynamik, 58

Effizienz, 58, 59
Entscheidungstabelle, 36
Erinnerung, 6, 9, 10, 56
Explanation Based Learning, 20

Fall, 10, 21, 70
fallbasierte Planung, 9, 53, 54
fallbasiertes Schließen, 5, 7
Fallbasis, 10, 11, 17, 21, 56, 58, 70
Fertigteil, 25
Fertigung, 23, 30, 48, 51
Fertigungsmittel, 30
Fertigungsproblem, 62
Formelement, 41
Formelementmodell, 40

Generalisierung, 7
generative Planung, 9, 10, 54
Generierungsprinzip, 33
Gestaltsmodell, 39

Hammond, K. J., 10
Handlungsplan, 2
hierarchische Planung, 4, 9, 10, 72
Hypothese, 65, 80

Index, 8, 58
Inductive Learning, 20
integriertes Datenmodell, 39
Investitionsplanung, 24

Kapazitätsplanung, 24
KEE, 51, 75
kognitiv adäquat, 59
Konstruktion, 23, 39–41, 48
Kosten, 27

Least Commitment Strategy, 3
Lernen, 6, 10, 11, 18, 22, 70, 72
LISP, 51

Materialdisposition, 24
Materialelement, 42
Materialplanung, 24
Memory of Failures, 17
Memory of Modifiers, 18
Memory of Repairs, 18
Methodenplanung, 24
Modifikation, 9–11, 60, 65, 67
MOLGEN, 4
Montage, 23
Montageplan, 24
MOP, 6

NC-Programm, 24, 39–41
Neuplanung, 35
nichtlineare Planung, 3, 9, 10, 72
NOAH, 3

Planung, 2, 48
PPS, 37
Produktionssteuerung, 39
Prüfplanung, 39–41
Purpose Directed Analogy, 6

Qualität, 30, 35

Relativkostenkatalog, 27
Repair, 10
Retrieval, 9–11, 22, 54, 56, 58, 72
Rohteil, 25, 58
Rohteilauswahl, 54

Sachmerkmalsleiste, 37, 57, 73
Schank, R. C., 6
Script, 6
Sollzeit, 30
STEP, 39
Storer, 11
STRIPS, 3, 18
Structure Mapping Theorie, 6

technische Auftragsabwicklung, 23, 35
Technologie, 27
Technologiemodell, 40
Terminplanung, 24
TOP, 18

Übergangselement, 47

Unstructured Fact, 76

Variante, 35, 54
Variantenplanung, 35
Vorgabezeit, 30

WAP, 39
Werkstattsteuerung, 24
Werkstück, 42
Werkstückrepräsentation, 40, 58, 63
Werkstückteil, 42
Wiederholplanung, 33, 35

Zeit, 30
Zielzustand, 2