Implementation einer musterorientierten Expertensystem-Shell und Kalküle für eine intuitionistische Modallogik als Basis des nicht-monotonen Schließens

Ulrich Koch

6. Juli 1989

Institut für
Numerische und instrumentelle Mathematik
Westfälische Wilhelms-Universität zu Münster

Einsteinstraße 62
D-4400 Münster
Tel.: +49 (251) 83-3097

ONM47@DMSWWU1A.BITNET

Inhaltsverzeichnis

Kapitel 1

Einleitung

Expertensysteme und Expertensystem-Shells sind das Resultat des Bestrebens, menschliches Problemlösen zu automatisieren. Um das Wissen menschlicher Experten zu konservieren und zu vervielfältigen, sind in den letzten Jahren eine Reihe von Expertensystemen entstanden. Viele von ihnen sind unter Rückgriff auf kommerzielle Entwicklungswerkzeuge, sogenannte Expertensystem-Shells entwickelt worden. Eine Auswahl der bekanntesten Expertensysteme und der ihnen zugrundeliegenden Konzepte wird im nächsten Abschnitt vorgestellt werden.

Der traditionelle Ansatz, Expertenwissen in einem Computer abzubilden, ist ein regelbasierter Formalismus. Aus einer Fülle von "wenn ... dann ..."-Regeln, die man aus Gesprächen mit einem oder mehreren Experten gewonnen hat, wird eine sogenannte Wissensbasis erzeugt. Eine Kontrollstruktur, die auf dieser Wissensbasis arbeitet, ist nun in der Lage, diese Regeln zu benutzen, um zu angemessenen Problemlösungen zu kommen.

Ein wesentlicher Einwand gegen diesen Ansatz ist, daß Experten oft nicht wirklich mit diesen Regeln arbeiten. Vielmehr besteht ihr Expertenwissen aus Erfahrungen und Heuristiken. Wenn Experten Probleme in ihrem Wissensgebiet lösen, erkennen sie neue Situationen als Spezialfälle von Dingen, mit denen sie bereits vertraut sind.[1] Die Beobachtung dieser menschlichen

[1] When experts solve problems in their area of expertise, they recognize new situations as instances of things with which they are already familiar [Wat86 S. 155]

Fähigkeit, in zu lösenden Problemen bekannte Muster wiederzufinden, war der Ausgangspunkt für die Entwicklung der Expertensystem-Shell *Donald*, die im folgenden detailliert beschrieben werden wird.

Statt mit Regeln arbeitet Donald mit gespeicherten Mustern. In den Eingabedaten wiedererkannte Strukturen steuern den Problemlösungsprozeß, indem sie Fragen nach detaillierteren Fakten auslösen und so das Erkennen speziellerer Muster ermöglichen. In mehreren Stufen nähert sich Donald so einer Lösung an. Dieses Vorgehen, in der Literatur häufig *successive refinement* genannt, wird in den nächsten Abschnitten noch beschrieben werden. Die Speicherung dieser Muster geschieht in Form einer *Vererbungshierarchie*.

Ein weiterer Unterschied zu herkömmlichen Expertensystemen ist, daß diese Lösung nicht in einer vorab festgelegten Menge von möglichen Lösungskandidaten "gefunden" wird, sondern aus Teilstücken, die den Stufen beim schrittweisen Annähern an die Lösung entsprechen, nach und nach zusammengesetzt wird.

Eine Besonderheit von Donald ist dabei das *nicht-monotone* Verhalten bei der Konstruktion der Lösung. Das heißt, daß auch bereits etablierte Arbeitshypothesen wieder rückgängig gemacht werden können, wenn neue Informationen das erforderlich machen.

Für den Mathematiker ergibt sich hier die Frage nach einer angemessenen formalen Repräsentation des nicht-monotonen Schließens in Gestalt eines nicht-monotonen Logikkalküls. Dem *nicht-monotonen Schließen* unter besonderer Berücksichtigung von *Modallogik* und *intuitionistischer Logik* ist deshalb die zweite Hälfte dieser Arbeit gewidmet.

Eine von Dov Gabbay vorgestellte aussagenlogische Semantik [Gab82] wird zu einem quantorenlogischen System (J4) erweitert, und es wird erstmals ein effektives Beweisverfahren in Form eines Sequenzenkalküls für diese Logik vorgestellt. Die Korrektheit und Vollständigkeit dieses Kalküls wird bewiesen.

Sequenzenkalküle sind gut geeignet für Untersuchungen über eine Logik. Zum automatischen Theorembeweisen, d.h. für die Implementation einer Logik, sind Matrixkalküle ([And81], [Bib81], [Wal87]) vorzuziehen, weil sie gewisse Redundanzen der Sequenzenkalküle (z.B. die Reihenfolgeabhängigkeit der Regelanwendungen) vermeiden.

Die grundlegenden Ideen des Matrixverfahrens, nämlich die Definition von *Pfaden* durch Formeln und Widerspruchspaaren in Pfaden, finden sich bei Dag Prawitz (*matrix reductions* [Pra60], [Pra70]).

Voraussetzung für den Einsatz eines auf J4 basierenden nicht-monotonen Schließens in Expertensystemen ist ein möglichst effektives Beweisverfahren. Analog zum Vorgehen bei anderen nicht-standard Logiken [Wal87] wird deshalb in Kapitel 6 ein Matrixkalkül für J4 definiert und in Kapitel 7 seine Korrektheit und Vollständigkeit bewiesen. Obendrein ergibt sich ein neuer Matrixkalkül für die intuitionistische Logik (J). Da anstelle der sonst üblichen auf Smullyan [Smu68] zurückgehenden *uniformen Darstellung* ([Fit83], [Wal87]) eine neue uniforme Darstellung benutzt wird, vermeidet der daraus resultierende einfachere Matrixkalkül für J den Umweg, J-Formeln zunächst in S4-Formeln zu übersetzen.

Daß die in Donald verwirklichten Ansätze in der Praxis erfolgreich sind, ist durch die Implementation des Expertensystems *Invest* ([HKC88],[Koc89]) bewiesen. Dabei handelt es sich um ein mit Donald erstelltes Expertensystem zur Wertpapieranlageberatung. Derzeit kann Invest Beratungen unter Berücksichtigung der rund 100 wichtigsten Gruppen von Wertpapieren durchführen. Dabei reicht das Spektrum von Sparbriefen über Aktien bis hin zu Optionen.

Die bei der Entwicklung von Invest beteiligten Experten hatten keine Mühe, ihr Wissen in den gespeicherten Mustern wiederzuerkennen. Auch später in der Entwicklungsphase hinzugezogene Experten waren rasch mit dem Musterformalismus vertraut und konnten konkrete Änderungsvorschläge machen. Das ist besonders wichtig für die Pflege der Wissensbasis, z.B. die Berücksichtigung neuer Typen von Wertpapieren.

Invest befindet sich zur Zeit im Testeinsatz bei einer Münsteraner Bank und hat bereits gezeigt, daß es in der Lage ist, mit menschlichen Kollegen zu konkurrieren.

Kapitel 2

Einführung in die Expertensystem-Technik

2.1 Formen der Repräsentation und Verarbeitung von Wissen

In diesem Abschnitt wird eine Übersicht über die wichtigsten Grundformen der Darstellung und Verarbeitung von Wissen durch Computer gegeben.

2.1.1 Produktionssysteme

Die häufigste Repräsentationsform für Wissen ist die *Produktionsregel*. Das ist eine Aussage der Gestalt:

Wenn *Bedingung*, dann *Aktion*.

Expertensysteme, die auf diesem Formalismus aufbauen, bestehen typischerweise aus einer großen Anzahl solcher Produktionsregeln, die ihre *Wissensbasis* bilden, einem *Regelinterpretierer*, der die Abarbeitung dieser Regeln steuert und einer *Datenbasis*, in der von vornherein bekannte und die durch Regelanwendung hergeleiteten Fakten gespeichert werden. Solche Systeme heißen *Produktionssysteme*.

Die Vorgehensweise beim Lösen eines Problems sieht so aus, daß der Regelinterpretierer durch Inspektion der Datenbasis prüft, für welche Regeln alle Vorbedingungen erfüllt sind. Ist das für mehrere Regeln der Fall, wählt er nach einer angegebenen Strategie eine von ihnen aus und bearbeitet den Aktionsteil der Regel, d.h. die dort angegebene Aktion wird ausgeführt. Eine solche Aktion kann zum Beispiel eine Veränderung der Datenbasis sein.

Dieser Ablauf aus Regelauswahl und Regelausführung läuft solange weiter, bis ein Abbruchkriterium erfüllt ist, oder keine Regel mehr vorhanden ist, deren Vorbedingungen erfüllt sind. Die Problemlösung ist dann aus der Datenbasis abzulesen.

Das folgende Beispiel aus [Alt84] illustriert die Arbeitsweise eines Produktionssystems:

1. Datenbasis:
   (a) $A(1) = 5, A(2) = 7, \ldots, A(10) = 21$
   (b) NUMA = 10, MAX = 0, MIN = 100000
   (c) VERSUCH = 1
2. Regeln:
   (a) Wenn VERSUCH > NUMA, dann schreibe (MAX, MIN) und HALT.
   (b) Wenn A(VERSUCH) > MAX, dann MAX := A(VERSUCH).
   (c) Wenn A(VERSUCH) < MIN, dann MIN := A(VERSUCH).
   (d) Wenn keine andere Regel ausführbar ist, dann VERSUCH = VERSUCH + 1.
3. Regelinterpretierer: Teste alle Regeln der Reihe nach bis für eine Regel alle Bedingungen erfüllt sind. Führe diese Regel aus. Dann gehe zurück zum Anfang der Liste und starte von neuem. Brich ab, wenn HALT signalisiert wird.

Dieses kleine Produktionssystem berechnet Maximum und Minimum der zehn in der Datenbasis gegebenen Zahlen $A(1), \ldots, A(10)$. Die zweite und dritte Regel manipulieren die Datenbasis, so daß dort am Ende die gesuchten Werte als MAX und MIN abzulesen sind.

Die in diesem Beispiel ausgeführte Strategie heißt *vorwärts Verketten*. Das bedeutet, daß die Regeln von links nach rechts bearbeitet werden: wenn der Bedingungsteil erfüllt ist, wird der Aktionsteil ausgeführt.

Abbildung 2.1 zeigt noch ein Beispiel [Wat86], in dem die Folgerung Z aus drei Regeln hergeleitet wird. $F \wedge B \rightarrow Z$ bezeichne die Regel: Wenn F und B in der Datenbasis enthalten sind, füge auch Z zur Datenbasis hinzu. Die ausführbare Regel und die Fakten, die ihren Bedingungsteil erfüllen, sind jeweils hervorgehoben.

Eine andere Strategie ist das *rückwärts Verketten*. Sie soll mit den gleichen Regeln und der gleichen Datenbasis wie das vorherige Beispiel erläutert werden. Das Ziel ist wieder, Z herzuleiten. Statt nun alle Regeln anzuwenden, bis Z schließlich in der Datenbasis auftaucht, sucht der Regelinterpretierer eine Regel aus, die Z beweisen würde, wenn alle ihre Bedingungen erfüllt wären. Diese Regel ist in diesem Fall $F \wedge B \rightarrow Z$. Wie eine Inspektion der Datenbasis ergibt, fehlt aber die Voraussetzung F. Der Regelinterpretierer setzt sich deshalb das neue Ziel, F zu beweisen, das heißt er sucht eine Regel, mit deren Hilfe F zur Datenbasis hinzugefügt würde. Er findet $C \wedge D \rightarrow F$. Also sind F und Z bewiesen, wenn es gelingt, D in die Datenbasis zu bekommen. Das besorgt gerade die Regel $A \rightarrow D$, deren Bedingung erfüllt ist, weil A schon bekannt ist. Damit ist Z bewiesen.

Vorwärts Verketten ist mit weniger Aufwand zu implementieren und liefert in vielen Fällen angemessene Resultate. Wenn aber die Bedingungsteile vieler Regeln erfüllt sind, wird unter Umständen eine große Anzahl von Fakten hergeleitet, die nichts zur Lösung beitragen. In diesem Fall ist das rückwärts Verketten effektiver, weil nur Regeln bearbeitet werden, die für die Lösung relevant sein könnten.

Regeln, wie in den oben gegebenen Beispielen, repräsentieren eher *prozedurales Wissen ("Wissen wie")*. Demgegenüber steht das *deklarative Wissen*, das *"Wissen was"*. Dazu gehört das Wissen über Objekte, ihre Eigenschaften und ihre Beziehungen zueinander. Zur Darstellung solcher Zusammenhänge werden häufig *semantische Netze* benutzt. Beide Sichtweisen sind prinzipiell äquivalent [Win75], aber je nach Anwendung ist die eine oder andere die

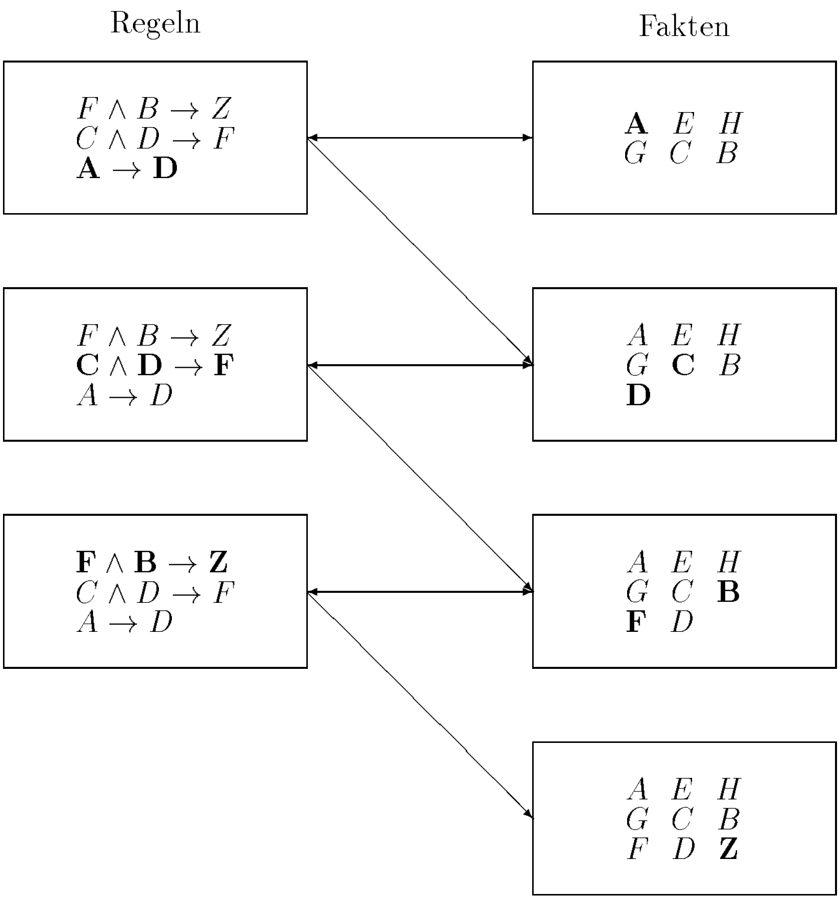


Abbildung 2.1: Vorwärts verketten

angemessenere.

2.1.2 Semantische Netze

Seit Mitte der sechziger Jahre haben sich semantische Netze als Hilfsmittel in der Psychologie (zum Modellieren menschlichen Erinnerungsvermögens) und in der Informatik (zum effizienten Speichern von Fakten für wissensbasierte Systeme) etabliert.

Semantische Netze bestehen aus Knoten, die für Objekte oder Konzepte stehen, und aus Kanten, die diese Knoten verbinden und die Relationen repräsentieren, in denen diese Knoten zueinander stehen. Zum Beispiel könnte ein semantisches Netz für den Satz "Produktionssyteme besitzen einen Regelinterpretierer" wie in Abbildung 2.2 aussehen.

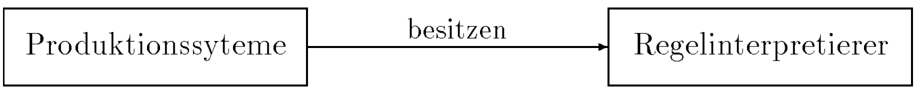


Abbildung 2.2: Ein einfaches semantisches Netz

Die Kanten eines semantischen Netzes sind gerichtet, um die gewünschte Subjekt-Objekt-Relation zwischen den Knoten darzustellen.

Quillian schlug 1966 ein semantisches Netzwerk zur Repräsentation der Bedeutung englischer Wörter vor. Dieses Netzwerk enthält Wörter als Knoten. Die gerichteten Kanten, die von einem Knoten ausgehen, zeigen auf andere Knoten, die zusammen eine Definition für das Wort, mit dem begonnen wurde, liefern. Darin spiegelt sich die Organisation eines Wörterbuches wider, das Begriffe durch Verweise auf andere Wörter, die an anderer Stelle im selben Wörterbuch stehen, definiert. Es war Quillians Auffassung, daß die "volle Bedeutung" eines Wortes alle Knoten umfasse, die von diesem Wort ausgehend über Kanten erreicht werden können ([Qui66], [BL85 S. 101]):

> ... a word's full concept is defined in the model memory to be

> all nodes that can be reached by an exhaustive tracing process, originating at its initial, patriarchical type node ...

Die objektive Bedeutung eines Wortes, das heißt seine Definition zusammen mit den Assoziationen, die es in unterschiedlichen Zusammenhängen bei einem Menschen auslöst, könnte durch seine Repräsentation in einem semantischen Netz dargestellt werden. Für Details siehe [Bra79,Qui66].

Das Interesse der Informatiker an semantischen Netzen ist zum Teil durch die eher technischen Aspekte effizienter Wissensspeicherung begründet. Dazu gehört die Möglichkeit des Aufbaus von Vererbungshierarchien durch *is-a-Kanten* (*ist-ein-Kanten*). Weitere Möglichkeiten sind *Demons* (angebundene Prozeduren), *defaults* (angenommene Standardsetzungen) und *Perspektiven*, die in einem der folgenden Abschnitte im Zusammenhang mit Frames erläutert werden werden.

Ist-ein-Kanten stellen in einem Netz eine partielle Ordnung her, indem sie eine Teilmengen- bzw. Elementrelation zwischen den Knoten darstellen. Das Beispiel in Abbildung 2.3 zeigt ein semantisches Netzwerk für das Konzept "Computer".

Dieses Netz enthält unter anderem die Aussagen: "Jeder Großrechner ist ein Computer" und: "Der Zentralrechner-1 ist ein Großrechner". Da die ist-ein-Relation, genau wie die hat-Teil-Relation, transitiv ist, ergibt sich, ohne daß es explizit angegeben wäre, daß der "Zentralrechner-1" ein Computer ist.

Die ist-ein-Relation und die hat-Teil-Relation erzeugen ein Vererbungsnetzwerk, in dem Knoten, die niedriger in der Hierarchie stehen, Eigenschaften von einem höher stehenden Knoten erben können. So muß eine Information, die auf alle Mitglieder einer Klasse zutrifft, nicht für jedes Mitglied wiederholt werden, sondern wird nur einmal, eben für diese Klasse gespeichert.

So kann aus "Ein Computer hat als Bestandteil ein Memory Board", "Ein Memory Board hat als Bestandteil Speicherchips" und "Der Zentralrechner-1 ist ein Computer" gefolgert werden, daß der "Zentralrechner-1" Speicherchips als Bestandteile hat.

Auch wenn in diesem sehr einfachen Beispiel nur ist-ein- und hat-Teil-Kanten auftauchen, so ist doch jede zweistellige Relation als Kante in einem semantischen Netz denkbar. Winston [Win84] benutzt zum Beispiel eine *ako-Kante*.

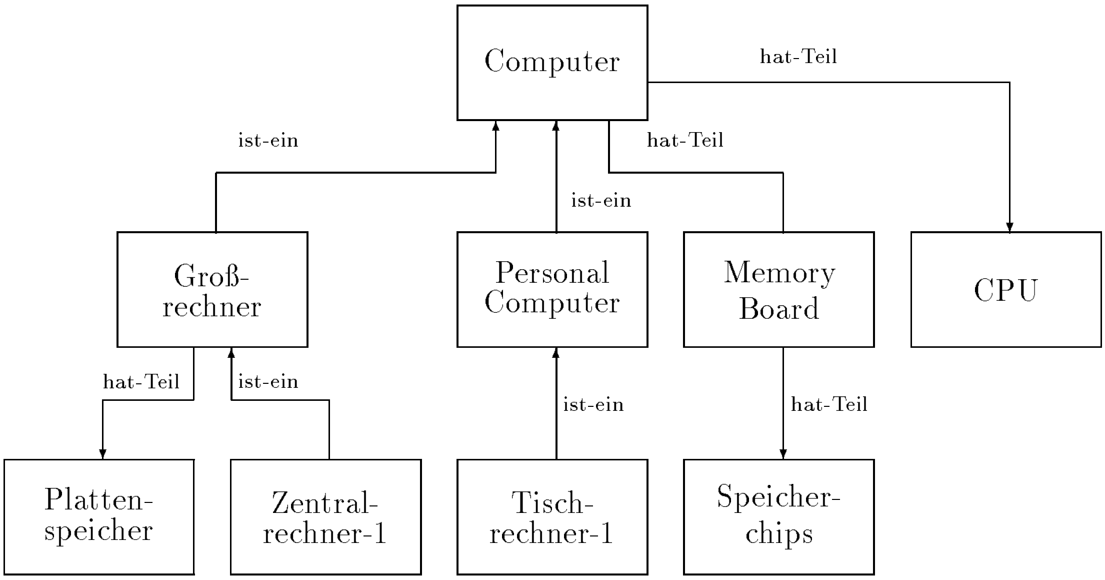


Abbildung 2.3: Ein semantisches Netz für *Computer*

"Ako" steht für "a kind of" ("eine Art von") und dient dazu, die Teilmengenrelation von der Elementrelation ("is-a") zu unterscheiden.

Semantische Netze sind besonders geeignet, um Wissensgebiete abzubilden, die ohnehin schon über eine gut gegliederte Hierarchie verfügen, wie zum Beispiel die Taxonomie biologischer Arten. Schwierigkeiten macht stets die Behandlung von Ausnahmen.

Eine der wichtigsten Anwendungen für semantische Netze ist die Analyse *natürlicher Sprache* (im Gegensatz zu einer Kunstsprache, etwa einer Computersprache). Ansätze in dieser Richtung versuchen typische Zusammenhänge durch ein Netz darzustellen, in dem die Knoten noch nicht mit Wörtern gefüllt sind, sondern Rollen darstellen, die von bestimmten Objekten oder Personen übernommen werden können.

Als Beispiel zeigt Abbildung 2.4 wie das Verb *tragen* durch ein semantisches Netz repräsentiert wird.

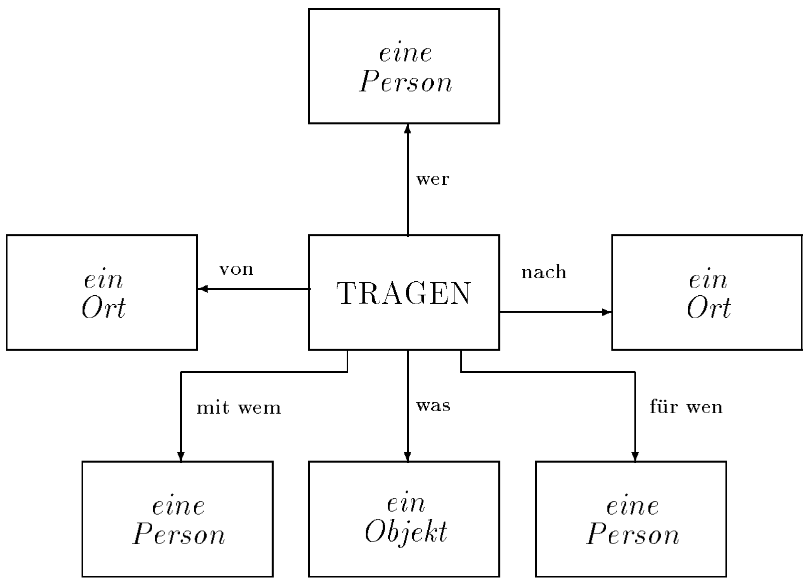


Abbildung 2.4: Ein semantisches Netz für das Wort *tragen*

Der Satz "Albert trägt mit Bernd für Constanze die Einkaufstasche vom

Markt nach Hause" wird durch ein Netz dargestellt, das aus dem in Abbildung 2.4 gezeigten Prototyp durch Einsetzen der entsprechenden Wörter entsteht. Eine Verallgemeinerung dieses Ansatzes besteht darin, das Verb *tragen* als Spezialfall eines allgemeinen Konzeptes von *agieren* zu verstehen.

So kommt man zu Prototyp-Netzen, in denen auch der Knoten, der das Verb darstellt, noch leer ist und durch ein Verb bestimmten Typs gefüllt werden kann. Schank und Rieger [SR74] identifizierten zwölf Prototypen für solche allgemeinen Aktionen, z.B.:

1. ATRANS: Der Transfer einer abstrakten Relation, wie Besitz oder Kontrolle
2. PTRANS: Der Transfer des physikalischen Ortes eines Objektes
3. PROPEL: Die Anwendung physikalischer Kraft auf ein Objekt

Zu jedem dieser Prototypen gehört das Gerüst eines semantischen Netzes, in dem Kanten für konzeptionelle Abhängigkeiten unter den Knoten stehen. Für weitere Details siehe [SR74] oder [Ric83].

2.1.3 Frames

Die enorme Leistungsfähigkeit und Geschwindigkeit menschlicher Denkprozesse waren für Marvin Minsky der Anlaß, daran zu zweifeln, daß menschliches Wissen eine Anhäufung vieler einzelner "Wissensbruchstücke" sein könnte. Er vertrat die Meinung, daß die Bausteine des Denkens, der Sprache, der Erinnerung und der Wahrnehmung größere stark strukturierte Einheiten sein müßten, deren faktischer und prozeduraler Inhalt stärker verknüpft sein muß, um die offensichtliche Stärke und Geschwindigkeit mentaler Aktivitäten zu erklären.[1]

Minskys Theorie soll hier lediglich als Motivation der Wissensrepräsentation in Donald dienen. Deshalb wird sie hier nur kurz vorgestellt (für eine ausführliche Darstellung siehe [Min75]):

[1] The "chunks" of reasoning, language, memory, and perception ought to be larger and more structured; their factual and procedural contents must be more intimately connected in order to explain the apparent power and speed of mental activities. [Min75 S. 246]

Wird man mit einer Situation konfrontiert, wählt man aus der Erinnerung eine geeignete Struktur, die Minsky *Frame* nennt. Das ist eine erinnerte prototypische Beschreibung, die je nach Notwendigkeit den Gegebenheiten der Realität angepaßt werden kann. Ein solcher Frame ist eine Datenstruktur, die eine Situation repräsentiert, wie am Schreibtisch zu sitzen oder am Strand zu liegen. Ein solcher Frame enthält Informationen unterschiedlicher Qualität. Da sind Informationen darüber, was in der angenommenen Situation stets zutrifft, was manchmal richtig ist und jetzt durch die Wahrnehmung geprüft werden sollte, was als nächstes passieren könnte, oder darüber, was in dieser Lage zu tun ist.

Normalerweise wird die vorgefundene Situation nicht nur einen Frame betreffen. Befindet man sich am Strand, so wird sicherlich der "Ich bin am Strand"-Frame erinnert, der Informationen enthält wie "der Boden besteht vermutlich aus hellem Sand", "ich befinde mich dicht am Wasser" oder "An Schutz gegen die Sonne denken!". Gleichzeitig werden auch Prototypen wie der "Ich bin draußen"- und der "Ich habe Freizeit"-Frame angesprochen, die ihrerseits andere Assoziationen, Erwartungen und Aktionen initiieren.

Bobrow und Winograd stellten mehrere Thesen zur effizienten Wissensrepräsentation auf, in denen Minskys Ideen unschwer wiederzuerkennen sind [BW77]:

- Wissen sollte rund um konzeptionelle Einheiten mit angebundenen Beschreibungen und Prozeduren organisiert werden.
- Eine Beschreibung muß in der Lage sein, unvollständiges Wissen über eine Einheit zu repräsentieren, und sie muß Platz für mehrere Deskriptoren bieten, die diese Einheit aus verschiedenen Perspektiven beschreiben.
- Eine wichtige Methode des Beschreibens ist der Vergleich mit einer bekannten Einheit und die weitere Spezifikation des zu beschreibenden Beispiels bezüglich des Prototyps.

In Übereinstimmung mit Minskys Theorie fordern Bobrow und Winograd, daß Schlußfolgern ein Prozeß des Wiedererkennens sein muß, in dem neue Objekte und Ereignisse mit gespeicherten Prototypen verglichen werden.

Ein Beispiel für die Umsetzung des Frame-Ansatzes in die Praxis ist FRL (*Frame Representation Language*) [RG77a], [RG77b]. FRL ist in Lisp (*LISt Processing*) [WH84],[Ste84],[Wil86] implementiert. Frames werden in FRL als geschachtelte Listen dargestellt, wie zum Beispiel der Frame in Abbildung 2.5, der den Studenten *Klaus* beschreibt.

```
(KLAUS (AKO (VALUE (STUDENT)))
       (JAHRGANG (VALUE (65)))
       (GRÖSSE (VALUE (180)))
       (HAARFARBE (VALUE (BLOND))))
```

Abbildung 2.5: Der Frame KLAUS

Die mit AKO, JAHRGANG, GRÖSSE und HAARFARBE beginnenden Listen heißen die *Slots* des Frames KLAUS. Die mit VALUE beginnenden Listen heißen *Facetten* des Slots, in dem sie stehen. In diesem Beispiel tauchen nur VALUE-Facetten auf. In FRL gibt es Funktionen (*fget, fput*), um die Inhalte der Slots zu erfragen oder zu verändern. Zum Beispiel würde der Befehl (FGET 'KLAUS 'JAHRGANG 'VALUE) die Antwort 65 ergeben.

Der AKO-Slot (*A-Kind-Of*) besagt, wie bei den semantischen Netzen, daß der Frame KLAUS eine Spezialisierung des allgemeineren Frames STUDENT ist. Durch die Anordnung von Frames in einer Hierarchie gilt schon gespeichertes Wissen über allgemeinere Objekte auch für die spezielleren, wenn es nicht ausdrücklich durch andere Werte überschrieben wird. Die Abbildung 2.6 zeigt eine solche Hierarchie von Frames.

FRL sieht, wie die meisten Frame-basierten Systeme, zwei Methoden vor, um unvollständiges Wissen zu bearbeiten. Das sind zum einen an Slots angebundene Prozeduren, sogenannte *Demons*, und zum anderen Standardwerte, sogenannte *Defaults*, die angenommen werden, wenn keine konkreteren Informationen vorhanden sind.

Demons sind Befehlfolgen, deren Abarbeitung durch einen Lese- oder Schreibzugriff auf den zugehörigen Slot eines Frames ausgelöst werden. Ein typischer Fall ist der Versuch den Inhalt eines Slots zu lesen, für den noch kein Wert ermittelt wurde. Ein mit diesem Slot assoziierter Demon, häufig *if-needed-*

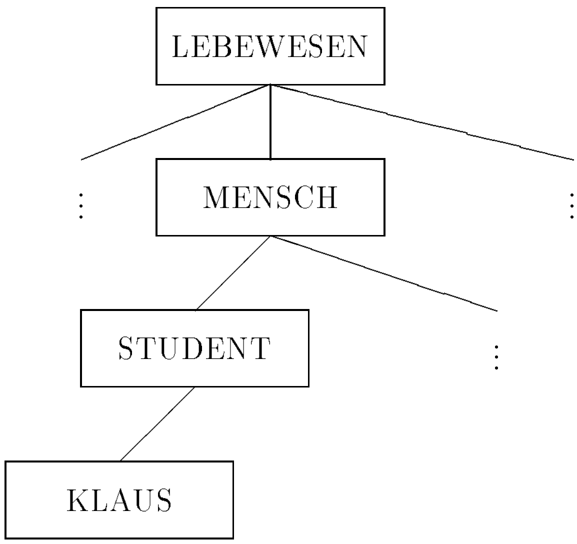


Abbildung 2.6: Eine Hierarchie von Frames

demon genannt, besteht dann aus einer Prozedur, die den gesuchten Wert aus anderen schon bekannten Werten errechnet oder, wenn das nicht möglich ist, den Programmbenutzer nach dem fehlenden Wert fragt.

Eine andere Möglichkeit ist, daß ein Demon durch das Eintragen eines Wertes in einem Slot ausgelöst wird. Ein solcher Demon könnte diesen (oder einen modifizierten) Wert an andere Slots im selben oder in anderen Frames weitergeben. Dadurch können andere Demons in anderen Slots ausgelöst werden.

Bei Expertensystemen, die einen Dialog mit dem Benutzer führen, wie zum Beispiel Invest, helfen die Demons, Fragen nach Werten, die sich aus schon Bekanntem schließen lassen, zu vermeiden.

Durch das Mittel der prozeduralen Anbindung geben Frame-basierte Systeme dem Expertensystementwickler die Möglichkeit sowohl zur prozeduralen als auch zur deklarativen Wissensrepräsentation.

Ein Beispiel: Die Anfrage (FGET 'KLAUS 'ALTER 'VALUE) nach dem Wert des Slots ALTER im Frame KLAUS findet keinen Wert im angegebenen Frame. Im Frame MENSCH (Abbildung 2.7), mit dem der Frame KLAUS über die AKO-Relation verbunden ist, existiert ein Slot ALTER, der einen Demon zur Berechnung des Alters enthält.

```
(MENSCH (AKO (VALUE (LEBEWESEN)))
            ...
            (ALTER (IF-NEEDED (Subtrahiere die Jahrgangsangabe
                               von der aktuellen Jahreszahl)))
            ...
            (GEWICHT (DEFAULT (70))))
```

Abbildung 2.7: Der Frame MENSCH

Eine Anfrage der Form (FGET-AS-NEEDED 'KLAUS 'ALTER) würde die ako-Hierarchie nach IF-NEEDED-Facetten durchsuchen und so das Resultat 24 liefern.

Die Anfrage (FGET 'KLAUS 'GEWICHT 'VALUE) würde auf gleichem Weg, da in den Frames KLAUS und STUDENT kein Slot GEWICHT auftaucht, den Slot GEWICHT im Frame MENSCH finden. Da auch dort keine VALUE-Facette vorhanden ist, würde FGET nach einer DEFAULT-Facette suchen und so den in der Abwesenheit besserer Informationen gültigen Wert 70 liefern. Zu einem späteren Zeitpunkt, wenn mehr über KLAUS bekannt ist, kann es sein, daß die VALUE-Facette des Slots GEWICHT mit einem konkreten Wert (z.B. 72) gefüllt ist. Dieselbe Anfrage, (FGET 'KLAUS 'GEWICHT 'VALUE), wird jetzt einen anderen Wert liefern. Durch das Anwachsen des Wissens über KLAUS werden zuvor gezogene Schlüsse ungültig. Diese Nicht-Monotonie, die im folgenden noch mehrfach Gegenstand mathematischer Betrachtung sein wird, ist das Haupthindernis bei der formalen Behandlung von Frame-Hierarchien, da sie im Gegensatz zur gewohnten Monotonie der klassischen Logik steht.

FRL ist ein gutes Beispiel, um sich mit der Arbeitsweise von Systemen, die auf Frames basieren, vertraut zu machen. Aber FRL unterstützt Minskys Forderung nach der Beschreibbarkeit eines Objektes aus verschiedenen Perspektiven nur unzureichend. Zwar kann man im AKO-Slot mehrere Frames als Vorgänger angeben und so zu einer Heterarchie von Frames kommen, aber ohne eine Namenskonvention ist dann nicht mehr erkennbar, welche der anderen Slots welcher Perspektive zuzuordnen sind, bzw. welche Slots neu, das heißt spezifisch für diesen Frame sind.

In Donald verfügen Frames über *Views*. Diese Views sind Datenstrukturen, die die Perspektive, aus der der Frame beschrieben wird, zusammen mit den für die Beschreibung des Frames nötigen Setzungen der Slots enthalten.

Das Zusammenfassen der Beschreibungen in Views ermöglicht die Behandlung von ganzen Gruppen von Slots als eine Einheit. Dadurch wird zum einen die Notation der Frames leichter lesbar, und zum anderen ergeben sich Vorteile beim Vergleichen von Frames und bei der Vererbung von Informationen an nachfolgende Frames.

Das aus Invest stammende Beispiel in Abbildung 2.8 soll als Illustration für die Realisierung des Frame-Konzeptes in Donald dienen. Es zeigt, daß Frames typischerweise in Hierarchien (oder auch in Heterarchien) angeordnet sind.

Innerhalb einer solchen Hierarchie geben weiter oben stehende Frames, in Donald *Classes* bzw. *SubClasses* genannt, die in ihnen gespeicherten Informationen automatisch an die tiefer stehenden Frames (zumeist *Instances*) weiter. So sind Laufzeit und Verzinsungsmodus für das Wertpapier *BSB-A-1985/11* erkennbar, ohne explizit für dieses Papier angegeben worden zu sein, weil sie im Frame BSB-A beschrieben sind.

Dieses Beispiel wird mit den entsprechenden Details im Abschnitt über Frames in Donald wiederaufgegriffen werden.

Mehr über die Vererbung von Informationen folgt in einem späteren dieses Thema behandelnden Abschnitt.

Die Kanten zwischen den Knoten in Abbildung 2.8 entsprechen den verschiedenen Perspektiven, aus denen ein Objekt beschrieben wird. Der Knoten *BSB-A-1985/11* wird durch seine Ähnlichkeiten mit den allgemeineren Konzepten *Bundesschatzbrief-A, Bundesschatzbrief* und *KursneutralesAnlagepapier* charakterisiert.

Die Darstellung von Objekten als Frames ist ein von vielen Expertensystemen benutzter Formalismus. Aber in allen dem Autor bekannten Fällen wird dieser Ansatz nur zur effektiven Speicherung von Daten eingesetzt. Die Hierarchie der Frames wird als Datenbank benutzt, die während des Pro-

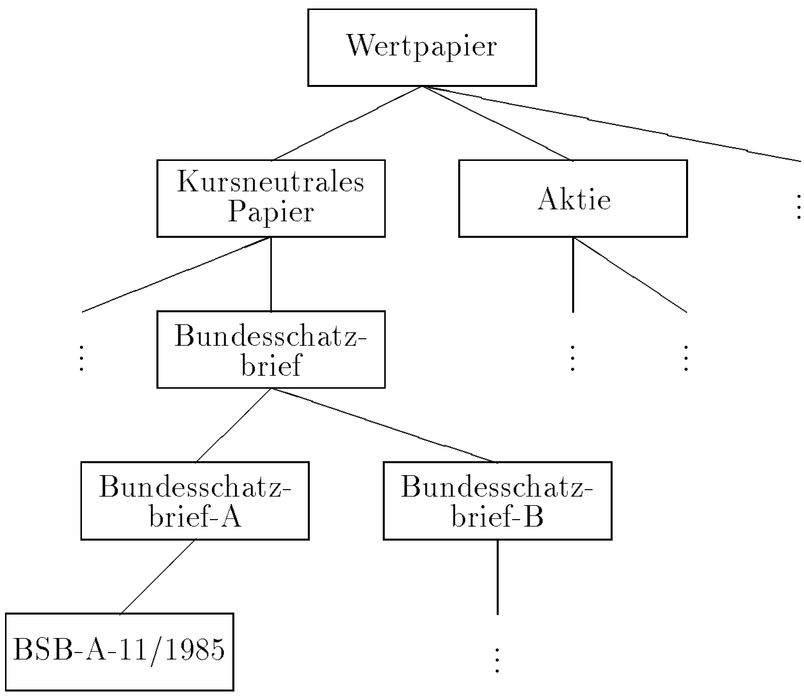


Abbildung 2.8: Eine Hierarchie von Wertpapieren

blemlösungsprozesses durch Produktionsregeln manipuliert wird.

Die Expertensystem-Shell Donald ist so konzipiert, daß sie fast völlig ohne Produktionsregeln auskommt. Die ganze Wissensbasis besteht aus Frames, die stark untereinander vernetzt sind. Auf diesen Frames arbeitet eine Kontrollstruktur, die in Analogie zu Minskys Ansatz dafür sorgt, daß

1. durch das Auftreten einer neuen Situation ein oder mehrere Frames erinnert werden,
2. noch bestehende Lücken, das heißt im erinnerten Frame noch offengelassene Stellen, gefüllt werden und
3. an den erinnerten Frame angebundene Prozeduren, d.h. Wissen darüber, was in dieser Situation zu tun ist, ausgeführt werden.

Damit soll nicht gesagt sein, daß Donald "denkt". Die in Donald angewandten Techniken sind lediglich das Resultat des Versuchs, die Vorbilder, von denen der Autor annimmt, daß sie wesentlich menschliches Denken bestimmen, nachzubilden.

Die Idee, die zur Auswahl dieses regelfreien Ansatzes führte, war, daß die Leistung eines Expertensystems der der Experten um so näher kommt, je ähnlicher die Problemlösungsstrategie des Computers der der Experten ist.

2.1.4 Logisches Programmieren

Unter *Logischem Programmieren* versteht man, die Prädikatenlogik der ersten Stufe als Programmiersprache zu benutzen. Die Grundlagen dazu sind entstanden durch das Bemühen, das Beweisen von Theoremen zu automatisieren. Eine solche Methode ist das Robinsonsche *Resolutions-Prinzip* [Rob65], das die Grundlage der Programmiersprache PROLOG (*PROgramming in LOGic*) bildet [CM81]. Die mathematischen Grundlagen des Resolutionsverfahrens werden hier ausgelassen, weil es in Donald keine Anwendung findet und weil es für die Bearbeitung der nicht-standard Logiken im zweiten Teil der Arbeit nicht einsetzbar ist. Eine ausführliche Darstellung findet sich z.B. in [Gal86], eine sehr anschauliche in [Tan87].

Der Einsatz der Prädikatenlogik zur Repräsentation von Wissen für Expertensysteme bringt Vor- und Nachteile mit sich. Das in der Datenbasis gespeicherte Wissen sollte unabhängig vom Expertensystem verständlich sein, um bei Bedarf leicht modifizierbar und erweiterbar zu sein. Dazu muß es eine klar definierte Bedeutung, eine formale Semantik, haben. Die Befürworter der Logischen Programmierung argumentieren, daß die einzig verfügbare Semantik die modell-theoretische ist, und daß die Prädikatenlogik der einzige vorhandene Formalismus ist, der eine solche Semantik besitzt [Rei87]. Dagegen ist einzuwenden (siehe z.B. [McD87]), daß Deduktion nicht die einzige Möglichkeit ist, zu Schlußfolgerungen zu kommen. Im Abschnitt 2.2.2 wird zum Beispiel *Abduktion* als andere, nicht deduktive Schlußweise erläutert werden. Weiterhin ist das *nicht-monotone Schließen* zu nennen, das ein ganz alltägliches Phänomen ist (sogenanntes *commonsense reasoning*), aber durch die Prädikatenlogik nicht effektiv repräsentiert werden kann. Ähnlich ist es mit *Vererbunghierarchien* (siehe Abschnitt 2.1.3), die auf einfache Weise nicht-monotone Schlüsse zulassen und auch über eine, wenn auch wenig formale, so doch um so einleuchtendere Semantik verfügen. Touretzky [Tou86 S. 15] schreibt dazu:

> Unfortunately, the nonmonotonic nature of inheritance when exceptions are present precludes any simple axiomatization of inheritance reasoning in first order logic.[2]

Patrick Hayes beschreibt in [Hay79], wie eine Frame-Repräsentation einer Vererbungshierarchie ohne Ausnahmen in der Prädikatenlogik nachgebildet werden kann, und schließt daraus, daß der Frame-Ansatz als Formalismus zur Darstellung von Wissen nichts beigesteuert hat, das über die Ausdrucksmöglichkeiten der Logik hinausgeht [Hay79 S.294]:

> [...] no new process of reasoning, no advance in expressive power.

Alles andere wäre auch eine Überraschung gewesen. Die deskriptive Universalität der klassischen Prädikatenlogik ist wohlbekannt: alles, was in irgendeiner formalen Sprache ausgedrückt werden kann, kann in irgendeiner

[2]Unglücklicherweise schließt die nicht-monotone Natur des Vererbens, wenn es Ausnahmen gibt, jede einfache Axiomatisierung des Schlußfolgerns mit Vererbungshierarchien in der Logik der ersten Sufe von vornherein aus.

Weise in der Prädikatenlogik ausgedrückt werden. Für die Behandlung von Frame-Hierarchien mit Defaults (zur Behandlung von Ausnahmen) schlägt Hayes eine Formalisierung vor, in der man sich *irgendwie* auf den derzeitigen Kenntnisstand ("NOW") und auf einen Begriff von *Herleitbarkeit* ("$\vdash$") beziehen kann. [3] Das Problem dabei ist jedoch, daß eine solche Darstellung in der klassischen Prädikatenlogik sehr unübersichtlich und ineffizient zu werden droht; zumindest ist bislang keine einfache Axiomatisierung bekannt. Hayes Darstellung eines Defaults als

$$(\exists y.\mathrm{NOW}\ \vdash \lceil A(y) \rceil) \vee A(b)$$

greift explizit zurück darauf, was mit dem derzeitigen Kenntnisstand *herleitbar* ist. Analog dazu benötigen Reiters Default-Regeln [Rei80] Informationen darüber, ob der Default-Schluß mit dem augenblicklichen Wissen *konsistent* ist. Für interessante Theorien (z.B. die Prädikatenlogik der ersten Stufe) sind diese Begriffe aber nicht *entscheidbar*, sondern bestenfalls, wie bei der Prädikatenlogik *semi-entscheidbar*. Im allgemeinen kann man also gar nicht wissen, ob ein Default-Schluß angewendet werden darf oder nicht. David Israel kommentiert das so [Isr80 S.55]:

> [...] the proofs, not just the theorems, are not recursively enumerable. Why such a beast is to be called a logic is somewhat beyond me.[4]

Im allgemeinen ist nicht-monotone Logik also nicht semi-entscheidbar. Wenn effektive Implementationen während des Versuchs, die Konsistenz eines Defaults zu zeigen, nicht für immer in Schleifen laufen sollen, müssen sie zwangsläufig manchmal Fehler machen. Das macht es erforderlich, die Datenbasis häufiger auf Widersprüche hin zu überprüfen. Dazu könnte ein *Truth Maintenance System* (Wahrheitserhaltungssystem) eingesetzt werden [Doy79].

[3] Let us suppose we can somehow refer to the current knowledge-state (denoted by NOW), and to a notion of derivability (denoted by the turnstile $\vdash$). Then we can express the default assumption by:

$$\exists y.\mathrm{NOW}\ \vdash \lceil \mathit{homeport}(\text{traveller}) = \text{y} \rceil \vee \mathit{homeport}(\text{traveller}) = \text{Palo Alto}.$$

[Hay79 S. 292]

[4] die Beweise, nicht nur die Theoreme, sind nicht rekursiv aufzählbar. Warum so ein Ungeheuer Logik genannt werden sollte, ist mir unbegreiflich.

Mit der Möglichkeit andere Logiken als die klassische zur Formulierung des nicht-monotonen Schließens einzusetzen, wird sich die zweite Hälfte dieser Arbeit beschäftigen.

2.2 Beispiele für Expertensysteme

In diesem Abschnitt folgt die Beschreibung einiger Expertensysteme und Shells, die entweder von besonderer Bedeutung für die Geschichte dieser Systeme sind, mit ähnlichen Techniken arbeiten wie Donald oder zu ähnlichen Wissensgebieten wie Donald und Invest erstellt wurden.

2.2.1 MYCIN

MYCIN ist ein Expertensystem zur Diagnose und Therapie bakterieller Infektionen, d.h. zur Identifikation der verursachenden Organismen und zur Auswahl einer Kombination von Wirkstoffen zur Behandlung der Infektion. MYCIN wurde zu Beginn der siebziger Jahre an der Stanford University entwickelt. Es ist sicherlich das bekannteste und meist beschriebene Expertensystem. Hier genügt deshalb eine kurze Betrachtung der Funktionsweise. Weitere Abhandlungen über MYCIN finden sich z.B. in [Sho76],[AC84],[Jac86] und [Ric83].

MYCIN ist ein regelbasiertes System, das die Regeln in der Wissensbasis zumeist rückwärts-verkettend auswertet. Die Regeln sind in Gruppen zusammengefaßt, die jeweils zu einem *Kontext* gehören. PATRULES, CULRULES und ORGRULES sind z.B. die Gruppen von Regeln, die Patienten, Kulturen bzw. Organismen betreffen. Durch die Kontextbildung wird gewährleistet, daß nicht zu viele Regeln auf Anwendbarkeit getestet werden müssen. Die Kontexte sind hierarchisch angeordnet, so daß sie in sinnvoller Weise aufeinander folgen.

Einer der wesentlichen Aspekte von MYCIN ist die Behandlung von unsicherem Wissen. Das sind Aussagen, die nicht immer zutreffen, sondern Ausnahmen zulassen, oder unter Umständen sogar nur gelegentlich zutreffen. MYCIN benutzt dazu die sogenannten *Certainty Factors*. Abbildung 2.9 zeigt eine typische Regel aus der MYCIN-Wissensbasis:

```
PREMISE: ($AND (SAME CNTXT GRAM GRAMNEG)
               (SAME CNTXT MORPH ROD)
               (SAME CNTXT AIR AEROBIC))
ACTION:  (CONCLUDE CNTXT CLASS ENTEROBACTERIACEAE
               TALLY .8)
```

Abbildung 2.9: Eine typische MYCIN-Regel

Die Symbole CNTXT und TALLY bezeichnen Variablen, die erst kurz vor, bzw. während der Behandlung der Regel einen Wert zugewiesen bekommen. CNTXT könnte z.B. den Wert ORGANISM-1 erhalten, wenn der derzeitige Kontext darin besteht, die Identität des ersten Organismus zu bestimmen. Benutzt werden dazu die Regeln aus der Klasse ORGRULES, für die die angegebene Regel ein Beispiel ist. Die Prämisse der Regel besteht aus der Konjunktion ($AND) dreier Bedingungen. Das Symbol SAME bezeichnet ein Prädikat, das auf Gleichheit prüft. Angenommen über ORGANISM-1 ist folgendes bereits bekannt:

```
ORGANISM-1
    GRAM   = (GRAMNEG 1.0)
    MORPH = (ROD .8) (COCCUS .2)
    AIR     = (AEROBIC .6) (FACUL .4)
```

Die Zahlen in den Klammern sind *Certainty Factors* (CF), die als Maß dafür dienen, mit welcher Sicherheit die angegebenen Werte "geglaubt" werden. Dabei bezeichnet 1.0 absolute Sicherheit, -1.0 völlige Verneinung. Um den CF der Konjunktion in der Prämisse zu bestimmen, bildet man das Minimum der Einzel-CF. (SAME ORGANISM-1 GRAM GRAMNEG) ergibt 1.0. Die beiden anderen Konjunktionsglieder ergeben .8 und .6. Der CF der Prämisse ist also .6. Dieser Wert wird der Variablen TALLY zugewiesen. Da der CF der Prämisse hoch genug ist, wird die Regel ausgeführt. Der CF der Konklusion ist das Produkt aus TALLY und dem im ACTION-Teil angegebenen Wert, in diesem Fall .8. Diese .8 gibt an, wie "überzeugt" MYCIN von der Konklusion gewesen wäre, vorausgesetzt, die Prämisse wäre mit völliger Sicherheit

(1.0) erfüllt gewesen. So schließt MYCIN, daß die Klasse des ORGANISM-1 *enterobacteriaceae* ist, mit Sicherheit .48 [Jac86].

Gibt es weitere Regeln, die zu demselben Schluß kommen, verrechnet MYCIN die verschiedenen CF zu einem neuen Wert. Zum Beispiel ist

$$CF_{NEU} = CF_1 + CF_2(1 - CF_1), \text{ falls } CF_1 > 0, CF_2 > 0.$$

Die Benutzung der Certainty Factors zur Behandlung unsicheren Wissens ist nicht unumstritten. Das liegt zum Teil daran, daß die Bewertungen der Regelkonklusionen nicht auf statistischen Daten, sondern auf subjektiven Einschätzungen der Experten beruhen. Zum anderen kann man zeigen [Ada76],[Jac86], daß die CF-Kalkulationen in einem einfachen Beispiel nicht mit der Wahrscheinlichkeitstheorie übereinstimmen. Dagegen ist zu halten, daß auf Bayes Theorem beruhende exakte wahrscheinlichkeitstheoretische Berechnungen in der Praxis wegen der fehlenden Werte für die *a priori*-Wahrscheinlichkeiten nicht durchführbar sind.

In Praxistests hat sich MYCIN als überzeugend kompetentes Expertensystem gegen menschliche Konkurrenten behaupten können. Daß es nicht routinemäßig eingesetzt wird, liegt unter anderem daran, daß das bearbeitete Wissensgebiet zu schmal ist und daß MYCIN zuviel Speicherplatz und Rechenzeit benötigt [Jac86].

2.2.2 INTERNIST

INTERNIST-I ist ein Expertensystem zur Inneren Medizin, das zu Beginn der siebziger Jahre von dem Internisten J. Myers und dem Informatiker H. Pople entwickelt worden ist. Im Unterschied zu anderen Systemen (z.B. MYCIN) ist INTERNIST der Versuch, menschliches diagnostisches Denken zu modellieren [Jac86],[AC84]. Die Strategie von INTERNIST besteht darin, zu vorgefundenen *Manifestationen*, d.h. Krankheitssymptomen, Laborbefunden u.a.m., eine oder mehrere Diagnosen zu finden, die diese Manifestationen erklären. Als Schlußfolgerungsmechanismus benutzt INTERNIST *Abduktion*. Unter Abduktion versteht man den Schluß von $\forall x(A(x) \rightarrow B(x))$ und der Beobachtung $B(a)$ auf $A(a)$. Dieser Schluß muß im Unterschied zu *Deduktions*schritten nicht korrekt sein, denn es kann auch andere "Gründe" für

$B(a)$ geben, z.B. $\forall x(C(x) \rightarrow B(x))$ und $C(a)$. Aber unter den Gründen für $B(a)$ kann man den heraussuchen, den man in einem bestimmten Sinn für den *einfachsten* Grund hält. Dieses Prinzip, einfachere Begründungen den komplizierteren vorzuziehen, heißt *Occam's Razor*. Pople selbst beschreibt Abduktion wie folgt [Jac86 S.117],[Pop73 S.147]:

> The essence of abductive inference is the generation of hypotheses, which if true, would explain some collection of observed facts.[5]

INTERNIST verfügt über eine strukturierte Darstellung der Teilgebiete der Inneren Medizin in Form eines *Krankheitsbaums*. Darin bezeichnet der Wurzelknoten die Menge aller inneren Erkrankungen, Zwischenknoten stehen für Gruppen von Erkrankungen (wie Lebererkrankungen, Lungenerkrankungen etc.). Die Blätter des Baums bezeichnen konkrete Krankheiten. Sie sind über zwei Relationen, EVOKE und MANIFEST, mit einer Liste aus allen dem Expertensystem bekannten Manifestationen verbunden. Die EVOKE-Relation verbindet eine Manifestation mit einer Krankheit. Dabei gibt ein Wert auf einer Skala von 0 bis 5 an, wie stark die Manifestation auf diese Krankheit hindeutet. Umgekehrt gibt es die MANIFEST-Relation und einen Wert auf einer Skala von 1 bis 5, der ein Maß dafür ist, wie häufig die bezeichnete Manifestation beim Vorliegen der angegebenen Krankheit auftritt.

Die Eingabe einer Liste von Manifestationen aktiviert Knoten im Krankheitsbaum. Diesen Knoten werden aufgrund der EVOKE- und MANIFEST-Relation numerische Werte zugewiesen. Zu dem Knoten mit der höchsten Punktzahl werden die Knoten, die mit ihm um die Erklärung derselben Manifestationen konkurrieren, gruppiert. Diese Knoten werden (in INTERNIST-I) als sich gegenseitig ausschließend angenommen. Unter ihnen wird je nach ihrer Anzahl durch verschiedene differential-diagnostische Strategien einer ausgewählt, der die plausibelste Erklärung für die Beobachtungen darstellt. Analog wird mit den Knoten verfahren, die um die Erklärung der übrigen noch unerklärten Manifestationen streiten. Weitere Symptome und Labordaten werden in Abhängigkeit von den bislang schon gefundenen Grobdiagnosen erfragt und der Prozeß wird iteriert, bis zu allen Manifestationen konkrete Erkrankungen als ihre Erklärungen gefunden worden sind.

[5] Die Essenz des abduktiven Schlußfolgerns ist das Erzeugen von Hypothesen, die, wenn sie wahr wären, eine gewisse Ansammlung von beobachteten Fakten erklärten.

Leider sind nur wenige auftretende Symptome hinreichende Bedingungen (pathognonomisch) für die Annahme einer bestimmten Diagnose. Aber es gibt Manifestationen, die für die Grobdiagnostik ausreichen und die Suche im Krankheitsbaum beschränken (z.B. Gelbsucht als Indiz für das Vorliegen einer Lebererkrankung).

Probleme mit INTERNIST-I, wie das anfängliche für den Benutzer unmotiviert erscheinende Hin- und Herspringen im Krankheitsbaum, wurden durch das Hinzufügen von diagnostisch-strategischem Wissen in INTERNIST-II (auch unter dem Namen CADUCEUS bekannt) ausgeräumt. INTERNIST hat in Tests mit komplizierten Fällen von Mehrfacherkrankungen Leistungen gezeigt, die mit denen menschlicher Experten vergleichbar sind [BF82]. Für eine detailliertere Beschreibung von INTERNIST sei auf [MPM82], [BF82], [AC84] und [Jac86] verwiesen.

2.2.3 PlanPower, GABI und EVA

PlanPower ist ein Expertensystem, das dazu gedacht ist, bei der Finanzplanung für Einzelpersonen oder Firmen zu assistieren. Das PlanPower-Projekt begann 1982. Von ersten Versuchen mit einem in LISP geschriebenen Produktionsregelinterpretierer, über den Einsatz von EMYCIN (der aus MYCIN entwickelten Expertensystem-Shell) [AC84] und Loops [BS83], ging die Entwicklung zurück zu einem direkt in LISP verfaßten Programm.

Die PlanPower-Wissensbasis besteht aus einer Frame- und einer Regeldatenbasis. Die Frames sind, wie in anderen Expertensystemen auch, hierarchisch angeordnet, um Vererbung zu ermöglichen, und sehen den Gebrauch von Defaultwerten und Demons vor. Eine Besonderheit von PlanPower ist die Berücksichtigung des Zeitfaktors durch die Repräsentation von Fakten als Zeitreihen. Sie stellen Werte eines Attributes zu verschiedenen Zeitpunkten dar und machen Defaultwerte als Ersatz für fehlende Angaben, bzw. Prozeduren zur Berechnung dieser Werte verfügbar [HH88]. PlanPowers Wissensbasis enthält über 200 Objekte, von denen ungefähr 120 Anlageobjekte sind. Die Regelbasis enthält ungefähr 6000 Regeln, die eng mit der Frame-Darstellung verknüpft sind und den *methods* in Objekt-orientierten Systemen (wie z.B. *Smalltalk* [GR83]) ähnlich sind. Die globale Strategie von PlanPower ist *successive refinement*, d.h. ein Lösungsvorschlag wird in mehreren Stufen, beginnend mit einer groben Skizze, verfeinert. Die Zeitreihen erzeugen eine beträchtliche Verzweigung bei der Planung. So benötigt PlanPower für eine Planung über einen Zeitraum von fünf Jahren ungefähr eine Stunde Rechenzeit [Ode87], [Ape87], [SG87] und [HH88].

Nicht unerwähnt sollen zwei weitere Prototypen bleiben, die sich ebenfalls mit der Wertpapieranlageberatung beschäftigen. Bei GABI (Geldanlage Beratungs- und Informationssystem) [Bal85] handelt es sich um ein in LISP und PEARL [DFW81] implementiertes Frame-basiertes System. Das Anlageberatungswissen ist in ungefähr 110 Regeln, die ebenfalls als Frames dargestellt sind, gespeichert. Zur Behandlung von unsicherem Wissen wird ein Certainty Factor-Mechanismus wie in MYCIN benutzt. Die Strategie von GABI ist es, in einem *Vordialog* Fakten über einen Kunden zu erfragen. Durch vorwärts verkettete Regeln werden aus diesen Kundendaten Anlagehypothesen generiert. Sie werden nach der Größe ihrer Evidenz sortiert. Dann versucht der Regelinterpretierer durch rückwärts verkettete Regeln den/die Spitzenreiter zu verifizieren, bis eine Anlagehypothese einen Evidenzwert hat, der größer als ein vorgewählter Schwellenwert ist. Diese Anlageform wird als Anlagevorschlag präsentiert. Die Frames, die die Anlagehypothesen darstellen, sind nicht in einer Hierarchie angeordnet, sondern stehen nebeneinander. Das bedeutet, daß schon nach dem Vordialog die Evidenzen für alle Anlagehypothesen berechnet werden müssen. Das könnte für eine große Wissensbasis einen erheblichen vermeidbaren Rechenaufwand bedeuten.

EVA (Expertensystem zur Vermögensanlageberatung) [Bac86] ist ein mit Hilfe der Shell BABYLON [GMD85] erstellter Expertensytem-Prototyp. Seine Strategie ist ähnlich zu der von GABI. Die Anlageobjekte sind als Frames, die Regeln in von BABYLON vorgesehenen *Rule-Sets* gespeichert.

In einer *Analysephase* werden fragebogenartig Kundendaten ermittelt. In der folgenden *Problemlösungsphase* werden durch vorwärts verkettete Regeln Hypothesen aufgestellt, die danach durch rückwärts verkettete Regeln überprüft werden. Aus den verifizierten Hypothesen wird daraufhin eine Anlageempfehlung aus 1–3 Anlageformen gegeben. Die möglichen Empfehlungen sind bislang nur ein Ausschnitt aus der Angebotspalette einer Bank. Abgesehen von umfassend bearbeiteten kursneutralen Anlagen wie Sparbriefen und Termineinlagen, gibt es lediglich die pauschalen Empfehlungen *Bundesanleihen*, bzw. *-obligationen* und *Aktien*, ohne diese Wertpapiergruppen detaillierter zu betrachten.

Kapitel 3

Die Expertensystem-Shell Donald

3.1 Motivation des Prototyp-Ansatzes

Ausgehend davon, daß es keinen Sinn hat eine Expertensystem-Shell konstruieren zu wollen, ohne ein konkretes Problem als Studien- und Testobjekt zu haben, wurde die Wertpapieranlageberatung als Thema gewählt.

Der Aufbau einer Wissensbasis erfordert intensive Interviews und Diskussionen mit den Experten, deren Wissen im Computer abgebildet werden soll.

In diesen Gesprächen zeigte sich ein auch in der Expertensystemliteratur beschriebenes Problem, nämlich daß das Expertenwissen nicht systematisch strukturiert vorliegt, sondern wie kompiliert erscheint, d.h. kollabiert und reduziert auf eine minimale und effiziente Form, die es schwieriger macht, es zu extrahieren.[1]

Waterman schreibt an anderer Stelle [Wat86 S. 153]:

> He [the expert] makes complex judgements rapidly, without laboriously reexamining and restating each step in his reasoning

[1] An expert's knowledge seems compiled, that is, collapsed and reduced to a minimal and efficient form, which makes it more difficult to extract. [Wat86 S. 155]

process. The pieces of basic knowledge are assumed and are combined so quickly that it is difficult for him to describe the process.[2]

Experten kommen zu richtigen Lösungen und haben um so mehr Mühe, den Lösungsweg zu beschreiben, je kompetenter sie sind. Das ist was Johnson die Paradoxie des Expertenwissens nennt.[3]

Der Versuch, aus der zu der Lösung gelieferten Argumentation Regeln zu bilden, mißlang fast durchweg. Regelformulierungen wurden von den Experten verworfen, weil sie zu allgemein waren und deshalb zu viele Ausnahmen hatten. Andererseits machen zu spezielle Regeln die Wissensbasis unnatürlich und schwer verständlich.

Vor allem aber ist es so, daß Experten beim Problemlösen solche "wenn-dann"-Regeln anscheinend nicht bewußt anwenden. Vielmehr sieht es so aus, als assoziierten sie bei der Schilderung eines Problems bestimmte in Frage kommende Musterfälle. Ein solcher Verdacht erzeugt dann konkrete Fragen, deren Beantwortung weitere speziellere Assoziationen auslöst.

Zwei Ideen bestimmten nun die weitere Vorgehensweise:

1. Der Aufbau der Wissensbasis wird um so leichter, je ähnlicher die Verarbeitung des Wissens im Computer den wirklichen Denkprozessen ist.

2. Effizienz und Kompetenz des Expertensystems sind um so größer, je ähnlicher das Computermodell die wirkliche Problemlösungsstrategie simuliert.

Logischerweise wurde nicht länger versucht, das Expertenwissen in Regelform zu bringen. Stattdessen wurden die vermuteten Musterfälle (fortan *Prototypen* genannt) identifiziert. Die Bearbeitung solcher Prototypen bildet nun ein Kernstück der Wissensverarbeitung in Donald.

[2]Er (der Experte) kommt rasch zu komplexen Einschätzungen, ohne mühsam jeden Schritt seines Denkprozesses wieder und wieder zu überprüfen. Die Bestandteile des Grundwissens werden so schnell vermutet und kombiniert, daß es für ihn schwierig ist, diesen Vorgang zu beschreiben.

[3]the paradox of expertise [Joh83], [Wat86 S. 154]

Die in Donald eingesetzte Lösungsstrategie ist das Resultat des Bemühens, eine möglichst natürliche Abbildung des Fachwissens, der Erfahrungen und der Strategien der Experten zu schaffen. Sie sieht wie folgt aus:

1. Donald fordert (durch Fragen) eine Problembeschreibung vom Benutzer.

2. Jede erhaltene Antwort kann einen oder mehrere Verdachtsmomente (fortan *Hypothesen* genannt) "wecken".

3. Jede Hypothese ist mit einem oder mehreren Prototypen verknüpft. Die Hypothesen werden geprüft, indem die Problembeschreibung mit den Prototypen verglichen wird. Eine Hypothese wird als bestätigt angesehen, wenn der oder die assoziierten Prototypen genügend Ähnlichkeit mit der Problembeschreibung aufweisen. Die bestätigten Hypothesen bestimmen, welche weiteren Fragen gestellt werden müssen. Darüberhinaus liefern sie die Mosaiksteine, aus denen die Lösung aufgebaut wird. Der Kreislauf beginnt wieder bei Punkt 1 und endet, wenn keine weiteren Fragen mehr erforderlich sind.

Dieser Ablauf wird später anhand eines konkreten Beispiels aus dem Expertensystem Invest detaillierter dargestellt werden. Abbildung 3.1 stellt den Ablauf einer Beratung mit Invest graphisch dar.

3.2 Wissensrepräsentation mit Frames in Donald

In diesem Abschnitt wird die Syntax für Frames in Donald definiert, und es werden Details der Implementation der Frame-Heterarchie erläutert.

Alle in Donald auftretenden Objekte sind als Frames gespeichert. Bei diesen Objekten handelt es sich zum einen um Begriffe der realen Welt, die für die konkrete Anwendung eines mit Donald erstellten Expertensystems von Bedeutung sind (im Fall der Anlageberatung sind das zum Beispiel Personen, Wertpapiere, wirtschaftliche Rahmenbedingungen), zum anderen um

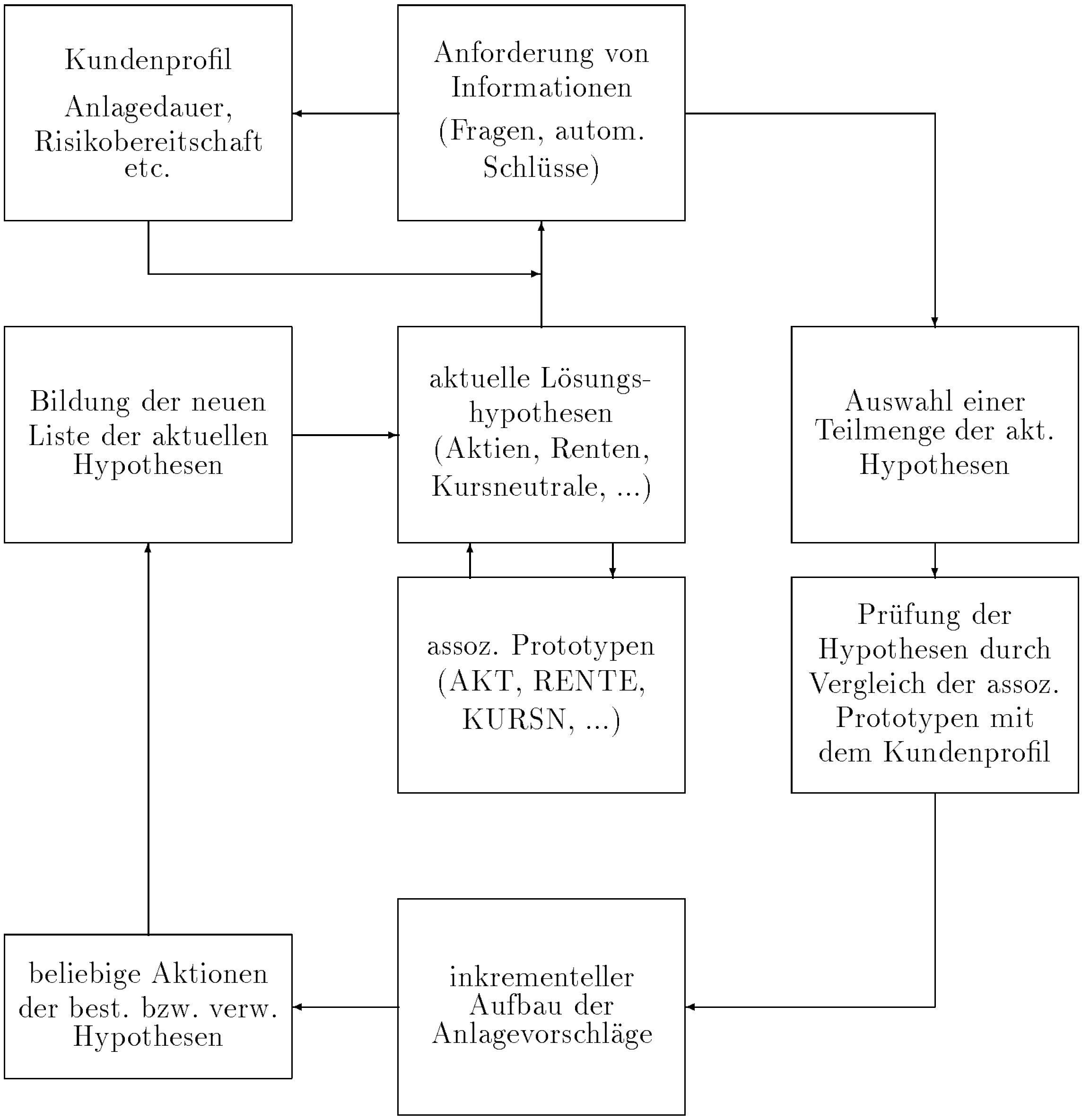


Abbildung 3.1: Flußdiagramm einer Beratung mit Invest

abstrakte Begriffe, die zu Donalds Inferenz- bzw. Wissensrepräsentationsmechanismus gehören (zum Beispiel Hypothesen, Lösungsvorschläge). Es folgen Beispiele verschiedener Typen von Objekten und ihre Darstellung als Frames.

3.2.1 Klassen, Teilklassen und Instanzen

Frames treten in Donald in drei Grundformen auf. Jeder Frame ist entweder eine *Klasse*, eine *Teilklasse* oder eine *Instanz*. Durch Klassen werden Objekte oder Begriffe beschrieben, die nicht als Spezialfälle eines allgemeineren, schon bekannten Begriffes definiert werden können. Klassen enthalten Definitionen für *Slots*, das heißt Definitionen der Merkmale, durch die die Mitglieder einer Klasse charakterisiert werden.

Im Gegensatz zu den Klassen stellen Instanzen konkrete Objekte dar, die keine weitere Spezialisierung mehr zulassen. Instanzen enthalten deshalb keine Slotdefinitionen, sondern *Views*. Ein View beschreibt einen Blickwinkel, aus dem eine Instanz als Mitglied einer Klasse betrachtet wird. Ein View ist die Aufzählung der in der zugehörigen Klasse definierten Slots zusammen mit den für die Instanz charakteristischen Werten. Instanzen, die einen View auf eine bestimmte Klasse besitzen, sind Elemente dieser Klasse.

Die Abbildungen 3.2–3.3 zeigen als Beispiel eine mögliche Darstellung der Klasse aller Aktien und als ein Mitglied dieser Klasse die Instanz "Beispiel-AG".

```
(Class Aktie
   (Slots are
          (Symbol Bezeichnung)
          (Real   Nominalwert)
          (Real   Kurs)
          (Real   KGV)))
```

Abbildung 3.2: Die Klasse Aktie

Da Klassen nicht mehr durch den Vergleich mit einem umfassenderen Konzept beschrieben werden können, besitzen sie keine Views. Umgekehrt haben

```
(Instance BAG
  (View Aktie with
        (Bezeichnung = Beispiel-AG)
        (Nominalwert = 50.0)
        (Kurs = 271.3)
        (KGV = 5.3)))
```

Abbildung 3.3: Die Instanz BAG

Instanzen, weil sie keine Elemente mehr haben können, keine Definitionen für Slots.

Die dritte Gruppe von Frames sind die *Teilklassen*. Sie enthalten sowohl Slotdefinitionen als auch Views. Teilklassen sind Teilmengen der Klassen (oder Teilklassen), für die sie Views besitzen. Sie ermöglichen eine strukturierte Darstellung einer Menge von Objekten und Begriffen mit vielen Abstufungen unterschiedlichen Abstraktionsgrades.

```
(Class Wertpapier
  (Slots are
        ((OneOf klein mittel hoch) Risiko)
        (Symbol                    Emittent)
        (Symbol                    Land)
        (Integer                   Nominalwert)
        ((OneOf ja nein)           boersengaengig)))
```

Abbildung 3.4: Die Klasse Wertpapier

```
(SubClass KursneutralesAnlagepapier
  (Slots are
        ((OneOf verzinslich abgezinst) Zinsmodus)
        (Real                           Nominalzins)
        (Real                           Rendite)
        (Integer                        Laufzeit))
  (View Wertpapier with
        (Risiko ist klein)
        (börsengängig = nein)))
```

Abbildung 3.5: Die Teilklasse Kursneutrales Anlagepapier

```
(SubClass Bundesschatzbrief
  (Slots are
        ((VectorOf Real) Nominalzinsen)
        ((OneOf A B)     Typ)))
  (View Wertpapier with
        (Emittent ist Bund)
        (Land ist BundesrepublikDeutschland)
        (Nominalwert ist 100)
  (View KursneutralesAnlagepapier))
```

Abbildung 3.6: Die Teilklasse Bundesschatzbrief

```
(SubClass Bundesschatzbrief-A
  (View KursneutralesAnlagepapier with
        (Zinsmodus ist verzinslich)
        (Laufzeit ist 6))
  (View Bundesschatzbrief with
        (Typ ist A)))
```

Abbildung 3.7: Die Teilklasse Bundesschatzbrief-A

```
(Instance BSB-A-1987/4
  (View KursneutralesAnlagepapier with
        (Rendite ist 5.84)
        (Rückzahlung ist 1.9.1993)
        (Zinstermine ist (List 1.9.)))
  (View Bundesschatzbrief with
        (Nominalzinsen sind (Vector 4.0 5.0 6.0 6.25 6.5 8.0)))
  (View Bundesschatzbrief-A))
```

Abbildung 3.8: Ein spezielles Wertpapier

3.2.2 Die Definition von Slots in Klassen und Teilklassen

Die Slots in Klassen und Teilklassen dienen nicht dem Speichern von Werten (obwohl auch das möglich ist), sondern der Definition der Merkmale, durch die sich die Mitglieder dieser Klasse unterscheiden lassen.

Jeder Slot hat einen Typ. Das ist eine Angabe darüber, welche Werte für diesen Slot erwartet werden. Mögliche Typen sind zum Beispiel *Integer* (ganze Zahlen), *Real* (Gleitkommazahlen), *Symbol* und andere mehr. Der Typangabe folgt der Name des Slots. Danach kann ein sogenannter *Default* angegeben werden. Das ist ein Wert, der für alle Instanzen und Teilklassen, die einen View mit diesem Slot haben, als Wert dieses Slots angenommen wird, sofern für sie nicht ausdrücklich ein anderer Wert angegeben wurde. Ein Beispiel dazu folgt im Abschnitt 3.4 über Vererbung. Die Abbildungen 3.4–3.8 illustrieren die Ausführungen in diesem Abschnitt.

3.2.3 Die Definition von Views in Teilklassen und Instanzen

Wie schon beschrieben, dienen Views dazu, neue Objekte oder Begriffe als spezielle Vertreter einer allgemeineren Klasse zu beschreiben.

Teilklassen und Instanzen können beliebig viele Views haben. Dabei müssen die (Teil-) Klassen, auf die die Views deuten, schon zuvor definiert sein. Das garantiert, daß der durch die Teilmengen- bzw. Elementrelation gebildete Graph aller Frames azyklisch ist.

Views sind Listen, die mit der Sequenz "View *Name* with" beginnen.

Um wichtige Gesichtspunkte von weniger wichtigen unterscheiden zu können, kann eine ganze Zahl $n \geq 1$ als Gewicht eines Views angegeben werden. Ohne diese Angabe wird als Gewicht 1 angenommen.

Darauf folgt optional eine Folge von Tripeln. Das sind Listen aus dem Namen eines Slots aus der zuvor durch *Name* bezeichneten (Teil-) Klasse, einem Zuweisungsoperator (z.B. "=") und einem Ausdruck, der der Wert dieses Slots in dieser Instanz bzw. Teilklasse ist.

3.2.4 Die Syntax für Frames in Donald

Die in den vorangegangenen Beispielen illustrierte Syntax für Frames wird in diesem Abschnitt formal definiert.

Frames werden als Sätze der Grammatik mitgeteilt, die in Tabelle 3.1 dargestellt ist.

Dabei sind nicht in Anführungszeichen (",") eingeschlossene Wörter terminale Symbole. Ein nachgestelltes Sternchen ($*$) heißt, daß der vorangegangene Ausdruck beliebig oft (auch keinmal) auftreten darf. Ein nachgestelltes Pluszeichen (+) heißt, daß der vorangegangene Ausdruck beliebig oft wiederholt werden darf, aber mindestens einmal auftreten muß. Eckige Klammern ([,]) umfassen Ausdrücke, die optional sind. Der senkrechte Strich (|) trennt Alternativen voneinander.

Kursiv gesetzte Begriffe bezeichnen Lisp-Atome. Diese Begriffe stellen selbst durch ihre Bedeutung die an das Lisp-Atom gestellte Bedingung dar. Zum Beispiel bezeichnet *Slotname* den Platz, an dem ein Lisp-Atom stehen darf, das der Name eines Slots ist.

"atom" und "s-expr" sind durch den zugrundeliegenden Lisp-Interpreter festgelegt und bezeichnen gültige Lisp-Atome bzw. Lisp-S-Ausdrücke [Wil86]

<table>
<tr><td>“frame”</td><td>::=</td><td>“class” | “subclass” | “instance”</td></tr>
<tr><td>“class”</td><td>::=</td><td>(Class Classname “slotdef”)</td></tr>
<tr><td>“slotdef”</td><td>::=</td><td>(Slots are “class-slot”+)</td></tr>
<tr><td>“class-slot”</td><td>::=</td><td>(“type” Slotname [“assignop” “value”] “facets”∗)</td></tr>
<tr><td>“subclass”</td><td>::=</td><td>(SubClass SubClassname [“slotdef”] “view”+)</td></tr>
<tr><td>“instance”</td><td>::=</td><td>(Instance Name “view”+)</td></tr>
<tr><td>“view”</td><td>::=</td><td>(View { Classname | SubClassname } [weight “w-integer”]
[with “slot”+])</td></tr>
<tr><td>“slot”</td><td>::=</td><td>(Slotname “assignop” “value” “facet”∗)</td></tr>
<tr><td>“assignop”</td><td>::=</td><td>= | is | are | ist | sind</td></tr>
<tr><td>“type”</td><td>::=</td><td>“atomic-type” | (OneOf “atom”+) |
(“ctype-tag” “atomic-type”)</td></tr>
<tr><td>“atomic-type”</td><td>::=</td><td>Integer | Real | Symbol | String |
Classname | SubClassname | Other</td></tr>
<tr><td>“ctype-tag”</td><td>::=</td><td>SetOf | ListOf | VectorOf | HListOn | AssocListOn</td></tr>
<tr><td>“value”</td><td>::=</td><td>“atom” | (Set “value”∗) | (List “value”∗) |
(Vector “value”∗) | (HList “h-pair”∗) |
(AssocList “pair”∗)</td></tr>
<tr><td>“pair”</td><td>::=</td><td>(“atom” . “value”)</td></tr>
<tr><td>“h-pair”</td><td>::=</td><td>(“atom” . “cf-integer”)</td></tr>
<tr><td>“facet”</td><td>::=</td><td>[to | if | when | falls] Facetname “s-expr”</td></tr>
</table>

Tabelle 3.1: Framesyntax in Donald

[Ste84].

"w-integer" bezeichnet eine natürliche Zahl ≥ 1. "cf-integer" ist eine ganze Zahl im Bereich von -1000 bis +1000.

Diese Darstellung von Frames heißt die *externe Form*. Sie ist, wie man der Grammatik ansehen kann, ein gültiger Lisp-S-Ausdruck und läßt sich als solcher mit den vom Lisp zur Verfügung gestellten Funktionen anzeigen, ausdrucken, auf Hintergrundspeicher (Platten, Disketten, Streamer) schreiben und auch wieder einlesen.

Eine FRL-ähnliche Implementation einer früherern Donald-Version hatte gezeigt, daß der Zugriff auf Frameinhalte durch die Darstellung der Frames als geschachtelte Listen verlangsamt wird.

Eine parallel entwickelte, auf einer Vektordarstellung beruhende Repräsentation der Frames [Heu87] ergab kürzere Zugriffszeiten und weniger Speicherplatzbedarf. Die Zugriffszeit auf eine Vektorkomponente ist konstant, während die Zugriffszeit auf ein Listenelement proportional zur Länge der Liste anwächst.

Das Bemühen, bei der Implementation CommonLisp-Kompabilität zu erzielen, legte es nahe, die *structure*-Möglichkeiten von CommonLisp zu benutzen. CommonLisp-structures sind Pascal-ähnliche Datenstrukturen. Bei der Definition einer solchen Struktur erzeugt der Interpreter bzw. der Compiler automatisch Zugriffsfunktionen auf die angegebenen Datenfelder und Funktionen zum Erstellen und Kopieren solcher Strukturen.

Die aktuelle Implementation rückt wieder ein wenig vom völlig an Vektoren orientierten Vorgehen ab. Zunächst war für jeden View einer Instanz bzw. einer Teilklasse ein Vektorfeld vorgesehen [Heu87]. Das macht das Verändern einer Instanz durch Hinzufügen oder Löschen eines Views kompliziert, da die Länge eines Instanzvektors ab dem Zeitpunkt der Definition der Instanz fest ist. Das gleiche gilt für Viewvektoren und die darin enthaltenen Slotvektoren.

Um hier wieder mehr Bewegungsraum zu schaffen, sind die Viewvektoren wieder zu einer Liste zusammengefaßt worden, die in einer Vektorzelle der Instanz gespeichert ist. Analog war das Vorgehen bei den Slotvektoren in den Views.

name	Name des Frames
bindings	derzeit unbenutzt
longname	lange Namensversion zu Ausgabezwecken
source-code	externe Form des Frames
created	Datum der Erzeugung des Frames
modified	Datum der letzten Änderung
version	Versionsnummer
database	die zugehörige Datenbasis
position	derzeit unbenutzt
backup	derzeit unbenutzt
flags	zur Zustandskontrolle des Frames
author	der Autor dieses Frames
views	alphabetische Liste der Viewvektoren
default*	Default-Setzungen der Slots
instances*	Instanzen-Liste der (Teil-) Klassen
subclasses*	Teilklassen-Liste der (Teil-) Klassen

Tabelle 3.2: Die interne Form der Frames

3.2.5 Die interne Darstellung der Frames

Die Transformation von der externen zur *internen* Darstellung geschieht durch das Einlesen der externen Form durch den Lisp-Interpreter. *Class, SubClass* und *Instance* sind Lisp-Makros, die beim Einlesen zu einem Aufruf der Funktion *create-frame* expandiert werden.

Die externe Form ist also gerade der Lisp-S-Ausdruck, dessen Auswertung als Seiteneffekt die Herstellung der internen Darstellung hat. Diese interne Form ist in Tabelle 3.2 dargestellt.

Die mit * bezeichneten Vektorfelder sind nur für Klassen und Teilklassen definiert.

Zu Tabelle 3.2 sind folgende Bemerkungen zu machen:

- Im Feld *database* steht der Name der Wissensbasis, zu der dieser Frame gehört.
- Flags markieren verschiedene Zustände des Frames. Sie zeigen zum Bei-

slots	Liste der Slotvektoren
upward	Zeiger auf den Framevektor
class	Name der definierenden Klasse des Views
definition	Zeiger auf den Klassenvektor
weight	Gewicht des Views
bindings	derzeit unbenutzt

Tabelle 3.3: Die interne Form der Views

name	Name des Slots
upward	Zeiger auf den Viewvektor
type	Typangabe des Slots
value	Wert des Slots
facets	Facetten des Slots
demons	Demons des Slots
origin	Herkunft des Slots

Tabelle 3.4: Die interne Form der Slots

spiel an, daß die interne und die externe Form nach einem Schreibzugriff auf den Frame nicht mehr übereinstimmen. Wird die externe Form benötigt und das entsprechende Flag ist gesetzt, so wird zunächst die neue externe Form aus der internen Darstellung berechnet, das Flag gelöscht und die neue Form weitergegeben.

- Auch Klassen haben aus Gründen der Uniformität ein *views*-Feld, das aber stets nur die leere Liste enthält.
- Das Feld *default* enthält einen Viewvektor mit den Default-Setzungen der (Teil-) Klasse.
- Die Felder *instances* und *subclasses* enthalten Listen mit den Namen der Instanzen bzw. der Teilklassen.

Ebenso wie Frames haben auch die Views und Slots eine interne Darstellung als Vektoren. Sie sind in den Tabellen 3.3 und 3.4 abgebildet.

Die Felder *upwards* enthalten einen Zeiger auf den Framevektor (Viewvektor), in dem der Viewvektor (Slotvektor) enthalten ist. Mit *Herkunft des Slots* ist gemeint, aus welcher Quelle der Inhalt dieses Slots stammt. Mögliche Werte sind :user, :demon, :default und nil. Diese Reihenfolge bildet auch die Priorität beim Schreibzugriff auf einen Slot: zum Beispiel kann eine vom Benutzer eingegebene Antwort einen Slotinhalt stets überschreiben, weil sie *origin = :user* hat. Umgekehrt kann ein Default-Wert weder einen von einem Demon errechneten Wert noch eine Benutzereingabe überschreiben.

3.3 Facetten und Demons in Donald

Demons und *Facetten* sind Begriffe, die in den vorangegangenen Abschnitten, vor allem im Zusammenhang mit FRL, schon erläutert worden sind. In diesem Abschnitt wird ihre Implementation und ihre Wirkungsweise beschrieben werden.

3.3.1 Facetten

Facetten sind in Slots auftretende Speicherplätze, auf deren Inhalt mit einem Schlüsselwort, dem Facettennamen (siehe Tabelle 3.2), zugegriffen werden kann. Sie sind als Lisp-*association-lists* [Wil86] im Vektorfeld *facets* gespeichert.

Im Unterschied zu FRL werden die Speicherplätze für den Wert des Slots (vgl. die VALUE-Facette) und für angebundene Prozeduren, die beim Zugriff auf den Slot ausgeführt werden, (vgl. die IF-NEEDED-Facette) nicht als Facetten bezeichnet.

In der derzeitigen Version von Donald sind die Facetten *ask, explain, trigger, weight* und *constraint* implementiert. Abbildung 3.9 zeigt eine Slotdefinition aus der Klasse *Anleger* und illustriert die Syntax für Facetten und Demons.

Die folgende Aufzählung erläutert die Bedeutung der Facetten im einzelnen.

- Die Facetten *ask* und *explain* enthalten einen String, der den Formatanforderungen der Lisp-Ausgabefunktion *format* [Wil86] genügt. Der ask-String wird benutzt, wenn eine Frage nach dem Wert des Slots

```
(Class Anleger
  (Slots are
    ...
    ((HListOn |ich weiss noch nicht| nein ja) AnlageVerfuegbarkeit
     to ask "Muss das Geld waehrend der Anlagedauer verfuegbar sein? "

     to trigger     (ja (.RENTE .INVESTMENT)
                    nein (.OPT .AUSL .AKT))
     to explain     "Verfuegbar sind Geldanlagen im weitesten Sinne im- ~@
                    mer, da man schlimmstenfalls die Anlage beleihen ~@
                    kann. Verfuegbar soll hier heissen, dass die Anlage ~@
                    liquidierbar ist unter (teilweisem) Verzicht auf ~@
                    die angestrebten Anlageziele, aber ohne zusaetz- ~@
                    liche Kosten wie Kreditzinsen. ~%"
     put> (InCase(eq (GetSlot AnlageVerfuegbarkeit) 'nein)
                    (Forward (KURSN-Anleger Liquiditaet/Rendite Rendite)
                             (BSB-Anleger Sperrfrist ja))
                    (Forward (KURSN-Anleger Liquiditaet/Rendite -)
                             (BSB-Anleger Sperrfrist -))))
    ...
     ))
```

Abbildung 3.9: Facetten und Demons eines Slots

“constraint”	::=	(“atomic-constr”) |
		([Or | And] (“atomic-constr”)+)
“atomic-constr”	::=	([not] “atomic-constr”) |
		(Range “number” “number”) |
		(“predicate” “s-expr”$*$)

Tabelle 3.5: Syntax für *constraints* in Donald

für den Benutzer formuliert wird. Ist diese Facette leer, wird ein Standardfragetext benutzt. Der explain-String wird ausgegeben, wenn der Benutzer auf eine Frage hin weitere Erläuterungen anfordert.

- Die *trigger*-Facette enthält eine Lisp-association-list. Die möglichen Slotwerte sind darin die Schlüssel. Die Werte dazu sind ein Hypothesenname oder eine Liste von Hypothesennamen. Auf eine Benutzerantwort hin wird an die entsprechende(n) Hypothese(n) je ein Triggerpunkt vergeben.

- Die Facette *weight* (siehe Abbildung 3.14) enthält eine natürliche Zahl $n \geq 1$, die für das Gewicht dieses Slots innerhalb des Views steht (siehe Abschnitt 3.5 über das Matchen).

- Die *constraint*-Facette ermöglicht die Angabe von Bedingungen, die ein für diesen Slot zulässiger Wert erfüllen muß. Die Angabe der *constraints* folgt der Syntax in Tabelle 3.5.

Ein Beispiel für eine constraint-Facette wäre

$$\text{constraint (And } (\geq 2)(\leq 4))$$

was der Angabe

$$\text{constraint (Range 2 4)}$$

entspricht und nur numerische Werte zwischen 2 und 4 als Slotinhalt zuläßt.

Facetten liefern Zusatzinformationen zu einem Slot, die in vorgegebenen Situationen vom Programm abgefragt werden können. Demons dagegen sind unabhängiger von der Kontrolle, die das Programm ausübt.

3.3.2 Demons

Demons sind Prozeduren, die zu jedem Zeitpunkt während des Programmablaufs durch einen Schreib- oder Lesezugriff auf einen Slot angestoßen werden können. Meist erfüllen sie Aufgaben wie, die Slotwerte zu aktualisieren oder leeren Slots einen hergeleiteten Wert zu geben. Da Demons ihrerseits wieder Zugriffe auf andere Slots enthalten können, kann es sein, daß größere Programmsequenzen, die in der Wissensbasis auftreten, unabhängig von der kontrollierenden Shell ablaufen.

Die Syntax für Demons entspricht der für Facetten (siehe Tabelle 3.1). Anstelle des Facettennamens steht eines der vier Lisp-Atome *>get, get>, >put* oder *put>*. Sie bezeichnen den Zeitpunkt der Ausführung des Demons. Dabei stehen *get* und *put* für einen Lese- bzw. Schreibzugriff. Das vorangestellte und nachgestellte >-Zeichen bedeutet *vor* bzw. *nach* dem angegebenen Zugriff. Der darauf folgende Lisp-S-Ausdruck kann ein beliebiges Stück Lisp-Quellcode sein. In Donald sind vier Demons bereits vorgesehen, nämlich *Forward, Fetch, GetSlot* und *InCase*. Die folgende Aufstellung erklärt die Bedeutung dieser Demons.

- Forward wird mit der Syntax

  (Forward (*Viewname Slotname* ["value"])+)

  aufgerufen. Forward gibt für jedes Paar bzw. Tripel den mit *value* bezeichneten Wert an den genannten Slot im View *Viewname* desselben Frames weiter. In die angesprochenen Slots geschrieben wird der Wert nur, wenn der bisherige *origin* der Slots es erlaubt (vgl. Tabelle 3.4). Ist kein Wert angegeben, benutzt Forward den Wert des Slots, in dem es selber steht.

- Fetch folgt der Syntax

  (Fetch (*Viewname Slotname*)+)

  Fetch durchsucht die durch *Viewname* und *Slotname* bezeichneten Slots desselben Frames bis ein Wert gefunden wird. Dieser Wert wird im Slot, in dem der Demon steht als Wert eingetragen.

- GetSlot-Demons werden durch einen Ausdruck der Form

  (GetSlot [*Slotname* | (*Slotname*) | (*Viewname Slotname*)])

  angegeben. GetSlot liefert als Ergebnis den Wert des Slots *Slotname* im selben View bzw. im View *Viewname*, falls ein Viewname angegeben ist. Ist weder ein Slot noch ein View angeführt, ist das Ergebnis der Wert des Slots, in dem der Demon steht.

- Der InCase-Demon ist eine Liste der Gestalt

  (InCase "*condition*" "*then-part*" ["*else-part*"])

  condition, then-part und *else-part* sind Lisp-S-Ausdrücke. Ist der Wert von *condition* nicht NIL, wird *then-part* evaluiert; sonst, falls vorhanden, der *else-part*.

Die Speicherung der Facetten und Demons als Lisp-association-lists macht den Zugriff auf sie etwas weniger effektiv, als wenn jede Facette und jeder Demon ein eigenes Vektorfeld in der internen Darstellung des Slots erhalten hätte. Da aber das Auftreten von Demons und Facetten eher die Ausnahme ist, hätte man dadurch in jedem Slot eine Reihe von Vektorfeldern reserviert, die meist leer geblieben wären. Die Speicherplatzersparnis hatte hier Vorrang vor der Uniformität der Darstellung.

3.4 Vererbung von Informationen in Donald

Im ersten Teil dieses Abschnitts wird ausgeführt werden, wie Donald Nutzen aus der hierarchischen Anordnung von Frames zieht. Das Thema *Vererbung von Informationen* war ja schon im vorangegangenen Kapitel im Zusammenhang mit semantischen Netzen und FRL angerissen worden.

3.4.1 Vererbungsstrategie in FRL

Zur Motivation der aktuellen Implementation soll hier ein kurzer Rückblick auf die Erfahrungen mit einer früheren FRL-basierten Donald-Version gegeben werden.

Im Abschnitt 2.1.3 wurde ein kleines Beispiel mit der Funktion FGET-AS-NEEDED gegeben, in dem Informationen, die in einem in der Ako-Hierarchie höher postierten Frame gespeichert waren, für einen tiefer stehenden ausgenutzt wurden. In FRL gibt es dazu insbesondere drei Funktionen: FGET-I, FGET-N und FGET-Z. Die gemeinsame Syntax ist

$$(\text{FGET-}x\ \textit{Framename Slotname}), x \in \{I, N, Z\}.$$

Ihre Funktionsweise ist wie folgt:

- FGET-I sucht in der Ako-Hierarchie aufwärts, bis es einen Wert in der VALUE-Facette eines Slots mit dem angegebenen Namen findet. (Slotnamen müssen für die ganze Wissensbasis eindeutig sein.)
- FGET-N sucht zunächst in der Ako-Hierarchie nach einem VALUE. Mißlingt das, beginnt eine neue Suche nach einer DEFAULT-Facette. Bleibt auch das ohne Erfolg, sucht FGET-N beim dritten Mal nach einer IF-NEEDED-Facette.
- FGET-Z sucht zunächst im angegebenen Frame nach einem VALUE, dann nach einem DEFAULT und schließlich nach der IF-NEEDED-Facette. Erst wenn das ohne Ergebnis bleibt, geht FGET-Z in der Ako-Hierarchie aufwärts.

Für weitere Details siehe [RG77a] und [RG77b].

3.4.2 Vererbungsstrategie in Donald

Das aufwendige Durchsuchen der Frame-Hierarchie nach brauchbaren Slotwerten während einer laufenden Konsultation von Donald machte die Antwortzeiten des Expertensystems zu lang. Ein weiteres Problem von FRL war, daß nicht klar war, in welcher Situation welches FGET-x anzuwenden war. Schließlich benötigen sie stark unterschiedliche Rechenzeiten und liefern unterschiedliche Ergebnisse. Die Semantik der Frame-Hierarchie war unklar:

je nach Suchstrategie wurde ein entfernter VALUE einem nahen DEFAULT vorgezogen oder umgekehrt.

Das Berechnen der ererbten Slotwerte *während* einer Beratung ist nur dann nötig, wenn das Expertenwissen sich im Laufe einer Konsultation ändert. Für die Expertensystem-Shell Donald wurde von der Prämisse ausgegangen, daß das in der Wissensbasis niedergeschriebene Wissen langlebiger ist und höchstens zwischen zwei Beratungen geändert werden muß. So können die Slotwerte, die sich für Instanzen und Teilklassen ergeben, im voraus errechnet werden. Während einer Beratung liegt die Datenbasis dann in *kompilierter* Form vor.

Genau diese Kompilation geschieht bei der Transformation eines Frames von seiner externen in seine interne Form. Das macht es möglich, den Quellcode eines Frames kurz und übersichtlich zu gestalten. Bei der Übersetzung wird dann automatisch die gesamte für einen Slot verfügbare Information gesammelt und in *einem* Vektor gespeichert. Nur noch *eine* Zugriffsfunktion ist nötig, um den Wert eines Slots zu erfahren:

(fget *Framename Viewname Slotname*).

Das Lesen eines Slots ist auf diese Weise wesentlich effizienter geworden. Das Vergleichen zweier Frames (das *Matchen*), das ja den Hauptinferenzmechanismus von Donald darstellt und das eine Fülle solcher Leseoperationen umfaßt, ist deutlich schneller geworden.

Die folgende Frame-Hierarchie soll den noch zu beschreibenen Vererbungsalgorithmus illustrieren.

```
(Class C1
   (Slots are
      (Real CS1 = 1.0)
      (Real CS2 = 2.0)))

(SubClass SC1
   (Slots are
      (Integer SCS1 = 1))
   (View C1 with
      (CS1 = 1.5)))
```

```
(Instance I1
   (View SC1))

(Instance I2
   (View SC1 with
     (SCS1 = 2))
   (View C1 with
     (CS1 = 2.0)
     (CS2 = 3.0)))
```

Die Instanz I1 hat via SC1 auch einen View auf C1 geerbt: (fget 'I1 'C1 'CS2) liefert den Default-Wert 2.0 als Ergebnis. (fget 'I1 'C1 'CS1) ergibt den aus SC1 ererbten Wert 1.5, der den Default 1.0 aus C1 überschreibt. (fget 'I2 'C1 'CS1) hat als Resultat 2.0, da sowohl der Default aus C1 als auch der ererbte Wert 1.5 aus SC1 durch die explizite Setzung des Slotwertes überschrieben wurden.

Bei der Generierung von Teilklassen und Instanzen liest Donald den Quellcode des Frames und transformiert zunächst den Quellcode der angegebenen (der *eigenen*) Views in Vektoren.

Im zweiten Schritt werden alle Views aus den Teilklassen, auf die eigene Views verweisen, in einer Liste gesammelt. Für jedes Element dieser Liste wird geprüft, ob der zu generierende Frame schon einen View gleichen Namens besitzt. Ist das nicht der Fall, wird der View für den neuen Frame übernommen. Andernfalls werden der eigene und der ererbte Viewvektor zu einem zusammengefaßt. Das geschieht Slot-weise so, daß die Informationen im eigenen View erhalten bleiben und lediglich die Lücken aus dem ererbten View gefüllt werden.

Im dritten Schritt werden noch verbliebenene Lücken in den eigenen Views aus den Default-Views der Klassen (vergl. Abschnitt 3.2.5), bzw. Teilklassen, auf die sie verweisen, gefüllt.

Der Algorithmus ist so konzipiert, daß die auf kürzerem Weg ererbten Daten sich gegen die, die auf längerem Weg erhalten wurden, durchsetzen. Sind die Wege gleich lang, setzen sich die Views gemäß ihrer Bearbeitungsreihenfolge bei der Transformation durch. Es ist deshalb meistens richtig, die Views auf speziellere (Teil-) Klassen im Quellcode vor die Views auf allgemeinere zu stellen. Durch eine solche Anordnung erhält man die von Touretzky vorgeschlagene *Inferential Distance Ordering* [Tou86]. Um größere Flexibilität zu erhalten, ist diese Vorgehensweise zur Erzeugung der internen Gestalt eines Frames nicht automatisiert worden. Die folgenden Frames bilden ein Beispiel für den *Kürzesten-Weg-Algorithmus*:

```
(Class Tier
  (Slots are
    ((Oneof ja nein) Kann-fliegen)))

(SubClass Vogel
  (Slots are
    ((Oneof ja nein) Hat-Fluegel = ja))
  (View Tier with
    (Kann-fliegen = ja)))

(SubClass Nicht-fliegender-Vogel
  (View Vogel)
  (View Tier with
    (Kann-fliegen = nein)))

(Instance Tweety
  (View Nicht-fliegender-Vogel))
```

Die Anfrage *(fget 'tweety 'tier 'kann-fliegen)* ergibt *nein*, weil der View auf *Tier* mit der Slotsetzung *nein* in einem Schritt geerbt wird, während der View auf *Tier* auf dem Weg über den geerbten View *Vogel* mit der Slotsetzung *ja* zwei Schritte entfernt ist.

3.4.3 Korrektheit der Vererbung

Die Ausarbeitung der mathematischen Grundlagen der Vererbung von Informationen in Donald erfolgt nicht mehr im Rahmen dieser Arbeit. Donald ist ein *Klassen/Eigenschaften-System, das multiple Vererbung mit Ausnahmen erlaubt.* Die Herstellung der *Orthogonalität*[4] der Wissensbasis ist nicht Voraussetzung für das Funktionieren, schützt aber vor unerwünschtem Vererbungsverhalten. Bei Berücksichtigung des Inferential Distance Ordering ergeben sich die erwarteten Ergebnisse.

3.5 Das Matchen von Frames

In einem der einleitenden Abschnitte wurde bereits auf die Bedeutung des Vergleichens von Frames als Schlußfolgerungsmechanismus hingewiesen:

> Reasoning is dominated by a process of recognition in which new objects and events are compared to stored sets of expected prototypes ... [BW77 S. 5].[5]

In diesem Abschnitt wird beschrieben werden, wie dieses Vergleichen in Donald geschieht.

Die Funktion *match* kann auf je zwei beliebige Lisp-Ausdrücke (z.B. Frames, Views, Slots, Listen oder Atome) angewandt werden. Um zwei Frames miteinander zu vergleichen, berechnet *match* rekursiv die Ähnlichkeit der gemeinsamen Views und darin die Ähnlichkeit der Slotsetzungen. Als Ergebnis liefert *match* eine ganze Zahl zwischen -1000 und $+1000$. Dabei bedeutet -1000, daß die Frames völlig verschieden sind, und $+1000$, daß sie, soweit es ihre gemeinsamen Views betrifft, gleich sind.

Der Algorithmus zum Vergleichen zweier Frames F_1 und F_2 sieht wie folgt aus:

1. Bilde die Listen VL_1 und VL_2 der Views von F_1 und F_2.

2. Jeder View ist mit einem Gewicht, d.h. einer Zahl $w \geq 1$ (Standardsetzung ist $w = 1$) versehen, die es ermöglicht, innerhalb eines Frames

[4] siehe [Tou86 S. 71f.]

[5] Denken ist ein Prozeß des Wiedererkennens, der bestimmt wird durch das Vergleichen neuer Objekte und Ereignisse mit gespeicherten Mengen erwarteter Prototypen.

nach wichtigen und weniger wichtigen Gesichtspunkten zu differenzieren (siehe Syntaxbeschreibung für Views).

Wende *match* auf alle Paare $[V_1, V_2]$ (mit $V_1 \in VL_1$ und $V_2 \in VL_2$) von Views auf dieselbe (Teil-) Klasse an. Das Resultat des Matchens zweier Views ist das gewichtete Mittel der Match-Resultate der Slots (s.u.). Das Gewicht des Ergebnisses ist das Maximum der Gewichte von V_1 und V_2.

Gibt es kein Paar $[V_1, V_2]$ von Views auf dieselbe (Teil-) Klasse, gib als Ergebnis 0 zurück.

Sonst bilde das gewichtete arithmetische Mittel aller Ergebnisse und gib es als Maß für die Ähnlichkeit von F_1 und F_2 zurück.

3. Jeder Slot ist mit einem Gewicht, d.h. einer Zahl $w \geq 1$ (Standardsetzung ist $w = 1$) versehen, die es ermöglicht, innerhalb eines Views nach wichtigen und weniger wichtigen Merkmalen zu differenzieren (siehe Beschreibung der Demons und Facetten).

   Wende *match* auf alle Paare $[S_1, S_2]$ (mit S_1 in V_1 und S_2 in V_2) von gleichnamigen, gesetzten Slots an, d.h. Slots, deren Werte ungleich NIL sind. Das Gewicht des Ergebnisses ist das Maximum der Gewichte von S_1 und S_2.

   Gibt es kein Paar $[S_1, S_2]$ von gleichnamigen gesetzten Slots, gib als Ergebnis 0 zurück.

   Sonst bilde das gewichtete arithmetische Mittel aller Ergebnisse und gib es als Maß für die Ähnlichkeit von V_1 und V_2 zurück.

4. Der Vergleich zweier Slots geschieht durch das Matchen der Werte der Slots. Je nach Typ des Slots gibt es dafür spezielle Funktionen (z.B. *match-string, match-integer, match-oneof, match-hlist*), die eigens zum Vergleich von Werten dieses Typs geschaffen worden sind.

Die Darstellung der Frames als Strukturen, die alphabetische Anordnung der Views nach den Namen ihrer definierenden Klasse und die Vorausberechnung aller ererbten Informationen dienen dazu, das Matchen möglichst schnell zu machen [KH86], [KH87].

3.6 Prototypen und Hypothesen

Im Abschnitt 3.1 ist der Einsatz von Prototypen und Hypothesen motiviert worden. Im folgenden wird die Art und Weise ihrer Implementation im Detail beschrieben werden.

3.6.1 Hypothesen

Unter Hypothesen in Donald sind Instanzen der Klasse *Hypothesis* zu verstehen. Das *Auslösen* und *Überprüfen* von Hypothesen ist der wichtigste Inferenzmechanismus der Expertensytem-Shell Donald.

Das Auslösen von Hypothesen (meist *Triggern* genannt) war ja schon im Zusammenhang mit den eingebauten *Facetten* angesprochen worden. Es geschieht während des Sammelns von Informationen. Zum Beispiel lösen Antworten des Benutzers auf Fragen des Expertensystems Hypothesen aus, d.h. diese Hypothesen werden vorgemerkt, um später überprüft zu werden. So soll dafür gesorgt werden, daß nur Hypothesen, für die zumindest ein Anhaltspunkt spricht, ausgewertet werden. Andernfalls verschwendete man Rechenzeit auf die Überprüfung völlig aussichtsloser Kandidaten.

Die Klasse *Hypothesis* ist in Abbildung 3.10 dargestellt.

Der Slot *TriggerPoints* nimmt als Wert eine ganze Zahl ≥ 0 auf, die angibt, wie oft die entsprechende Hypothese ausgelöst worden ist.

Die Slots *MoreGeneral* und *MoreSpecific* dienen dazu, für jede Hypothese die Listen mit den Namen der Vorgänger und der Nachfolger, d.h. der allgemeineren bzw. der spezielleren Hypothesen, aufzunehmen.

Aus den einfachen Hypothesen, die vom Expertensystem für richtig erachtet werden, versucht es durch Kombination komplexere Gruppen aufzubauen, die zu Lösungsvorschlägen führen. Um diese Gruppierungen zu ermöglichen, muß bekannt sein, welche Hypothesen mit welchen anderen "verträglich" sind. Dazu dient der Slot *IncompatibleWith*, der zu jeder Hypothese eine unter Umständen leere Liste der Hypothesen aufnehmen kann, die nicht mit ihr kombinierbar sind.

```
(Class Hypothesis
   (Slots are
          ((ListOf Hypothesis) MoreGeneral)
          ((ListOf Hypothesis) MoreSpecific)
          ((ListOf Hypothesis) IncompatibleWith)
          (Symbol              Prototype)
          ((ListOf Symbol)     Proposition)
          (Symbol              CompareWith)
          (Other               IfConfirmed)
          (Other               IfDisproved)
          (Integer             TriggerPoints)
          (Integer             ConfidenceFactor)))
```

Abbildung 3.10: Die Klasse Hypothesis

Der Slot *Prototype* enthält einen oder mehrere Namen von Frames, die zur Überprüfung der Hypothese benutzt werden sollen.

Ist dort nur ein Name angegeben, wird der Frame mit diesem Namen mit dem Frame, der im Slot *CompareWith* genannt ist, durch die Funktion *match* verglichen (siehe Abschnitt 3.5). Das Ergebnis dieses Vergleiches wird im Slot *ConfidenceFactor* gespeichert und repräsentiert die Größe der Gewißheit, mit der das Expertensystem diese Hypothese “glaubt”.

Ist der Inhalt des Prototype-Slots von der Form (And *Liste von Framenamen*) bzw. (Or *Liste von Framenamen*), werden alle angegebenen Frames wie zuvor durch *match* überprüft und das Minimum bzw. das Maximum der Ergebnisse als Konfidenzfaktor gespeichert.

Ist der Konfidenzfaktor größer bzw. kleiner als ein voreingestellter oberer bzw. unterer Schwellenwert, gilt die Hypothese als bestätigt bzw. als verworfen.

In den Slots *IfConfirmed* und *IfDisproved* kann beliebiger Lisp-Quellcode gespeichert werden, der für den Fall, daß die Hypothese bestätigt bzw. verworfen wurde, ausgeführt wird.

Der Slot *Proposition* enthält eine Liste von Views, der Syntax für Views in Instanzen und Teilklassen entsprechend, die zum stückweisen Aufbau von Lösungvorschlägen benutzt werden, wenn die Hypothese bestätigt worden ist. Mehr über die Konstruktion von Lösungen folgt in einem späteren Abschnitt.

Die Abbildung 3.11 zeigt eine Hypothese aus dem Expertensystem Invest. Dabei handelt es sich um die Hypothese *rentenf*, d.h. den Vorschlag, einen Anteil an einem Investmentfonds, der auf festverzinsliche Wertpapiere spezialisiert ist, zu kaufen.

```
(Instance .RENTENF
  (LongName is "Rentenfonds")
  (View Hypothesis with
        (MoreGeneral are (List .INVESTMENT .AUSLF .THESF))
        (MoreSpecific are (List .RENTENFKL .RENTENFLL))
        (IncompatibleWith are (List .AKTIENF .MISCHF))
        (Prototype is RENTENF)
        (Proposition is (List (View Rentenfonds)))
        (ConfidenceFactor is 0)))
```

Abbildung 3.11: Die Instanz .RENTENF

Die Einordnung dieser Hypothese in einen größeren Kontext zeigt Abbildung 3.12, die einen Ausschnitt aus einem semantischen Netz zeigt, daß durch die Verweise in den Slots *MoreGeneral, MoreSpecific, IncompatibleWith* und *Prototype* geknüpft wird. In der Abbildung sind der Übersichtlichkeit halber nicht alle Kanten eingetragen.

Das Beispiel zeigt, daß Frame-basierte Systeme eine echte Verallgemeinerung des Ansatzes mit semantischen Netzen sind. Die *is-a-* bzw. *ako*-Kanten (in Donald sind das die Kanten zwischen Instanzen, Teilklassen und Klassen) sind zwar wegen ihrer besonderen Bedeutung für die Vererbung hervorgehoben, aber auch andere Arten von Kanten lassen sich durch entsprechende Einträge in Slots leicht realisieren.

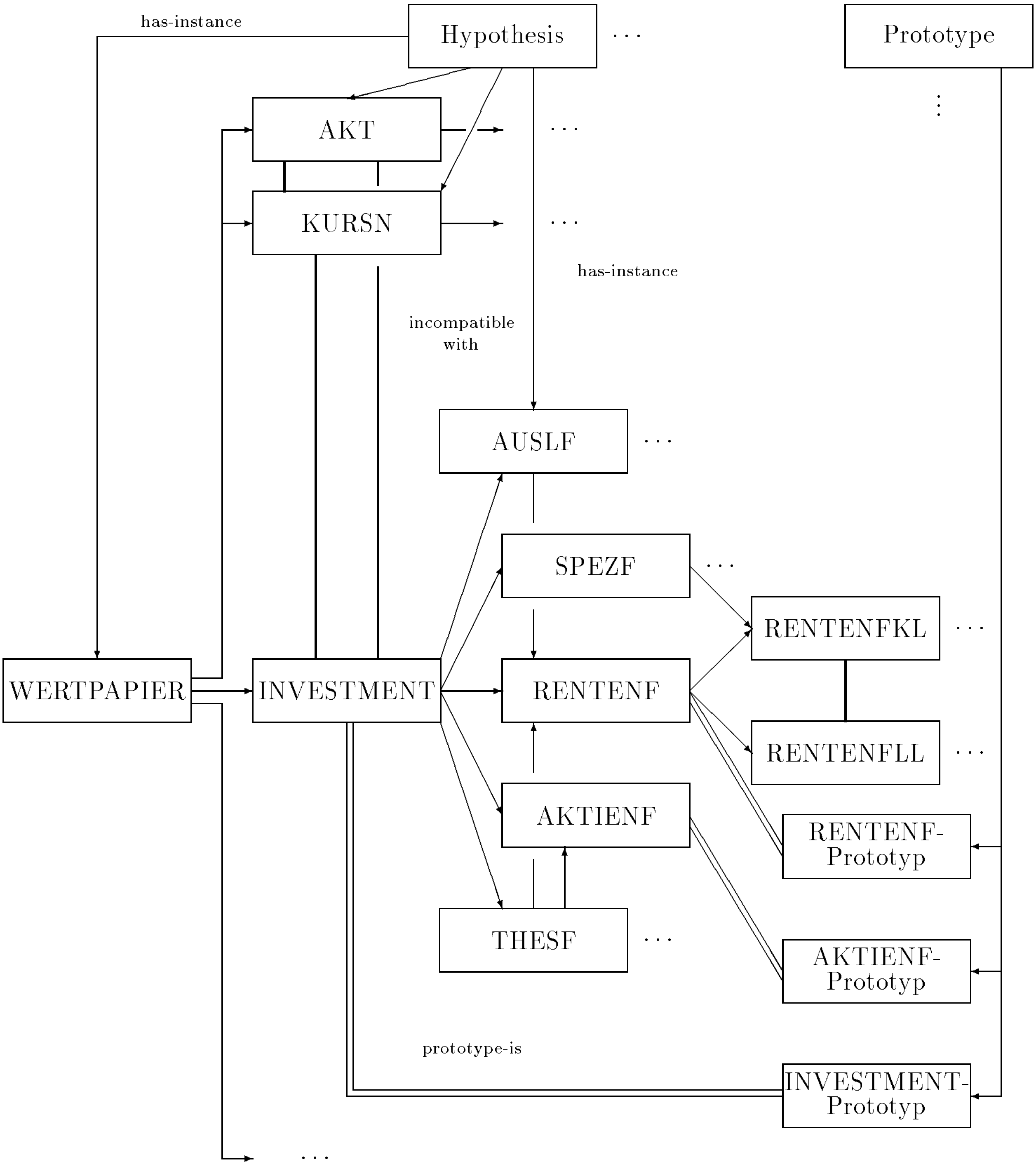


Abbildung 3.12: Ein semantisches Netz von Hypothesen und Prototypen

3.6.2 Prototypen

Die Prototypen in Donald sind Instanzen von Klassen, die Situationen abbilden.

Ein Beispiel aus Invest soll das verdeutlichen: Der Benutzer von Invest ist ein potentieller Wertpapierkäufer, ein Anleger. Die Klasse *Anleger* enthält die Slotdefinitionen, die nötig sind, um die näheren Umstände darzustellen, in denen sich der Benutzer befindet, unter dem Gesichtspunkt, daß er ein Anleger ist.

Zu Beginn einer Konsultation wird ein neuer Frame mit einem vom Benutzer gewählten Namen geschaffen. Dieser Name wird in den getriggerten Hypothesen in den CompareWith-Slot eingetragen. Zunächst enthält dieser Frame nur einen View auf die Klasse Anleger. Dieser View wird durch Antworten des Benutzers gefüllt und gibt einen ersten Eindruck der Benutzersituation wieder.

Prototypen sind Instanzen, die spezielle Situationen verkörpern. Zum Beispiel gibt es in Invest Prototypen, die ebenfalls einen View auf die Klasse Anleger haben und für bestimmte Anlagearten typische Situationen darstellen. So bilden zum Beispiel die Instanzen *AKT, RENTE* und *KURSN* prototypische Anleger für Aktien, Rentenpapiere und kursneutrale Anlagepapiere ab.

Die getriggerten Hypothesen bestimmen durch ihre Prototype-Slots, welche Prototypen mit dem Frame, der die Benutzersituation zeigt, ge*match*t werden. Die Größe der Ähnlichkeit zwischen Benutzer-Frame und den Prototyp-Frames ist entscheidend dafür, welche Hypothesen bestätigt bzw. verworfen werden. Abbildung 3.13 zeigt einen typischen Benutzer-Frame nach der ersten Runde einer Konsultation.

Abbildung 3.14 zeigt den Prototyp *KURSN*.

3.6.3 Der Datentyp HList

Mit den im Abschnitt 3.5 beschriebenen Datentypen liefert die Funktion *match* auf Slots entweder den Wert -1000 oder +1000.

```
(Instance Kunde-1
  (View Anleger with
        (AnlageDauer = mittel)
        (AnlageBetrag = mittel)
        (AnlageZweck = Vermoegensstreuung)
        (KundenVorstellung = Zinspapiere)
        (RenditeErwartung = normal)
        (RisikoBereitschaft = klein)
        (AnlageVerfuegbarkeit = nein)
        (KundenSteuerproblem = ja)))
```

Abbildung 3.13: Die Instanz Kunde-1

Bei der Entwicklung von Invest zeigte sich, daß das in vielen Situationen ein zu grobes Maß ist. Auch bei Slots sollte auf Ähnlichkeit und nicht auf völlige Übereinstimmung geprüft werden.

Ein Beispiel aus Invest soll das veranschaulichen: eine der Einstiegsfragen bei jeder Konsultation von Invest ist die Frage nach dem Zweck der geplanten Anlage in Wertpapieren. Die Antwortmöglichkeiten sind *Vermögensansammlung, Vermögensstreuung* und *Spekulation.*

Ist nun die Wertpapiergattung A besonders gut zum Ansparen von Vermögen geeignet, würde der Slot *AnlageZweck* des A-Prototypen den Wert *Vermögensansammlung* enthalten. Es ist aber nicht unwahrscheinlich, daß die Wertpapiere der Gruppe A auch zum Risikostreuen, zwar nicht optimal, aber gut geeignet sind. Die Benutzerantwort *Vermögensstreuung* würde aber ohne Rücksicht darauf beim Matchen der Slots -1000 ergeben.

Die Lösung, im A-Prototyp in diesem Slot mit *(Or Vermögensansammlung Vermögensstreuung)* beide Alternativen vorzusehen, würde dazu führen, daß bei beiden Antworten das Ergebnis des Matchens +1000 wäre. So kommt aber das Wissen, daß Wertpapiere der Gattung A zum Ansammeln besser geeignet sind als zum Streuen, nicht zum Ausdruck.

Für diese Fälle ist der Datentyp *HList* (Heuristic List) vorgesehen. Eine HList ist eine Liste von Paaren, deren erstes Element ein möglicher Slotinhalt ist

```
(Instance KURSN
 (LongName is "kursneutral")
 (View Anleger with
        (AnlageDauer ist
            (HList (ich weiss noch nicht . 350)
            (lang . 250)
            (mittel . 1000)
            (kurz . 1000))
            weight 2)
        (AnlageZweck ist
            (HList (Vermoegensansammlung . 1000)
            (Vermoegensstreuung . 500)
            (Spekulation . -500))
            weight 1)
        (KundenVorstellung ist
            (HList (nein . 250)
            (Optionen . -1000)
            (Investment . -500)
            (Zinspapiere . 1000)
            (Aktien . -1000))
            weight 3)
        (RenditeErwartung ist
            (HList (ich weiss noch nicht . 250)
            (ueberdurchschnittlich . -750)
            (normal . 1000))
            weight 2)
        (RisikoBereitschaft ist
            (HList (ich weiss noch nicht . 0)
            (ueberdurchschnittlich . -500)
            (normal . 500)
            (klein . 1000))
            weight 4)))
```

Abbildung 3.14: Der Prototyp KURSN

und deren zweites Element der korrespondierende Wert der Match-Funktion ist (vergleiche Abbildung 3.14 in Abschnitt 3.6.2).

Beim Matchen dieses Prototyps gegen den Frame Kunde-1 (siehe Abbildung 3.13) steuert der Slot *AnlageZweck* einfach gewichtete 500 Punkte bei.

Die numerischen Werte in den HLists sind mit einer groben Semantik versehen. Die neun Vielfachen von 250 (-1000, -750, ..., 750, 1000) repräsentieren die Einschätzungen "paßt überhaupt nicht" bis "paßt sehr gut".

Durch HLists wird das Matchen aussagefähiger. Sie definieren "Umgebungen" von Antworttupeln um den idealen Prototypen. Alle Tupel, die in einer solchen Umgebung liegen, gelten als zum Prototyp ähnlich. Mit HLists wird zum Ausdruck gebracht, daß eine Reihe von schwächeren Ähnlichkeiten ebenso gut auf einen Protoypen hinweisen können wie einige wenige starke Übereinstimmungen.

Schon die Antworten auf die Einstiegsfragen in Invest lassen circa 17000 verschiedene Antwortkombinationen zu. Die HLists machen es möglich diese Kombinationen mit einer relativ geringen Anzahl von Prototypen zu klassifizieren. Ein aus Gründen der graphischen Darstellbarkeit stark vereinfachtes Modell, in dem Anlageempfehlungen nur von der Anlagedauer, dem Anlagezweck und der Risikobereitschaft abhängig gemacht werden, zeigt Abbildung 3.15. Die darin angegebenen Zahlenwerte sind grob gerundete *ConfidenceFactors*.

HLists sind vor allem für Antwortauswahlen vorgesehen, die sich nicht auf einer Skala anordnen lassen. Bei Fragen nach Quantitäten (Dauer, Betrag, Rendite, Gewicht, Länge etc.) wurden schon spezielle Match-Funktionen implementiert. Zum Beispiel können Slots vom Typ *Integer* und *Real* mit der speziellen Match-Funktion *nummatch* verglichen werden.

Für zwei Zahlen x und y ist

$$\text{nummatch}(x, y) \;:=\; \begin{cases} 1000 & \text{falls } x = 0, y = 0 \\ 1000 * [1 - 2\frac{|x-y|}{|x|+|y|}] & \text{sonst} \end{cases}$$

Diese Funktion berücksichtigt nicht nur die Differenz der beiden Zahlen, sondern auch ihre Größenordnung. So ist nummatch(1,100) = -960 und nummatch(1001,1100) = +906.

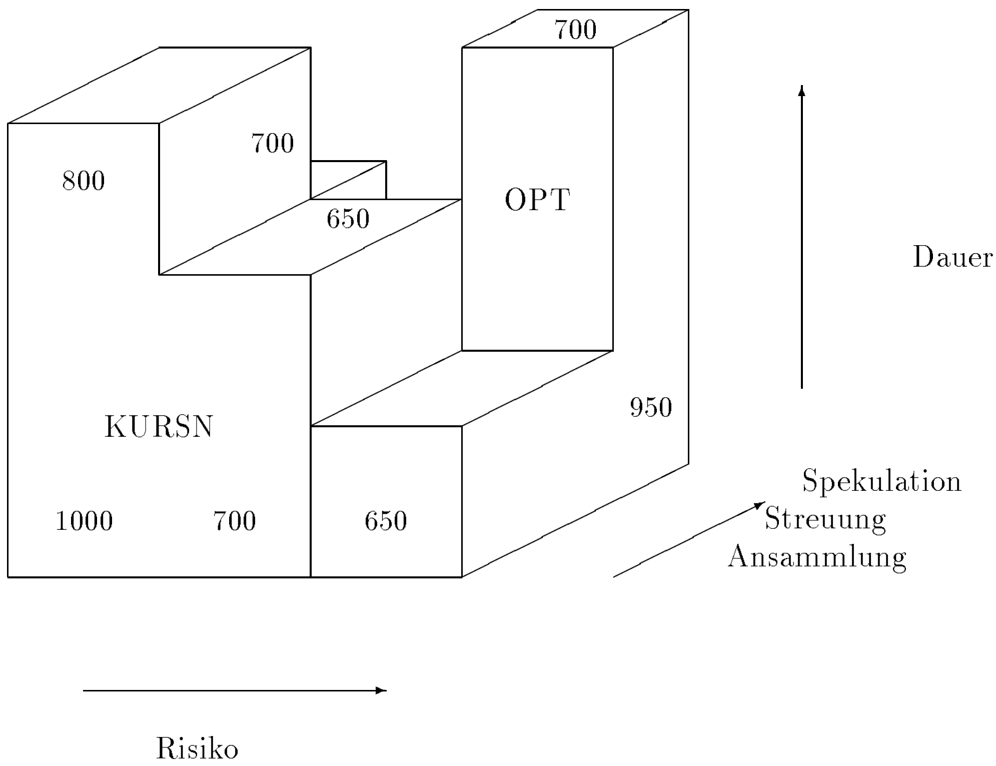


Abbildung 3.15: Umgebungen der Prototypen OPT und KURSN

Um Slots auf Ähnlichkeit zu prüfen gibt es also zwei Möglichkeiten. Entweder man definiert eine Match-Funktion speziell für Daten eines besonderen Typs, oder man benutzt HLists. HLists lassen sich als Spezialfall der ersten Möglichkeit auffassen, weil sie einer Wertetabelle für eine spezielle Match-Funktion entsprechen.

HLists sind vielleicht die weniger elegante Lösung, aber sie sind ein allgemein anwendbarer Formalismus, der die für das Programmieren einer speziellen Match-Funktion nötigen Lisp-Kenntnisse entbehrlich macht.

3.7 Die Synthese der Lösung

Im Abschnitt 3.6.1 wurde bereits erwähnt, daß Donald versucht, aus den einfachen bestätigten Hypothesen größere Gruppen aufzubauen. Dies ist einer der Punkte, in denen Donald sich von herkömmlichen Expertensystem-Shells zur Diagnose und Klassifikation unterscheidet, deren Ziel es ist, ein Objekt aus einer im voraus bekannten Menge zu finden (Fehlerdiagnose) oder ein Objekt einer im voraus bekannten Klasse zuzuordnen.

Donald konstruiert aus den Hypothesen-Bausteinen Lösungsvorschläge, wobei die Freiheit dieser Konstruktion nur durch das Wissen darüber, welche Hypothesen sich gegenseitig ausschließen, beschränkt wird. Die Überlegung, die zu dieser Konzeption geführt hat, war, daß es ökonomischer ist, nicht alle möglichen Kombinationen von Hypothesen als Objekte zu repräsentieren, sondern lieber das Wissen darüber, wie zulässige Kombinationen gebildet werden können, abzubilden.

So können zum Beispiel in Invest aus den circa 30 terminalen Hypothesen (i.e. den Hypothesen, die keine spezielleren Nachfolger mehr haben) die rund 100 wichtigsten Gruppen von Wertpapieren geformt werden.

In diesem Abschnitt wird beschrieben werden, wie der Aufbau von Lösungen vollzogen wird.

Eine Hypothese V heißt *Vorgänger* der Hypothese H, wenn sie im Slot *MoreGeneral* von H aufgeführt ist.

Eine Hypothese A heißt *Ahne* der Hypothese H, falls

1. A ein Vorgänger von H ist, oder falls
2. es einen Vorgänger von H gibt, dessen Ahne A ist.

Zwei Hypothesen H_1 und H_2 heißen *inkompatibel*, wenn

1. im Slot *IncompatibleWith* der Hypothese H_i die Hypothese H_j aufgeführt ist, wobei $i, j \in \{1, 2\}, i \neq j$; oder
2. wenn im *IncompatibleWith*-Slot eines Ahnen H_A von H_i eine Hypothese H_B genannt ist, die ein Ahne von H_j ist, $i, j \in \{1, 2\}, i \neq j$.

Ein (partieller) Lösungsvorschlag LV ist eine Liste, die

1. die Liste mit dem Namen der allgemeinsten Hypothese (der *Wurzelhypothese*) als einzigem Element enthält, und
2. die $n (n \geq 0)$ weitere Elemente enthält, die geordnete Paare (Listen aus zwei Elementen) von Namen bestätigter Hypothesen sind. Dabei ist der erste Name der Name des Paares und gibt an, für welche Hypothese das Paar steht. Der zweite Name ist der Name des Vorgängers, d.h. der Hypothese, als deren Spezialisierung die erste bestätigt wurde.
3. Jede Hypothese, die in einem Paar als zweites Element auftritt, muß einmal auch als erstes Element eines Paares auftreten (*Fundiertheit*).
4. Unter allen in LV vorkommenden Hypothesen dürfen nicht zwei auftreten, die inkompatibel sind (*Konsistenz*).

Eine Liste, die diesen Bedingungen genügt, heißt *fundierter, konsistenter, (partieller) Lösungsvorschlag*. Ein Lösungsvorschlag LV heißt *maximal-konsistent*, wenn es kein noch nicht in LV enthaltenes Paar $P = (H, V)$ aus einer bestätigten Hypothese H und ihrem Vorgänger V gibt, so daß P zu allen Paaren in LV kompatibel wäre.

Zu Beginn einer Konsultation von Donald besteht die Liste der partiellen Lösungsvorschläge nur aus einem Element. Das ist der Lösungsvorschlag, der als einziges Element die Liste enthält, die nur aus der Wurzelhypothese WH besteht. Diese gibt durch ihren *MoreSpecific*-Slot die Hypothesen

$NH_1, ..., NH_n$ an, die in der ersten Runde der Konsultation überprüft werden sollen. Aus den bestätigten Hypothesen $NH_{i_1}, ..., NH_{i_m}, m \leq n$, und ihren Vorgängern (in diesem Fall stets WH) werden die Paare

$$(NH_{i_1}, WH), \ldots, (NH_{i_m}, WH)$$

gebildet, die durch die partiellen Lösungsvorschläge erweitert werden sollen. Das geschieht dadurch, daß die Paare der Reihe nach mit dem *Propose*-Algorithmus bearbeitet werden.

Die Routine *Propose* nimmt als Argumente ein Paar $P = (H, V)$ aus einer bestätigten Hypothese H und ihrem Vorgänger V, sowie die Liste der partiellen Lösungsvorschäge LLV, soweit sie bisher aufgebaut worden ist.

Für jeden partiellen Lösungsvorschlag LV in LLV versucht Propose, LV um P zu erweitern. Taucht V in LV nicht auf, oder ist P schon in LV vertreten, geht Propose weiter zum nächsten noch nicht bearbeiteten Lösungsvorschlag.

Andernfalls bildet Propose aus allen in LV auftretenden Hypothesen, die zu H inkompatibel sind, die Konflikt-Menge CS und aus den übrigen die Differenz-Menge DS. Ist CS leer, wird P zum Lösungsvorschlag LV hinzugefügt. Im anderen Fall wird DS um P erweitert. Ist diese neue Liste fundiert und nicht bereits in LLV enthalten, so wird sie als weiterer Lösungsvorschlag zu LLV hinzugefügt.

Sind alle Paare bearbeitet worden, folgt ein weiterer Test: enthält ein Lösungsvorschlag LV ein Paar (H, V), für das kein Nachfolgepaar (N, H) in LV existiert, obwohl alle direkten Nachfolger N (d.h. die im *MoreSpecific*-Slot von H aufgeführten Hypothesen) schon geprüft worden sind, so wird LV aus LLV entfernt. So werden bestätigte Hypothesen, deren sämtliche Nachfolger bei der Überprüfung unbestätigt geblieben sind, von der Konstruktion der Lösungsvorschläge ausgeschlossen. Ebenso werden Lösungsvorschläge, die aufgrund von Inkompabilitäten nicht mehr erweitert werden können, aus der Liste der Lösungsvorschläge gelöscht.

So kann es sein, daß zu Beginn und in der Mitte einer Konsultation die Anzahl der Lösungsvorschläge anwächst und gegen Ende wieder abnimmt oder sogar 0 wird. Diese Fähigkeit, Arbeitshypothesen zu bilden, und sie auch wieder verwerfen zu können, diese sogenannte *Nicht-Monotonie*, ist eines der wichtigsten Konstruktionsmerkmale der Expertensystem-Shell Donald und wird noch Gegenstand eines folgenden Abschnitts sein.

Das *nicht-monotone Schließen*, d.h. die Fähigkeit eines formalen Systems, einmal gezogene Schlußfolgerungen zu revidieren, wird ebenfalls noch Gegenstand folgender Kapitel sein.

Der Propose-Algorithmus liefert nach jeder Konsultationsrunde eine Liste von maximal-konsistenten, fundierten (partiellen) Lösungsvorschlägen. Die Views, die in den *Proposition*-Slots der Hypothesen, die in einem Lösungsvorschlag auftreten, angeführt sind, werden zur externen Form je einer Instanz *Proposal*-n, $n \geq 1$, zusammengefaßt. Durch die Auswertung dieser Darstellung entsteht zu jedem Lösungsvorschlag ein neues Objekt, das eine vom Expertensystem vorgeschlagene Lösung ist.

3.8 Ein Beispieldialog

Dieser Abschnitt zeigt eine komplette Konsultation von Donald, bzw. Invest. Das Beispiel demonstriert das nicht-monotone Schlußfolgern, den schrittweisen Aufbau der Lösung, die Möglichkeit, sich Fragen und Lösungen erläutern zu lassen und anderes mehr. (Text hinter einem Semikolon wurde nachträglich als Erläuterung hinzugefügt.)

```
Wie heisst der Kunde?
> Kunde-1                                     ; Text hinter dem ">" sind
                                              ; Eingaben des Benutzers.
        **** Rangliste ****

No. Bestaetigt S  Hypothese                   ; S steht fuer "Status".
 1.  100%      +  Wertpapier                  ; "+" ("-") heisst: diese
                                              ; Hypothese wird momentan
                                              ; geglaubt (verworfen).

Konsultation>
Bitte waehlen Sie aus:
  (1)  rangliste             (8)  aendere Eintrag
  (2)  weiter                (9)  warum
  (3)  bestaetigt           (10)  warum nicht
```

```
  (4)  vorschlaege          (11)  bisherige Antworten
  (5)  kandidaten           (12)  neue Beratung
  (6)  bestaetige           (13)  ende
  (7)  verwerfe

> 2
--> weiter                             ; fahre in der Konsulta-
                                       ; tion fort.

Es folgt das Fuellen des Aspekts --Anleger-- fuer Kunde-1:

Wie lange moechten Sie Ihr Geld anlegen?
Bitte waehlen Sie aus:
  (1)  ich weiss noch nicht   (3)  mittel
  (2)  lang                   (4)  kurz

> k                                    ; als Antwort reicht eine
--> kurz                               ; Nummer oder ein eindeu-
                                       ; tiges Anfangsstueck.

Wie hoch ist der Betrag, den Sie anlegen wollen?
Bitte waehlen Sie aus:
  (1)  regelmaessig   (3)  mittel
  (2)  gross          (4)  klein

> k
--> klein

Zu welchem Zweck moechten Sie Ihr Geld anlegen?
Bitte waehlen Sie aus:
  (1)  spekulation            (3)  vermoegensansammlung
  (2)  vermoegensstreuung

> s
--> spekulation

Haben Sie schon eigene Vorstellungen bzgl. der Anlageform?
```

```
Bitte waehlen Sie aus:
  (1)  nein         (3)  investment    (5)  aktien
  (2)  optionen     (4)  zinspapiere

> o
--> optionen

Wie hoch ist Ihre Renditeerwartung?
Bitte waehlen Sie aus:
  (1)  ich weiss noch nicht    (3)  normal
  (2)  ueberdurchschnittlich

> ?                                     ; Gib weitere Erlaeute-
                                        ; rungen.

Bei der Angabe der Renditeerwartung sollte man
nicht vergessen, dass Rendite und Risiko stets
einander bedingen.  Eine hoehere  Rendite muss
immer mit Nachteilen bei der  Liquiditaet oder
beim Risiko erkauft werden.

Wie hoch ist Ihre Renditeerwartung?
Bitte waehlen Sie aus:
  (1)  ich weiss noch nicht    (3)  normal
  (2)  ueberdurchschnittlich

> u
--> ueberdurchschnittlich

Wie gross ist Ihre Risikobereitschaft?
Bitte waehlen Sie aus:
  (1)  ich weiss noch nicht    (3)  normal
  (2)  ueberdurchschnittlich   (4)  klein

> u
--> ueberdurchschnittlich
```

```
Muss das Geld waehrend der Anlagedauer verfuegbar sein?
Bitte waehlen Sie aus:
  (1)  ich weiss noch nicht   (3)  ja
  (2)  nein

> ?

Verfuegbar sind Geldanlagen im weitesten Sinne im-
mer, da man  schlimmstenfalls  die Anlage beleihen
kann. Verfuegbar soll hier heissen, dass die Anlage
liquidierbar  ist  unter (teilweisem) Verzicht auf
die angestrebten  Anlageziele,  aber ohne zusaetz-
liche Kosten wie Kreditzinsen.

Muss das Geld waehrend der Anlagedauer verfuegbar sein?
Bitte waehlen Sie aus:
  (1)  ich weiss noch nicht   (3)  ja
  (2)  nein

> 3
--> ja

Muessen steuerliche Aspekte schon jetzt beachtet werden?
Bitte waehlen Sie aus:
  (1)  nein   (2)  ja

> 1
--> nein
                                      ; nach einer Runde werden
        **** Rangliste ****           ; diese Hypothesen ge-
                                      ; glaubt bzw. verworfen.
No. Bestaetigt S  Hypothese
 1.  100%      +  auslaendisch
 2.   96%      +  Option/Optionsscheine
 3.   72%      +  Aktien
 4.   55%         Investmentfonds
 5.  -22%      -  Rentenpapier
```

```
 6.  -41%       -  kursneutral
 7.  -76%       -  Immobilienfonds


Konsultation>
Bitte waehlen Sie aus:
  (1)  rangliste            (8)  aendere Eintrag
  (2)  weiter               (9)  warum
  (3)  bestaetigt          (10)  warum nicht
  (4)  vorschlaege         (11)  bisherige Antworten
  (5)  kandidaten          (12)  neue Beratung
  (6)  bestaetige          (13)  ende
  (7)  verwerfe

> 9
--> warum                              ; erklaere mir, warum
                                       ; Du glaubst, dass ...
Warum welche Hypothese?
Bitte waehlen Sie aus:
  (1)  auslaendisch             (5)  Rentenpapier
  (2)  Option/Optionsscheine    (6)  kursneutral
  (3)  Aktien                   (7)  Immobilienfonds
  (4)  Investmentfonds          (8)  Wertpapier

> 3                                    ; ... Aktien in Frage
--> Aktien                             ; kommen.

--------------------------------------------------
Name:    Aktien
Konfidenz: 72.6%
--------------------------------------------------
--------------------------------------------------

Die Antwort auf die wichtige  Frage:
Wie lange moechten Sie Ihr Geld anlegen?
war: kurz
Das spricht sehr fuer  den Vorschlag
```

```
"Aktien"
(750 Punkte)
--------------------------------------------------
Soll ich weitere Gruende nennen? (J oder N) Ja
--------------------------------------------------

Die Antwort auf die  Frage:
Zu welchem Zweck moechten Sie Ihr Geld anlegen?
war: spekulation
Das spricht absolut fuer  den Vorschlag
"Aktien"
(1000 Punkte)
--------------------------------------------------
Soll ich weitere Gruende nennen? (J oder N) Nein


Konsultation>
Bitte waehlen Sie aus:
  (1)  rangliste              (8)  aendere Eintrag
  (2)  weiter                 (9)  warum
  (3)  bestaetigt            (10)  warum nicht
  (4)  vorschlaege           (11)  bisherige Antworten
  (5)  kandidaten            (12)  neue Beratung
  (6)  bestaetige            (13)  ende
  (7)  verwerfe

> 2
--> weiter

Es folgt das Fuellen des Aspekts --AKT-Anleger-- fuer Kunde-1:

In welchem Umfang (in DM) besitzen Sie schon Aktien?
Bitte waehlen Sie aus:
  (1)  keine                (3)  zw. 10 u. 50 Tsd.
  (2)  unter 10 Tsd.        (4)  ueber 50 Tsd.

> 3
```

```
--> zw. 10 u. 50 Tsd.

Besitzen Sie schon Aktien auslaendischer AG's?
Bitte waehlen Sie aus:
  (1)  ja    (2)  nein

> n
--> nein

Was ist/waere Ihr Ziel bei der Aktienanlage?
Bitte waehlen Sie aus:
  (1)  schneller begrenzter Kursgewinn
  (2)  mittelfr. bedeutender Kursgewinn
  (3)  vermoegensstreuung

> 1
--> schneller begrenzter Kursgewinn

Wieviel Zeit haben Sie, sich um Kurse etc.
zu kuemmern?
Bitte waehlen Sie aus:
  (1)  keine Zeit   (2)  wenig Zeit   (3)  viel Zeit

> w
--> wenig Zeit

Wie gross ist Ihre Risikofreudigkeit bzgl. Aktien?
Bitte waehlen Sie aus:
  (1)  klein   (2)  mittel   (3)  gross

> g
--> gross



Es folgt das Fuellen des Aspekts --OPT-Anleger-- fuer Kunde-1:
```

```
Wie schaetzen Sie die Kursentwicklung fuer die
naechste Zeit ein?
Bitte waehlen Sie aus:
  (1)  fallend          (3)  steigend
  (2)  konstant         (4)  ich weiss nicht

> 1
--> fallend

Wie gross ist Ihre Risikobereitschaft
bei Optionen oder Optionsscheinen?
Bitte waehlen Sie aus:
  (1)  klein    (2)  mittel   (3)  gross

> g
--> gross

Wie lange soll die Option bestehen?
Bitte waehlen Sie aus:
  (1)  wenige Monate   (2)  wenige Jahre

> 1
--> wenige Monate

        **** Rangliste ****

No. Bestaetigt S  Hypothese
 1.  100%      +  Option
 2.   45%         Standardwert
 3.   24%         Turnaround
 4.   20%         Optionsschein
 5.   18%      -  ausl. Aktien
 6.    3%      -  Wachstumswert


Konsultation>
Bitte waehlen Sie aus:
```

```
  (1)  rangliste             (8)  aendere Eintrag
  (2)  weiter                (9)  warum
  (3)  bestaetigt           (10)  warum nicht
  (4)  vorschlaege          (11)  bisherige Antworten
  (5)  kandidaten           (12)  neue Beratung
  (6)  bestaetige           (13)  ende
  (7)  verwerfe

> vor
--> vorschlaege

(instance proposal-1                   ; Die bestaetigten Hypo-
   (view proposal)                     ; thesen "auslaendisch"
   (view wertpapier with               ; und "Option/Options-
        (herkunft = ausland)           ; schein" haben Views zu
        (risiko = enorm))              ; diesem Proposal beige-
   (view boersengaengigeswertpapier)   ; tragen.
   (view terminpapier))


Soll ich mehr zeigen? (J oder N) Ja

(instance proposal-2                   ; Die bestaetigten Hypo-
   (view proposal)                     ; thesen "auslaendisch"
   (view wertpapier with               ; und "Aktien" haben
        (herkunft = ausland)           ; Views zu diesem Propo-
        (nominalwert = 50)             ; sal beigetragen.
        (boersengaengig = ja)
        (deckungsstockfaehig = nein)
        (lombardfaehig = nein)
        (muendelsicher = nein)
        (absicherung = sonstiges))
   (view aktie)
   (view boersengaengigeswertpapier))

Soll ich mehr zeigen? (J oder N) Ja
```

```
Konsultation>
Bitte waehlen Sie aus:
  (1)  rangliste            (8)  aendere Eintrag
  (2)  weiter               (9)  warum
  (3)  bestaetigt          (10)  warum nicht
  (4)  vorschlaege         (11)  bisherige Antworten
  (5)  kandidaten          (12)  neue Beratung
  (6)  bestaetige          (13)  ende
  (7)  verwerfe

> 10
--> warum nicht                      ; erklaere mir, warum
                                     ; Du glaubst, dass ...
Warum nicht welche Hypothese?
Bitte waehlen Sie aus:
  (1)  Option                 (8)  Option/Optionsscheine
  (2)  Standardwert           (9)  auslaendisch
  (3)  Turnaround            (10)  Wertpapier
  (4)  Optionsschein         (11)  Investmentfonds
  (5)  ausl. Aktien          (12)  Immobilienfonds
  (6)  Wachstumswert         (13)  kursneutral
  (7)  Aktien                (14)  Rentenpapier

> 4
--> Optionsschein                    ; ... Optionsscheine
                                     ; nicht in Frage kommen.

--------------------------------------------------
Name:    Optionsschein
Konfidenz: 20.8%
--------------------------------------------------
--------------------------------------------------

Die Antwort auf die entscheidene  Frage:
Wie schaetzen Sie die Kursentwicklung fuer die
naechste Zeit ein?
```

```
war: fallend
Das spricht absolut gegen  den Vorschlag
"Optionsschein"
(-1000 Punkte)
--------------------------------------------------
Soll ich weitere Gruende nennen? (J oder N) Nein


Konsultation>
Bitte waehlen Sie aus:
  (1)  rangliste              (8)  aendere Eintrag
  (2)  weiter                 (9)  warum
  (3)  bestaetigt            (10)  warum nicht
  (4)  vorschlaege           (11)  bisherige Antworten
  (5)  kandidaten            (12)  neue Beratung
  (6)  bestaetige            (13)  ende
  (7)  verwerfe

> 10                                        ; ausl. Aktien sahen
--> warum nicht                             ; nach der ersten Runde
                                            ; doch vielversprechend
Warum nicht welche Hypothese?               ; aus.
Bitte waehlen Sie aus:
  (1)  Option                   (8)  Option/Optionsscheine
  (2)  Standardwert             (9)  auslaendisch
  (3)  Turnaround              (10)  Wertpapier
  (4)  Optionsschein           (11)  Investmentfonds
  (5)  ausl. Aktien            (12)  Immobilienfonds
  (6)  Wachstumswert           (13)  kursneutral
  (7)  Aktien                  (14)  Rentenpapier

> 5
--> ausl. Aktien
--------------------------------------------------
Name:    ausl. Aktien
Konfidenz: 18.6%
--------------------------------------------------
```

```
--------------------------------------------------

Die Antwort auf die entscheidene  Frage:
Was ist/waere Ihr Ziel bei der Aktienanlage?
war: schneller begrenzter Kursgewinn
Das spricht sehr gegen  den Vorschlag
"ausl. Aktien"
(-800 Punkte)
--------------------------------------------------
Soll ich weitere Gruende nennen? (J oder N) Ja


Konsultation>
Bitte waehlen Sie aus:
  (1)  rangliste             (8)  aendere Eintrag
  (2)  weiter                (9)  warum
  (3)  bestaetigt           (10)  warum nicht
  (4)  vorschlaege          (11)  bisherige Antworten
  (5)  kandidaten           (12)  neue Beratung
  (6)  bestaetige           (13)  ende
  (7)  verwerfe

> 2
--> weiter

        **** Rangliste ****             ; die Rangliste bleibt
                                        ; leer: das Ende der
No. Bestaetigt S  Hypothese             ; Hypothesenhierarchie
                                        ; ist erreicht.

Konsultation>
Bitte waehlen Sie aus:
  (1)  rangliste             (8)  aendere Eintrag
  (2)  weiter                (9)  warum
  (3)  bestaetigt           (10)  warum nicht
  (4)  vorschlaege          (11)  bisherige Antworten
  (5)  kandidaten           (12)  neue Beratung
```

```
  (6)  bestaetige           (13)  ende
  (7)  verwerfe

> vo
--> vorschlaege

(instance proposal-1                    ; Die Hypothesen "aus-
   (view proposal)                      ; laendisch"und "Aktien"
   (view wertpapier with                ; wurden revidiert. Die
        (herkunft = inland)             ; bestaetigten Hypothesen
        (risiko = enorm)                ; "Option" und "Option/Op-
        (boersengaengig = ja))          ; tionsschein" haben Views
   (view boersengaengigeswertpapier)    ; zu diesem Proposal bei-
   (view terminpapier)                  ; getragen.
   (view optionen))

Soll ich mehr zeigen? (J oder N) Ja


Konsultation>
Bitte waehlen Sie aus:
  (1)  rangliste             (8)  aendere Eintrag
  (2)  weiter                (9)  warum
  (3)  bestaetigt           (10)  warum nicht
  (4)  vorschlaege          (11)  bisherige Antworten
  (5)  kandidaten           (12)  neue Beratung
  (6)  bestaetige           (13)  ende
  (7)  verwerfe

> 13
--> ende

Konsultation beendet
```

3.9 Nicht-monotones Schließen in Donald

In diesem Abschnitt wird das nicht-monotone Vorgehen der Expertensystem-Shell Donald noch einmal näher untersucht.

Das nicht-monotone Verhalten beruht auf im Programm berücksichtigten Heuristiken. Somit existiert keine formale Syntax, nach der dieses Schließen stattfände, abgesehen von der formalen Darstellung in der Programmiersprache. Zum Verständnis dieses Prozesses wäre eine formale Semantik wünschenswert, und zur Rechtfertigung der Schlüsse eine dazu passende Syntax. In den folgenden Kapiteln wird, basierend auf einer von Dov Gabbay [Gab82] vorgeschlagenen Semantik, die Logik J4 definiert, die als Grundlage einer nicht-monotonen Logik dienen soll.

Unter Vorwegnahme der dort formal definierten Begriffe soll an dieser Stelle die Nicht-Monotonie von Donald (eher informell) beschrieben werden. Die Problematik inkompatibler Lösungen soll dabei der Einfachheit halber ausgelassen werden. Wie im Invest-Beispiel soll eine Menge von Hypothesen als Baum angeordnet sein (siehe Abbildung 3.16).

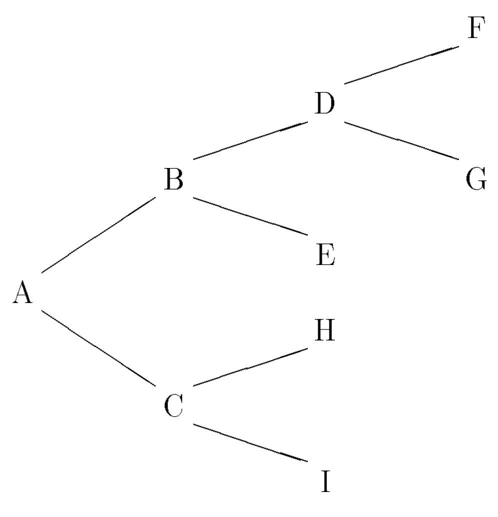


Abbildung 3.16: Hierarchie von Hypothesen

Diese Baumstruktur wird durch die folgenden Axiome repräsentiert:

1. $A \to B \vee C$
2. $B \to D \vee E$
3. $C \to H \vee I$
4. $D \to F \vee G$

Das bedeutet z.B.: wenn die Hypothese A bewiesen ist, ist B oder C bewiesen.

Als neue Bezeichnung dafür, daß eine Hypothese A "geglaubt" wird, wird $\Diamond A$ verwendet. Syntaktisch steht $\Diamond A$ als Abkürzung dafür, daß das Matchen des (oder der) Prototypen für A mit dem Vergleichs-Frame einen hinreichend großen Wert ergeben hat. $\Diamond A$ soll deshalb gelesen werden als: "es ist widerspruchsfrei, A anzunehmen".

Zu den oben angeführten Axiomen kommt noch für jeden Knoten das Axiom $\Diamond K \to K$ hinzu. Diese Axiome entsprechen in etwa den *normal defaults* in Reiters *default reasoning* [Rei80].

Eine Formel A soll *nicht-monoton herleitbar* heißen (in Zeichen $\mid\!\sim A$), wenn sie aus den Axiomen unter Rückgriff auf Annahmen der Form $\Diamond B$ bewiesen werden kann. Dabei müssen die Annahmen mit dem bisher Bewiesenen konsistent sein.

Zu Beginn einer Donald-Sitzung (zum Zeitpunkt t_0) wird nur die Start-Hypothese geglaubt:

$$t_0 : \Diamond A$$

Wegen $\Diamond A \to A, \Diamond A \vdash A$ gilt also $\mid\!\sim A$.

Zum nächsten Zeitpunkt (t_1) seien auch B und C durch *match* bestätigt worden:

$$t_1 : \Diamond B, \Diamond C \Longrightarrow \mid\!\sim B \text{ und } \mid\!\sim C$$

Zum Zeitpunkt t_2 bestätigt *match* die Hypothese D, lehnt aber E, H und I ab:

$$t_2 : \Diamond D, \neg\Diamond E, \neg\Diamond H, \neg\Diamond I$$

Aus $\neg\Diamond K$ folgt $\neg K$ (was leicht zu beweisen ist, wenn die Syntax von J4 erst einmal verfügbar ist). Also gilt:

$$\neg H \wedge \neg I \rightarrow \neg C \text{ (Kontraposition von Axiom 3)}$$

und damit $\vdash \neg C$. Dadurch wird zum Zeitpunkt t_2 die Anwendung der Annahme $\Diamond C$ blockiert: sie ist nicht mehr widerspruchsfrei. Jetzt ist C nicht mehr nicht-monoton herleitbar:

$$\not\mid\!\sim C$$

Der nächste Schritt ergebe

$$t_3 : \Diamond F, \neg\Diamond G, \text{ also } \mid\!\sim F, \not\mid\!\sim G$$

Zum Zeitpunkt t_3 gilt deshalb:

$$\mid\!\sim A \wedge B \wedge D \wedge F$$

Dieses Beispiel demonstriert, wie durch im Laufe der Zeit anwachsendes Wissen die Rücknahme ehemals gültiger Schlüsse erzwungen wird.

Kapitel 4

Einführung in das nicht-monotone Schließen

4.1 Einleitung

Die klassische Prädikatenlogik (PL) und die intuitionistische Logik (J) sind monotone Logiken. Das bedeutet, daß, was immer aus einer Voraussetzung A gefolgert werden kann, wahr bleibt, wenn man neue Voraussetzungen zu A hinzufügt.

Im Gegensatz zu diesen formalen Systemen gibt der Alltag häufig Beispiele für nicht-monotones Schlußfolgern.

Ein Beispiel soll einen Eindruck davon vermitteln, worum es sich beim nicht-monotonen Schließen handelt [Gab82]:

Das World Travel Agency (WTA) bietet Flüge von New York via Paris nach Stuttgart an. Jeden Sonntag startet in New York eine Reisegruppe. Manche Passagiere steigen in Paris aus, die übrigen in Stuttgart. Das Stuttgarter WTA-Büro arrangiert die Hotelreservierungen und Ausflüge in Europa.

Um 12 Uhr Stuttgarter Zeit, drei Stunden nach dem planmäßigen Start einer WTA-Reisegruppe in New York, erfährt das Stuttgarter Büro, daß Terroristen den Pariser Zielflughafen überfallen haben und daß alle Flüge mit Ziel Paris umgeleitet werden. Alle Telefonleitungen nach New York sind besetzt.

So muß das Stuttgarter Büro entscheiden, was mit der WTA-Reisegruppe zu tun ist, ohne mit dem New Yorker Büro Kontakt aufnehmen zu können.

Aufgrund der üblichen Praxis des WTA kommt man zu dem Schluß, daß das Flugzeug nach London umgeleitet werden wird, daß die Passagiere mit Ziel Paris in London übernachten werden und daß die übrigen am Abend aus London kommend in Stuttgart eintreffen werden.

Zusammengefaßt sehen die Überlegungen des Stuttgarter Büros um 12 Uhr so aus:

1. Folgendes ist um 12 Uhr als wahr erkannt:
   (a) A = die WTA-Maschine ist in New York gestartet
   (b) B = es hat ein Überfall auf den Pariser Flughafen stattgefunden
2. Aufgrund der üblichen Praxis und ohne Hinweis auf das Gegenteil wird um 12 Uhr angenommen, daß

   C = der Flug über London umgeleitet wird.
3. Schlußfolgerung um 12 Uhr

   D = die Passagiere mit Ziel Stuttgart werden heute abend in Stuttgart eintreffen

Zwei Stunden später, um 14 Uhr, kann das Stuttgarter Büro Kontakt mit New York aufnehmen und erfährt, daß sich der Start der Maschine um 2 1/2 Stunden verspätet hat, weil noch eine große Gruppe von Paris-Touristen mitfliegen wollte. Das New Yorker Büro weiß nicht, was der Flugkapitän entschieden hat, als er schon in der Luft von dem Terroristenüberfall gehört hat.

Wiederum aufgrund der üblichen Praxis nimmt man nun in Stuttgart an, daß der Kapitän das kurze Stück nach New York zurückgeflogen ist, damit die Passagiere zu Hause übernachten können.

Zusammengefaßt sehen die Überlegungen des Stuttgarter Büros um 14 Uhr so aus:

1. Folgendes ist um 14 Uhr als wahr erkannt:

   (a) A = die WTA-Maschine ist in New York gestartet
   (b) B = es hat ein Überfall auf den Pariser Flughafen stattgefunden
   (c) B' = der Start in New York hat sich verspätet. Eine große Gruppe von Paris-Touristen ist an Bord

2. Aufgrund der üblichen Praxis und ohne Hinweis auf das Gegenteil wird um 14 Uhr angenommen, daß

   C' = das Flugzeug nach New York zurückkehrt

3. Schlußfolgerung um 14 Uhr

   $\neg D$ = die Passagiere mit Ziel Stuttgart werden heute abend nicht in Stuttgart eintreffen

Um dieses Beispiel formal fassen zu können, benötigt man

- einen Herleitungsbegriff: $\vdash$
- eine Formalisierung von: "Man kann annehmen, daß etwas stimmt, weil es üblicherweise so ist und nichts Gegenteiliges bekannt ist": M
- eine Definition dafür, was eine nicht-monotone Herleitung ist: $\mid\!\sim$

McDermott und Doyle [McD80] fügen M als neues Zeichen zur zugrundeliegenden formalen Sprache hinzu. Eine Formel MA soll gelesen werden als: "es führt nicht zu einem Widerspruch, wenn man A als wahr annimmt".

Bei Berücksichtigung dieser Schreibweisen erhält man:

1. Um 12 Uhr:

   (a) Bekannt sind A und B. C wird angenommen.
   (b) $A \wedge B \wedge \mathrm{M}C \vdash D$
   (c) $A \wedge B \mid\!\sim D$

2. Um 14 Uhr:

   (a) Bekannt sind A, B und B'. C' wird angenommen.

(b) $A \wedge B \wedge B' \wedge \mathrm{M}C' \vdash \neg D$

(c) $A \wedge B \wedge B' \mid\!\sim \neg D$

Daraus folgt, daß $\mid\!\sim$ sich nicht-monoton verhält.

4.2 McDermott und Doyles Logik

Die nicht-monotone Logik von McDermott und Doyle basiert auf der klassischen Prädikatenlogik und einer Fixpunktkonstruktion.

Ausgangspunkt ist die Sprache $\mathcal{L}$ aus Aussagezeichen, Konstanten, Variablen und Prädikatszeichen. Dazu kommen Zeichen für die üblichen logischen Verknüpfungen, sowie das schon erwähnte M.

Aussagezeichen oder Zeichenketten der Form $g(x_1, ..., x_n)$ mit einem Prädikatszeichen g und Variablen oder Konstanten $x_1, ..., x_n$ heißen atomare Formeln.

Die Formeln sind atomare Formeln oder für Formeln A und B und eine Variable x Ausdrücke der Form:

$\mathrm{M}A$, $\neg A$, $A \rightarrow B$ und $\forall x A$

Dazu kommen die üblichen Abkürzungen:

$A \wedge B, A \vee B, \exists x A$.

Die Axiome sind:

1. $A \rightarrow (B \rightarrow A)$
2. $(A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$
3. $(\neg B \rightarrow \neg A) \rightarrow ((\neg B \rightarrow A) \rightarrow B)$
4. $\forall x A(x) \rightarrow A(t)$, t für x in A einsetzbar.
5. $\forall x(A \rightarrow B) \rightarrow (A \rightarrow \forall x B(x))$, falls A kein freies Auftreten von x enthält

Die Schlußregeln sind:

1. Modus Ponens: aus A und $A \to B$ schließe auf B (MP)
2. Verallgemeinerung: aus A schließe auf $\forall x A(x)$ (V)

Wenn S eine Menge von Formeln ist, und A folgt aus S und den Axiomen der Theorie T durch (MP) und (V), schreibt man

$S \vdash_T A$

Statt $S \vdash_{PL} A$ schreibt man nur $S \vdash A$.

Definition 4.2.1 *$Th(S) := \{A \mid S \vdash A\}$ heißt die Menge der Theoreme von S.*

Der Operator Th hat die folgenden Eigenschaften:

1. $T \subseteq Th(T)$
2. Aus $T1 \subseteq T2$ folgt $Th(T1) \subseteq Th(T2)$
3. $Th(Th(T)) = Th(T)$ (Idempotenz)

Man kann Punkt 3 als Fixpunktgleichung auffassen. Sie sagt aus, daß die Menge der Theoreme, die monoton aus einer Theorie herleitbar sind, ein Fixpunkt des Operators ist, der den Abschluß einer Menge von Formeln bezüglich der monotonen Herleitungsregeln berechnet. $Th(T)$ ist der kleinste Fixpunkt dieses Prozeßes.

Um mit nicht-monotoner Logik arbeiten zu können, braucht man eine neue Schlußregel. Sie sollte ungefähr aussagen, daß eine Formel A geglaubt werden kann, wenn $\neg A$ nicht herleitbar ist. Die Regel

- Wenn nicht $\vdash_T \neg A$, dann $\vdash_T \mathrm{M}A$

ist jedoch nicht brauchbar, weil Paradoxa auftreten können: es ist denkbar, daß $\mathrm{M}A \to \neg A$ in T herleitbar wäre; dann ist $\neg A$ herleitbar, wenn es nicht herleitbar ist.

Definition 4.2.2 $\vdash$ *soll fortan "monoton herleitbar" bedeuten.*

Definition 4.2.3 *Für jede Theorie T erster Ordnung und jede Menge S von Formeln aus $\mathcal{L}$ sei*

- $NM_T(S) := Th(T \cup An_T(S))$ *wobei* $An_T(S)$ *die Menge der Annahmen aus S ist:*
- $An_T(S) := \{\mathrm{M}A \mid A \in \mathcal{L}, \neg A \notin S\} \setminus Th(T)$

Theoreme von T der Form MA gelten nicht als Annahmen. NM nimmt als Argumente eine Theorie T und eine Menge S und berechnet daraus eine neue Menge, die $Th(T)$ und alle Folgerungen aus der erweiterten Menge von Axiomen und Annahmen. Diese erweiterte Menge ist die ganze ursprüngliche Theorie zusammen mit den Annahmen, die nicht von S ausgeschlossen werden.

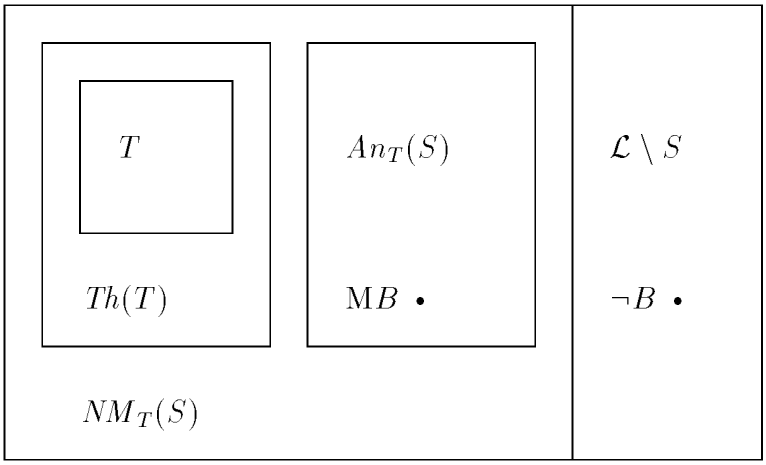
Es wäre wünschenswert $TH(T)$, die Menge aller nicht-monoton herleitbaren Theoreme in Analogie zum monotonen Fall als den kleinsten Fixpunkt des Operators NM zu definieren. Ein Fixpunkt $NM_T(S) = S$ ist in Abbildung 4.1 dargestellt.

Leider ist im allgemeinen die Existenz eines Fixpunktes nicht zwingend. Wenn dennoch Fixpunkte von NM existieren, muß nicht notwendig ein kleinster Fixpunkt existieren.

Ein Beispiel dafür ist McDermotts und Doyles Theorie $T1$.

$$T1 := PL \cup \{\mathrm{M}C \to \neg D, \mathrm{M}D \to \neg C\}$$

$T1$ hat zwei Fixpunkte $F1$ und $F2$. $F1$ enthält $\neg C$ und kann deshalb nicht MC und auch nicht $\neg D$ enthalten. $F2$ enthält $\neg D$, aber nicht MD und nicht $\neg C$. Weder der Schnitt von $F1$ und $F2$ noch die Vereinigung von $F1$ und $F2$ sind Fixpunkte von $NM(T1)$. Da weder $\neg C$ noch $\neg D$ im Schnitt von $F1$ und $F2$ liegen, sind MC und MD in $NM_{T1}(F1 \cap F2)$ und damit auch $\neg C$ und $\neg D$. Also ist $F1 \cap F2 \neq NM_{T1}(F1 \cap F2)$. Sowohl $\neg C$, als auch $\neg D$ sind in $NM_{T1}(F1 \cup F2)$. Das bedeutet, daß die Anwendung von NM auf $F1 \cup F2$ eine kleinere Menge ergibt.



$$T \subseteq Th(T)$$

$$Th(T) \cup An_T(S) \subseteq NM_T(S)$$
$$Th(T) \cap An_T(S) = \emptyset$$
$$(\mathcal{L} \cap S) \setminus NM_T(S) = \emptyset$$

Abbildung 4.1: Der Fixpunkt $NM_T(S) = S$

Ein zweites Beispiel ist die Theorie $T2 := PL \cup \{\mathrm{M}C \to \neg C\}$. $T2$ hat gar keinen Fixpunkt, da jede Anwendung auf irgendeine Menge S abwechselnd neue Mengen erzeugt, die entweder $\mathrm{M}C$ und $\neg C$ oder keinen von beiden enthalten.

Definition 4.2.4

$$TH(T) := \bigcap(\{\mathcal{L}\} \cup \{S \mid NM_T(S) = S\})$$

Die nicht-monoton beweisbaren Formeln sind der Schnitt aller Fixpunkte von NM oder die ganze Sprache, wenn keine Fixpunkte existieren.

Davis beschreibt die Menge $TH(T)$ (ohne Beweis der Äquivalenz) wie folgt [Dav80]:

Definition 4.2.5 *Es sei $F = \{A_i \mid i = 1, 2, ...\}$ eine Aufzählung der Formeln der gegebenen Theorie (z.B. $T1$). S sei eine Menge von Prämissen. Dann setze:*

$$\begin{array}{lcl} S_0 & := & S \\ S_{i+1} & := & \begin{cases} F & \textit{falls } \exists j A_j \in F, \\ & \mathrm{M}A_j \in S_i \\ & \textit{und } S_i \vdash \neg A_j \\ S_i \cup \{\mathrm{M}A_i\} & \textit{falls } S_i \cup \{A_i\} \\ & \textit{konsistent ist} \\ S_i & \textit{sonst} \end{cases} \end{array}$$

Schließlich sei $S^* := \bigcup S_i$.

S^* hängt dabei von der Reihenfolge der Aufzählung ab. Nicht-monoton herleitbar ($\mid\!\sim$) heißen bei McDermott und Doyle die Formeln aus $TH(T)$. Davis definiert $S \mid\!\sim B \Longleftrightarrow S^* \vdash B$ für alle Aufzählungen von F.

Mit dieser Definition ist weder $\mathrm{M}C$ noch $\mathrm{M}D$ ein Theorem von $T1$, aber $\mathrm{M}C \vee \mathrm{M}D \in TH(T1)$.

Eine Konsequenz dieser Definition von nicht-montoner Beweisbarkeit ist, daß das Deduktionstheorem für $\mid\!\sim$ nicht gilt.

Die nicht-montone Herleitbarkeit einer Formel beruht darauf, daß alle zur Verfügung stehenden Axiome zur Herleitung betrachtet wurden. Wäre das Deduktionstheorem gültig, würde es gültige Implikationen produzieren, die unabhängig von neu hinzugefügten Axiomen wären, die eine Wirkung auf die Herleitung haben könnten.

McDermott und Doyle beschreiben drei Probleme, die ihre hier geschilderte nicht-monotone Logik aufwirft:

Bemerkung 4.2.6 *1. $T13 := PL \cup \{\mathrm{M}C \to D, \neg D\}$ ist inkonsistent, da $\neg C$ nicht in $T13$ herleitbar ist.*

2. $\mathrm{M}C$ folgt nicht aus $\mathrm{M}(C \wedge D)$.

3. $T14 := PL \cup \{\mathrm{M}C, \neg C\}$ ist nicht inkonsistent.

4.3 Die Semantik

Zur Lösung der drei angesprochenen Probleme ist es notwendig, daß in der zu konstruierenden Logik die folgenden Regeln gelten:

1. $$\frac{\mathrm{M}A \to B \quad \neg B}{\neg A}$$

   Für $B = \mathrm{M}A$ heißt das
   $$\frac{\neg \mathrm{M}A}{\neg A}$$

2. Die fehlende Distributivität wird durch die zweite Regel gesichert:
   $$\frac{\mathrm{M}(A \wedge B)}{\mathrm{M}A}$$

3. Drittens muß erzwungen werden, daß $\mathrm{M}A \wedge \neg A$ inkonsistent ist, was von der Regel
   $$\frac{\neg A}{\neg \mathrm{M}A}$$
   geleistet wird.

In der klassischen Prädikatenlogik bedeuteten die erste und dritte Regel zusammen, daß A und $\mathrm{M}A$ äquivalent sind. Somit ist die klassische Prädikatenlogik als Grundlage der im folgenden beschriebenen nicht-monotonen Logik ausgeschlossen.

Bei der Konstruktion des nicht-monotonen Systems soll eine zeitliche Abfolge von Zuständen "wachsenden Wissens" modelliert werden.

Die Sprache L umfasse

1. für jedes n abzählbar viele n-stellige Prädikatszeichen $P^n, Q^n, \ldots$
2. abzählbar viele Variablen $x, y, z, \ldots$
3. unendlich viele Konstanten $c, d, \ldots$
4. Junktoren $(\neg, \wedge, \vee, \rightarrow)$
5. den gewohnten Satz von Quantoren $(\forall, \exists)$ sowie
6. den neuen einstelligen Junktor M.

Primformeln und Formeln sind wie gewöhnlich definiert. Für M ist intuitiv an die Regel $w \not\Vdash \neg A \Longrightarrow w \Vdash \mathrm{M}A$ gedacht. Das soll heißen, daß es statthaft ist, A anzunehmen, wenn zum Zeitpunkt w nicht $\neg A$ gilt (nicht erzwungen wird).

Definition 4.3.1 $(T, \leq, D, \bar{D})$ *heißt J4-Kripke-Rahmen, wenn*

1. *T eine nicht-leere Menge ist,*
2. *$\leq$ eine reflexive, transitive Relation auf T ist,*
3. *D eine nicht-leere Menge ist und*
4. *$\bar{D} : T \longrightarrow \wp(D)$ ist eine Abbildung von D auf nicht-leere Teilmengen von D, so daß gilt:*
$$D = \bigcup_{w \in T} \bar{D}(w)$$
und $\bar{D}(w) \subseteq \bar{D}(v)$, wenn $w \leq v$.

Definition 4.3.2 *Sei $(T, \leq, D, \bar{D})$ ein J4-Kripke-Rahmen. Sei $\Vdash$ eine Relation zwischen den Elementen von T und den geschlossenen Formeln von L. Dann ist $(T, \leq, D, \bar{D}, \Vdash)$ ein J4-Kripke-Modell, wenn für alle $w \in T$,*

1. $w \Vdash P$ *und* $w' \geq w \Longrightarrow w' \Vdash P$, *für Primformeln* P
2. $w \Vdash A \wedge B \Longleftrightarrow w \Vdash A$ *und* $w \Vdash B$
3. $w \Vdash A \vee B \Longleftrightarrow w \Vdash A$ *oder* $w \Vdash B$
4. $w \Vdash \neg A \Longleftrightarrow \forall w' \geq w\, w' \nVdash A$
5. $w \Vdash A \rightarrow B \Longleftrightarrow \forall w' \geq w\, (w' \Vdash A \Longrightarrow w' \Vdash B)$
6. $w \Vdash \mathrm{M}A \Longleftrightarrow \exists w' \geq w\, w' \Vdash A$
7. $w \Vdash \exists x F(x) \Longleftrightarrow \exists c \in \bar{D}(w)\, w \Vdash F(c)$
8. $w \Vdash \forall x F(x) \Longleftrightarrow \forall w' \geq w\, \forall c \in \bar{D}(w')\, w' \Vdash F(c)$

Bemerkung 4.3.3 *Die Relation $w \Vdash A$ wird gelesen als: w erzwingt A.*

Definition 4.3.4 *Es sei $\mathcal{A} = (T, \leq, D, \bar{D}, \Vdash)$. Eine Formel C gilt in diesem Kripke-Modell $\mathcal{A}$ (Schreibweise: $\mathcal{A} \models C$), wenn für alle w aus T $w \Vdash C$ gilt. $A_1, \ldots, A_n \models B$ soll heißen, daß B in allen Modellen gilt, in denen $A_1, \ldots, A_n$ gelten.*

Lemma 4.3.5 *Für* M*-freie Formeln C folgt aus $\alpha \Vdash C$, daß für alle $\beta \geq \alpha$ $\beta \Vdash C$ gilt.*

Beweis: Induktion nach dem Aufbau von C.

Bemerkung 4.3.6 *Weitere Anforderungen an die $\leq$-Ordnung von $(T, \leq)$ führen zu unterschiedlichen Semantiken und induzieren dadurch verschiedene syntaktische Systeme.*

Gabbay gibt einige Beispiele für Regeln und Schemata, die bezüglich dieser Semantik korrekt sind [Gab82, S. 266]:

Beispiele 4.3.7 *1.* $\Vdash \mathrm{M}A \vee \neg A$

2. $\Vdash \neg A \Longleftrightarrow \Vdash \neg \mathrm{M}A$

3. $\Vdash \mathrm{M}A \rightarrow B \Longleftrightarrow \Vdash \neg A \vee B$

4. $\Vdash \mathrm{M}A \rightarrow \neg A \Longleftrightarrow \Vdash \neg A$

5. Wenn $A \wedge \mathrm{M}B \Vdash C$ *gilt, kann es trotzdem sein, daß sich* $\neg \mathrm{M}B$ *herausstellt, so daß auch* $A \wedge \neg B \nVdash C$ *möglich ist.*

Lemma 4.3.8 *Bezüglich der in 4.3.2 angegebenen Semantik gelten die Regeln*

$$\frac{\neg \mathrm{M}C}{\neg C}$$

und

$$\frac{\mathrm{M}(A \wedge B)}{\mathrm{M}A}$$

Hingegen ist

$$\mathrm{M}C \wedge \neg C$$

ungültig.

Beweis:

1.

$$\begin{array}{rcl} \alpha \Vdash \neg \mathrm{M}C & \Longrightarrow & \forall \beta \geq \alpha\, \beta \nVdash \mathrm{M}C \\ & \Longrightarrow & \neg \exists \beta \geq \alpha\, \beta \Vdash C \\ & \Longrightarrow & \forall \beta \geq \alpha\, \beta \nVdash C \\ & \Longrightarrow & \alpha \Vdash \neg C \end{array}$$

2.

$$\begin{array}{rcl} \alpha \Vdash \mathrm{M}(A \wedge B) & \Longrightarrow & \exists \beta \geq \alpha\, \beta \Vdash A \wedge B \\ & \Longrightarrow & \exists \beta \geq \alpha\, \beta \Vdash A \\ & \Longrightarrow & \alpha \Vdash \mathrm{M}A \end{array}$$

3.

$$\begin{array}{rcl}\alpha \Vdash \mathrm{M}C \wedge \neg C & \Longrightarrow & \alpha \Vdash \mathrm{M}C \text{ und } \alpha \Vdash \neg C \\ & \Longrightarrow & \exists \beta \geq \alpha\, \beta \Vdash C \\ & \mathit{und} & \forall \beta \geq \alpha\, \beta \nVdash C \end{array}$$

was offensichtlich ein Widerspruch ist.

Das zeigt, daß die hier vorgestellte Theorie, bei der es sich im wesentlichen um Gabbays μ handelt [Gab82], die von McDermott und Doyle aufgeworfenen Probleme löst.

4.4 Die Syntax von μ

Gabbay [Gab82] präsentiert die Syntax von μ folgendermaßen [Gab82, S. 270]:

Definition 4.4.1 *1. Axiome der intuitionistischen Logik*

(a) $A \to (B \to A)$

(b) $[A \to (B \to C)] \to [(A \to B) \to (A \to C)]$

(c) $A \to (\neg A \to B)$

(d) $(A \to \neg B) \to (B \to \neg A)$

(e) $A \to (B \to A \wedge B)$

(f) $(A \wedge B) \to A$

(g) $(A \wedge B) \to B$

(h) $(A \to C) \to ((B \to C) \to (A \vee B) \to C))$

(i) $A \to A \vee B$

(j) $B \to A \vee B$

(k) $\forall x A(x) \to A(y); A(y) \to \exists x A(x)$

2. Regeln der intuitionistischen Logik

(a) $A, A \to B \vdash B$

(b) $A \to B(x) \vdash A \to \forall x B(x)$*, falls x nicht frei in A auftritt.*

(c) $B(x) \to A \vdash \exists x B(x) \to A$*, falls x nicht frei in A auftritt.*

3. *Axiome für Formeln mit* M

   (a) $A \to \mathrm{M}A$

   (b) $\mathrm{MM}A \to \mathrm{M}A$

   (c) $\neg(A \wedge \mathrm{M}B) \leftrightarrow (A \to \neg B)$

   (d) $(A \wedge \mathrm{M}B \to C \vee \mathrm{M}D) \leftrightarrow (A \wedge \neg D \to \neg B \vee C)$

   (e) $\forall x[A \vee \mathrm{M}B(x)] \leftrightarrow \forall x[\neg B(x) \to A]$

   (f) $\neg \exists y[A \wedge \mathrm{M}B(y)] \leftrightarrow \forall y[A \to \neg B(y)]$

Dabei gelten die Axiome und Regeln in 1. und 2. nur für M*-freie Formeln.*

Gleichheitsgesetze werden von Gabbay nicht angegeben. Bei der expliziten Definition der Semantik beschränkt er sich auf den aussagenlogischen Teil.

Von den anfangs erwähnten Aufgaben sind zwei geleistet: der formale Herleitungsbegriff ($\vdash$) und die Formalisierung von M werden durch die Angabe der Axiome und Regeln für μ geliefert. Die Semantik dazu wurde in Definition 4.3.2 angegeben. Es fehlt noch eine Definition von *nicht-monoton herleitbar* ($\mid\!\sim$).

Definition 4.4.2 *Aus $A_1, \ldots, A_n$ folgt nicht-monoton B, in Zeichen*

$$A_1, \ldots, A_n \rightsquigarrow B,$$

wenn für Formeln $P_1, \ldots, P_k$ die Menge $A_1, \ldots, A_n, \mathrm{M}P_1, \ldots, \mathrm{M}P_k$ konsistent ist, und

$$A_1, \ldots, A_n, \mathrm{M}P_1, \ldots, \mathrm{M}P_k \models B$$

gilt.

Nicht-monotone Herleitbarkeit $\mid\!\sim$ ist dann wie folgt definiert:

$$A_1, \ldots, A_n \mid\!\sim X,$$

wenn es Formeln $B_1, \ldots, B_m$ *gibt, so daß*

$$\begin{gathered} A_1, \ldots, A_n \leadsto B_1 \\ A_1, \ldots, A_n, B_1 \leadsto B_2 \\ \vdots \\ A_1, \ldots, A_n, B_1, \ldots, B_m \leadsto X \end{gathered}$$

gilt.

Das nicht-monotone Schließen ist also ähnlich wie bei Reiters *default reasoning* definiert [Rei80],[Tur84].

Diese Definition unterscheidet sich etwas von der in [CG88], da dort $A \models B$ anders definiert ist. Dort bedeutet $A \models B$, daß bei festem Kripke-Rahmen $(T, \leq)$ für alle Interpretationen der Aussagezeichen, B an allen Knoten $w \in T$ erzwungen wird, an denen A erzwungen wird.

Beispiel 4.4.3

$$(\mathrm{M}C \to C) \wedge \mathrm{M}C \vdash C$$

Also gilt

$$(\mathrm{M}C \to C) \mid\!\sim C.$$

$$\{(\mathrm{M}P \to P), (P \wedge \mathrm{M}Q \to R)\} \mid\!\sim R$$

denn

$$\{(\mathrm{M}P \to P), (P \wedge \mathrm{M}Q \to R), \mathrm{M}P\} \models P$$

und

$$\{(\mathrm{M}P \to P), (P \wedge \mathrm{M}Q \to R), P, \mathrm{M}Q\} \models R$$

([CG88, S. 171])

Für weitere Beispiele siehe [Gab82] und [CG88].

Für nicht-monotone Schlüsse ist es gemäß der angegebenen Definition also notwendig zu entscheiden, ob eine gegebene Menge von Formeln konsistent ist oder nicht. Da auch die intuitionistische Prädikatenlogik unentscheidbar ist, bedeutet das, daß man im Zweifel unter Umständen auf einen gültigen Schluß verzichtet, oder aber einen möglicherweise inkorrekten Schluß vollzieht. Ein effektiv arbeitendes System müßte also mit einem *Truth Maintenance System* [Doy79] ausgestattet sein, das eventuelle Widersprüche aufdeckt und gegebenenfalls inkorrekte Schlüsse wieder rückgängig macht.

Zur Vermeidung solcher Schwierigkeiten betrachten Clarke und Gabbay in [CG88] nur den aussagenlogischen Teil der o.a. Semantik und beschränken sich auf lineare Kripke-Rahmen. Unter diesen zusätzlichen Annahmen soll die Regel *restricted monotonicity*

$$\frac{A \rightsquigarrow B \qquad A \rightsquigarrow X}{A, B \rightsquigarrow X}$$

gelten. Das heißt, wenn man aus A nicht-monoton B und X folgern kann, sind A und B konsistent und erlauben die nicht-monotone Folgerung X.

Der angegebene Beweis ist (zumindest für den Verfasser dieser Arbeit) nicht einleuchtend. Ein Gegenbeispiel: $(\mathrm{M}P \rightarrow P) \rightsquigarrow P$ und $(\mathrm{M}P \rightarrow P) \rightsquigarrow \neg P$ sind gültige Schlüsse, denn $(\mathrm{M}P \rightarrow P), \mathrm{M}P \models P$ und $(\mathrm{M}P \rightarrow P), \mathrm{M}(\neg P) \models \neg P$ sind korrekt. Aber $\{(\mathrm{M}P \rightarrow P), P, \mathrm{M}(\neg P)\}$ ist nicht konsistent.

Für gerichtete, und damit insbesondere für lineare Kripke-Modelle ist $\mathrm{M}A$ äquivalent zu $\neg\neg A$. Wie in [Koc88] gezeigt wird, sind schon die Logiken $J + TND^*$ bzw. $J + AIS$[1] bei gerichteten bzw. linearen Kripke-Rahmen für den aussagenlogischen Teil der o.a. Semantik vollständig. Die in Definition 4.4.1 angegebene Axiomatisierung ist für diese Fälle also deutlich zu vereinfachen.

Im folgenden sollen wieder Kripke-Rahmen beliebigen Aussehens betrachtet werden. Auch dann ist die angegebene Axiomatisierung für eine tatsächliche Implementation der Logik wenig geeignet. In den folgenden Kapiteln wird deshalb zunächst ein schnittfreier Sequenzenkalkül und schließlich ein Matrixkalkül für die gegebene Semantik vorgestellt werden.

[1] TND^* bezeichnet das Axiomenschema $\neg A \vee \neg\neg A$, AIS bezeichnet das Axiomenschema $(B \rightarrow C) \vee (C \rightarrow B)$

Kapitel 5

Die intuitionistische Modallogik J4

In diesem Kapitel wird eine bezüglich der von Gabbay [Gab82] angegebenen Semantik vollständige und korrekte Syntax in Gestalt eines Sequenzenkalküls definiert.

5.1 Die Semantik von J4

Um dieses Kapitel in sich geschlossen zu machen, werden die Definitionen der Sprache L und der Semantik noch einmal wiederholt.

Die Sprache L umfasse

1. für jedes n abzählbar viele n-stellige Prädikatszeichen $P^n, Q^n, \ldots$
2. abzählbar viele Variablen $x, y, z, \ldots$
3. unendlich viele Konstanten $c, d, \ldots$
4. Junktoren $(\neg, \wedge, \vee, \rightarrow)$
5. den gewohnten Satz von Quantoren $(\forall, \exists)$ sowie

6. den neuen einstelligen Junktor $\Diamond$.

Primformeln und Formeln sind wie gewöhnlich definiert. Für $\Diamond$ ist intuitiv an die Regel $w \not\Vdash \neg A \Longrightarrow w \Vdash \Diamond A$ gedacht. Das soll heißen, daß es statthaft ist, A anzunehmen, wenn zum Zeitpunkt w nicht $\neg A$ gilt (nicht erzwungen wird).

Definition 5.1.1 *$(T, \leq, D, \bar{D})$ heißt J4-Kripke-Rahmen, wenn*

1. *T eine nicht-leere Menge ist,*
2. *$\leq$ eine reflexive, transitive Relation auf T ist,*
3. *D eine nicht-leere Menge ist,*
4. *$\bar{D} : T \longrightarrow \wp(D)$ ist eine Abbildung von D auf nicht-leere Teilmengen von D, so daß gilt:*
$$D = \bigcup_{w \in T} \bar{D}(w)$$
und $\bar{D}(w) \subseteq \bar{D}(v)$, wenn $w \leq v$ ist.

Definition 5.1.2 *Sei $(T, \leq, D, \bar{D})$ ein J4-Kripke-Rahmen. Sei $\Vdash$ eine Relation zwischen den Elementen von T und den geschlossenen Formeln von L. Dann ist $(T, \leq, D, \bar{D}, \Vdash)$ ein J4-Kripke-Modell, wenn für alle $w \in T$ gilt:*

1. *$w \Vdash P$ und $w' \geq w \Longrightarrow w' \Vdash P$, für Primformeln P*
2. *$w \Vdash A \wedge B \Longleftrightarrow w \Vdash A$ und $w \Vdash B$*
3. *$w \Vdash A \vee B \Longleftrightarrow w \Vdash A$ oder $w \Vdash B$*
4. *$w \Vdash \neg A \Longleftrightarrow \forall w' \geq w \, w' \not\Vdash A$*
5. *$w \Vdash A \to B \Longleftrightarrow \forall w' \geq w \, (w' \Vdash A \Longrightarrow w' \Vdash B)$*
6. *$w \Vdash \Diamond A \Longleftrightarrow \exists w' \geq w \, w' \Vdash A$*
7. *$w \Vdash \exists x F(x) \Longleftrightarrow \exists c \in \bar{D}(w) \, w \Vdash F(c)$*

8. $w \Vdash \forall x F(x) \Longleftrightarrow \forall w' \geq w \, \forall c \in \bar{D}(w') \, w' \Vdash F(c)$

Bemerkung 5.1.3 *Die Relation $w \Vdash A$ wird gelesen als: w erzwingt A.*

Definition 5.1.4 *Es sei $\mathcal{A} = (T, \leq, D, \bar{D}, \Vdash)$. Eine Formel C gilt in diesem Kripke-Modell $\mathcal{A}$ (Schreibweise: $\mathcal{A} \models C$), wenn für alle w aus T $w \Vdash C$ gilt.*

Bemerkung 5.1.5 *Die Sprache L hängt ab von der Wahl der Variablen, der Prädikatszeichen und der Konstanten. Die Auswahl der Variablen und Prädikatszeichen soll im folgenden fest bleiben, aber die Festlegung der Konstanten sei frei. Für eine Menge C von Konstanten bezeichne $L(C)$ die Sprache L, wie oben definiert, aber mit Konstanten aus C. Für den Rest dieses Textes sollen zwei abzählbare, unendliche Mengen von Konstanten bzw. Parametern reserviert werden: C^* und P^*. $L(C^*)$ ist die formale Sprache von J4. $L(C^* \cup P^*)$ ist die formale Sprache der Beweise für Sätze aus $L(C^*)$. Im folgenden bezeichne D^* die Vereinigung $C^* \cup P^*$. Schließlich sei für ein Modell $(T, \leq, D, \bar{D}, \Vdash)$ mit dem Individuenbereich D $L(D)$ die Sprache dieses Modells.*

Definition 5.1.6 *Sei $(T, \leq, D, \bar{D}, \Vdash)$ ein J4-Kripke-Modell. Sei D_0 eine nicht-leere zu D disjunkte Menge von Konstanten. Eine Interpretation einer Sprache über D_0 in diesem Modell ist eine Abbildung $\iota : D_0 \to D$ und $\iota : \{P \mid P \text{ Relationszeichen}\} \times T \to \{R \mid R \text{ Relation auf } D\}$. Für ein n-stelliges Prädikatszeichen P^n ist $\iota(P^n, w) = R^n \subseteq \bar{D}(w) \times \ldots \times \bar{D}(w)$. Eine geschlossene Formel A aus der Sprache $L(D_0)$ gilt unter dieser Interpretation ι in dem Modell genau dann, wenn $\iota(A)$ in dem Modell gültig ist.*

Definition 5.1.7 T *und* F *seien zwei neue Symbole. Ist A eine Formel, so heißen* $\mathrm{T}A$ *und* $\mathrm{F}A$ signierte Formeln. T *bzw.* F *heißt das* Signum *der Formel. Für ein J4-Modell $(T, \leq, D, \bar{D}, \Vdash)$ soll $w \Vdash \mathrm{T}A$ gelten, wenn $w \Vdash A$ und $w \Vdash \mathrm{F}A$, wenn $w \nVdash A$.*

Die zur Definition von $\Vdash$ äquivalenten Bedingungen für signierte Formeln ergeben sich leicht. Für $\mathrm{T}(A \wedge B)$ und $\mathrm{F}(A \wedge B)$ gilt zum Beispiel:

- $$\begin{aligned} w \Vdash \mathrm{T}(A \wedge B) &\iff w \Vdash A \wedge B \\ &\iff w \Vdash A \text{ und } w \Vdash B \\ &\iff w \Vdash \mathrm{T}A \text{ und } w \Vdash \mathrm{T}B \end{aligned}$$

- $$\begin{aligned} w \Vdash \mathrm{F}(A \wedge B) &\iff w \nVdash A \wedge B \\ &\iff w \nVdash A \text{ oder } w \nVdash B \\ &\iff w \Vdash \mathrm{F}A \text{ oder } w \Vdash \mathrm{F}B \end{aligned}$$

Die so erhaltenen zehn Bedingungen lassen sich durch die im folgenden beschriebene sogenannte *uniforme Darstellung* zusammenfassen. Smullyan [Smu68] präsentierte ein System der uniformen Darstellung für die klassische Logik, das bei der Regelformulierung zu einer deutlichen Vereinfachung führt. In [Fit83] findet sich die Erweiterung der uniformen Darstellung auf modale Logiken. Hier wird ein System der uniformen Darstellung für *intuitionistische* Modallogik vorgestellt:

Definition 5.1.8 *Als Formeln vom Typ α (oder kurz α-Formeln) werden signierte Formeln der Form $\mathrm{T}(A \wedge B)$ und $\mathrm{F}(A \vee B)$ bezeichnet. Zu jeder α-Formel sind, wie in der folgenden Tabelle angegeben, Unterformeln vom Typ α_1 bzw. α_2 definiert:*

α	α_1	α_2
$\mathrm{T}(A \wedge B)$	$\mathrm{T}A$	$\mathrm{T}B$
$\mathrm{F}(A \vee B)$	$\mathrm{F}A$	$\mathrm{F}B$

Definition 5.1.9 *Als Formeln vom Typ β (oder kurz β-Formeln) werden signierte Formeln der Form $\mathrm{T}(A \vee B)$ und $\mathrm{F}(A \wedge B)$ bezeichnet. Zu jeder β-Formel sind, wie in der folgenden Tabelle angegeben, Unterformeln vom Typ β_1 bzw. β_2 definiert:*

β	β_1	β_2
$\mathrm{T}(A \vee B)$	$\mathrm{T}A$	$\mathrm{T}B$
$\mathrm{F}(A \wedge B)$	$\mathrm{F}A$	$\mathrm{F}B$

Die Schreibweise $A[a/x]$ bezeichne das Einsetzen von a für alle Auftreten von x in A.

Definition 5.1.10 *Als Formeln vom Typ γ (oder kurz γ-Formeln) werden signierte Formeln der Form* $\mathrm{F}\exists xA$ *bezeichnet. Zu jeder γ-Formel und zu jedem $a \in D^*$ ist, wie in der folgenden Tabelle angegeben, eine Unterformel vom Typ $\gamma_0(a)$ definiert:*

γ	$\gamma_0(a)$
$\mathrm{F}\exists xA$	$\mathrm{F}A[a/x]$

Definition 5.1.11 *Als Formeln vom Typ δ (oder kurz δ-Formeln) werden signierte Formeln der Form* $\mathrm{T}\exists xA$ *bezeichnet. Zu jeder δ-Formel und zu jedem $a \in D^*$ ist, wie in der folgenden Tabelle angegeben, eine Unterformel vom Typ $\delta_0(a)$ definiert:*

δ	$\delta_0(a)$
$\mathrm{T}\exists xA$	$\mathrm{T}A[a/x]$

Definition 5.1.12 *Als Formeln vom Typ π (oder kurz π-Formeln) werden signierte Formeln der Form* $\mathrm{F}P$ *für Primformeln P,* $\mathrm{T}\Diamond A, \mathrm{F}\neg A$, $\mathrm{F}(A \to B)$ *und* $\mathrm{F}\forall xA$ *bezeichnet. Zu den Teilformeln der π-Formel sind die Sekundärtypen*[1] $\pi_0(a)$, π_1 *bzw.* π_2, *wie in der folgenden Tabelle angegeben, definiert:*

π	$\pi_0(a)$	π_1	π_2
$\mathrm{T}\Diamond A$		$\mathrm{T}A$	$\mathrm{T}A$
$\mathrm{F}\neg A$		$\mathrm{T}A$	$\mathrm{T}A$
$\mathrm{F}(A \to B)$		$\mathrm{T}A$	$\mathrm{F}B$
$\mathrm{F}\forall xA$	$\mathrm{F}A[a/x]$		

Definition 5.1.13 *Als Formeln vom Typ ν (oder kurz ν-Formeln) werden signierte Formeln der Form* $\mathrm{T}P$ *für Primformeln P,* $\mathrm{F}\Diamond A, \mathrm{T}\neg A$, $\mathrm{T}(A \to B)$ *und* $\mathrm{T}\forall xA$ *bezeichnet. Zu den Teilformeln der π-Formel sind die Sekundärtypen*[2] $\nu_0(a)$, ν_1 *bzw.* ν_2, *wie in der folgenden Tabelle angegeben, defi-*

[1] Primformeln P haben im Fall von J4 den angegebenen Typ, im Fall von J keinen Typ (vgl. dazu Abschnitt 6.1.2)

[2] dito

niert:

ν	$\nu_0(a)$	ν_1	ν_2
$\mathrm{F}\Diamond A$		$\mathrm{F}A$	$\mathrm{F}A$
$\mathrm{T}\neg A$		$\mathrm{F}A$	$\mathrm{F}A$
$\mathrm{T}(A \to B)$		$\mathrm{F}A$	$\mathrm{T}B$
$\mathrm{T}\forall x A$	$\mathrm{T}A[a/x]$		

Im Gegensatz zu den üblichen Modallogiken, die auf der klassischen Prädikatenlogik aufbauen, sind auch $\neg$-, $\to$- und $\forall$-Formeln modal. Das zeigt sich in der vom klassischen Fall abweichenden Definition der Typen ν und π. Insbesondere der $\forall$-Quantor macht Probleme bei der Festlegung seines Typs. Zum einen muß er entsprechend seiner Modalität, zum anderen gewissen Variablenbedingungen entsprechend behandelt werden. Die Definition von drei Untertypen für ν- bzw. π-Formeln scheint überzogen, da doch ohnehin nur im Fall der Implikation mehr als eine direkte Teilformel existiert. Die Motivation für die ν_0- und π_0-Typen war, die Analogie zu den Typen γ und δ möglichst weitgehend aufrecht zu erhalten.

Bemerkung 5.1.14 *Die Typen $\alpha, \beta, \gamma, \ldots$ werden als* Primärtypen *(kurz: PTypen), $\alpha_1, \alpha_2, \ldots, \nu_0, \nu_1, \ldots$ als* Sekundärtypen *(kurz: STypen) bezeichnet.*

Bemerkung 5.1.15 (Bezeichnungsmißbrauch) *$\alpha, \alpha_1, \alpha_2, \beta, \ldots, \delta$, $\delta_0(a)$ etc. werden im folgenden anstelle von Variablen für Formeln des entsprechenden Typs benutzt werden.*

Durch die Ausnutzung der Typenschreibweise kann die Definition von $\Vdash$ für signierte Formeln kürzer so dargestellt werden:

Korollar 5.1.16 *Sei $(T, \leq, D, \bar{D}, \Vdash)$ ein J4-Kripke-Modell. Dann gilt für jedes $w \in T$:*

1. $w \Vdash \alpha \iff w \Vdash \alpha_1$ *und* $w \Vdash \alpha_2$
2. $w \Vdash \beta \iff w \Vdash \beta_1$ *oder* $w \Vdash \beta_2$
3. $w \Vdash \gamma \iff \forall a \in \bar{D}(w)\, w \Vdash \gamma_0(a)$

4. $w \Vdash \delta \Longleftrightarrow \exists a \in \bar{D}(w)\, w \Vdash \delta_0(a)$

5. $w \Vdash \nu \Longleftrightarrow \forall w' \geq w\, \forall a \in \bar{D}(w')\, w' \Vdash \nu_0(a), w' \Vdash \nu_1$ *oder* $w' \Vdash \nu_2$

6. $w \Vdash \pi \Longleftrightarrow \exists w' \geq w\, \exists a \in \bar{D}(w')\, w' \Vdash \pi_0(a), w' \Vdash \pi_1$ *und* $w' \Vdash \pi_2$

Das Korollar 5.1.16 zeigt weitgehende Übereinstimmung mit den entsprechenden Aussagen bezüglich klassischer Modallogik. Auffällig ist nur die Behandlung der Typen ν und π. Die klassischen Fälle für T$\Diamond A$ und F$\Diamond A$ sind jedoch darin wiederzuerkennen, weil für diese Formeln ν_1 und ν_2 (bzw. π_1 und π_2) identisch sind und der zweite Quantor ($\forall a$ bzw. $\exists a$) sich auf eine im folgenden nicht auftretende Variable bezieht und deshalb ausgelassen werden kann.

Definition 5.1.17 *Sei M eine Menge von signierten Formeln. M wird an einem Punkt $w \in T$ erzwungen, wenn $w \Vdash A$ für alle $A \in M$. Die Bezeichnung dafür ist: $w \Vdash M$.*

Eine Menge M von signierten Formeln der Sprache über D_0 heißt erfüllbar, wenn es ein Modell $(T, \leq, D, \bar{D}, \Vdash)$ und eine Interpretation $\iota : D_0 \to D$ sowie ein $w \in T$ gibt, so daß $w \Vdash \iota(M)$ gilt.

Soviel zur Semantik von J4. Der nächste Abschnitt wird sich mit der Syntax, d.h. mit Herleitungen für Formeln in J4 beschäftigen.

5.2 Ein Sequenzenkalkül für J4

Im folgenden sei $D^* := C^* \cup P^*$.

Definition 5.2.1 *Eine* Sequenz *ist ein geordnetes Paar $\langle \Gamma, \Delta \rangle$ von Mengen von Formeln. Statt $\langle \Gamma, \Delta \rangle$ schreibt man $\Gamma \Rightarrow \Delta$.*

Eine Sequenz $\Gamma \Rightarrow \Delta$ gilt *in einem Kripke-Modell $(T, \leq, D, \bar{D}, \Vdash)$, wenn für alle $w \in T$ gilt: wenn $w \Vdash \Gamma$, dann gibt es ein $A \in \Delta$ mit $w \Vdash A$.*

Eine Sequenz $\Gamma \Rightarrow \Delta$ *über* D^* gilt *im Kripke-Modell* $(T, \leq, D, \bar{D}, \Vdash)$ *unter der Interpretation* $\iota : D^* \to D$, *wenn* $\iota(\Gamma) \Rightarrow \iota(\Delta)$ *in diesem Modell gültig ist.*

Eine Sequenz $\Gamma \Rightarrow \Delta$ *heißt* gültig, *wenn sie unter jeder Interpretation in jedem Modell gültig ist.*

Eine Sequenz $\Gamma \Rightarrow \Delta$ *definiert eine Menge* $M_\Gamma \cup M_\Delta$ *von signierten Formeln, wobei*

$$\begin{array}{rcl} M_\Gamma & := & \{\mathrm{T}A \mid A \in \Gamma\} \\ M_\Delta & := & \{\mathrm{F}A \mid A \in \Delta\} \end{array}$$

Diese Menge heißt die assoziierte Menge *der Sequenz.*

Bemerkung 5.2.2 *Sei* $\Gamma \Rightarrow \Delta$ *eine Sequenz.* Γ *bzw.* Δ *heißen* Antezedens *bzw.* Sukzedens *der Sequenz.* Γ, A *steht als Abkürzung für* $\Gamma \cup \{A\}$.

Das folgende Lemma ist ein erster Schritt auf dem Weg zum Korrektheitssatz. Es stellt einen Zusammenhang zwischen der Gültigkeit einer Sequenz und der (Nicht-) Erfüllbarkeit der zugehörigen assoziierten Menge her.

Lemma 5.2.3 *Eine Sequenz* $\Gamma \Rightarrow \Delta$ *ist gültig genau dann, wenn ihre assoziierte Menge nicht erfüllbar ist.*

Beweis: Für jedes Modell $(T, \leq, D, \bar{D}, \Vdash)$ und jede Interpretation ι der Konstanten aus $\Gamma \Rightarrow \Delta$ in diesem Modell gilt: für alle $w \in T$

$$\begin{array}{rl} & w \Vdash \iota(\Gamma) \Longrightarrow \text{es gibt } A \in \Delta : w \Vdash \iota(A) \\ \Longleftrightarrow & w \Vdash \iota(M_\Gamma) \Longrightarrow \text{es gibt } \mathrm{F}A \in M_\Delta : w \nVdash \iota(\mathrm{F}A) \\ \Longleftrightarrow & w \nVdash \iota(M_\Gamma \cup M_\Delta) \end{array}$$

Das bedeutet, daß die assoziierte Menge nicht erfüllbar ist. □

Im folgenden werden jetzt die Axiome und Regeln eines Sequenzenkalküls für J4 definiert. Die Regeln dieses Kalküls sind besser von unten nach oben zu lesen. Die Motivation für die Regelformulierung ist, daß die Erfüllbarkeit der Konklusion die Erfüllbarkeit einer der Prämissen impliziert.

$$\Gamma, A \Rightarrow A, \Delta$$

$$\frac{\Gamma \Rightarrow A, \Delta \quad \Gamma, B \Rightarrow \Delta}{\Gamma, A \to B \Rightarrow \Delta} \to\Rightarrow \qquad \frac{\Gamma^\star, A \Rightarrow B, \Delta^\star}{\Gamma \Rightarrow A \to B, \Delta} \Rightarrow\to$$

$$\frac{\Gamma, A, B \Rightarrow \Delta}{\Gamma, A \wedge B \Rightarrow \Delta} \wedge \Rightarrow \qquad \frac{\Gamma \Rightarrow A, \Delta \quad \Gamma \Rightarrow B, \Delta}{\Gamma \Rightarrow A \wedge B, \Delta} \Rightarrow \wedge$$

$$\frac{\Gamma, A \Rightarrow \Delta \quad \Gamma, B \Rightarrow \Delta}{\Gamma, A \vee B \Rightarrow \Delta} \vee \Rightarrow \qquad \frac{\Gamma \Rightarrow A, B, \Delta}{\Gamma \Rightarrow A \vee B, \Delta} \Rightarrow \vee$$

$$\frac{\Gamma \Rightarrow A, \Delta}{\Gamma, \neg A \Rightarrow \Delta} \neg \Rightarrow \qquad \frac{\Gamma^\star, A \Rightarrow \Delta^\star}{\Gamma \Rightarrow \neg A, \Delta} \Rightarrow \neg$$

$$\frac{\Gamma, A[c/x] \Rightarrow \Delta}{\Gamma, \forall x A \Rightarrow \Delta} \forall \Rightarrow \qquad \frac{\Gamma^\star \Rightarrow A[a/x], \Delta^\star}{\Gamma \Rightarrow \forall x A, \Delta} \Rightarrow \forall$$

$$\frac{\Gamma, A[a/x] \Rightarrow \Delta}{\Gamma, \exists x A \Rightarrow \Delta} \exists \Rightarrow \qquad \frac{\Gamma \Rightarrow A[c/x], \Delta}{\Gamma \Rightarrow \exists x A, \Delta} \Rightarrow \exists$$

$$\frac{\Gamma^\star, A \Rightarrow \Delta^\star}{\Gamma, \Diamond A \Rightarrow \Delta} \Diamond \Rightarrow \qquad \frac{\Gamma \Rightarrow A, \Delta}{\Gamma \Rightarrow \Diamond A, \Delta} \Rightarrow \Diamond$$

Für $\Rightarrow \forall$ und $\exists \Rightarrow$ darf der Parameter a nicht in der Konklusion auftreten.

$$\begin{aligned} \Gamma^\star &:= \{A \in \Gamma \mid A = \neg B, B \to C, \forall x B \text{ oder } A \text{ ist Primformel}\} \\ \Delta^\star &:= \{\Diamond A \mid \Diamond A \in \Delta\} \end{aligned}$$

Tabelle 5.1: Ein Sequenzenkalkül für J4

Definition 5.2.4 *Sequenzen der Form $\Gamma, A \Rightarrow A, \Delta$ heißen Axiome oder Basissequenz.*

Lemma 5.2.5 *Axiome sind stets gültig.*

Beweis: Die assoziierte Menge eines Axioms enthält für eine Formel A sowohl $\mathrm{T}A$ als auch $\mathrm{F}A$ und ist deshalb nicht erfüllbar. □

Tabelle 5.1 zeigt die Regeln des Sequenzenkalküls für J4.

Wie erwähnt, bestimmt jede Sequenz eine Menge von signierten Formeln, die sogenannte assoziierte Menge. Durch Ausnutzung der Typenschreibweise kann die Anzahl der Regeln des Sequenzenkalküls für J4 stark verringert werden. Das Umformen einen Sequenzenregel in Typenschreibweise wird am Beispiel der Regel

$$\frac{\Gamma, A, B \Rightarrow \Delta}{\Gamma, A \wedge B \Rightarrow \Delta} \wedge \Rightarrow$$

dargestellt werden. Die assoziierte Menge der Prämisse ist

$$M_\Gamma \cup M_\Delta \cup \{\mathrm{T}A, \mathrm{T}B\},$$

oder kürzer

$$M_\Gamma, M_\Delta, \mathrm{T}A, \mathrm{T}B$$

Die assoziierte Menge der Konklusion ist

$$M_\Gamma, M_\Delta, \mathrm{T}(A \wedge B)$$

$\mathrm{T}(A \wedge B)$ ist entsprechend der angegebenen Klassifizierung eine α-Formel. Die Regel sieht also mit Typen geschrieben so aus:

$$\frac{M, \alpha_1, \alpha_2}{M, \alpha}$$

wobei $M := M_\Gamma \cup M_\Delta$ ist.

Auf diese Weise werden aus den vier Regeln für $\wedge$ und $\vee$

$$\frac{M, \alpha_1, \alpha_2}{M, \alpha}\alpha \quad \frac{M, \beta_1 \quad M, \beta_2}{M, \beta}\beta$$

In Typenschreibweise ist $M^\star$ die Menge

$$\{A \in M \mid A \text{ vom Typ } \nu\}$$

Aus den zwei Regeln für den Existenzquantor wird

$$\frac{M, \gamma_0(a)}{M, \gamma}\gamma \quad \frac{M, \delta_0(a)}{M, \delta}\delta$$

Dabei ist a ein Element von D^*, das nicht in der Konklusion der δ-Regel auftreten darf.

Aus den acht Regeln für $\neg, \rightarrow, \forall$ und $\Diamond$ wird

$\dfrac{M, \nu_0(a)}{M, \nu}\nu$	mit $a \in D^*$, falls ν eine $\forall$-Formel ist
$\dfrac{M, \nu_1 \quad M, \nu_2}{M, \nu}\nu$	sonst
$\dfrac{M^\star, \pi_0(a), \pi_1, \pi_2}{M, \pi}\pi$	$a \in D^*$ darf nicht in der Konklusion auftreten.

Auffällig ist das z.B. bei $\Diamond$-Formeln vom Typ π jetzt zwei gleiche Teilformeln in der Regelprämisse auftreten, da π_1 und π_2 gleich sind. Die doppelten Auftreten dürfen gestrichen werden, weil die Prämisse eine Menge von Formeln ist. Analog soll im Fall, daß ν_1 gleich ν_2 ist, nur M, ν_1 als Prämisse auftreten.

Zum Beweis der Korrektheit wird nun mit den folgenden Hilfssätzen und durch Kontraposition der Aussage von Lemma 5.2.3 gezeigt werden, daß die Konklusionen der Regeln gültig sind, wenn die Prämissen gültig sind. Da, wie bereits in Lemma 5.2.5 gezeigt, Axiome stets gültig sind, folgt daraus der Korrektheitssatz. Es sei daran erinnert, daß die Gültigkeit einer Sequenz gleichbedeutend mit der Nicht-Erfüllbarkeit ihrer assoziierten Menge ist.

Lemma 5.2.6 *Wenn M, α erfüllbar ist, ist auch M, α_1, α_2 erfüllbar.*

Beweis: Wegen der Voraussetzung gibt es ein Modell $(T, \leq, D, \bar{D}, \Vdash)$, eine Interpretation $\iota : D^* \rightarrow D$ und ein $w \in T$, so daß $w \Vdash \iota(M)$ und $w \Vdash \iota(\alpha)$. Wegen Korollar 5.1.16 gilt dann auch $w \Vdash \iota(\alpha_1)$ und $w \Vdash \iota(\alpha_2)$. Also ist auch M, α_1, α_2 erfüllbar. □

Durch Kontraposition ergibt sich daraus, daß die Regeln $\wedge \Rightarrow$ und $\Rightarrow \vee$ korrekt sind, d.h. aus der Gültigkeit der Prämisse folgt die Gültigkeit der Konklusion.

Lemma 5.2.7 *Wenn M, β erfüllbar ist, ist M, β_1 oder M, β_2 erfüllbar.*

Beweis: Wegen der Voraussetzung gibt es ein Modell $(T, \leq, D, \bar{D}, \Vdash)$, eine Interpretation $\iota : D^* \to D$ und ein $w \in T$, so daß $w \Vdash \iota(M)$ und $w \Vdash \iota(\beta)$. Wegen Korollar 5.1.16 gilt dann $w \Vdash \iota(\beta_1)$ oder $w \Vdash \iota(\beta_2)$. Also ist M, β_1 oder M, β_2 erfüllbar. □

Durch Kontraposition ergibt sich daraus, daß die Regeln $\vee \Rightarrow$ und $\Rightarrow \wedge$ korrekt sind.

Lemma 5.2.8 *Wenn M, γ erfüllbar ist, ist für eine Konstante oder einen Parameter $c \in D^*$ auch $M, \gamma_0(c)$ erfüllbar.*

Beweis: Wegen der Voraussetzung gibt es ein Modell $(T, \leq, D, \bar{D}, \Vdash)$, eine Interpretation $\iota : D^* \to D$ und ein $w \in T$, so daß $w \Vdash \iota(M)$ und $w \Vdash \iota(\gamma)$. Wegen Korollar 5.1.16 gilt dann $\forall b \in \bar{D}(w) : w \Vdash \iota(\gamma_0(b))$. Insbesondere gibt es ein $b \in \bar{D}(w)$ mit $w \Vdash \iota(\gamma_0(b))$. Es sind zwei Fälle zu unterscheiden:

1. c tritt bereits in M, γ auf. Dann ist $\iota(c) \in \bar{D}(w)$ und $M, \gamma_0(c)$ wird unter der Interpretation ι von w erfüllt.

2. sonst: definiere eine neue Interpretation ι' wie folgt:

$$\iota'(x) := \begin{cases} b & \text{falls } x = c \\ \iota(x) & \text{sonst} \end{cases}$$

   Dann gilt:

$$\begin{aligned} & w \Vdash \iota'(\gamma_0)(b) \\ \iff & w \Vdash \iota'(\gamma_0)(\iota'(c)) \\ \iff & w \Vdash \iota'(\gamma_0(c)) \end{aligned}$$

   $w \Vdash \iota'(M)$, da c in M nicht auftritt.

Also ist $M, \gamma_0(c)$ erfüllbar. $\Box$

Durch Kontraposition ergibt sich daraus, daß die Regel $\Rightarrow \exists$ korrekt ist.

Lemma 5.2.9 *Wenn M, δ erfüllbar ist, so ist für einen Parameter $a \in P^* \subseteq D^*$, der nicht in M, δ auftritt, $M, \delta_0(a)$ erfüllbar.*

Beweis: Wegen der Voraussetzung gibt es ein Modell $(T, \leq, D, \bar{D}, \Vdash)$, eine Interpretation $\iota : D^* \to D$ und ein $w \in T$, so daß $w \Vdash \iota(M)$ und $w \Vdash \iota(\delta)$. Wegen Korollar 5.1.16 gilt dann $\exists b \in \bar{D}(w) : w \Vdash \iota(\delta_0(b))$. Definiere eine neue Interpretation ι' wie folgt:

$$\iota'(x) := \begin{cases} b & \text{falls } x = a \\ \iota(x) & \text{sonst} \end{cases}$$

$w \Vdash \iota'(M)$, da a in M nicht auftritt, und es gilt:

$$\begin{aligned} & w \Vdash \iota'(\delta_0)(b) \\ \iff & w \Vdash \iota'(\delta_0)(\iota'(a)) \\ \iff & w \Vdash \iota'(\delta_0(a)) \end{aligned}$$

Also ist $M, \delta_0(a)$ erfüllbar. $\Box$

Durch Kontraposition ergibt sich daraus, daß die Regel $\Rightarrow \exists$ korrekt ist.

Lemma 5.2.10 *Wenn M, ν erfüllbar ist, sind zwei Fälle zu unterscheiden:*

1. *ν ist eine $\forall$-Formel. Dann ist für einen Parameter oder eine Konstante $c \in D^*$ $M, \nu_0(c)$ erfüllbar.*
2. *Andernfalls ist M, ν_1 oder M, ν_2 erfüllbar.*

Beweis:

1. Der Beweis ist der gleiche wie im Fall von Lemma 5.2.8.

2. Wegen der Voraussetzung gibt es ein Modell $(T, \leq, D, \bar{D}, \Vdash)$, eine Interpretation $\iota : D^* \to D$ und ein $w \in T$, so daß $w \Vdash \iota(M)$ und $w \Vdash \iota(\nu)$. Wegen Korollar 5.1.16 gilt dann $w \Vdash \iota(\nu_1)$ oder $w \Vdash \iota(\nu_2)$. Also ist M, ν_1 oder M, ν_2 erfüllbar. □

Durch Kontraposition ergibt sich daraus, daß die Regeln $\Rightarrow \Diamond$, $\neg \Rightarrow$, $\forall \Rightarrow$ und $\to\Rightarrow$ korrekt sind.

Lemma 5.2.11 *Sei $(T, \leq, D, \bar{D}, \Vdash)$ ein Kripke-Modell, M eine Menge von (signierten) Formeln, $w \in T$ und $w' \in T$ mit $w \leq w'$. Dann gilt:*

1. $w \Vdash \nu \Rightarrow$
   (a) $w' \Vdash \nu_0(a)$ $\quad$ *($a \in \bar{D}(w')$ falls ν eine $\forall$-Formel ist),*
   (b) $w' \Vdash \nu_1$ *oder*
   (c) $w' \Vdash \nu_2$
2. $w \Vdash M \Rightarrow w' \Vdash \{\nu \mid \nu \in M\}$
3. $w \Vdash \nu \Rightarrow$
   (a) $w \Vdash \nu_0(a)$ $\quad$ *($a \in \bar{D}(w)$ falls ν eine $\forall$-Formel ist),*
   (b) $w \Vdash \nu_1$ *oder*
   (c) $w \Vdash \nu_2$

Beweis:

1. folgt direkt aus Korollar 5.1.16 und Definition 5.1.7.
2. Sei $v \in T$ beliebig mit $w' \leq v$. Da "$\leq$" transitiv ist, gilt $w \leq v$. Aus $w \Vdash \nu$ folgt also $\forall a \in \bar{D}(v)\, v \Vdash \nu_0(a)$, $v \Vdash \nu_1$ oder $v \Vdash \nu_2$. Da v beliebig war, folgt $w' \Vdash \nu$.
3. folgt direkt aus 1. mit $w = w'$. □

Lemma 5.2.12 *Wenn M, π erfüllbar ist, sind zwei Fälle zu unterscheiden:*

1. *π ist eine $\forall$-Formel. Dann ist für einen Parameter $a \in P^*$ $M^\star, \pi_0(a)$ erfüllbar.*

2. *Andernfalls ist $M^\star, \pi_1, \pi_2$ erfüllbar.*

Beweis:

1. Wegen der Voraussetzung gibt es ein Modell $(T, \leq, D, \bar{D}, \Vdash)$, eine Interpretation $\iota : D^* \to D$ und ein $w \in T$, so daß $w \Vdash \iota(M)$ und $w \Vdash \iota(\pi)$. Wegen Korollar 5.1.16 gilt dann $\exists v \geq w\, \exists b \in \bar{D}(v) : v \Vdash \iota(\pi_0(b))$. Wegen Lemma 5.2.11 gilt $v \Vdash M^\star$. Definiere eine neue Interpretation $\iota' : D^* \to D$ wie folgt:
$$\iota'(x) := \begin{cases} b & \text{falls } x = a \\ \iota(x) & \text{sonst} \end{cases}$$
$v \Vdash \iota'(M^\star)$, da a in $M^\star$ nicht auftritt, und es gilt:
$$\begin{aligned} & v \Vdash \iota'(\pi_0)(b) \\ \iff & v \Vdash \iota'(\pi_0)(\iota'(a)) \\ \iff & v \Vdash \iota'(\pi_0(a)) \end{aligned}$$
Also ist $M^\star, \pi_0(a)$ erfüllbar.

2. Wegen der Voraussetzung gibt es ein Modell $(T, \leq, D, \bar{D}, \Vdash)$, eine Interpretation ι und ein $w \in T$, so daß $w \Vdash \iota(M)$ und $w \Vdash \iota(\pi)$. Wegen Korollar 5.1.16 gibt es dann ein $v \geq w$ mit $v \Vdash \iota(\pi_1)$ und $v \Vdash \iota(\pi_2)$. $v \Vdash \iota(M^\star)$ wegen Lemma 5.2.11. Also ist auch $M^\star, \pi_1, \pi_2$ erfüllbar. $\square$

Durch Kontraposition ergibt sich daraus, daß die Regeln $\diamond \Rightarrow$, $\Rightarrow \neg$, $\Rightarrow \forall$ und $\Rightarrow \rightarrow$ korrekt sind.

Definition 5.2.13 *Eine Formel A (Sequenz $\Gamma \Rightarrow \Delta$) der Sprache $L(C^*)$ heißt in J4* bewiesen*, wenn es eine mit Instanzen von Axiomen beginnende Herleitung der Sequenz $\Rightarrow A$ ($\Gamma \Rightarrow \Delta$) gibt, die nur durch Anwendung der Regeln des Sequenzenkalküls für J4 entstanden ist.*

Satz 5.2.14 (Korrektheitssatz für J4) *Jede mit dem J4-Sequenzenkalkül bewiesene Formel (Sequenz) ist gültig.*

Beweis: Lemma 5.2.3, Lemma 5.2.5 und Lemma 5.2.6–5.2.12 $\square$

5.3 Vollständigkeit von J4 und Einbettung in S4

In diesem Abschnitt wird gezeigt werden, daß sich J4 ähnlich wie J als Teillogik von S4 auffassen läßt. Dadurch erhält man einen zweiten Beweis der Korrektheit von J4 sowie die Vollständigkeit von J4.

Zunächst werden Semantik und Syntax von S4 definiert.

Definition 5.3.1 *Die Sprache von S4 ist die Sprache von J4 erweitert um den neuen einstelligen Junktor* $\Box$.

$(T, \leq, D, \bar{D})$ *ist ein S4-Kripke-Rahmen, wenn es ein J4-Kripke-Rahmen ist.*

Definition 5.3.2 *Sei* $(T, \leq, D, \bar{D})$ *ein S4-Kripke-Rahmen. Sei* $\Vdash$ *eine Relation zwischen den Elementen von* T *und den geschlossenen Formeln von* L. *Dann ist* $(T, \leq, D, \bar{D}, \Vdash)$ *ein S4-Kripke-Modell, wenn für alle* $w \in T$,

1. $w \Vdash A \land B \iff w \Vdash A$ *und* $w \Vdash B$
2. $w \Vdash A \lor B \iff w \Vdash A$ *oder* $w \Vdash B$
3. $w \Vdash \neg A \iff w \nVdash A$
4. $w \Vdash A \to B \iff w \nVdash A$ *oder* $w \Vdash B$
5. $w \Vdash \Diamond A \iff \exists w' \geq w\; w' \Vdash A$
6. $w \Vdash \Box A \iff \forall w' \geq w\; w' \Vdash A$
7. $w \Vdash \exists x F(x) \iff \exists c \in \bar{D}(w)\; w \Vdash F(c)$
8. $w \Vdash \forall x F(x) \iff \forall c \in \bar{D}(w)\; w \Vdash F(c)$

Definition 5.3.3 *Die Axiome von S4 sind Sequenzen der Form* $\Gamma, A \Rightarrow A, \Delta$.

$$\Gamma, A \Rightarrow A, \Delta$$

$$\frac{\Gamma \Rightarrow A, \Delta \quad \Gamma, B \Rightarrow \Delta}{\Gamma, A \to B \Rightarrow \Delta} \to\Rightarrow \qquad \frac{\Gamma, A \Rightarrow B, \Delta}{\Gamma \Rightarrow A \to B, \Delta} \Rightarrow\to$$

$$\frac{\Gamma, A, B \Rightarrow \Delta}{\Gamma, A \wedge B \Rightarrow \Delta} \wedge \Rightarrow \qquad \frac{\Gamma \Rightarrow A, \Delta \quad \Gamma \Rightarrow B, \Delta}{\Gamma \Rightarrow A \wedge B, \Delta} \Rightarrow \wedge$$

$$\frac{\Gamma, A \Rightarrow \Delta \quad \Gamma, B \Rightarrow \Delta}{\Gamma, A \vee B \Rightarrow \Delta} \vee \Rightarrow \qquad \frac{\Gamma \Rightarrow A, B, \Delta}{\Gamma \Rightarrow A \vee B, \Delta} \Rightarrow \vee$$

$$\frac{\Gamma \Rightarrow A, \Delta}{\Gamma, \neg A \Rightarrow \Delta} \neg \Rightarrow \qquad \frac{\Gamma, A \Rightarrow \Delta}{\Gamma \Rightarrow \neg A, \Delta} \Rightarrow \neg$$

$$\frac{\Gamma, A[c/x] \Rightarrow \Delta}{\Gamma, \forall x A \Rightarrow \Delta} \forall \Rightarrow \qquad \frac{\Gamma \Rightarrow A[a/x], \Delta}{\Gamma \Rightarrow \forall x A, \Delta} \Rightarrow \forall$$

$$\frac{\Gamma, A[a/x] \Rightarrow \Delta}{\Gamma, \exists x A \Rightarrow \Delta} \exists \Rightarrow \qquad \frac{\Gamma \Rightarrow A[c/x], \Delta}{\Gamma \Rightarrow \exists x A, \Delta} \Rightarrow \exists$$

$$\frac{\Gamma *, A \Rightarrow \Delta *}{\Gamma, \Diamond A \Rightarrow \Delta} \Diamond \Rightarrow \qquad \frac{\Gamma \Rightarrow A, \Delta}{\Gamma \Rightarrow \Diamond A, \Delta} \Rightarrow \Diamond$$

$$\frac{\Gamma, A \Rightarrow \Delta}{\Gamma, \Box A \Rightarrow \Delta} \Box \Rightarrow \qquad \frac{\Gamma * \Rightarrow A, \Delta *}{\Gamma \Rightarrow \Box A, \Delta} \Rightarrow \Box$$

Für $\Rightarrow \forall$ und $\exists \Rightarrow$ darf der Parameter a nicht in der Konklusion auftreten.

$$\begin{aligned} \Gamma * &:= \{\Box A \mid \Box A \in \Gamma\} \\ \Delta * &:= \{\Diamond A \mid \Diamond A \in \Delta\} \end{aligned}$$

Tabelle 5.2: Ein Sequenzenkalkül für S4

Die Tabelle 5.2 zeigt die Regeln eines Sequenzenkalküls für S4.

Um J4 in S4 einzubetten wird jeder Formel A von J4 eine Formel $\overline{A}$ von S4 zugeordnet, so daß $\vdash_{J4} A \Longleftrightarrow \vdash_{S4} \overline{A}$ gilt. Daß J auf diese Weise in S4 abgebildet werden kann, ist ein Resultat von K. Gödel. Für J4 wird hier eine Erweiterung der in [Fit83] angegebenen Abbildung benutzt:

Definition 5.3.4 *Zu jeder J4-Formel A ist $\overline{A}$ wie folgt induktiv definiert:*

1. $\overline{A} = \Box A$ *für Primformeln A.*
2. $\overline{A \wedge B} = \overline{A} \wedge \overline{B}$
3. $\overline{A \vee B} = \overline{A} \vee \overline{B}$
4. $\overline{A \to B} = \Box(\overline{A} \to \overline{B})$
5. $\overline{\neg A} = \Box \neg \overline{A}$
6. $\overline{\Diamond A} = \Diamond \overline{A}$
7. $\overline{\forall x A} = \Box \forall x \overline{A}$
8. $\overline{\exists x A} = \exists x \overline{A}$

Weil J4- und S4-Modelle dieselben Kripke-Rahmen haben, liegt der Verdacht nahe, daß es eine Beziehung zwischen den Modellen gibt.

Lemma 5.3.5 *Seien $\mathcal{A} = (T, \leq, D, \bar{D}, \Vdash_{S4})$ ein S4-Kripke-Modell und $\mathcal{B} = (T, \leq, D, \bar{D}, \Vdash_{J4})$ ein J4-Kripke-Modell, so daß für alle Primformeln P und alle $w \in T$ gilt*

$$w \Vdash_{J4} P \Leftrightarrow w \Vdash_{S4} \overline{P}$$

Dann gilt für alle Formeln A aus $L(C^*)$

$$w \Vdash_{J4} A \Leftrightarrow w \Vdash_{S4} \overline{A}$$

Beweis: Der Beweis wird induktiv nach dem Aufbau von A geführt.

1. Ist A eine Primformel gilt die Behauptung trivialerweise.

2. $A = B \wedge C$

$$\begin{aligned} w \Vdash_{S4} \overline{B \wedge C} &\iff w \Vdash_{S4} \overline{B} \text{ und } w \Vdash_{S4} \overline{C} \\ &\overset{IV}{\iff} w \Vdash_{J4} B \text{ und } w \Vdash_{J4} C \\ &\iff w \Vdash_{J4} B \wedge C \end{aligned}$$

3. $A = B \vee C$

$$\begin{aligned} w \Vdash_{S4} \overline{B \vee C} &\iff w \Vdash_{S4} \overline{B} \text{ oder } w \Vdash_{S4} \overline{C} \\ &\overset{IV}{\iff} w \Vdash_{J4} B \text{ oder } w \Vdash_{J4} C \\ &\iff w \Vdash_{J4} B \vee C \end{aligned}$$

4. $A = \neg B$

$$\begin{aligned} w \Vdash_{S4} \overline{\neg B} &\iff w \Vdash_{S4} \Box \neg \overline{B} \\ &\iff \forall w' \geq w \; w' \Vdash_{S4} \neg \overline{B} \\ &\iff \forall w' \geq w \; w' \nVdash_{S4} \overline{B} \\ &\overset{IV}{\iff} \forall w' \geq w \; w' \nVdash_{J4} B \\ &\iff w \Vdash_{J4} \neg B \end{aligned}$$

5. $A = B \to C$

$$\begin{aligned} w \Vdash_{S4} \overline{B \to C} &\iff w \Vdash_{S4} \Box(\overline{B} \to \overline{C}) \\ &\iff \forall w' \geq w \; w' \nVdash_{S4} \overline{B} \text{ oder } w' \Vdash_{S4} \overline{C} \\ &\overset{IV}{\iff} \forall w' \geq w \; w' \nVdash_{J4} B \text{ oder } w' \Vdash_{J4} C \\ &\iff w \Vdash_{J4} B \to C \end{aligned}$$

6. $A = \Diamond B$

$$\begin{aligned} w \Vdash_{S4} \overline{\Diamond B} &\iff w \Vdash_{S4} \Diamond \overline{B} \\ &\iff \exists w' \geq w \; w' \Vdash_{S4} \overline{B} \\ &\overset{IV}{\iff} \exists w' \geq w \; w' \Vdash_{J4} B \\ &\iff w \Vdash_{J4} \Diamond B \end{aligned}$$

7. $A = \forall x B$

$$
\begin{aligned}
w \Vdash_{S4} \overline{\forall x B} &\iff w \Vdash_{S4} \Box \forall x \overline{B} \\
&\iff \forall w' \geq w \, \forall c \in \bar{D}(w') \, w' \Vdash_{S4} \overline{B(c)} \\
&\overset{IV}{\iff} \forall w' \geq w \, \forall c \in \bar{D}(w') \, w' \Vdash_{J4} B(c) \\
&\iff w \Vdash_{J4} \forall x B
\end{aligned}
$$

8. $A = \exists x B$

$$
\begin{aligned}
w \Vdash_{S4} \overline{\exists x B} &\iff w \Vdash_{S4} \exists x \overline{B} \\
&\iff \exists c \in \bar{D}(w') \, w' \Vdash_{S4} \overline{B(c)} \\
&\overset{IV}{\iff} \exists c \in \bar{D}(w') \, w' \Vdash_{J4} B(c) \\
&\iff w \Vdash_{J4} \exists x B
\end{aligned}
$$

□

Lemma 5.3.6 *Gibt es ein J4-Modell, in dem die Formel A nicht gilt, so gibt es ein S4-Modell, in dem $\overline{A}$ nicht gilt.*

Beweis: Sei $(T, \leq, D, \bar{D}, \Vdash_{J4})$ ein J4-Kripke-Modell, so daß für ein $w \in T$ $w \not\Vdash_{J4} A$ gilt. Das S4-Modell sei $(T, \leq, D, \bar{D}, \Vdash_{S4})$, wobei $\Vdash_{S4}$ für Primformeln P durch $w \Vdash_{S4} P :\Leftrightarrow w \Vdash_{J4} P$ und für die übrigen Formeln wie üblich definiert ist. Für Primformeln P gilt:

$$
\begin{aligned}
w \Vdash_{S4} \overline{P} &\iff w \Vdash_{S4} \Box P \\
&\iff \forall w' \geq w \, w' \Vdash_{S4} P \\
&\iff \forall w' \geq w \, w' \Vdash_{J4} P \\
&\iff w \Vdash_{J4} P
\end{aligned}
$$

Die Behauptung folgt aus Lemma 5.3.5. □

Lemma 5.3.7 *Gibt es ein S4-Modell, in dem die Formel $\overline{A}$ nicht gilt, so gibt es ein J4-Modell, in dem A nicht gilt.*

Beweis: Sei $(T, \leq, D, \bar{D}, \Vdash_{S4})$ ein S4-Kripke-Modell, so daß für ein $w \in T$ $w \not\Vdash_{S4} \overline{A}$ gilt. Das J4-Modell sei $(T, \leq, D, \bar{D}, \Vdash_{J4})$, wobei $\Vdash_{J4}$ für Primformeln P durch $w \Vdash_{J4} P :\Leftrightarrow w \Vdash_{S4} \overline{P}$ und für die übrigen Formeln wie üblich definiert ist. Die Behauptung folgt aus Lemma 5.3.5. □

Satz 5.3.8 *Sei A eine Formel aus $L(C^*)$. Dann gilt* $\models_{S4} \overline{A} \Longleftrightarrow \models_{J4} A$

Beweis: Lemma 5.3.6 und Lemma 5.3.7. □

Lemma 5.3.9 (Abschwächungslemma) *Ist in S4 oder J4 die Sequenz $\Gamma \Rightarrow \Delta$ herleitbar, so ist auch die Sequenz $\Lambda, \Gamma \Rightarrow \Delta, \Theta$ herleitbar.*

Beweis: Herleitungsinduktion. □

Satz 5.3.10 *Sei in S4 eine Sequenz $\hat{\Sigma}$ herleitbar, für die gilt:*

1. *in $\hat{\Sigma}$ treten nur Formeln der Gestalt*
   (a) $\overline{A}$
   (b) $\neg\overline{A}, \overline{A} \to \overline{B}, \forall x \overline{A}$ *und Primformeln auf.*
2. *tritt eine Formel aus 1b im Sukzedens von $\hat{\Sigma}$ auf, so besteht der Antezedens von $\hat{\Sigma}$ nur aus Formeln der Gestalt $\Box B$, der übrige Sukzedens nur aus Formeln der Gestalt $\Diamond B$.*

Dann ist die Sequenz Σ, die durch Ersetzen aller Formelteile $\overline{A}$ durch ihre Urbilder A entsteht, in J4 herleitbar.

Beweis: Der Beweis wird durch S4-Herleitungsinduktion geführt. Mit $\hat{A}$ bzw. $\hat{\Gamma}$ und $\hat{\Delta}$ werden Formeln bzw. Mengen von Formeln bezeichnet, die Punkt 1 der Voraussetzungen genügen. A, Γ und Δ bezeichnen dann die entsprechenden (Mengen von) Formeln, in denen alle Auftreten von Formeln der Gestalt $\overline{B}$ durch ihre Urbilder B ersetzt worden sind.

1. Induktionsanfang: $\hat{\Sigma}$ ist ein S4-Axiom. $\hat{\Sigma} = \hat{\Gamma}, \hat{A} \Rightarrow \hat{A}, \hat{\Delta}$. Dann ist $\Sigma = \Gamma, A \Rightarrow A, \Delta$ ein J4-Axiom.

2. Induktionsschritt:

   (a) Der letzte Schluß war $\Rightarrow \vee$

   $$\frac{\hat{\Gamma} \Rightarrow \overline{A}, \overline{B}, \hat{\Delta}}{\hat{\Gamma} \Rightarrow \overline{A} \vee \overline{B}, \hat{\Delta}}$$

   Nach Induktionsvoraussetzung ist dann $\Gamma \Rightarrow A, B, \Delta$ in J4 herleitbar und man erhält $\Gamma \Rightarrow A \vee B, \Delta$ mit demselben Schluß.

   (b) Für $\Rightarrow \wedge$, $\Rightarrow \exists$, $\Rightarrow \Diamond$, $\vee \Rightarrow$, $\exists \Rightarrow$ und $\wedge \Rightarrow$ ist die Argumentation dieselbe.

   (c) Der letzte Schluß war $\Box \Rightarrow$. Dann ist $\hat{\Sigma} = \hat{\Gamma}, \hat{C} \Rightarrow \hat{\Delta}$ mit $C = \overline{P}$ für Primformel P, $\overline{\neg A}, \overline{A \to B}$ oder $\overline{\forall x A}$. Sei $C = \overline{\neg A}$ (die anderen Fälle sind analog zu behandeln).

   $$\frac{\hat{\Gamma}, \neg\overline{A} \Rightarrow \hat{\Delta}}{\hat{\Gamma}, \Box\neg\overline{A} \Rightarrow \hat{\Delta}}$$

   Nach Induktionsvoraussetzung ist dann $\Gamma, \neg A \Rightarrow \Delta$ J4-herleitbar. Das ist gerade Σ.

   (d) Der letzte Schluß war $\Rightarrow \Box$. Dann ist $\hat{\Sigma} = \hat{\Gamma} \Rightarrow \overline{C}, \hat{\Delta}$ mit $C = \overline{P}$ für Primformel P, $\overline{\neg A}, \overline{A \to B}$ oder $\overline{\forall x A}$. Sei $C = \overline{\neg A}$ (die beiden anderen Fälle sind analog zu behandeln).

   $$\frac{\hat{\Gamma}* \Rightarrow \neg\overline{A}, \hat{\Delta}*}{\hat{\Gamma} \Rightarrow \Box\neg\overline{A}, \hat{\Delta}}$$

   $\hat{\Gamma}*$ enthält nur Formeln der Form $\Box B$. Also müssen alle Formeln in $\hat{\Gamma}*$ die Form $\overline{P}$ für Primformel P, $\overline{\neg B}, \overline{\forall x B}$, oder $\overline{B \to C}$ haben. $\hat{\Delta}*$ enthält nur Formeln der Form $\Diamond B$. Deshalb sind alle Elemente von $\hat{\Delta}*$ Formeln der Gestalt $\overline{\Diamond B}$. Es sind also die Bedingungen 1 und 2 erfüllt. Nach Induktionsvoraussetzung ist also die Sequenz $\Gamma* \Rightarrow \neg A, \Delta*$ in J4 herleitbar. Mit derselben Herleitung ist dann auch die Abschwächung Σ zu beweisen.

   (e) Der letzte Schluß war $\Diamond \Rightarrow$. Wie der vorangegangene Fall zu behandeln.

(f) Der letzte Schluß war $\Rightarrow \neg$.

$$\frac{\hat{\Gamma}*, \overline{A} \Rightarrow \hat{\Delta}*}{\hat{\Gamma} \Rightarrow \neg\overline{A}, \hat{\Delta}}$$

Dieser Schluß kann wegen der ersten Voraussetzung nur Formeln der Form $\neg\overline{A}$ ergeben. Wegen der zweiten Voraussetzung bestehen der Antezedens und Sukzedens der Konklusion (abgesehen von der Hauptformel des Schlusses) nur aus $\Box$– bzw. $\Diamond$–Formeln. Mit der Induktionsvoraussetzung erhält man wie oben die Herleitbarkeit von $\Gamma*, A \Rightarrow \Delta*$.

$\hat{\Gamma}*$ kann nur Formeln der Gestalt $\Box B$ enthalten. Deshalb müssen alle Formeln in $\hat{\Gamma}*$ die Form $\overline{P}$ für Primformel P, $\overline{\neg B}, \overline{\forall x B}$, oder $\overline{B \to C}$ haben. $\hat{\Delta}*$ enthält nur Formeln der Form $\Diamond B$. Also sind alle Elemente von $\hat{\Delta}*$ Formeln der Gestalt $\overline{\Diamond B}$.

Aus diesem Grund haben $\Gamma*$ und $\Delta*$ schon die erforderliche Form für den J4-Schluß $\Rightarrow \neg$, und es ergibt sich $\Gamma* \Rightarrow \neg A, \Delta*$. Mit derselben Herleitung ist dann auch $\Gamma \Rightarrow \neg A, \Delta$ J4-beweisbar.

(g) Der letzte Schluß war $\Rightarrow\to$ oder $\Rightarrow \forall$. Diese Fälle sind genau wie der vorangegangene zu behandeln.

(h) Der letzte Schluß war $\neg \Rightarrow$, $\forall \Rightarrow$ oder $\to\Rightarrow$. Diese Schlüsse sind wieder einfach abzuhandeln. Als Beispiel diene

$$\frac{\hat{\Gamma} \Rightarrow \overline{A}, \hat{\Delta}}{\hat{\Gamma}, \neg\overline{A} \Rightarrow \hat{\Delta}}$$

Nach Induktionsvoraussetzung ist $\Gamma \Rightarrow A, \Delta$ J4-herleitbar. Mit $\neg \Rightarrow$ ist dann auch $\Gamma, \neg A \Rightarrow \Delta$ J4-herleitbar. Die anderen Fälle verlaufen analog. $\Box$

Korollar 5.3.11 $S4 \vdash \overline{A} \Longrightarrow J4 \vdash A$

Beweis: Setze in Satz 5.3.10 $\hat{\Sigma} = \quad \Rightarrow \overline{A}$. $\Box$

Korollar 5.3.12 (Vollständigkeit von J4) *J4 ist vollständig bezüglich der in 5.1.2 angegebenen Semantik.*

Beweis: Sei A eine J4-Formel mit $\models_{J4} A$. Wegen Satz 5.3.8 gilt $\models_{S4} \overline{A}$. Aus der Vollständigkeit von S4 folgt $\vdash_{S4} \overline{A}$. Mit Korollar 5.3.11 erhält man $\vdash_{J4} A$. $\Box$

Lemma 5.3.13 *Seien* * *und* $*$ *die in den Sequenzenkalkülen für J4 bzw. S4 definierten Operatoren. Sei* $\Gamma \Rightarrow \Delta$ *eine J4-Sequenz. Dann gilt:* $\overline{\Gamma^\star} \subseteq (\overline{\Gamma})*$ *und* $\overline{\Delta^\star} \subseteq (\overline{\Delta})*$

Beweis:

1. Sei $C \in \overline{\Gamma^\star}$. Dann ist $C = \overline{B}$ und B ist eine Primformel, $\neg$–, $\rightarrow$–, oder $\forall$–Formel. $C = \overline{B} \in \overline{\Gamma}$ hat also die Form $\Box D$. Also ist $C \in \overline{\Gamma}*$.
2. Sei $C \in \overline{\Delta^\star}$. Dann ist $C = \overline{B}$ und B ist eine $\Diamond$–Formel. $C = \overline{B} \in \overline{\Delta}$ hat also die Form $\Diamond D$. Also ist $C \in \overline{\Delta}*$. $\Box$

Satz 5.3.14 *Ist die Sequenz* Σ *in J4 herleitbar, so ist* $\overline{\Sigma}$ *in S4 herleitbar.*

Beweis: Der Beweis wird durch Herleitungsinduktion geführt.

1. Σ ist ein J4-Axiom. Dann ist $\overline{\Sigma}$ ein S4-Axiom.
2. Der letzte Schluß in der Herleitung von Σ war $\wedge \Rightarrow$, $\vee \Rightarrow$, $\exists \Rightarrow$, $\Rightarrow \wedge$, $\Rightarrow \vee$, $\Rightarrow \exists$ oder $\Rightarrow \Diamond$. Dann wird $\overline{\Sigma}$ wegen der Induktionsvoraussetzung mit demselben Schluß hergeleitet.
3. Der letzte Schluß war $\neg \Rightarrow$

$$\frac{\Gamma \Rightarrow A, \Delta}{\Gamma, \neg A \Rightarrow \Delta}$$

Nach Induktionsvoraussetzung ist dann $\overline{\Gamma} \Rightarrow \overline{A}, \overline{\Delta}$ in S4 herleitbar. Der Induktionsschritt ist dann

$$\cfrac{\cfrac{\cfrac{\overline{\Gamma} \Rightarrow \overline{A}, \overline{\Delta}}{\overline{\Gamma}, \neg\overline{A} \Rightarrow \overline{\Delta}}}{\overline{\Gamma}, \Box\neg\overline{A} \Rightarrow \overline{\Delta}}}{\overline{\Gamma}, \overline{\neg A} \Rightarrow \overline{\Delta}}$$

4. Genauso werden die Fälle $\to\Rightarrow$ und $\forall\Rightarrow$ behandelt.

5. Der letzte Schluß war $\Rightarrow\neg$

$$\frac{\Gamma^\star, A \Rightarrow \Delta^\star}{\Gamma \Rightarrow \neg A, \Delta}$$

Nach Induktionsvoraussetzung ist $\overline{\Gamma^\star}, \overline{A} \Rightarrow \overline{\Delta^\star}$ S4-herleitbar. Wegen Lemma 5.3.13 ist auch $\overline{\Gamma}*, \overline{A} \Rightarrow \overline{\Delta}*$ S4-herleitbar. Der Induktionsschritt ist dann

$$\cfrac{\cfrac{\cfrac{\overline{\Gamma}*, \overline{A} \Rightarrow \overline{\Delta}*}{\overline{\Gamma}* \Rightarrow \neg\overline{A}, \overline{\Delta}*}}{\overline{\Gamma} \Rightarrow \Box\neg\overline{A}, \overline{\Delta}}}{\overline{\Gamma} \Rightarrow \overline{\neg A}, \overline{\Delta}}$$

6. Die Fälle $\Rightarrow\to$, $\Rightarrow\forall$ und $\Diamond\Rightarrow$ werden analog abgehandelt. $\Box$

Korollar 5.3.15 $J4 \vdash A \Longrightarrow S4 \vdash \overline{A}$

Beweis: Setze in Satz 5.3.14 $\Sigma = \quad \Rightarrow A$. $\Box$

Daraus folgt ein zweiter Beweis der Korrektheit von J4.

Korollar 5.3.16 (Korrektheit von J4) *Jede mit dem J4-Sequenzenkalkül bewiesene Formel (Sequenz) ist gültig.*

Beweis: Sei A eine J4-Formel mit $\vdash_{J4} A$. Wegen Korollar 5.3.15 gilt $\vdash_{S4} \overline{A}$. Aus der Korrektheit von S4 folgt $\models_{S4} \overline{A}$. Mit Satz 5.3.8 erhält man $\models_{J4} A$. $\Box$

5.4 Eigenschaften von J4

In diesem Abschnitt werden verschiedene Ergebnisse über J4 zusammengetragen.

Korollar 5.4.1 (Schnittregel) *Für J4 gilt die Regel*

$$\frac{\Gamma \Rightarrow B, \Delta \quad \Lambda, B \Rightarrow \Theta}{\Gamma, \Lambda \Rightarrow \Delta, \Theta}$$

Beweis: Aus der Herleitbarkeit von $\Gamma \Rightarrow B, \Delta$ und $\Lambda, B \Rightarrow \Theta$ folgt die Gültigkeit der Prämissen und daraus die Gültigkeit von $\Gamma, \Lambda \Rightarrow \Delta, \Theta$. Wegen der Vollständigkeit von J4 ist die Konklusion ebenfalls herleitbar. $\Box$

Korollar 5.4.2 *Für J4 gilt: aus* $\vdash A \to B$ *und* $\vdash B \to C$ *folgt* $\vdash A \to C$, *d.h. die Schnittregel ist für J4 zulässig.*

Beweis: Aus $\Vdash A \to B$ und $\Vdash B \to C$ folgt $\Vdash A \to C$. Wegen der Vollständigkeit von J4 ist das äquivalent zu $\vdash A \to C$. $\Box$

Definition 5.4.3 *Die* $\Diamond$*–schwachen Formeln der Sprache für J4 sind wie folgt induktiv definiert:*

1. *Primformeln,* $\neg$*–,* $\to$*– und* $\forall$*–Formeln sind* $\Diamond$*–schwach.*
2. *Sind* A *und* B $\Diamond$*–schwach, so sind auch* $A \wedge B$, $A \vee B$ *und* $\exists x A$ $\Diamond$*–schwach.*

Bemerkung 5.4.4 *Die syntaktische Eigenschaft,* $\Diamond$*-schwach zu sein, spiegelt sich in der Semantik wider: ist* A *eine* $\Diamond$*-schwache Formel, so gilt für ein Modell* $(T, \leq, D, \bar{D}, \Vdash)$ *von J4 und für* $w \in T$

$$w \Vdash A \Longrightarrow \forall w' \geq w\, w' \Vdash A$$

Beweis: Induktion nach der Definition der $\Diamond$-schwachen Formeln. $\Box$

Üblicherweise gehört zu einer Modallogik (wie bei S4 gesehen) ein zweiter modaler Operator, nämlich $\Box$. Die Bedeutung von $\Box A$ soll sein, daß A in einem gewissen Sinn *notwendig* ist. In Analogie zu den Definitionen 5.1.2/6 und 5.3.2/6 soll für J4 gelten: $w \Vdash \Box A \Longleftrightarrow \forall w' \geq w\, w' \Vdash A$.

Definition 5.4.5 *Sei $\top$ eine Abkürzung für eine herleitbare J4-Formel, z.B. $P_0 \to P_0$ für ein ausgezeichnetes Aussagezeichen P_0. $\Box A$ soll fortan als Abkürzung für $\top \to A$ stehen.*

Lemma 5.4.6 $w \Vdash \Box A \iff \forall w' \geq w\, w' \Vdash A$.

Beweis:

$$\begin{aligned} w \Vdash \Box A &\iff w \Vdash \top \to A \\ &\iff \forall w' \geq w\, w' \nVdash \top \text{ oder } w' \Vdash A \\ &\iff \forall w' \geq w\, w' \Vdash A \end{aligned}$$

Satz 5.4.7 (Deduktionstheorem für J4) *Gilt B in allen Modellen, in denen A gilt, so ist die Implikation $\Box A \to B$ in J4 herleitbar und umgekehrt:*

$$A \models B \iff \vdash \Box A \to B$$

Beweis:

"$\Longleftarrow$" Sei $\mathcal{M} = (T, \leq, D, \bar{D}, \Vdash)$ ein Modell, in dem A gilt: $\mathcal{M} \models A$. Zu zeigen ist $\mathcal{M} \models B$. Es gilt $v \Vdash A$ für alle $v \in T$, weil A in $\mathcal{M}$ gilt. Deshalb gilt auch $v \Vdash \Box A$ für alle $v \in T$. Sei $w \in T$ beliebig.

$$\begin{array}{lll} & w \Vdash \Box A \to B & \text{nach Voraussetzung} \\ \iff & \forall w' \geq w\, w' \nVdash \Box A \text{ oder } w' \Vdash B & \text{nach Definition} \\ \Longrightarrow & \forall w' \geq w\, w' \Vdash B & \text{da } \mathcal{M} \models A \\ \Longrightarrow & w \Vdash B & \end{array}$$

Da $\mathcal{M}$ und w beliebig waren, folgt $A \models B$.

"$\Longrightarrow$" Es gilt $A \models B$. Angenommen, es gilt $\nvdash \Box A \to B$. Dann gibt es ein Modell $\mathcal{M} = (T, \leq, D, \bar{D}, \Vdash)$, in dem $\Box A \to B$ nicht gilt. Das heißt, es gibt ein $w \in T$ mit $w \Vdash \Box A$ und $w \nVdash B$. Es sei $T' = T \cap \{u \mid u \in T, u \geq w\}$. Dann ist $\mathcal{M}' = (T', \leq, D, \bar{D}, \Vdash)$ ein Modell, in dem A gilt. Aus $\mathcal{M}' \models A$ folgt wegen der Voraussetzung $\mathcal{M}' \models B$. Also gilt $w \Vdash B$ im Widerspruch zur Annahme. $\Box$

Bemerkung 5.4.8 *Für $\Diamond$–schwaches A gilt:*

$$A \models B \Longrightarrow \ \vdash A \to B$$

Beweis: Ähnlich wie "$\Longrightarrow$" aus dem vorangegangenen Satz.

Lemma 5.4.9 1. $\dfrac{\Gamma, A \Rightarrow \Delta}{\Gamma, \Box A \Rightarrow \Delta} \Box \Rightarrow$ *ist eine für J4 korrekte Regel.*

2. $\dfrac{\Gamma^\star \Rightarrow A, \Delta^\star}{\Gamma \Rightarrow \Box A, \Delta} \Rightarrow \Box$ *ist eine für J4 korrekte Regel. Dabei gilt:*

$$\begin{array}{lll} \Gamma^\star & := & \{A \in \Gamma \mid A = \neg B, B \to C, \forall x B \text{ oder } A \text{ ist eine Primformel}\} \\ \Delta^\star & := & \{\Diamond A \mid \Diamond A \in \Delta\} \end{array}$$

Beweis:

1. Sei $\Gamma, A \Rightarrow \Delta$ herleitbar. $\Gamma \Rightarrow \top, \Delta$ ist stets herleitbar, weil $\top$ eine herleitbare Formel ist. Mit einem $\to\Rightarrow$–Schluß erhält man

$$\dfrac{\dfrac{\Gamma \Rightarrow \top, \Delta \quad \Gamma, A \Rightarrow \Delta}{\Gamma, \top \to A \Rightarrow \Delta}}{\Gamma, \Box A \Rightarrow \Delta}$$

2. Sei $\Gamma^\star \Rightarrow A, \Delta^\star$ herleitbar. Dann ist mit denselben Schlüssen auch $\Gamma^\star, \top \Rightarrow A, \Delta^\star$ herleitbar. Mit einem $\Rightarrow\to$–Schluß erhält man:

$$\dfrac{\dfrac{\Gamma^\star, \top \Rightarrow A, \Delta^\star}{\Gamma \Rightarrow \top \to A, \Delta}}{\Gamma \Rightarrow \Box A, \Delta}$$

Die folgenden Lemmata beschreiben das Verhalten von $\Box$ und $\Diamond$, das nicht immer mit dem für andere Modallogiken bekannten übereinstimmt.

Lemma 5.4.10 1. $\vdash \neg\Box\neg A \to \Diamond A$

2. $\vdash \Box A \to \neg\Diamond\neg A$

Beweis:

1.

$$
\begin{array}{cl}
A \Rightarrow A & \\ \hline
A \Rightarrow \Diamond A & \Rightarrow \Diamond \\ \hline
\Rightarrow \Diamond A, \neg A & \Rightarrow \neg \\ \hline
\Rightarrow \Diamond A, \Box \neg A & \Rightarrow \Box \\ \hline
\neg \Box \neg A \Rightarrow \Diamond A & \neg \Rightarrow \\ \hline
\Rightarrow \neg \Box \neg A \rightarrow \Diamond A & \Rightarrow \rightarrow
\end{array}
$$

2.

$$
\begin{array}{cl}
A \Rightarrow A & \\ \hline
A, \neg A \Rightarrow & \neg \Rightarrow \\ \hline
\Box A, \neg A \Rightarrow & \Box \Rightarrow \\ \hline
\Box A, \Diamond \neg A \Rightarrow & \Diamond \Rightarrow \\ \hline
\Box A \Rightarrow \neg \Diamond \neg A & \Rightarrow \neg \\ \hline
\Rightarrow \Box A \rightarrow \neg \Diamond \neg A & \Rightarrow \rightarrow
\end{array}
$$

Lemma 5.4.11 *1.* $\vdash \neg \Diamond \neg A \leftrightarrow \neg \neg A$

2. $\vdash \neg \Box \neg A \leftrightarrow \neg \neg A$

Beweis: Nur der Fall $\vdash \neg \Box \neg A \rightarrow \neg \neg A$ wird gezeigt.

$$
\begin{array}{cl}
\neg A \Rightarrow \neg A & \\ \hline
\neg A \Rightarrow \Box \neg A & \Rightarrow \Box \\ \hline
\neg \Box \neg A, \neg A \Rightarrow & \neg \Rightarrow \\ \hline
\neg \Box \neg A \Rightarrow \neg \neg A & \Rightarrow \neg \\ \hline
\Rightarrow \neg \Box \neg A \rightarrow \neg \neg A & \Rightarrow \rightarrow
\end{array}
$$

Die anderen drei Herleitungen verlaufen ähnlich. $\Box$

In J4 sind $\Box A$ und $\Diamond A$ nicht wechselseitig definierbar, d.h. im Gegensatz zu z.B. S4 gilt weder $\Diamond A \leftrightarrow \neg \Box \neg A$, noch $\Box A \leftrightarrow \neg \Diamond \neg A$. Deshalb ergeben sich Unterschiede bei der Herleitbarkeit von Formeln, die in S4 äquivalent sind.

Bemerkung 5.4.12 *1. Für Formeln A der Form $\Diamond B$ gilt:*

$$\vdash \neg\neg A \to A$$

2. Für das S5-Axiom gilt:

$$\nvdash \Diamond A \to \Box\Diamond A$$

Lemma 5.4.13 *Für $\Diamond$–schwache Formeln A gilt $\vdash A \leftrightarrow \Box A$.*

Beweis: nach der induktiven Definition von $\Diamond$–schwachen Formeln.

"$\to$" (a) Sei $A = P$ für eine Primformel P, $A = \neg B$, $A = B \to C$ oder $A = \forall x B(x)$. Dann ist

$$\dfrac{\dfrac{A \Rightarrow A}{A \Rightarrow \Box A}\;{\Rightarrow}\,\Box}{\Rightarrow A \to \Box A}\;{\Rightarrow}{\to}$$

eine korrekte Herleitung.

(b) Sei $A = B \wedge C$, $A = B \vee C$ oder $\exists x B(x)$.

$$\dfrac{\dfrac{\dfrac{\dfrac{\text{Ax.} \qquad \dfrac{\dfrac{\dfrac{\dfrac{C, B \Rightarrow B \qquad C, B \Rightarrow C}{C, B \Rightarrow B \wedge C}\;{\Rightarrow}\wedge}{C, \Box B \Rightarrow B \wedge C}\;\Box{\Rightarrow}}{\Box C, \Box B \Rightarrow B \wedge C}\;\Box{\Rightarrow}}{\Box C, \Box B, B, C \Rightarrow \Box(B \wedge C)}\;{\Rightarrow}\Box}{\Box C, B \to \Box B, B, C \Rightarrow \Box(B \wedge C) \qquad \text{Ax.}}\;{\to}{\Rightarrow}}{B \to \Box B, C \to \Box C, B, C \Rightarrow \Box(B \wedge C)}\;{\to}{\Rightarrow}}{C \to \Box C, B, C \Rightarrow \Box(B \wedge C)}\;(*)}{\dfrac{B, C \Rightarrow \Box(B \wedge C)}{B \wedge C \Rightarrow \Box(B \wedge C)}\;\wedge{\Rightarrow}}\;(*)$$

Bei den mit (*) gekennzeichneten Stellen geht jeweils die Induktionsvoraussetzung und eine Anwendung der Schnittregel ein; Bei den mit “Ax.” bezeichneten Stellen hat sich jeweils eine Basissequenz, also ein Axiom, ergeben. Die Beweise für die beiden anderen Fälle verlaufen analog.

“$\leftarrow$” klar. $\Box$

Bemerkung 5.4.14 *Das vorangegangene Lemma rechtfertigt eine alternative Definition für $\Gamma^\star$ im Sequenzenkalkül für J4: statt $\Gamma^\star := \{A \mid A \in \Gamma, A = \neg B, B \to C, \forall x B$ oder Primformel$\}$ könnte man auch definieren $\Gamma^\star := \{A \mid A$ ist $\Diamond$–schwach$\}$.*

Ein Blick auf den Sequenzenkalkül für J4 zeigt, daß er für $\Diamond$–freie Formeln derselbe ist, wie der für J. Das heißt, daß jede $\Diamond$–freie Formel, die in J4 herleitbar ist, auch in J herleitbar ist.

Die Barcan-Formel $\forall x \Box A(x) \to \Box \forall x A(x)$ ist in S4 nicht herleitbar. Die Hinzunahme der Barcan-Formel zu S4 als Axiomenschema führt zu Modellen mit *konstanten Universen*, d.h. für Modelle $(T, \leq, D, \bar{D}, \Vdash)$ von S4 + Barcan-Formel gilt $\bar{D}(v) = \bar{D}(w)$ für alle $v, w \in T$.

Lemma 5.4.15 *Für die Barcan-Formel gilt in J4:*

1. $\vdash \forall x \Box A(x) \to \Box \forall x A(x)$
2. $\nvdash \forall x \neg \Diamond \neg A(x) \to \neg \Diamond \neg \forall x A(x)$

Beweis:

1.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{A(a) \Rightarrow A(a)}{\Box A(a) \Rightarrow A(a)}\;\Box\Rightarrow}{\forall x \Box A(x) \Rightarrow A(a)}\;\forall\Rightarrow}{\forall x \Box A(x) \Rightarrow \forall x A(x)}\;\Rightarrow\forall}{\forall x \Box A(x) \Rightarrow \Box \forall x A(x)}\;\Rightarrow\Box}{\Rightarrow \forall x \Box A(x) \to \Box \forall x A(x)}\;\Rightarrow\to$$

2. Wegen Lemma 5.4.11 ist die Barcan-Formel

$$\forall x \neg \Diamond \neg A(x) \to \neg \Diamond \neg \forall x A(x)$$

äquivalent zu DNS[3], $\forall x \neg\neg A(x) \to \neg\neg \forall x A(x)$, einem bekannten intuitionistischen Nicht-Theorem. Wegen der vorangegangenen Bemerkung ist DNS nicht in J4 herleitbar.

5.5 Effizienzprobleme beim Sequenzenkalkül

Sequenzenkalküle suchen, einer vorgegebenen Strategie folgend, nach einer Herleitung für eine Formel ohne Einsicht in ihren syntaktischen Aufbau. Dabei treten in erster Linie zwei Probleme auf: das Relevanzproblem und das Problem der Reihenfolgeabhängigkeit der Schlußregeln.

Als Beispiel für das Relevanzproblem betrachte man die gültige Formel $(Q \to A) \wedge (A \wedge P) \to (Q \vee P)$. Darin seien P und Q Primformeln und A beliebig, aber so, daß P und Q keine Teilformeln von A sind. Eine Herleitung dieser Formel kann so beginnen:

$$\cfrac{\cfrac{\cfrac{Q \to A, A \wedge P \Rightarrow Q, P}{(Q \to A) \wedge (A \wedge P) \Rightarrow Q, P}}{(Q \to A) \wedge (A \wedge P) \Rightarrow (Q \vee P)}}{\Rightarrow (Q \to A) \wedge (A \wedge P) \to (Q \vee P)}$$

Jetzt besteht die Wahl zwischen der Reduktion von $Q \to A$ und $A \wedge P$. Ohne Rücksicht auf die innere Struktur der Formel wird ein Algorithmus, der z.B. alle Ausdrücke stets von links nach rechts abarbeitet, nicht bemerken, daß $Q \to A$ nichts zur Bildung einer Basissequenz beitragen kann, weil Q darin mit dem gleichen Signum auftritt, wie das Q auf der rechten Seite. Ist A obendrein eine komplexe Formel, wird der Algorithmus viel Zeit damit verschwenden, an ungeeigneter Stelle nach Material zur Bildung von Basissequenzen zu suchen.

[3]Double Negation Shift

Ein zweites Problem der Sequenzenkalküle ist die Reihenfolgeabhängigkeit der Regelanwendungen. Schon bei der modalen bzw. intuitionistischen Aussagenlogik kann es passieren, daß die Regeln, die zur Herleitung einer Formel führen, in anderer Reihenfolge keinen Beweis mehr liefern. Diese Reihenfolgeabhängigkeit resultiert zum einen aus den Variablenbedingungen bei den δ- und π-Regeln und zum anderen aus der Streichungsvorschrift bei der (inversen) Anwendung der π-Regel. Als Beispiel diene die Herleitung von $\Diamond\forall x A(x) \rightarrow \exists x \Diamond A(x)$:

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{A(c) \Rightarrow A(c)}{\forall x A(x) \Rightarrow A(c)}}{\forall x A(x) \Rightarrow \Diamond A(c)}}{\Diamond\forall x A(x) \Rightarrow \Diamond A(c)}}{\Diamond\forall x A(x) \Rightarrow \exists x \Diamond A(x)}}{\Rightarrow \Diamond\forall x A(x) \rightarrow \exists x \Diamond A(x)}$$

Diese Herleitung enthält mehrere Stellen, an denen eine sture Suche ("blind search") ohne *backtracking* ergebnislos bleiben würde: zum Beispiel muß der $\Diamond$-Operator im Antezedens *nach* dem $\exists$-Quantor, aber *vor* dem $\Diamond$-Operator im Sukzedens reduziert werden. Sonst ergäbe sich

$$\cfrac{\cfrac{\cfrac{\forall x A(x) \Rightarrow}{\Diamond\forall x A(x) \Rightarrow A(c)}}{\Diamond\forall x A(x) \Rightarrow \Diamond A(c)}}{\Rightarrow \Diamond\forall x A(x) \rightarrow \exists x \Diamond A(x)}$$

und damit keine Herleitung.

Matrixkalküle bringen bei den beiden geschilderten Problemen eine Effizienzverbesserung [Bib82b], [Wal87]. Im folgenden Kapitel wird ein Matrixkalkül für J4 definiert und gezeigt, wie das Relevanz- und das Reihenfolgeproblem durch das Unifizieren von Ausdrücken bearbeitet werden. Am Ende des Kapitels wird die eben untersuchte Formel noch einmal als Beispiel aufgegriffen.

Kapitel 6

Ein Matrixkalkül für J4

In diesem Kapitel wird ein Beweisverfahren für J4 vorgestellt, das effizienter ist als der im vorangegangen Kapitel beschriebene Sequenzenkalkül. Es wird ein Matrixkalkül für J4 (und damit auch für J) entwickelt, der unabhängig von der Einbettung von J4 (J) in S4 arbeitet. Wenn nicht ausdrücklich auf Unterschiede hingewiesen wird, kann man alles in diesem und den folgenden Kapiteln über J4 Gesagte auch für J übernehmen, in dem man die Aussagen nur für $\Diamond$-freie Formeln liest.

6.1 Formelbäume

Die Darstellung von Formeln als Baumstrukturen ist einer der Grundgedanken des Matrixkalküls. Der Einblick in den inneren Aufbau einer Formel ist von wesentlicher Bedeutung für die Effizienz des Matrixkalküls.

6.1.1 Formelbäume für geschlossene Formeln

Definition 6.1.1 (direkte Teilformel) *Seien A und B J4-Formeln.*

1. *A ist direkte Teilformel von $\neg A$ und $\Diamond A$.*
2. *A und B sind direkte Teilformeln von $A \wedge B$, $A \vee B$ und $A \rightarrow B$.*

3. A ist direkte Teilformel von $\exists x A$ und $\forall x A$.

Definition 6.1.2 (Teilformel) *Seien A, B und C J4-Formeln.*

1. *B ist eine Teilformel von A, wenn B direkte Teilformel von A oder gleich A ist.*
2. *Ist A Teilformel von B, B Teilformel von C, so ist A auch Teilformel von C.*

Definition 6.1.3 *Sei A eine geschlossene Formel. Ein Formelbaum für A enthält für jede Teilformel von A einen Knoten. Der Wurzelknoten entspricht der Formel A. Die Nachfolger jedes Knotens sind seine direkten Teilformeln.*

Als Bezeichnung für die Knoten eines Formelbaums, im folgenden Positionen *gennant, dienen $a_0, a_1, \ldots, b_0, b_1$, usw.*

form(a) bezeichnet die der Position a entsprechende Teilformel, d.h. die zu der Position gehörende Formel. Als Abkürzung für die Teilformelordnung schreibt man $a_i \gg a_j$, falls die Position a_i oberhalb der Position a_j im Formelbaum auftritt, d.h. form(a_j) ist Teilformel von form(a_i).

Die Bezeichnung $a_i \gg a_j$ soll andeuten, daß a_i in gewissem Sinne "größer" ist als a_j, wenn die a_j entsprechende Formel eine Teilformel der zu a_i gehörenden ist. Diese Definitionen werden am besten durch ein Beispiel erläutert. Abbildung 6.1 zeigt einen Formelbaum für die Formel $\Diamond(A \wedge B) \rightarrow \Diamond A \vee \Diamond B$.

6.1.2 Formelbäume für signierte Formeln

Das besondere Interesse gilt Formelbäumen für signierte Formeln der Gestalt FA, denn sie entsprechen Sequenzen der Form $\Rightarrow A$.

Definition 6.1.4 *Formelbäume für signierte Formeln enthalten an jeder Position außer der Teilformel selbst noch das Signum der Teilformel (*T *oder* F*), sowie den Primärtyp (α, β, γ, usw.) und den Sekundärtyp (α_1, α_2, usw.) der Teilformel entsprechend der Definition der Typen für die uniforme Darstellung in den Definitionen 5.1.8–5.1.13.*

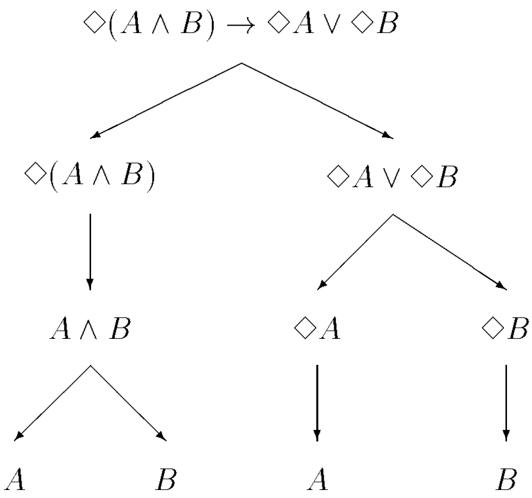


Abbildung 6.1: Ein Formelbaum für die Formel $\Diamond(A \wedge B) \rightarrow \Diamond A \vee \Diamond B$

Der Startknoten erhält den Sekundärtyp π_1, wenn er mit F signiert ist, und ν_1 sonst.

Abbildung 6.2 zeigt den Formelbaum für $\mathrm{F}(\Diamond(A \wedge B) \rightarrow \Diamond A \vee \Diamond B)$. Im folgenden werden in Formelbäumen an jedem Knoten zusätzliche Informationen notiert, und zwar Primär- und Sekundärtyp links oben bzw. unten, Formel und Signum in spitzen Klammern und die Bezeichnung der Position rechts oben. Positionen mit Sekundärtyp γ_0 oder ν_i für $i \in \{0, 1, 2\}$ werden durch Überstreichen hervorgehoben.

Definition 6.1.5 *Die Funktion* $\mathrm{sign}(a)$ *liefert die Polarität (das Signum* T *oder* F*) einer Position.* $\mathrm{sform}(a)$ *liefert sign(a)form(a), also die zur Position a gehörige signierte Formel.*

Für J macht es keinen Unterschied, ob die Blätter eines Formelbaums mit Primformeln oder komplexeren Formeln besetzt sind. In herleitbaren Formelschemata sind beliebige Formeln für die Platzhalter einsetzbar. In J4 gibt es Formelschemata, die nur für bestimmte Formelklassen herleitbar sind, z.B. nur für $\Diamond$-schwache Formeln. Ein Beispiel für eine solche Formel ist $A \rightarrow \Box A$. Deshalb spielt es für J4 eine Rolle, ob die Blätter eines Formelbaums mit

$$
\begin{tikzcd}[column sep=small]
 & & {}^{\pi}_{\pi_1}\langle \Diamond(A \wedge B) \rightarrow \Diamond A \vee \Diamond B, \mathrm{F}\rangle^{a_0} \arrow[dl] \arrow[dr] & & \\
 & {}^{\pi}_{\pi_1}\langle \Diamond(A \wedge B), \mathrm{T}\rangle^{a_1} \arrow[d] & & {}^{\alpha}_{\pi_2}\langle \Diamond A \vee \Diamond B, \mathrm{F}\rangle^{a_5} \arrow[dl] \arrow[dr] & \\
 & {}^{\alpha}_{\pi_1}\langle A \wedge B, \mathrm{T}\rangle^{a_2} \arrow[dl] \arrow[d] & {}^{\nu}_{\alpha_1}\langle \Diamond A, \mathrm{F}\rangle^{a_6} \arrow[d] & & {}^{\nu}_{\alpha_2}\langle \Diamond B, \mathrm{F}\rangle^{a_8} \arrow[d] \\
 {}_{\alpha_1}\langle A, \mathrm{T}\rangle^{a_3} & {}_{\alpha_2}\langle B, \mathrm{T}\rangle^{a_4} & {}_{\nu_1}\langle A, \mathrm{F}\rangle^{\bar{a}_7} & & {}_{\nu_1}\langle B, \mathrm{F}\rangle^{\bar{a}_9}
\end{tikzcd}
$$

Abbildung 6.2: Ein Formelbaum für eine signierte Formel

Primformeln besetzt sind oder nicht. In der Abbildung 6.2 sollen die Blätter nicht prim sein. Da nichts mehr über den inneren Aufbau bekannt ist, haben sie keinen Primärtyp. Anders ist die Situation, wenn A und B Primformeln darstellen. Dann haben die Blätter, anders als bei Smullyans uniformer Darstellung, den Typ ν bzw. π. In diesem Fall wird der Formelbaum um je eine Position ergänzt, und zwar erhält jede Primformel-Position einen Nachfolger vom Typ ν_1 bzw. π_1 mit derselben Formel und demselben Signum. Abbildung 6.3 zeigt den so erweiterten Formelbaum.

6.2 Multiplizitäten für Formelbäume

In manchen Herleitungen mit einem Sequenzenkalkül (z.B. dem für J4) ist es notwendig bestimmte Formeln (zum Beispiel Formeln der Gestalt $\forall x A(x)$ im Antezedens oder $\exists x A(x)$ im Sukzedens) mehrfach zu reduzieren, um Teilformeln mit verschiedenen Konstanten anstelle des x zu erhalten. Ein einfaches Beispiel ist $\forall x(A(x) \wedge B(x)) \rightarrow (A(c) \wedge B(d))$ für Konstanten c und d.

Formeln, die Anlaß zur Vervielfachung ihrer Teilformeln geben heißen *generativ*. Bei den generativen Formeln handelt es sich genau um die Formeln

$$
\begin{tikzcd}[column sep=small]
 & {}^{\pi}_{\pi_1}\langle \Diamond(A \wedge B) \to \Diamond A \vee \Diamond B, \mathrm{F}\rangle^{a_0} \arrow[d] \arrow[dr] & & \\
 & {}^{\pi}_{\pi_1}\langle \Diamond(A \wedge B), \mathrm{T}\rangle^{a_1} \arrow[d] & {}^{\alpha}_{\pi_2}\langle \Diamond A \vee \Diamond B, \mathrm{F}\rangle^{a_5} \arrow[d] \arrow[dr] & \\
 & {}^{\alpha}_{\pi_1}\langle A \wedge B, \mathrm{T}\rangle^{a_2} \arrow[dl] \arrow[d] & {}^{\nu}_{\alpha_1}\langle \Diamond A, \mathrm{F}\rangle^{a_6} \arrow[d] & {}^{\nu}_{\alpha_2}\langle \Diamond B, \mathrm{F}\rangle^{a_8} \arrow[d] \\
{}^{\nu}_{\alpha_1}\langle A, \mathrm{T}\rangle^{a_3} \arrow[d] & {}^{\nu}_{\alpha_2}\langle B, \mathrm{T}\rangle^{a_4} \arrow[d] & {}^{\pi}_{\nu_1}\langle A, \mathrm{F}\rangle^{\bar{a}_7} \arrow[d] & {}^{\pi}_{\nu_1}\langle B, \mathrm{F}\rangle^{\bar{a}_9} \arrow[d] \\
{}_{\nu_1}\langle A, \mathrm{T}\rangle^{\bar{b}_3} & {}_{\nu_1}\langle B, \mathrm{T}\rangle^{\bar{b}_4} & {}_{\pi_1}\langle A, \mathrm{F}\rangle^{b_7} & {}_{\pi_1}\langle B, \mathrm{F}\rangle^{b_9}
\end{tikzcd}
$$

Abbildung 6.3: Erweiterter Formelbaum für eine signierte Formel vom Typ ν und γ.

Bemerkung 6.2.1 (zur Notation) *Im folgenden bezeichnen* Γ_0, Δ_0, Π_i *und* $\boldsymbol{\nu}_i$ *die Mengen von Positionen eines Formelbaums vom Sekundärtyp* γ_0, δ_0, π_i *bzw.* ν_i, $i \in \{0, 1, 2\}$.

Die Anzahl der möglichen Wiederholungen der Teilformeln einer generativen Formel wird durch eine Funktion $\mu : \boldsymbol{\nu}_0 \cup \boldsymbol{\nu}_1 \cup \boldsymbol{\nu}_2 \cup \Gamma_0 \longrightarrow I\!N$ bestimmt. Sie setzt sich zusammen aus einer Funktion μ_Q, der quantorenlogischen Multiplizität, und μ_M, der modalen Multiplizität. μ_M ordnet jeder Position vom Sekundärtyp ν_1 bzw. ν_2 eine natürliche Zahl zu, die angibt, wieviele unterschiedliche Exemplare der Teilformel vom Typ ν_1 bzw. ν_2 im Formelbaum als Nachfolger der generativen ν-Formel auftreten dürfen. μ_Q ordnet jeder Position vom Sekundärtyp γ_0 und ν_0 eine natürliche Zahl zu, die angibt, wieviele unterschiedliche Exemplare der Teilformel vom Typ γ_0 bzw. ν_0 im Formelbaum als Nachfolger der generativen γ- bzw. ν-Formel auftreten dürfen.

Ist a eine Position in einem Formelbaum, so bezeichnet $\mu(a)$ also

$$\mu(a) \quad := \quad \begin{cases} \mu_M(a), & a \in \boldsymbol{\nu}_1 \cup \boldsymbol{\nu}_2 \\ \mu_Q(a), & a \in \boldsymbol{\nu}_0 \cup \Gamma_0 \end{cases}$$

Ist μ eine Multiplizität für die Formel A und a eine Position im Formelbaum für A, so unterscheidet man verschiedene Auftreten der Formel form(a) durch einen an die Position angefügten hochgestellten Index, z.B. a^λ. Dabei ist λ eine Folge von natürlichen Zahlen. Bei jedem neuen Auftauchen einer generativen Position wird diese Folge um eine Stelle verlängert, mit der die verschiedenen Auftreten der Teilformeln der generativen Formel gezählt werden. Der Formelbaum für die Formel A und die Multiplizität μ heißt *indizierter* Formelbaum und wird mit A^μ bezeichnet.

Definition 6.2.2 *Sei* A *eine Formel,* μ *eine Multiplizität für* A, a *eine Position im Formelbaum für* A *und* $a_1 \gg a_2 \gg \ldots \gg a_n \gtrdot a$ *die Positionen vom Typ* γ_0 *und* $\nu_i, i \in \{0, 1, 2\}$, *die oberhalb von* a *liegen. Sei* λ *eine Folge von natürlichen Zahlen. Dann ist* a^λ *eine* indizierte Position *im Formelbaum* A^μ, *wenn*

1. *$\mu(a_i) \neq 0, 1 \leq i \leq n$*

2. *$\lambda = \lambda_1\lambda_2 \ldots \lambda_n$, wobei $1 \leq \lambda_i \leq \mu(a_i), 1 \leq i \leq n$ ist.*

Die $\gg$-Ordnung wird wie folgt zu einer Ordnung $\gg^\mu$ auf dem indizierten Formelbaum ausgedehnt:

Definition 6.2.3 *Seien a^λ und b^κ Positionen in einem indizierten Formelbaum. Es gilt $a^\lambda \gg^\mu b^\kappa$, falls $a \gg b$ gilt und λ ein Anfangsstück von κ (in Zeichen: $\lambda \preceq \kappa$) ist.*

Primärtyp, Sekundärtyp und Signum einer indizierten Position sind die gleichen wie die der nicht indizierten Position.

Um die Schreibweise nicht zu schwerfällig werden zu lassen, wird das μ in $\gg^\mu$ fortan ausgelassen, wenn klar ist, daß es sich um die Ordnung auf einem durch μ indizierten Formelbaum handelt.

Die Funktion *form* ist für indizierte Formelbäume wie folgt definiert:

Definition 6.2.4 *Sei A eine Formel, und A^μ ein dazugehöriger indizierter Formelbaum.*

1. *form$(a^\lambda) = A$, wenn a die Wurzelposition des Baums ist.*

2. *Ist form$(a^\lambda) = B \wedge C$ und sind a_1^λ und a_2^λ die Nachfolger von a^λ, so ist form$(a_1^\lambda) = B$ und form$(a_2^\lambda) = C$.*

3. *Ist form$(a^\lambda) = B \vee C$ und sind a_1^λ und a_2^λ die Nachfolger von a^λ, so ist form$(a_1^\lambda) = B$ und form$(a_2^\lambda) = C$.*

4. *Ist form$(a^\lambda) = B \rightarrow C$ und sind a_1^κ und a_2^κ die Nachfolger von a^λ für ein κ mit $\lambda \preceq \kappa$, so ist form$(a_1^\kappa) = B$ und form$(a_2^\kappa) = C$.*

5. *Ist form$(a^\lambda) = \neg B$ und ist a_1^κ der Nachfolger von a^λ für ein κ mit $\lambda \preceq \kappa$, so ist form$(a_1^\kappa) = B$.*

6. *Ist form$(a^\lambda) = \Diamond B$ und ist a_1^κ der Nachfolger von a^λ für ein κ mit $\lambda \preceq \kappa$, so ist form$(a_1^\kappa) = B$.*

7. *Ist form*(a^λ) $= \exists x B(x)$ *und ist* a_1^κ *der Nachfolger von* a^λ *für ein* κ *mit* $\lambda \preceq \kappa$*, so ist form*(a_1^κ) $= B(a_1^\kappa)$.

8. *Ist form*(a^λ) $= \forall x B(x)$ *und ist* a_1^κ *der Nachfolger von* a^λ *für ein* κ *mit* $\lambda \preceq \kappa$*, so ist form*(a_1^κ) $= B(a_1^\kappa)$.

Anders als beim Sequenzenkalkül, wo Parameter verwendet werden, wird beim Matrixkalkül bei der Reduktion von Quantoren der Name der Position selbst anstelle der zuvor gebundenen Variablen in der Formel der Position eingesetzt.

Abbildung 6.4 zeigt den indizierten Formelbaum für $\mathrm{F}\forall x \Diamond A(x) \to \Diamond \forall x \Diamond A(x)$ mit $\mu(a_2) = 2$ und μ konstant 1 sonst.

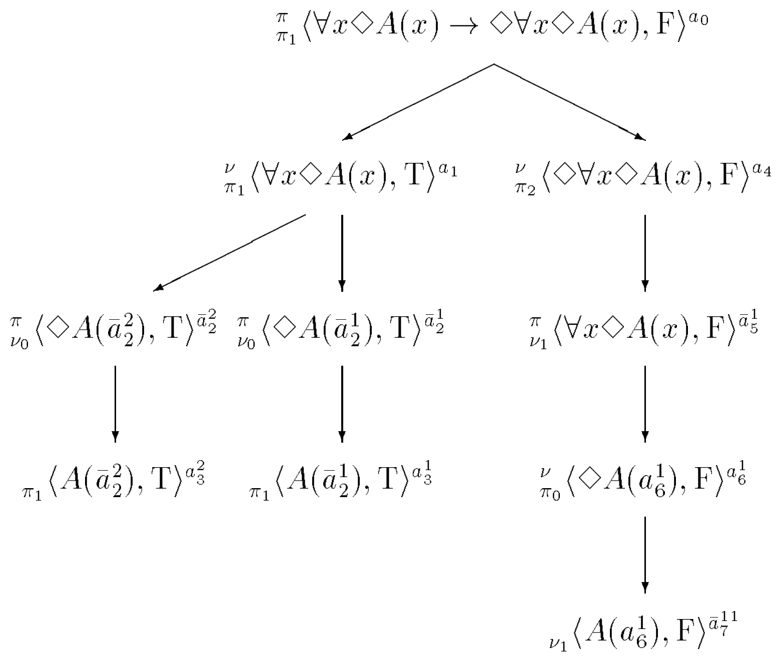


Abbildung 6.4: Ein indizierter Formelbaum für eine signierte Formel

Bemerkung 6.2.5 (zur Notation) *Für eine Multiplizität* μ *und einen indizierten Formelbaum* A^μ *bezeichne* $\Gamma_0(\mu)$ *die Positionen vom Typ* γ_0 *in* A^μ*. Die Schreibweise wird analog auch für die anderen Typen benutzt werden.*

6.3 Pfade und Konnektionen

Sei A^μ ein indizierter Formelbaum. Unter einem Pfad durch A^μ versteht man eine Teilmenge der Menge der Knoten des dazugehörigen indizierten Formelbaums. Pfade sind wie folgt induktiv definiert:

Definition 6.3.1 *1. Ist a_0 der Wurzelknoten des indizierten Formelbaums für A, so ist $\{a_0^\lambda\}$ ein Pfad;*

2. *ist $m \cup \{\alpha^\lambda\}$ ein Pfad, so ist auch $(m \setminus \{\alpha^\lambda\}) \cup \{\alpha_1^\lambda, \alpha_2^\lambda\}$ ein Pfad;*
3. *ist $m \cup \{\beta^\lambda\}$ ein Pfad, so sind auch $(m \setminus \{\beta^\lambda\}) \cup \{\beta_1^\lambda\}$ und $(m \setminus \{\beta^\lambda\}) \cup \{\beta_2^\lambda\}$ Pfade;*
4. *ist $m \cup \{\gamma^\lambda\}$ ein Pfad, so ist auch $m \cup \{\gamma^\lambda, \gamma_0^{\lambda l}\}$ ein Pfad für jedes l mit $1 \leq l \leq \mu(\gamma_0)$;*
5. *ist $m \cup \{\delta^\lambda\}$ ein Pfad, so ist auch $(m \setminus \{\delta^\lambda\}) \cup \{\delta_0^\lambda\}$ ein Pfad;*
6. *(a) ist $m \cup \{\nu^\lambda\}$ ein Pfad, so ist, wenn form(ν^λ) eine $\forall$-Formel ist, auch $m \cup \{\nu^\lambda, \nu_0^{\lambda l}\}$ ein Pfad für jedes l mit $1 \leq l \leq \mu(\nu_0)$;*

   (b) ist $m \cup \{\nu^\lambda\}$ ein Pfad, so ist, wenn form(ν^λ) eine $\neg$- oder $\Diamond$-Formel oder eine Primformel[1] ist, auch $m \cup \{\nu^\lambda, \nu_1^{\lambda l}\}$ ein Pfad für jedes l mit $1 \leq l \leq \mu(\nu_0)$;

   (c) ist $m \cup \{\nu^\lambda\}$ ein Pfad und ist form(ν^λ) eine $\rightarrow$-Formel, so sind auch $m \cup \{\nu^\lambda, \nu_1^{\lambda l}\}$ und $m \cup \{\nu^\lambda, \nu_2^{\lambda k}\}$ Pfade für jedes l und k mit $1 \leq l \leq \mu(\nu_1)$ und $1 \leq k \leq \mu(\nu_2)$;[2]
7. *(a) ist $m \cup \{\pi^\lambda\}$ ein Pfad, so ist, wenn form(π^λ) eine $\forall$-Formel ist, auch $(m \setminus \{\pi^\lambda\}) \cup \{\pi_0^\lambda\}$ ein Pfad;*

   (b) ist $m \cup \{\pi^\lambda\}$ ein Pfad, so ist, wenn form(π^λ) eine $\neg$ oder $\Diamond$-Formel oder eine Primformel[3] ist, auch $(m \setminus \{\pi^\lambda\}) \cup \{\pi_1^\lambda\}$ ein Pfad;

   (c) ist $m \cup \{\pi^\lambda\}$ ein Pfad und ist form(π^λ) eine $\rightarrow$-Formel, so ist $(m \setminus \{\pi^\lambda\}) \cup \{\pi_1^\lambda, \pi_2^\lambda\}$ ein Pfad.

---

[1]Der Primformel-Fall gilt nur für J4. In Formelbäumen für Formeln aus J haben Primformel-Positionen keine Nachfolger mehr. (vgl. Abschnitt 6.1.2)

[2]Man könnte ohne Einschränkung fordern, daß stets $\mu(\nu_2) = \mu(\nu_1)$ ist.

[3]siehe Anmerkung für Primformeln vom Typ ν

Wie aus der Definition abzulesen ist, bleiben die generativen Positionen, d.h. die Positionen vom Typ ν und γ, in einem Pfad bei ihrer Reduktion erhalten, während die anderen Positionen durch ihre Nachfolger ersetzt werden. Die Entwicklung der Pfade, die sich bei der Reduktion der Beispielformel $\mathrm{F}\forall x \Diamond A(x) \rightarrow \Diamond \forall x \Diamond A(x)$ aus Abbildung 6.4 ergeben, ist in der Abbildung 6.5 dargestellt.

$$
\begin{array}{ll}
\{a_0\} & \\
 & \pi\text{-Reduktion von } a_0 \\
\{a_1, a_4\} & \\
 & \nu\text{-Reduktion von } a_1 \\
\{a_1, \bar{a}_2^1, a_4\} & \\
 & \nu\text{-Reduktion von } a_1 \\
\{a_1, \bar{a}_2^1, \bar{a}_2^2, a_4\} & \\
 & \pi\text{-Reduktion von } \bar{a}_2^1 \\
\{a_1, a_3^1, \bar{a}_2^2, a_4\} & \\
 & \pi\text{-Reduktion von } \bar{a}_2^2 \\
\{a_1, a_3^1, a_3^2, a_4\} & \\
 & \nu\text{-Reduktion von } a_4 \\
\{a_1, a_3^1, a_3^2, a_4, \bar{a}_5^1\} & \\
 & \pi\text{-Reduktion von } \bar{a}_5^1 \\
\{a_1, a_3^1, a_3^2, a_4, a_6^1\} & \\
 & \nu\text{-Reduktion von } a_6^1 \\
\{a_1, a_3^1, a_3^2, a_4, a_6^1, \bar{a}_7^{11}\} &
\end{array}
$$

Abbildung 6.5: Pfade durch $\mathrm{F}\forall x \Diamond A(x) \rightarrow \Diamond \forall x \Diamond A(x)$

Zu jedem Pfad m gehören zwei Mengen von Positionen, die *assoziierte Menge* $\mathcal{A}(m)$ und die *dominierte Menge* $\mathcal{D}(m)$, die wie folgt definiert sind.

Definition 6.3.2 *Sei m ein Pfad durch einen indizierten Formelbaum. Die assoziierte Menge von m ist die Menge aller Positionen, die im Formelbaum über den Elementen von m liegen, und die Elemente von m selbst:*

$$\mathcal{A}(m) := \{b \mid b \geqslant a, a \in m\}$$

$\mathcal{A}(m)$ bezeichnet alle Positionen, die bei der Entstehung des Pfades m bereits reduziert worden sind und ihre durch diese Reduktion in den Pfad eingeführten direkten Nachfolger.

Die dominierte Menge $\mathcal{D}(m)$ wird aus den noch nicht reduzierten Positionen gebildet, die ausgehend von den Positionen des Formelbaums in $\mathcal{A}(m)$ noch erreicht werden können. Die *minimalen* Positionen eines Pfades sind die Positionen, die noch gar nicht reduziert worden sind, und die Eltern der Positionen, die noch nicht ihrer Multiplizität entsprechend oft in den Pfad eingeführt worden sind:

Definition 6.3.3 *Eine Position a^λ in einem Pfad m in einem indizierten Formelbaum A^μ heißt* minimal für *m,*

1. *wenn a^λ nicht vom Typ ν oder γ ist (denn in diesem Fall gibt es kein $b^\tau \in m$ gibt, so daß $a^\lambda \gg b^\tau$ gälte).*
2. *wenn a^λ vom Typ ν oder γ ist und es ein $l, 1 \leq l \leq \mu(\nu)$ bzw. $\mu(\gamma)$ gibt und $\nu_i^{\lambda l}, i \in \{0,1,2\}$ bzw. $\gamma_0^{\lambda l}$ noch nicht in $\mathcal{A}(m)$ ist.*

Die Menge der minimalen Positionen in m werde mit $\mathcal{M}(m)$ bezeichnet.

Definition 6.3.4 *Die Menge $\mathcal{D}(m)$ der durch den Pfad m dominierten Positionen eines indizierten Formelbaums ist*

$$\mathcal{D}(m) := \{a \mid b \gg a, \textit{ für ein } b \in \mathcal{M}(m) \textit{ und } a \notin \mathcal{A}(m)\}$$

Aus diesen Definitionen folgt sofort, daß für jeden Pfad m

$$\mathcal{A}(m) \cap \mathcal{D}(m) = \emptyset$$

gilt.

Definition 6.3.5 *Ein Pfad m durch einen indizierten Formelbaum heißt* atomar*, wenn $\mathcal{D}(m) = \emptyset$ ist.*

Die soeben definierten Begriffe sollen am Beispiel der Pfade in Abbildung 6.5 erläutert werden. Als Beispiel soll der dritte Pfad, $m = \{a_1, \bar{a}_2^1, a_4\}$ betrachtet werden. Für diesen Pfad ist $\mathcal{M}(m) = \{a_1, \bar{a}_2^1, a_4\}$, $\mathcal{A}(m) = \{a_0, a_1, a_4, \bar{a}_2^1\}$ und $\mathcal{D}(m) = \{\bar{a}_2^2, a_3^1, a_3^2, \bar{a}_5^1, a_6^1, \bar{a}_7^{11}\}$

Definition 6.3.6 *Eine* Konnektion *ist eine Teilmenge eines Pfades, die aus zwei Positionen besteht, für die die Formeln bis auf die Bezeichnungen der freien Variablen und Konstanten identisch sind und die unterschiedliches Signum haben. Eine Menge M von Konnektionen heißt* Basis *für die signierte Formel A, wenn für eine Multiplizität μ jeder atomare Pfad durch den indizierten Formelbaum A^μ eine Konnektion aus M enthält.*

Für das Beispiel in Abbildung 6.4 sind $\{a_3^1, \bar{a}_7^{11}\}$ und $\{a_3^2, \bar{a}_7^{11}\}$ Konnektionen.

6.4 Komplementarität

Für die klassische Aussagenlogik ist die Existenz einer Basis für eine signierte Formel FA schon gleichbedeutend mit der Herleitbarkeit von A [Wal87]. Für die intuitionistische Logik (sowohl J als auch J4) kann das keine hinreichende Bedingung sein, denn die Formel F$(A \vee \neg A)$ besitzt eine Basis, $A \vee \neg A$ ist aber nicht herleitbar. Vielmehr werden an die Elemente einer Konnektion noch weitere Bedingungen bezüglich ihres modalen Kontextes gestellt.

Definition 6.4.1 *Sei A eine Formel, μ eine Multiplizität für A. $T_M(\mu)$ bezeichnet die Menge aller Positionen des indizierten Formelbaums A^μ vom Typ ν_i und $\pi_i, i \in \{0, 1, 2\}$.*

Definition 6.4.2 *Sei b eine Position in einem indizierten Formelbaum A^μ. Seien $b_1, \ldots, b_n, n \geq 1$ die Positionen aus $T_M(\mu)$, die oberhalb von b liegen, d.h. $b_1 \gg \ldots \gg b_n \gneq b$.*

Die Zeichenkette $b_1 \ldots b_n$ ist das Präfix *der Position b.*

Das Präfix einer Position b wird mit *pre(b)* abgekürzt. $P(\mu)$ bezeichnet die Menge aller Präfixe der Positionen eines indizierten Formelbaums.

Sind p und q aus $P(\mu)$ so steht $p \preceq q$ ($p \prec q$) dafür, daß p ein (echtes) Anfangsstück von q ist, und $u \in p$ dafür, daß $u \in T_M(\mu)$ ist und u in p auftritt.

$T_M(\mu)^\star$ bezeichnet die Menge aller Zeichenketten, die aus Elementen von $T_M(\mu)$ gebildet werden können, einschließlich der leeren Zeichenkette ($\emptyset$).

Das Präfix jeder Position wird in Formelbäumen im folgenden an der noch freien Stelle rechts unten notiert. Als Beispiel dient nochmals der Formelbaum für $\forall x \Diamond A(x) \rightarrow \Diamond \forall x \Diamond A(x)$ in Abbildung 6.6.

$$
\begin{array}{ccc}
 & {}^{\pi}_{\pi_1}\langle \forall x \Diamond A(x) \rightarrow \Diamond \forall x \Diamond A(x), \mathrm{F}\rangle^{a_0}_{a_0} & \\
 & \swarrow \qquad \searrow & \\
 & {}^{\nu}_{\pi_1}\langle \forall x \Diamond A(x), \mathrm{T}\rangle^{a_1}_{a_0 a_1} & {}^{\nu}_{\pi_2}\langle \Diamond \forall x \Diamond A(x), \mathrm{F}\rangle^{a_4}_{a_0 a_4} \\
 \swarrow & \downarrow & \downarrow \\
{}^{\pi}_{\nu_0}\langle \Diamond A(\bar{a}_2^2), \mathrm{T}\rangle^{\bar{a}_2^2}_{a_0 a_1 \bar{a}_2^2} & {}^{\pi}_{\nu_0}\langle \Diamond A(\bar{a}_2^1), \mathrm{T}\rangle^{\bar{a}_2^1}_{a_0 a_1 \bar{a}_2^1} & {}^{\pi}_{\nu_1}\langle \forall x \Diamond A(x), \mathrm{F}\rangle^{\bar{a}_5^1}_{a_0 a_4 \bar{a}_5^1} \\
\downarrow & \downarrow & \downarrow \\
{}_{\pi_1}\langle A(\bar{a}_2^2), \mathrm{T}\rangle^{a_3^2}_{a_0 a_1 \bar{a}_2^2 a_3^2} & {}_{\pi_1}\langle A(\bar{a}_2^1), \mathrm{T}\rangle^{a_3^1}_{a_0 a_1 \bar{a}_2^1 a_3^1} & {}^{\nu}_{\pi_0}\langle \Diamond A(a_6^1), \mathrm{F}\rangle^{a_6^1}_{a_0 a_4 \bar{a}_5^1 a_6^1} \\
 & & \downarrow \\
 & & {}_{\nu_1}\langle A(a_6^1), \mathrm{F}\rangle^{\bar{a}_7^{11}}_{a_0 a_4 \bar{a}_5^1 a_6^1 \bar{a}_7^{11}}
\end{array}
$$

Abbildung 6.6: Ein indizierter Formelbaum mit Präfixen

Definition 6.4.3 (modale Substitution) *Sei μ eine Multiplizität und σ_M eine Abbildung von $T_M(\mu)$ nach $T_M(\mu)^\star$, die auf $T_M(\mu) \setminus (\boldsymbol{\nu}_0(\mu) \cup \boldsymbol{\nu}_1(\mu) \cup \boldsymbol{\nu}_2(\mu))$ die Identität ist.*

$\sigma_M^\#$ sei die homomorphe Erweiterung von σ_M auf Zeichenketten aus $T_M(\mu)^\star$.

$\sigma_M(\mu)$ *heißt* modale Substitution, *wenn stets gilt:*

$$\text{aus } u, v \in T_M(\mu), u \in \sigma_M(v) \text{ folgt } \sigma_M(u) = u$$

Diese letzte Bedingung ist keine starke Forderung. Sie besagt lediglich, daß die Substitution endlich durchführbar ist und nicht in einem unendlichen Zirkel läuft.

Definition 6.4.4 *Eine modale Substitution σ_M führt zu den beiden Relationen $\cong_M$ und $\rightsquigarrow_M$ auf $T_M(\mu) \times T_M(\mu)$. Dabei ist $\cong_M$ eine Äquivalenzrelation und*

1.

$$\left.\begin{array}{l} a \in \boldsymbol{\nu}_i(\mu), i \in \{0,1,2\}, \\ b \in \boldsymbol{\nu}_i(\mu), i \in \{0,1,2\} \\ \text{und } \sigma_M(a) = b \end{array}\right\} \Longrightarrow a \cong_M b$$

2.

$$\left.\begin{array}{l} a \in \boldsymbol{\nu}_i(\mu), i \in \{0,1,2\} \\ p \notin \boldsymbol{\nu}_i(\mu), i \in \{0,1,2\} \\ \text{und } \sigma_M(a) = p \end{array}\right\} \Longrightarrow b \rightsquigarrow_M a \text{ für alle } b \in p, b \notin \bigcup_{i=0}^{2} \boldsymbol{\nu}_i(\mu)$$

3. Gilt $b \rightsquigarrow_M a$ und $a \cong_M a'$, so folgt $b \rightsquigarrow_M a'$.

4. Sei a vom Typ ν und a_1, a_2 vom Sekundärtyp ν_1 bzw. ν_2 die Nachfolger von a. Für $i, j \in \{1,2\}, i \neq j$ folgt aus $b \rightsquigarrow_M a_i$, daß $b \rightsquigarrow_M a_j$.

Definition 6.4.5 *Eine modale Substitution σ_M heißt J4-zulässig (J-zulässig), wenn die transitive Hülle der Vereinigung der Ordnung auf dem indizierten Formelbaum und der $\rightsquigarrow_M$-Relation, kurz*

$$\rhd_M := (\gg \cup \rightsquigarrow_M)^+$$

irreflexiv ist.

Definition 6.4.6 (σ_M-Komplementarität) *Sei A^μ ein indizierter Formelbaum, σ_M eine J4- bzw. J-zulässige modale Substitution und $\{a, b\}$ eine Konnektion in A^μ. Diese Konnektion heißt* σ_M-komplementär, *wenn*

$$\sigma_M^\#(pre(a)) = \sigma_M^\#(pre(b))$$

sofern man die Namen der π_2-Positionen mit denen der zugehörigen π_1-Positionen identifiziert.

Ein Pfad durch A^μ heißt σ_M-komplementär, *wenn er eine σ_M-komplementäre Konnektion enthält.*

Im Fall des Sequenzenkalküls ist es wichtig, eine bestimmte Reihenfolge bei der Anwendung der (inversen) Regeln einzuhalten, um zu verhindern, daß Formelteile, die zur Bildung von Basissequenzen (Axiomen) nötig sind, zu früh gestrichen werden. Dieser Reihenfolge entspricht beim Matrixkalkül die $\triangleright_M$-Ordnung.

Nun müssen noch die im Formelbaum A^μ auftretenden Konstanten und die durch die Reduktion der Quantoren in die Formeln der Positionen eingeführten Positionsnamen berücksichtigt werden.

Definition 6.4.7 (quantorenlogische Substitution) *Sei A eine Formel und μ eine Multiplizität für A. Es sei*

$$T_Q(\mu) := \Gamma_0(\mu) \cup \Delta_0(\mu) \cup \boldsymbol{\nu}_0(\mu) \cup \Pi_0(\mu).$$

Eine quantorenlogische Substitution *ist eine Abbildung*

$$\sigma_Q : T_Q(\mu) \longrightarrow T_Q(\mu) \cup C,$$

wobei C die Menge der Konstanten aus der zugrundeliegenden Sprache L ist, die in A auftreten. Auf $\Delta_0(\mu) \cup \Pi_0(\mu)$ ist σ_Q die Identität.

Beim Sequenzenkalkül ist es wichtig, zunächst die Parameter, die durch Anwendung der (inversen) δ- und π-Regeln entstehen, in die Herleitung einzubringen, bevor die (inversen) γ- bzw. ν-Regeln angewandt werden, die Teilformeln ergeben, die denselben Parameter erhalten sollen. Beim Matrixkalkül erzeugen quantorenlogische Substitutionen Ordnungen, die entsprechenden Bedingungen genügen müssen.

Definition 6.4.8 *Eine quantorenlogische Substitution erzeugt eine Äquivalenzrelation* $\cong_Q$ *und eine Relation* $\rightsquigarrow_Q$ *auf* $T_Q \times T_Q$ *und zwar:*

1. $a, b \in \boldsymbol{\nu}_0(\mu) \cup \Gamma_0(\mu), \sigma_Q(a) = b \Longrightarrow a \cong_Q b$.
2. $a \in \boldsymbol{\nu}_0(\mu) \cup \Gamma_0(\mu), b \notin \boldsymbol{\nu}_0(\mu) \cup \Gamma_0(\mu), \sigma_Q(a) = b \Longrightarrow b \rightsquigarrow_Q a$.
3. *Gilt* $b \rightsquigarrow_Q a$ *und* $a \cong_Q a'$, *so folgt* $b \rightsquigarrow_Q a'$.

Das Paar (σ_M, σ_Q) heißt *zusammengesetzte Substitution* und wird im folgenden nur kurz mit σ bezeichnet.

Die Relation $\rightsquigarrow_M$ stellt eine Beziehung zwischen den Positionen vom Typ $\Pi_0 \cup \Pi_1 \cup \Pi_2$ und denen vom Typ $\boldsymbol{\nu}_0 \cup \boldsymbol{\nu}_1 \cup \boldsymbol{\nu}_2$ her. Entsprechend stellt $\rightsquigarrow_Q$ eine Beziehung zwischen den Positionen vom Typ $\Delta_0 \cup \Pi_0$ und denen vom Typ $\Gamma_0 \cup \boldsymbol{\nu}_0$ her.

Zur Veranschaulichung werden diese Relationen als gestrichelte Pfeile (siehe Abbildung 6.7) im indizierten Formelbaum eingezeichnet.

Definition 6.4.9 σ *ist J4-zulässig (J-zulässig), wenn*

1. *die transitive Hülle der Vereinigung der Ordnung auf dem indizierten Formelbaum, der* $\rightsquigarrow_M$*-Relation und der* $\rightsquigarrow_Q$*-Relation, kurz*

   $$\rhd := (\gg \cup \rightsquigarrow_M \cup \rightsquigarrow_Q)^+$$

   irreflexiv ist. $\rhd$ *heißt die* Reduktionsordnung.
2. *für* $a, b \in T_Q(\mu), \sigma_Q(a) = b$ *gilt* $\sigma_M^{\#}(pre(b)) \preceq \sigma_M^{\#}(pre(a))$

Definition 6.4.10 *Ist* σ *eine zulässige zusammengesetzte Substitution für einen Formelbaum* A^μ, *so ist eine Konnektion* $\{a, b\}$ *in* A^μ σ-komplementär, *wenn*

1. $\sigma_M^{\#}(pre(a)) = \sigma_M^{\#}(pre(b))$, *wenn man* π_2*-Positionen mit den dazugehörigen* π_1*-Positionen identifiziert.*

2. $\sigma_Q(form(a)) = \sigma_Q(form(b))$.[4]

Damit die Schreibweise nicht zu unübersichtlich wird, wird im folgenden, wenn klar ist was gemeint ist, das #-Zeichen in $\sigma_M^\#$ ausgelassen.

Definition 6.4.11 *Ein Pfad durch einen indizierten Formelbaum A^μ heißt* σ-komplementär, *wenn er eine σ-komplementäre Konnektion enthält.*

Definition 6.4.12 (Matrixkalkül-Herleitung) *Eine Formel A aus $L(C^*)$ heißt* mit dem Matrixkalkül für J bzw. J4 hergeleitet, *wenn es eine Multiplizität μ und eine zulässige Substitution σ gibt, so daß zu A^μ, dem indizierten Formelbaum für* FA, *eine Basis aus σ-komplementären Konnektionen existiert.*

Als Beispiel soll der Formelbaum für $\mathrm{F}\Diamond\forall x A(x) \to \exists x \Diamond A(x)$ betrachtet werden, mit μ konstant gleich 1, $\sigma_M(\bar{a}_6^{11}) = a_2$ und $\sigma_M(\bar{a}_3^1)$ gleich der leeren Zeichenkette.

Abbildung 6.7 zeigt den Formelbaum und die $\rhd$-Relation. Die $\leadsto$-Relation wird darin durch eine gestrichelte Linie dargestellt. Die π_2-Position a_4 wird mit a_1 identifiziert.

Es gibt nur einen atomaren Pfad durch diesen Formelbaum, nämlich

$$\{a_2, a_4, \bar{a}_5^1, \bar{a}_3^1, \bar{a}_6^{11}\}$$

Setzt man $\sigma_Q(\bar{a}_3^1) := \bar{a}_5^1$, so ist σ eine zulässige Substitution, und der Pfad ist σ-komplementär.

Die Formel im Beispiel ist bewiesen, weil es eine Multiplizität und eine zulässige Substitution σ gibt, so daß alle atomaren Pfade (in diesem Fall gibt es nur einen) durch den indizierten Formelbaum $(\Diamond\forall x A(x) \to \exists x \Diamond A(x))^\mu$ für $\mathrm{F}(\Diamond\forall x A(x) \to \exists x \Diamond A(x))$ eine σ-komplementäre Konnektion enthalten.

Wie das nächste Kapitel zeigen wird, folgt daraus, daß diese Formel herleitbar ist.

[4] Die Anwendung von σ_Q auf Formeln bezeichne einfach die Ersetzung der auftretenden Variablen durch ihre Bilder unter σ_Q.

$$\begin{tikzcd}
& {}^{\pi}_{\pi_1}\langle \Diamond\forall x A(x) \to \exists x \Diamond A(x), \mathrm{F}\rangle^{a_0}_{a_0} \arrow[dl] \arrow[dr] & \\
{}^{\pi}_{\pi_1}\langle \Diamond\forall x A(x), \mathrm{T}\rangle^{a_1}_{a_0 a_1} \arrow[d] & & {}^{\gamma}_{\pi_2}\langle \exists x \Diamond A(x), \mathrm{F}\rangle^{a_4}_{a_0 a_4} \arrow[d] \\
{}^{\nu}_{\pi_1}\langle \forall x A(x), \mathrm{T}\rangle^{a_2}_{a_0 a_1 a_2} \arrow[d] \arrow[drr, dashed] & & {}^{\nu}_{\gamma_0}\langle \Diamond A(\bar{a}^1_5), \mathrm{F}\rangle^{\bar{a}^1_5}_{a_0 a_4} \arrow[d] \\
{}_{\nu_0}\langle A(\bar{a}^1_3), \mathrm{T}\rangle^{\bar{a}^1_3}_{a_0 a_1 a_2 \bar{a}^1_3} & & {}_{\nu_1}\langle A(\bar{a}^1_5), \mathrm{F}\rangle^{\bar{a}^{11}_6}_{a_0 a_4 \bar{a}^{11}_6}
\end{tikzcd}$$

Abbildung 6.7: Ein indizierter Formelbaum mit $\rhd$-Relation

Bemerkung 6.4.13 *Die soeben bearbeitete Formel ist in Abschnitt 5.5 schon einmal betrachtet worden. Es ist interessant zu beobachten, daß von den beiden Reihenfolgebedingungen der Regelanwendung für den Sequenzenkalkül beim Matrixkalkül nur eine übrig geblieben ist: die Reduktionsordnung ($\rhd$) bestimmt lediglich, daß der $\Diamond$-Operator im Antezedens vor dem $\Diamond$-Operator im Sukzedens reduziert werden muß.*

Kapitel 7

Korrektheit und Vollständigkeit

In diesem Kapitel wird zunächst die Korrektheit und danach die Vollständigkeit des im vorangegangenen Kapitel definierten Matrixkalküls für J4 bzw. J gezeigt.

7.1 Korrektheit des Matrixkalküls

In diesem Abschnitt sei A die Formel aus L(C^*), deren Gültigkeit bzw. Herleitbarkeit untersucht wird, μ eine Multiplizität für FA, A^μ der indizierte Formelbaum für FA und σ eine zusammengesetzte J4-zulässige Substitution für A^μ.

Das Ziel dieses Abschnitts ist es zu zeigen, daß die Existenz einer Basis für A aus σ-komplementären Konnektionen gleichbedeutend ist mit der Gültigkeit von A. Im folgenden wird vorausgesetzt, daß eine solche Basis existiert.

7.1.1 Ordentliche Reduktionen

Die Reduktion eines Pfades entspricht ungefähr der Anwendung einer Regel im Sequenzenkalkül. Die dort vorhandende Reihenfolgeabhängigkeit findet man beim Matrixkalkül in Form der Reduktionsordnung wieder. Ein Pfad verträgt sich mit der Reduktionsordnung, wenn die bisherigen Reduktionen die Reduktionsordnung berücksichtigt haben. Das heißt, daß es keinen noch erreichbaren Knoten gibt, der in der Reduktionsordnung vor einem bereits bearbeiteten Knoten liegt.

Definition 7.1.1 *Sei m eine Menge von Positionen in A^μ. Eine Position $a \in m$ heißt $\rhd$-maximal*[1] *in m, wenn es kein $b \in m$ gibt, so daß $b \rhd a$ gilt.*

Mit $\mathcal{D}(m)$ wurde die Menge der durch den Pfad m dominierten Positionen bezeichnet. Das sind die Positionen, die durch weitere Reduktionen noch erreicht werden können. $\mathcal{A}(m)$ bezeichnete die Menge der bei der Entstehung des Pfades m bereits bearbeiteten Positionen.

Definition 7.1.2 *Sei m ein Pfad in A^μ. Ein Element $a \in \mathcal{D}(m)$ heißt* unbeschränkt*, wenn es kein $b \in \mathcal{A}(m)$ gibt, für das $a \rhd b$ gilt.*

Ein Pfad heißt $\rhd$-verträglich*, wenn jedes Element von $\mathcal{D}(m)$ unbeschränkt ist.*

Ein nicht-atomarer, $\rhd$-verträglicher Pfad m heißt ordentlich reduzierbar *(kurz: o.r.) bei Position $a \in m$, wenn der/die bei dieser Reduktion in den neuen Pfad eingeführten Nachfolger von a $\rhd$-maximale Elemente von $\mathcal{D}(m)$ sind.*

Solche $\rhd$-maximalen Elemente existieren, weil $\mathcal{D}(m) \neq \emptyset$ ist und $\rhd$ nicht reflexiv ist.

Lemma 7.1.3 *Der Startpfad $\{a_0\}$ ist $\rhd$-verträglich.*

Beweis: $\mathcal{A}(\{a_0\})$ ist $\{a_0\}$. $\mathcal{D}(\{a_0\})$ besteht aus allen übrigen Positionen im Formelbaum. Gälte für irgendeine Position $b \in \mathcal{D}(\{a_0\}) \quad b \rhd a_0$, so wäre $\rhd$ zyklisch, denn es gilt auf jeden Fall $a_0 \gg b$ und damit $a_0 \rhd b$. $\square$

Nach Definition sind ordentlich reduzierbare Pfade immer $\rhd$-verträglich. Die Eigenschaft eines Pfades, $\rhd$-verträglich zu sein, wird bei ordentlichen Reduktionen an die entstehenden Nachfolgepfade vererbt.

[1] lies: Reduktionsordnungs-maximal

Lemma 7.1.4 *Sei m ein nicht-atomarer, $\rhd$-verträglicher Pfad. Dann gibt es ein $a \in m$, bei dem m o.r. ist, und die bei der Reduktion entstehenden Pfade sind ebenfalls $\rhd$-verträglich.*

Beweis: $\mathcal{D}(m)$ ist nicht leer und es gibt $\rhd$-maximale Elemente, da $\rhd$ nicht reflexiv ist. Sei E die Menge aller direkten Vorgänger (aller Eltern) dieser $\rhd$-maximalen Elemente. Wählt man eines dieser $\rhd$-maximalen Elemente aus und reduziert seinen Vorgänger gerade zu diesem ausgewählten $\rhd$-maximalen Element, so ist nicht von vornherein klar, daß diese Reduktion ordentlich ist: außer dem ausgewählten Element können noch weitere Positionen in den neuen Pfad eingeführt werden (z.B. bei der Reduktion von α-Positionen) oder mehrere Pfade entstehen (z.B. bei Reduktion von β-Positionen). Es muß also gezeigt werden, daß auch die Geschwister dieser $\rhd$-maximalen Position, die bei der Reduktion in den (die) neuen Pfad(e) eingeführt werden, $\rhd$-maximal sind.

Behauptung: alle Nachfolgepositionen, die durch eine Reduktion einer Positionen aus E zu einem $\rhd$-maximalen Element von $\mathcal{D}(m)$ erhalten werden, sind ebenfalls $\rhd$-maximal in $\mathcal{D}(m)$.

Begründung: für alle Positionen aus E vom Typ δ und π, die nicht $\rightarrow$-Formeln sind, ist die Behauptung klar, da sie nur einen Nachfolger haben, der nach Voraussetzung $\rhd$-maximal in $\mathcal{D}(m)$ ist. Für die generativen Positionen aus E vom Typ γ und ν, außer den $\rightarrow$-Formeln, existiert nach Voraussetzung ein $\rhd$-maximaler Nachfolger, so daß die Reduktion ordentlich ist.

Sei a eine Position aus E vom Typ α, β oder π. Zu zeigen ist, daß alle Nachfolger $\rhd$-maximale Elemente von $\mathcal{D}(m)$ sind. Sei a_i ein solcher Nachfolger vom Typ α_i, β_i oder π_i mit $i \in \{1, 2\}$.

Angenommen a_i ist nicht $\rhd$-maximal in $\mathcal{D}(m)$, d.h. es gibt ein $b \in \mathcal{D}(m)$ mit $b \rhd a_i$. $b \rhd a_i$ bedeutet

1. $b \rightsquigarrow_M a_i$ oder
2. $b \rightsquigarrow_Q a_i$ oder
3. $b \gg a_i$ oder
4. $\exists c : b \rhd c$ und

(a) $c \rightsquigarrow_M a_i$ oder

(b) $c \rightsquigarrow_Q a_i$ oder

(c) $c \gg a_i$

Da $a_i \notin \mathcal{V}_0 \cup \mathcal{V}_1 \cup \mathcal{V}_2 \cup \Gamma_0$, muß der der dritte oder vierte Fall zutreffen. Aus $b \gg a_i$ folgt $b \geqslant a$, also $b \in \mathcal{A}(m)$. Das ergibt einen Widerspruch, denn $\mathcal{A}(m) \cap \mathcal{D}(m) = \emptyset$. Die Fälle 4a) und 4b) sind wiederum ausgeschlossen, weil $a_i \notin \mathcal{V}_0 \cup \mathcal{V}_1 \cup \mathcal{V}_2 \cup \Gamma_0$ ist. Aus 4c) folgt, daß $b \rhd a$ ist, was der $\rhd$-Verträglichkeit von m widerspricht.

Bleibt noch der Fall, daß a eine Position vom Typ ν ist, wobei form(a) eine $\rightarrow$-Formel ist. Seien a_1, a_2 die beiden Nachfolger vom Typ ν_1 bzw. ν_2. Sei $b \in \mathcal{D}(m)$ und $b \rhd a_i, i \in \{1, 2\}$. Mit einer zum vorangegangenen Fall analogen Argumentation erhält man, daß die Fälle 2, 3, 4b) und 4c) nicht eintreten können. Also muß gelten : $\exists c : b \underline{\rhd} c$ und $c \rightsquigarrow_M a_i$ gelten. Dann folgt wegen Definition 6.4.4, daß auch $c \rightsquigarrow_M a_{3-i}$ gilt. Also folgt $b \rhd a_1$ und $b \rhd a_2$. Das widerspricht der Voraussetzung, daß mindestens einer der Nachfolger von a $\rhd$-maximal für $\mathcal{D}(m)$ ist.

Sei jetzt a ein Element von m, bei dem m o.r. ist. Es ist noch zu zeigen, daß die daraus resultierenden Pfade $\rhd$-verträglich sind. Sei m' einer dieser Pfade, die durch die ordentliche Reduktion von m bei a entstanden sind. $\mathcal{D}(m')$ ist eine Teilmenge von $\mathcal{D}(m)$ und $\mathcal{A}(m')$ besteht aus $\mathcal{A}(m)$ erweitert um die Nachfolger a_i von a, die bei der Reduktion von a in den neuen Pfad m' aufgenommen worden sind. Eine Position b aus $\mathcal{D}(m')$ kann deshalb nur dann beschränkt sein, wenn für eines dieser a_i $b \rhd a_i$ gilt. Dann wäre aber im Widerspruch zur oben bewiesenen Behauptung a_i nicht $\rhd$-maximal für $\mathcal{D}(m)$ gewesen. Daraus folgt, daß auch m' $\rhd$-verträglich ist. $\square$

Bemerkung 7.1.5 *Da ein indizierter Formelbaum nur endlich viele Knoten hat, erhält man beginnend mit dem Startpfad durch wiederholte Reduktionen in endlich vielen Schritten alle atomaren Pfade durch A^μ. Ist die Reduktionsordnung nicht zyklisch, kann man die Reduktionen in einer Reihenfolge ausführen, so daß alle Reduktionen ordentlich sind. Wegen Lemma 7.1.3 und 7.1.4 sind dann alle dabei auftretenden Pfade $\rhd$-verträglich.*

7.1.2 Korrektheit ordentlicher Reduktionen

In diesem Abschnitt wird definiert, was es bedeutet, daß ein Pfad J4-erfüllbar ist. Es wird gezeigt, daß ordentliche Reduktionen diese Eigenschaft an zumindest einen der resultierenden Pfade weitergeben. σ-komplementäre Pfade werden sich als nicht erfüllbar herausstellen, woraus folgt, daß der Startpfad für einen Baum, der eine σ-komplementäre Basis besitzt, nicht erfüllbar ist.

Definition 7.1.6 *Sei m ein Pfad im Formelbaum A^μ. $\mathcal{P}(\mu, m)$ bezeichne die Menge aller Präfixe der Positionen in der assoziierten Menge von m:*

$$\mathcal{P}(\mu, m) := \{p \mid p = pre(q), q \in \mathcal{A}(m)\}$$

$\mathcal{P}_\sigma(\mu, m)$ bezeichne die Menge aller Anfangsstücke der Bilder von $\mathcal{P}(\mu, m)$ unter σ_M:

$$\mathcal{P}_\sigma(\mu, m) := \{p \mid p \preceq q, q = \sigma_M(pre(a)), a \in \mathcal{A}(m)\}$$

Definition 7.1.7 *Sei m ein Pfad im Formelbaum A^μ. $\mathcal{C}(\mu, m)$ bezeichne die Menge der Positionen vom STyp $\gamma_0, \delta_0, \pi_0$ und ν_0, die in $\mathcal{A}(m)$ liegen:*

$$\mathcal{C}(\mu, m) := \mathcal{A}(m) \cap T_Q(\mu)$$

$\mathcal{C}_\sigma(\mu, m)$ bezeichne die Bilder von $\mathcal{C}(\mu, m)$ unter σ_Q zusammen mit den Konstanten, die in der Formel A auftreten:

$$\mathcal{C}_\sigma(\mu, m) := \{a \mid a = \sigma_Q(b), b \in \mathcal{C}(\mu, m)\} \cup \{c \mid c \textit{ Konstante in } A\}$$

Definition 7.1.8 *Sei $(T, \leq, D, \bar{D}, \Vdash)$ ein J4-Modell und $(T_0, \preceq, D_0, \bar{D}_0)$ ein J4-Kripke-Rahmen mit $T \cap T_0 = \emptyset$. Sei D_0 eine Menge von Konstanten mit $D \cap D_0 = \emptyset$. ι ist eine Interpretation von $P_0 \subseteq T_0$ in T und $C_0 \subseteq D_0$ in D, wenn für alle $p, q \in P_0$ gilt:*

$$p \preceq q \Longrightarrow \iota(p) \leq \iota(q)$$

Definition 7.1.9 (Erfüllbarkeit) *Ein Pfad m heißt* J4-erfüllbar unter σ, *wenn es ein J4-Modell $(T, \leq, D, \bar{D}, \Vdash)$ und eine Interpretation ι von $\mathcal{P}_\sigma(\mu, m)$ in T und $\mathcal{C}_\sigma(\mu, m)$ in D gibt, so daß*

1. *für jede Position $a \in \mathcal{A}(m)$ mit $\sigma_M(pre(a)) = p$ und $\sigma_Q(sform(a)) = B$ alle Konstanten von $\iota(B)$ in $\bar{D}(\iota(p))$ liegen und $\iota(p) \Vdash \iota(B)$*

2. *für jedes $c \in \mathcal{C}_\sigma(\mu, m) \cap T_Q(\mu)$ gilt:*

$$\iota(c) \in \bar{D}(\iota(\sigma_M(pre(c)))).$$

Diese zweite Bedingung heißt die Parameter-Bedingung.

Die Parameter-Bedingung besagt, daß für jedes Element von $\mathcal{C}_\sigma(\mu, m)$, das eine Position im Formelbaum A^μ ist, die Interpretation dieses Elements in der "Welt" liegt, die durch die Interpretation des Präfixes dieser Position gegeben ist. Die Parameter-Bedingung wird bei den γ- und ν-Reduktionen erfüllbarer Pfade eine Rolle spielen.

Die nächsten Lemmata dienen dazu zu zeigen, daß die Parameter-Bedingung bei ordentlichen Reduktionen erhalten bleibt.

Lemma 7.1.10 *Sei m ein Pfad durch A^μ, der bei der Position $a \in m$ o.r. ist und der unter σ durch ι J4-erfüllt wird. Sei m' ein Pfad, der durch die ordentliche Reduktion von m bei der Position $a \in m$ entstanden ist. Wenn $\mathcal{C}_\sigma(\mu, m') = \mathcal{C}_\sigma(\mu, m)$ ist, erfüllt auch m' mit ι die Parameter-Bedingung.*

Beweis: Wie das folgende Lemma zeigen wird, gilt für alle $a \in \mathcal{C}_\sigma(\mu, m)$ mit $p = \sigma_M(\text{pre}(a))$, daß $a \in \mathcal{P}_\sigma(\mu, m)$ liegt. Die Parameter-Bedingung gilt für m', da sie nach Voraussetzung für m gilt und $\mathcal{C}_\sigma(\mu, m') = \mathcal{C}_\sigma(\mu, m)$ ist. $\square$

Das folgende Lemma wird zum Nachweis der Gültigkeit der Parameter-Bedingung im Fall der π- und ν-Reduktionen benötigt.

Lemma 7.1.11 *Sei m ein Pfad durch A^μ. Sei $p \in T_M(\mu)^\star$. Wenn $p \notin \mathcal{P}_\sigma(\mu, m)$ ist, gibt es kein $a \in \mathcal{C}_\sigma(\mu, m)$ mit $\sigma_M(pre(a)) = p$.*

Beweis: Angenommen für $a \in \mathcal{C}_\sigma(\mu, m)$ ist $\sigma_M(\text{pre}(a)) = p$ und $p \notin \mathcal{P}_\sigma(\mu, m)$. Wäre $a \in \mathcal{C}(\mu, m)$, so gälte $a \in \mathcal{A}(m)$ und deshalb $p \in \mathcal{P}_\sigma(\mu, m)$. Also gilt $a \notin \mathcal{C}(\mu, m)$. Das heißt, daß es ein $b \in \mathcal{C}(\mu, m)$ gibt, mit $\sigma_Q(b) = a$.

Sei $q := \sigma_M(\text{pre}(b))$. Nach Definition ist $q \in \mathcal{P}_\sigma(\mu, m)$. Weil σ J4-zulässig ist, folgt aus $a = \sigma_Q(b)$, daß $p = \sigma_M(\text{pre}(a)) \preceq \sigma_M(\text{pre}(b)) = q$ gelten muß (Definition 6.4.9). Nach Definition von $\mathcal{P}_\sigma(\mu, m)$ gilt dann $p \in \mathcal{P}_\sigma(\mu, m)$, was der Voraussetzung $p \notin \mathcal{P}_\sigma(\mu, m)$ widerspricht. □

Lemma 7.1.12 *Ein Pfad m in A^μ, der σ-komplementär ist, ist nicht J4-erfüllbar.*

Beweis: m enthalte die σ-komplementäre Konnektion $\{a, b\}$. Nach Definition 6.4.10 bedeutet das:

1. $\sigma_M(\text{pre}(a)) = \sigma_M(\text{pre}(b)) = p$, wenn man π_2-Positionen mit den dazugehörigen π_1-Positionen identifiziert.

2. $\sigma_Q(\text{form}(a)) = \sigma_Q(\text{form}(b)) = B$ für eine Formel B.[2]

Angenommen $(T, \leq, D, \bar{D}, \Vdash)$ sei ein Modell, in dem $\{a, b\}$ unter der Interpretation ι J4-erfüllbar ist. Nach Definition 7.1.9 bedeutet das, daß alle Konstanten von $\iota(B)$ in $\bar{D}(\iota(p))$ liegen und

$$\iota(p) \Vdash \iota(\sigma_Q(\text{sform}(a)))$$

und

$$\iota(p) \Vdash \iota(\sigma_Q(\text{sform}(b)))$$

Weil $\{a, b\}$ eine Konnektion ist, haben $\text{sform}(a)$ und $\text{sform}(b)$ unterschiedliches Signum. Daraus folgt $\iota(p) \Vdash \iota(\text{T}B)$ und $\iota(p) \Vdash \iota(\text{F}B)$. Aus diesem Widerspruch folgt, daß m nicht J4-erfüllbar ist. □

Als nächster Schritt auf dem Weg zum Beweis der Korrektheit des Matrixkalküls wird jetzt gezeigt, daß ein Pfad die Eigenschaft, erfüllbar zu sein, bei ordentlichen Reduktionen an wenigstens einen seiner Nachfolgepfade weitergibt.

[2]Die Erweiterung von σ und ι auf (signierte) Formeln ist klar:
$\sigma_Q(\text{sign}(a)\text{form}(a)) := \text{sign}(a)\sigma_Q(\text{form}(a))$
$\iota(\text{sign}(a)\text{form}(a)) := \text{sign}(a)\iota(\text{form}(a))$

Bemerkung 7.1.13 *Im folgenden wird der Bezeichnungsmißbrauch aus Bemerkung 5.1.15 von Formeln auf Positionen ausgedehnt.* $\alpha, \beta, \ldots, \pi_1, \pi_2$ *etc. sollen auch für Positionen des entsprechenden Typs stehen.*

Lemma 7.1.14 *Sei* m *ein Pfad durch* A^μ*, der unter* σ *J4-erfüllbar ist und bei der Position* $\alpha \in m$ *o.r. ist. Dann ist auch der daraus resultierende Pfad unter* σ *J4-erfüllbar.*

Beweis: Seien α_1, α_2 die Nachfolgepositionen von α. Sei m' der Nachfolgepfad von m:

$$\begin{array}{lcl} m' & = & (m \setminus \{\alpha\}) \cup \{\alpha_1, \alpha_2\} \\ \mathcal{A}(m') & = & \mathcal{A}(m) \cup \{\alpha_1, \alpha_2\} \\ \mathcal{P}_\sigma(\mu, m') & = & \mathcal{P}_\sigma(\mu, m) \\ \mathcal{C}_\sigma(\mu, m') & = & \mathcal{C}_\sigma(\mu, m), \end{array}$$

da bei der Reduktion von α die Präfixe der $\alpha_i, i \in \{1,2\}$, gleich dem von α sind und keine neuen Konstanten in $\mathrm{sform}(\alpha_i), i \in \{1,2\}$, gelangen. Wegen Lemma 7.1.10 erfüllt m' die Parameter-Bedingung. Sei $p = \sigma_M(\mathrm{pre}(\alpha))$, $B = \sigma_Q(\mathrm{sform}(\alpha))$ und $B_i = \sigma_Q(\mathrm{sform}(\alpha_i)), i \in \{1,2\}$. Nach Voraussetzung gibt es ein Modell $(T, \leq, D, \bar{D}, \Vdash)$ und eine Interpretation ι mit $\iota(p) \Vdash \iota(B)$. Wegen Korollar 5.1.16 folgt daraus $\iota(p) \Vdash \iota(B_i), i \in \{1,2\}$. Also ist m' J4-erfüllbar unter σ. □

Lemma 7.1.15 *Sei* m *ein Pfad durch* A^μ*, der unter* σ *J4-erfüllbar ist und bei der Position* $\beta \in m$ *o.r. ist. Dann ist auch einer der daraus resultierenden Pfade unter* σ *J4-erfüllbar.*

Beweis: Seien β_1, β_2 die Nachfolgepositionen von β. Seien $m_i, i \in \{1,2\}$ die Nachfolgepfade von m:

$$\begin{array}{lcl} m_i & = & (m \setminus \{\beta\}) \cup \{\beta_i\}, i \in \{1,2\} \\ \mathcal{A}(m_i) & = & \mathcal{A}(m) \cup \{\beta_i\}, i \in \{1,2\} \\ \mathcal{P}_\sigma(\mu, m_i) & = & \mathcal{P}_\sigma(\mu, m) \\ \mathcal{C}_\sigma(\mu, m_i) & = & \mathcal{C}_\sigma(\mu, m), \end{array}$$

da bei der Reduktion von β die Präfixe der $\beta_i, i \in \{1,2\}$, gleich dem von β sind und keine neuen Konstanten in $\mathrm{sform}(\beta_i), i \in \{1,2\}$, gelangen. Wegen

Lemma 7.1.10 erfüllt m' die Parameter-Bedingung. Sei $p = \sigma_M(\text{pre}(\beta))$, $B = \sigma_Q(\text{sform}(\beta))$ und $B_i = \sigma_Q(\text{sform}(\beta_i)), i \in \{1,2\}$. Nach Voraussetzung gibt es ein Modell $(T, \leq, D, \bar{D}, \Vdash)$ und eine Interpretation ι mit $\iota(p) \Vdash \iota(B)$. Wegen Korollar 5.1.16 folgt daraus $\iota(p) \Vdash \iota(B_1)$, oder $\iota(p) \Vdash \iota(B_2)$. Also ist m_1 oder m_2 J4-erfüllbar unter σ. □

Das folgende Lemma stellt sicher, daß bei ordentlichen Reduktionen von δ- bzw. π-Positionen zu δ_0- bzw. π_0-Positionen neue Konstanten in den Pfad eingeführt werden. Das entspricht der Einhaltung der Variablenbedingung beim Sequenzenkalkül.

Lemma 7.1.16 *Sei m ein Pfad, der in δ (π) o.r. ist. Dann ist δ_0 (π_0) nicht in $\mathcal{C}_\sigma(\mu, m)$.*

Beweis: m ist $\rhd$-verträglich, d.h. alle Elemente von $\mathcal{D}(m)$ sind unbeschränkt. Angenommen es gälte $\delta_0 \in \mathcal{C}_\sigma(\mu, m)$ ($\pi_0 \in \mathcal{C}_\sigma(\mu, m)$). Da $\delta_0 \in \mathcal{D}(m)$ ($\pi_0 \in \mathcal{D}(m)$) liegt, ist $\delta_0 \notin \mathcal{A}(m)$ ($\pi_0 \notin \mathcal{A}(m)$). Also gibt es ein $a \in \mathcal{C}(\mu, m) = T_Q(\mu) \cap \mathcal{A}(m)$ mit $\sigma_Q(a) = \delta_0$ ($\sigma_Q(a) = \pi_0$). Daraus folgt $\delta_0 \rhd a$ ($\pi_0 \rhd a$), was der Unbeschränktheit von δ_0 (π_0) widerspricht. □

Der Streichungsvorschrift im Sequenzenkalkül bei der (inversen) Anwendung der π-Regel entspricht, wenn auch weniger offensichtlich, die Aussage des folgenden Lemmas. Es besagt, daß bei der ordentlichen Reduktion einer π-Position ein neues Präfix in den Pfad eingeführt wird. Das entspricht dem Übergang zu einer neuen "Welt".

Lemma 7.1.17 *Sei m ein Pfad durch A^μ, der in π o.r. ist. Dann ist*

$$\sigma_M(\mathit{pre}(\pi_i)) \notin \mathcal{P}_\sigma(\mu, m) \text{ für } i \in \{0,1,2\}$$

Beweis: Sei $p := \text{pre}(\pi_i)$. Angenommen $\sigma_M(p) \in \mathcal{P}_\sigma(\mu, m)$. Dann gibt es ein $q \in \mathcal{P}(\mu, m)$ mit $\sigma_M(p) \preceq \sigma_M(q)$. Es gilt:

$$\sigma_M(p) = \sigma_M(\text{pre}(\pi))\pi_i \preceq \sigma_M(q)$$

Also gibt es ein $e \in q \in \mathcal{P}(\mu, m)$ mit $\pi_i \in \sigma_M(e)$. Gemäß Definition 6.4.4 bedeutet das $\pi_i \rhd e \in \mathcal{A}(m)$. Das ist ein Widerspruch zur $\rhd$-Verträglichkeit von m. □

Lemma 7.1.18 *Sei m ein Pfad durch A^{μ}, der unter σ J4-erfüllbar ist und bei der Position $\delta \in m$ o.r. ist. Dann ist auch der daraus resultierende Pfad unter σ J4-erfüllbar.*

Beweis: Sei δ_0 die Nachfolgeposition von δ. Sei m' der Nachfolgepfad von m:

$$\begin{array}{lcl} m' & = & (m \setminus \{\delta\}) \cup \{\delta_0\} \\ \mathcal{A}(m') & = & \mathcal{A}(m) \cup \{\delta_0\} \\ \mathcal{P}_\sigma(\mu, m') & = & \mathcal{P}_\sigma(\mu, m) \\ \mathcal{C}_\sigma(\mu, m') & = & \mathcal{C}_\sigma(\mu, m) \cup \{\delta_0\}, \end{array}$$

Sei $p = \sigma_M(\mathrm{pre}(\delta))$, $B = \sigma_Q(\mathrm{sform}(\delta))$ und $B_0(\delta_0) = \sigma_Q(\mathrm{sform}(\delta_0))$. Nach Voraussetzung gibt es ein Modell $(T, \leq, D, \bar{D}, \Vdash)$ und eine Interpretation ι, so daß alle Konstanten von $\iota(B)$ in $\bar{D}(\iota(p))$ liegen und $\iota(p) \Vdash \iota(B)$. Wegen Korollar 5.1.16 gibt es ein $c \in \bar{D}(\iota(p))$, so daß $\iota(p) \Vdash \iota(B_0)(c)$. Wegen Korollar 7.1.16 ist $\delta_0 \notin \mathcal{C}_\sigma(\mu, m)$. Deshalb ist ι für δ_0 noch undefiniert. ι kann aber wie folgt auf $\mathcal{C}_\sigma(\mu, m')$ zu einer Interpretation ι' ausgedehnt werden:

$$\iota'(x) \;\; := \;\; \begin{cases} c, & \text{falls } x = \delta_0 \\ \iota(x), & \text{sonst} \end{cases}$$

Da $\mathcal{P}_\sigma(\mu, m') = \mathcal{P}_\sigma(\mu, m)$, sei $\iota'(p) = \iota(p)$ für $p \in \mathcal{P}_\sigma(\mu, m')$.

Es bleibt zu zeigen, daß der neue Pfad m' unter σ durch ι' erfüllt wird.

1. Es gilt $\iota'(\delta_0)) = c \in \bar{D}(\iota'(p))$ Außerdem gilt $\iota'(p) = \iota(p)$ für $p \in \mathcal{P}_\sigma(\mu, m')$ und $\iota'(B_0) = \iota(B_0)$, da ι' und ι auf den Elementen von $\mathcal{C}_\sigma(\mu, m)$ übereinstimmen. Es folgt:

$$\begin{array}{lrcl} & \iota'(p) & \Vdash & \iota'(B_0)(c) \\ \Longleftrightarrow & \iota'(p) & \Vdash & \iota'(B_0)(\iota'(\delta_0)) \\ \Longleftrightarrow & \iota'(p) & \Vdash & \iota'(B_0(\delta_0)) \\ \Longleftrightarrow & \iota'(\sigma_M(\mathrm{pre}(\delta_0))) & \Vdash & \iota'(\sigma_Q(\mathrm{sform}(\delta_0))) \end{array}$$

   Für die übrigen Positionen in $\mathcal{A}(m')$ gilt diese Bedingung, weil sie schon für $\mathcal{A}(m)$ gegolten hat. Damit ist die erste Bedingung für J4-Erfüllbarkeit aus Definition 7.1.9 gezeigt.

2. Die zweite Bedingung, die Parameter-Bedingung, fordert, daß für $\delta_0 \in \mathcal{C}_\sigma(\mu, m'), p = \sigma_M(\text{pre}(\delta_0))$ gilt:

$$\iota'(\delta_0) \in \bar{D}(\iota'(p))$$

Das ist gerade durch die Konstruktion von ι' erfüllt. Für die anderen Elemente aus $\mathcal{C}_\sigma(\mu, m')$ gilt diese Bedingung, weil sie für $\mathcal{C}_\sigma(\mu, m)$ schon gegolten hat.

Daraus folgt, daß m' J4-erfüllbar ist. □

Lemma 7.1.19 *Sei m ein Pfad durch A^μ, der unter σ J4-erfüllbar ist und bei der Position $\gamma \in m$ o.r. ist. Dann ist auch der daraus resultierende Pfad unter σ J4-erfüllbar.*

Beweis: Die Position γ hat der Multiplizität von γ_0 (dem Nachfolger von γ im nicht-indizierten Formelbaum) entsprechend viele Nachfolger. Daß m bei γ o.r ist, bedeutet, daß zumindest einer dieser Nachfolger ein $\rhd$-maximales Element von $\mathcal{D}(m)$ ist. γ_0 bezeichne diese Nachfolgeposition von γ. Sei m' der Nachfolgepfad von m:

$$\begin{array}{lcl} m' & = & m \cup \{\gamma_0\} \\ \mathcal{A}(m') & = & \mathcal{A}(m) \cup \{\gamma_0\} \\ \mathcal{P}_\sigma(\mu, m') & = & \mathcal{P}_\sigma(\mu, m) \\ \mathcal{C}_\sigma(\mu, m') & = & \mathcal{C}_\sigma(\mu, m) \cup \{\sigma_Q(\gamma_0)\}, \end{array}$$

Sei $p := \sigma_M(\text{pre}(\gamma))$, $B := \sigma_Q(\text{sform}(\gamma))$, $t := \sigma_Q(\gamma_0)$, $B_0(t) := \sigma_Q(\text{sform}(\gamma_0))$. Nach Voraussetzung gibt es ein Modell $(T, \leq, D, \bar{D}, \Vdash)$ und eine Interpretation ι, so daß alle Konstanten von $\iota(B)$ in $\bar{D}(\iota(p))$ liegen und $\iota(p) \Vdash \iota(B)$. Wegen Korollar 5.1.16 gilt für alle $c \in \bar{D}(\iota(p))$, daß $\iota(p) \Vdash \iota(B_0)(c)$.

Sei $q := \sigma_M(\text{pre}(t))$.[3] Wegen der Zulässigkeit von σ (Definition 6.4.9) folgt aus $\sigma_Q(\gamma_0) = t$, daß $q \preceq p$ ist. Also ist $q \in \mathcal{P}_\sigma(\mu, m)$. Da ι eine Interpretation für $\mathcal{P}_\sigma(\mu, m)$ ist, folgt aus $q \preceq p$, daß $\iota(q) \leq \iota(p)$ ist und, weil J4 wachsende Universen hat, daß $\bar{D}(\iota(q)) \subseteq \bar{D}(\iota(p)$.

Zwei Fälle sind zu unterscheiden:

[3] Ist t eine Konstante von $\mathrm{L}(C^*)$, die in A auftritt, so ist $\iota(t) \in \bar{D}(\iota(p))$, $t \in \mathcal{C}_\sigma(\mu, m')$ laut Definition. Setze $\iota' = \iota$. Die Parameter-Bedingung gilt wegen Lemma 7.1.10

1. $t \in \mathcal{C}_\sigma(\mu, m)$. Dann ist $\mathcal{C}_\sigma(\mu, m') = \mathcal{C}_\sigma(\mu, m)$. An dieser Stelle sorgt nun die Parameter-Bedingung dafür, daß t unter ι durch ein Element interpretiert wird, das aus der "Welt" stammt, die der Interpretation des σ_M-Bildes des Präfixes von t entspricht: da für m die Parameter-Bedingung gilt, folgt
$$\iota(t) \in \bar{D}(\iota(q)) \subseteq \bar{D}(\iota(p))$$
und damit
$$\iota(p) \Vdash \iota(B_0(t)).$$
Also ist die erste Bedingung aus Definition 7.1.9 auch für $\gamma_0 \in \mathcal{A}(m')$ erfüllt. Die Parameter-Bedingung gilt wegen Lemma 7.1.10.

2. $t \notin \mathcal{C}_\sigma(\mu, m)$. ι kann wie folgt auf $\mathcal{C}_\sigma(\mu, m')$ zu einer Interpretation ι' ausgedehnt werden. Für ein $c \in \bar{D}(\iota(q))$ sei
$$\iota'(x) \quad := \quad \begin{cases} c, & \text{falls } x = t \\ \iota(x), & \text{sonst} \end{cases}$$
Da $\mathcal{P}_\sigma(\mu, m') = \mathcal{P}_\sigma(\mu, m)$, sei $\iota'(p) = \iota(p)$ für $p \in \mathcal{P}_\sigma(\mu, m')$. Daraus folgt, das ι' eine Interpretation ist. Bedingung 1 aus 7.1.9 ist erfüllt, da $\iota'(t) \in \bar{D}(\iota'(p))$ und weil wegen Korollar 5.1.16 $\iota'(p) \Vdash \iota'(B_0(t))$. Die Parameter-Bedingung gilt für ι' und m', da sie für ι und m galt und wegen der Konstruktion von ι'. □

Lemma 7.1.20 *Sei m ein Pfad durch A^μ, der unter σ J4-erfüllbar ist und bei der Position $\pi \in m$ o.r. ist. Dann ist auch der daraus resultierende Pfad unter σ J4-erfüllbar.*

Beweis: Seien $\pi_i, i \in \{1, 2\}$ bzw. π_0 die Nachfolgepositionen von π und $p := \sigma_M(\mathrm{pre}(\pi))$. Sei m' der Nachfolgepfad von m[4]:

$$\begin{array}{lcll} m' & = & (m \setminus \{\pi\}) \cup \{\pi_1, \pi_2\} & [\text{ bzw. } (m \setminus \{\pi\}) \cup \{\pi_0\}] \\ \mathcal{A}(m') & = & \mathcal{A}(m) \cup \{\pi_1, \pi_2\} & [\text{ bzw. } \mathcal{A}(m) \cup \{\pi_0\}] \\ \mathcal{P}_\sigma(\mu, m') & = & \mathcal{P}_\sigma(\mu, m) \cup \{p\pi_1, p\pi_2\} & [\text{ bzw. } \mathcal{P}_\sigma(\mu, m) \cup \{p\pi_0\}] \\ \mathcal{C}_\sigma(\mu, m') & = & \mathcal{C}_\sigma(\mu, m) & [\text{ bzw. } \mathcal{C}_\sigma(\mu, m) \cup \{\pi_0\}] \end{array}$$

[4]Text in eckigen Klammern "[,]" bezieht sich im folgenden auf den Fall, daß π eine $\forall$-Formel ist.

Weiterhin sei $B := \sigma_Q(\text{sform}(\pi))$ und $B_i := \sigma_Q(\text{sform}(\pi_i)), i \in \{1,2\}$, bzw. $B_0(\pi_0) := \sigma_Q(\text{sform}(\pi_0))$.

Nach Voraussetzung gibt es ein Modell $(T, \leq, D, \bar{D}, \Vdash)$ und eine Interpretation ι, so daß

- alle Konstanten von $\iota(B)$ in $\bar{D}(\iota(p))$ liegen und
- $\iota(p) \Vdash \iota(B)$.

Wegen Korollar 5.1.16 gibt es ein $w \geq \iota(p)$ [und ein $c \in \bar{D}(w)$], so daß

$$w \Vdash \iota(B_1) \text{ und } w \Vdash \iota(B_2)$$

bzw.

$$w \Vdash \iota(B_0)(c)$$

Wegen Lemma 7.1.17 ist

$$p\pi_i \notin \mathcal{P}_\sigma(\mu, m),\ i \in \{0,1,2\}.$$

Deshalb ist ι für $p\pi_i$ undefiniert. ι kann aber wie folgt auf $\mathcal{P}_\sigma(\mu, m')$ zu einer Interpretation ι' ausgedehnt werden:

$$\iota'(x) \ := \ \begin{cases} w, & \text{falls } x = p\pi_i,\ i \in \{0,1,2\} \\ \iota(x), & \text{für } x \in \mathcal{P}_\sigma(\mu, m) \end{cases}$$

Für den Fall, daß π eine $\forall$-Formel ist muß ι auch auf $\mathcal{C}_\sigma(\mu, m')$ ausgeweitet werden, weil wegen Lemma 7.1.16 $c \notin \mathcal{C}_\sigma(\mu, m)$ ist:

$$\iota'(x) \ := \ \begin{cases} c, & \text{falls } x = \pi_0 \\ \iota(x), & \text{für } x \in \mathcal{C}_\sigma(\mu, m) \end{cases}$$

Es bleibt noch zu zeigen, daß

1. Bedingung 1 aus Definition 7.1.9 erfüllt ist,
2. die Parameter-Bedingung erfüllt ist und daß
3. ι' eine Interpretation für $\mathcal{P}_\sigma(\mu, m')$ und $\mathcal{C}_\sigma(\mu, m')$ ist.

Die Punkte werden der Reihe nach behandelt:

1. Für $a \in \mathcal{A}(m)$ gilt

$$\iota'(\sigma_M(\mathrm{pre}(a))) \Vdash \iota'(\sigma_Q(\mathrm{sform}(a))),$$

da ι und ι' auf $\mathcal{P}_\sigma(\mu, m)$ und $\mathcal{C}_\sigma(\mu, m)$ gleich sind, und alle Konstanten von $\sigma_Q(\mathrm{sform}(a))$ nach Voraussetzung in $\bar{D}(\iota(\sigma_M(\mathrm{pre}(a)))$ liegen.

Für $a = \pi_i$, $i \in \{1, 2\}$ gilt

$$\iota'(p\pi_i) \Vdash \iota'(B_i)$$

und für $i = 0$

$$\iota'(p\pi_0) \Vdash \iota'(B_0(\pi_0))$$

nach Konstruktion von ι' und

$$\iota'(\pi_0) = c \in \bar{D}(\iota'(p\pi_0)) = \bar{D}(\iota'(\sigma_M(\mathrm{pre}(\pi_0)))).$$

2. Sei $a \in \mathcal{C}_\sigma(\mu, m')$ mit $q := \sigma_M(\mathrm{pre}(a))$. Zu zeigen ist:

$$\iota'(a) \in \bar{D}(\iota'(q))$$

Für $i = 1, 2$ gilt: $\mathcal{C}_\sigma(\mu, m') = \mathcal{C}_\sigma(\mu, m)$. Die Parameter-Bedingung gilt also, weil sie für m und ι gegolten hat, und ι' und ι auf $\mathcal{C}_\sigma(\mu, m)$ übereinstimmen (Lemma 7.1.10).

Für $i = 0$ gilt: $\pi_0 \in \mathcal{C}_\sigma(\mu, m'), p\pi_0 = \sigma_M(\mathrm{pre}(\pi_0))$. Die Parameter-Bedingung ist für π_0 erfüllt, weil $\iota'(\pi_0) = c \in \bar{D}(\iota'(p\pi_0)) = \bar{D}(w)$ nach Konstruktion von ι'. Für die anderen $a \in \mathcal{C}_\sigma(\mu, m')$ gilt die Parameter-Bedingung, weil sie für m und ι gegolten hat, und ι' und ι auf $\mathcal{C}_\sigma(\mu, m)$ übereinstimmen (Lemma 7.1.10).

3. Zu zeigen ist: für alle $p_0, q_0 \in \mathcal{P}_\sigma(\mu, m')$ gilt:

$$p_0 \preceq q_0 \Longrightarrow \iota'(p_0) \leq \iota'(q_0).$$

Da ι und ι' auf $\mathcal{P}_\sigma(\mu, m)$ übereinstimmen, gilt die Behauptung, wenn p_0 und q_0 aus $\mathcal{P}_\sigma(\mu, m)$ sind. Sei $i \in \{0, 1, 2\}$. Der Fall $p\pi_i \preceq q_0$ für ein $q_0 \in \mathcal{P}_\sigma(\mu, m)$ kann nicht eintreten, da dann $p\pi_i \in \mathcal{P}_\sigma(\mu, m)$ folgen würde. Bleibt der Fall $p_0 \preceq p\pi_i$. Für $p_0 = p\pi_i$ folgt trivialerweise $\iota'(p_0) = \iota'(p\pi_i)$. Aus $p_0 \prec p\pi_i$ folgt $p_0 \preceq p$ und damit $\iota'(p_0) \leq \iota'(p)$, weil ι und ι' in diesem Fall übereinstimmen. Es gilt also $\iota'(p_0) \leq \iota'(p\pi_i)$. $\square$

Lemma 7.1.21 *Sei m ein Pfad durch A^μ, der unter σ J4-erfüllbar ist und bei der Position $\nu \in m$ o.r. ist. Dann ist auch wenigstens einer der daraus resultierenden Pfade unter σ J4-erfüllbar.*

Beweis: Die Position ν hat der Multiplizität von ν_0, ν_1 bzw. ν_2 (den Nachfolgern von ν im nicht-indizierten Formelbaum) entsprechend viele Nachfolger. Daß m bei ν o.r ist, bedeutet, daß zumindest einer dieser Nachfolger (bzw. ein Nachfolgepaar) $\rhd$-maximal für $\mathcal{D}(m)$ ist. ν_i, $i \in \{1,2\}$ bzw. ν_0 bezeichne diese Nachfolgeposition/en von ν je nachdem, ob ν $\forall$-, $\neg$, $\rightarrow$-Formel oder Primformel ist. Es seien $p := \sigma_M(\mathrm{pre}(\nu))$, $q_i := \sigma_M(\nu_i)$ für $i \in \{0,1,2\}$ und ggf. $t_0 := \sigma_Q(\nu_0)$. Seien m_i der/die Nachfolgepfad(e) von m[5]:

$$\begin{array}{lcl} m_i & = & m \cup \{\nu_i\} \\ \mathcal{A}(m_i) & = & \mathcal{A}(m) \cup \{\nu_i\} \\ \mathcal{P}_\sigma(\mu, m_i) & = & \mathcal{P}_\sigma(\mu, m) \cup \{r \mid r \preceq pq_i\},\ i \in \{0,1,2\} \\ \mathcal{C}_\sigma(\mu, m_i) & = & \mathcal{C}_\sigma(\mu, m)\ [\text{ bzw. } \mathcal{C}_\sigma(\mu, m) \cup \{\sigma_Q(\nu_0)\}] \end{array}$$

Weiterhin sei $B := \sigma_Q(\mathrm{sform}(\nu))$ und $B_i := \sigma_Q(\mathrm{sform}(\nu_i)), i \in \{1,2\}$, bzw. $B_0(\nu_0) := \sigma_Q(\mathrm{sform}(\nu_0))$.

Nach Voraussetzung gibt es ein Modell $(T, \leq, D, \bar{D}, \Vdash)$ und eine Interpretation ι, so daß

- alle Konstanten von $\iota(B)$ in $\bar{D}(\iota(p))$ liegen,
- $\iota(p) \Vdash \iota(B)$ und
- für alle $a \in \mathcal{C}_\sigma(\mu, m)$ mit $r := \sigma_M(\mathrm{pre}(a))$ gilt: $\iota(a) \in \bar{D}(\iota(r))$.

Wegen Korollar 5.1.16 gilt für alle w mit $w \geq \iota(p)$, daß $w \Vdash \iota(B_1)$ oder $w \Vdash \iota(B_2)$ [bzw. für alle $c \in \bar{D}(w)$ $w \Vdash \iota(B_0)(c)$].

Gesucht wird eine Erweiterung von ι zu einer Interpretation ι' von $\mathcal{P}_\sigma(\mu, m_i)$ und $\mathcal{C}_\sigma(\mu, m_i)$ unter σ, so daß

[5] Text in eckigen Klammern "[,]" bezieht sich im folgenden auf den Fall, daß ν eine $\forall$-Formel ist.

- für $p', q' \in \mathcal{P}_\sigma(\mu, m_i)$ folgt:

  $$p' \preceq q' \Longrightarrow \iota'(p') \leq \iota'(q')$$

  Insbesondere soll gelten

  $$\iota'(p) \leq \iota'(pq_i) \text{ für } i \in \{0, 1, 2\}$$

  bzw.

  $$\iota'(t_0) \in \bar{D}(\iota'(pq_0))$$

- die Parameter-Bedingung erfüllt ist.

Ist eine solche Interpretation gefunden, so ist das Lemma bewiesen, denn wegen Korollar 5.1.16 folgt

$$\iota'(pq_1) \Vdash \iota'(B_1) \text{ oder } \iota'(pq_2) \Vdash \iota'(B_2)$$

bzw.

$$\iota'(pq_0) \Vdash \iota'(B_0(t_0))],$$

d.h.

$$\iota'(\sigma_M(\text{pre}(\nu_i)) \Vdash \iota'(\sigma_Q(\text{sform}(\nu_i)))$$

für $i = 0, 1$ oder 2.

Bei der Erweiterung der Interpretation sind zwei Fälle zu unterscheiden. Der Einfachheit halber steht im folgenden nur q anstelle von q_0, q_1 bzw. q_2.

1. Sei $q = \sigma_M(\nu_i) = \emptyset$. Dann ist $\mathcal{P}_\sigma(\mu, m_i) = \mathcal{P}_\sigma(\mu, m)$. In diesem Fall läuft der Beweis wie im Fall von Lemma 7.1.15 für β-Reduktionen [Lemma 7.1.19 für γ-Reduktionen].

2. Sei $q = a_1 \ldots a_n, n \geq 1$. Es wird eine Folge von J4-Interpretationen $(\iota_i)_{0 \leq i \leq n}$ für die Mengen $\mathcal{P}^i_\sigma(\mu, m)$ und $\mathcal{C}^i_\sigma(\mu, m), 0 \leq i \leq n$ wie folgt definiert:

   $$\begin{array}{lcl} \mathcal{P}^0_\sigma(\mu, m) & := & \mathcal{P}_\sigma(\mu, m) \\ \mathcal{P}^{j+1}_\sigma(\mu, m) & := & \mathcal{P}^j_\sigma(\mu, m) \cup \{pa_1 \ldots a_{j+1}\}, 0 \leq j < n \\ \mathcal{C}^j_\sigma(\mu, m) & := & \mathcal{C}_\sigma(\mu, m),\ 0 \leq j \leq n \end{array}$$

- $i = 0$. Setze $\iota_0 = \iota$. ι_0 ist eine Interpretation für $\mathcal{P}_\sigma(\mu, m)$ und $\mathcal{C}_\sigma(\mu, m)$ nach Voraussetzung.
- $i \to i + 1$.
  (a) $pa_1 \dots a_{i+1} \in \mathcal{P}^i_\sigma(\mu, m)$. Dann ist ι_i bereits auf $pa_1 \dots a_{i+1}$ definiert. Setze $\iota_{i+1} = \iota_i$. ι_{i+1} ist eine Interpretation mit

  $$\iota_{i+1}(pa_1 \dots a_i) \le \iota_{i+1}(pa_1 \dots a_{i+1}),$$

  da $\iota_{i+1} = \iota_i$ und $pa_1 \dots a_i \preceq pa_1 \dots a_{i+1}$. Die Parameter-Bedingung gilt wegen Lemma 7.1.10, da

  $$\mathcal{C}^{i+1}_\sigma(\mu, m) = \mathcal{C}^i_\sigma(\mu, m).$$

  (b) $pa_1 \dots a_{i+1} \notin \mathcal{P}^i_\sigma(\mu, m)$. Da "$\le$" reflexiv ist, gibt es stets ein $w \in T$ mit

  $$\iota_i(pa_1 \dots a_i) \le w.$$

  Definiere ι_{i+1} für $\mathcal{P}^{i+1}_\sigma(\mu, m)$ wie folgt:

  $$\iota_{i+1}(x) \quad := \quad \begin{cases} w & \text{für } x = pa_1 \dots a_{i+1} \\ \iota_i(x) & \text{für } x \in \mathcal{P}^i_\sigma(\mu, m) \end{cases}$$

  Es ist zu zeigen, daß ι_{i+1} tatsächlich eine Interpretation für $\mathcal{P}^{i+1}_\sigma(\mu, m)$ und $\mathcal{C}^{i+1}_\sigma(\mu, m)$ ist: für $p', q' \in \mathcal{P}^i_\sigma(\mu, m)$ gilt

  $$p' \preceq q' \Longrightarrow \iota_{i+1}(p') \le \iota_{i+1}(q'),$$

  weil ι_{i+1} und ι_i auf $\mathcal{P}^i_\sigma(\mu, m)$ gleich sind. Für $p' \in \mathcal{P}^i_\sigma(\mu, m)$ kann nicht $pa_1 \dots a_{i+1} \preceq p'$ gelten, da sonst $pa_1 \dots a_{i+1} \in \mathcal{P}^i_\sigma(\mu, m)$ gälte. Sei also $p' \preceq pa_1 \dots a_{i+1}$ für $p' \in \mathcal{P}^{i+1}_\sigma(\mu, m)$.
  - 1. Fall: $p' = pa_1 \dots a_{i+1}$. Dann ist

    $$\iota_{i+1}(p') \le \iota_{i+1}(pa_1 \dots a_{i+1}),$$

    weil "$\le$" reflexiv ist.
  - 2. Fall: $p' \prec pa_1 \dots a_{i+1}$. Dann ist

    $$p' \preceq pa_1 \dots a_i,$$

    und damit

    $$\iota_{i+1}(p') \le \iota_{i+1}(pa_1 \dots a_i) \le w = \iota_{i+1}(pa_1 \dots a_{i+1})$$

Die Parameter-Bedingung gilt wegen Lemma 7.1.10, weil

$$\mathcal{C}_\sigma^i(\mu, m) = \mathcal{C}_\sigma^{i+1}(\mu, m).$$

Für ι_n gilt $\iota_n(pa_1 \dots a_i) \leq \iota_n(pa_1 \dots a_{i+1}), 0 \leq i < n$ nach Konstruktion der ι_{i+1}, und weil ι_n auf $\mathcal{P}_\sigma^{i+1}(\mu, m)$ mit $\iota_{i+1}, 0 \leq i < n$, übereinstimmt. Wegen der Transitivität von "$\leq$" folgt $\iota_n(p) \leq \iota_n(pq)$. Für den Fall, daß ν keine $\forall$-Formel ist, ist das Lemma mit $\iota' := \iota_n$ bewiesen.

Falls ν eine $\forall$-Formel ist bleibt noch, ι_n zu ι' auf $\mathcal{C}_\sigma(\mu, m_0)$ und $\mathcal{P}_\sigma(\mu, m_0)$ zu erweitern.

Aus der J4-Zulässigkeit von σ (Definition 6.4.9) folgt wegen $\sigma_Q(\nu_0) = t_0$, daß $r := \sigma_M(\text{pre}(t_0)) \preceq \sigma_M(\text{pre}(\nu_0)) = pq_0$.

Daraus folgt, daß $r \in \mathcal{P}_\sigma^n(\mu, m)$ ist. Da ι_n eine Interpretation für $\mathcal{P}_\sigma^n(\mu, m)$ ist, folgt $\iota_n(r) \leq \iota_n(pq_0)$ und deshalb $\bar{D}(\iota_n(r)) \subseteq \bar{D}(\iota_n(pq_0))$.

- Angenommen $t_0 \in \mathcal{C}_\sigma^n(\mu, m) = \mathcal{C}_\sigma(\mu, m)$. Dann ist $\mathcal{C}_\sigma(\mu, m_0) = \mathcal{C}_\sigma(\mu, m)$. Da $r \in \mathcal{P}_\sigma^n(\mu, m)$ ist und die Parameter-Bedingung gilt, folgt $\iota_n(t_0) \in \bar{D}(\iota_n(r)) \subseteq \bar{D}(\iota_n(pq_0))$. Definiere jetzt $\iota' := \iota_n$. Die Parameter-Bedingung gilt wegen Lemma 7.1.10, weil sie für m erfüllt ist und $\mathcal{C}_\sigma^n(\mu, m) = \mathcal{C}_\sigma(\mu, m_0)$ ist.
- Angenommen $t_0 \notin \mathcal{C}_\sigma^n(\mu, m) = \mathcal{C}_\sigma(\mu, m)$. Sei dann $c \in \bar{D}(\iota_n(r))$ und definiere $\iota' = \iota_n$ auf $\mathcal{P}_\sigma(\mu, m_0)$ und ι' auf $\mathcal{C}_\sigma(\mu, m_0)$ wie folgt:

  $$\iota'(x) \quad := \quad \begin{cases} c, & \text{falls } x = t_0 \\ \iota_n(x), & \text{für } x \in \mathcal{C}_\sigma^n(\mu, m) \end{cases}$$

  $\iota'(t_0) = c \in \bar{D}(\iota'(r)) \subseteq \bar{D}(\iota'(pq_0))$ nach Konstruktion, weil ι_n und ι' auf $\mathcal{P}_\sigma^n(\mu, m) = \mathcal{P}_\sigma(\mu, m_0)$ übereinstimmen. Die Parameter-Bedingung gilt, denn: für $t_0 \in \mathcal{C}_\sigma(\mu, m_0), r = \sigma_M(\text{pre}(t_0))$ gilt $\iota'(t_0) = c \in \bar{D}(\iota'(r))$. $\square$

Satz 7.1.22 (Korrektheitssatz) *Sei A eine Formel aus $L(C^*)$, der Sprache von J4, μ eine Multiplizität für* FA, *A^μ ein indizierter Formelbaum für* FA, *σ eine J4-zulässige Substitution und es gebe eine Basis für A^μ aus σ-komplementären Konnektionen. Dann ist A eine für J4 gültige Formel.*

Beweis: Angenommen, die Voraussetungen sind erfüllt, aber A ist keine J4-gültige Formel. Sei C die Menge der in A auftretenden Konstanten. Weil A nicht gültig ist, gibt es ein Modell $(T, \leq, D, \bar{D}, \Vdash)$ und eine Interpretation ι, so daß für ein $w \in T$ die Bilder aller Konstanten aus A unter ι in $\bar{D}(w)$ liegen und es gilt:

$$w \nVdash \iota(A) \text{ d.h. } w \Vdash \iota(\mathrm{F}A)$$

Es läßt sich deshalb eine Interpretation ι' konstruieren, so daß der Startpfad durch den Formelbaum für FA erfüllbar ist: es sei der Startpfad $m := \{a_0\}$, d.h. $\mathrm{sform}(a_0) = \mathrm{F}A$.

$$\begin{array}{lcl} \mathcal{A}(m) & := & \{a_0\} \\ \mathcal{P}_\sigma(\mu, m) & := & \{a_0\} \\ \mathcal{C}_\sigma(\mu, m) & := & C \end{array}$$

Definiere ι' wie folgt:

$$\iota'(x) \quad := \quad \begin{cases} w & \text{falls } x = a_0 \\ \iota(x) & \text{falls } x \in \mathcal{C}_\sigma(\mu, m) \end{cases}$$

ι' ist eine Interpretation, da $\mathcal{P}_\sigma(\mu, m)$ als einziges Element nur a_0 enthält. Die Parameter-Bedingung gilt, weil alle Konstanten von A in $\bar{D}(w)$ liegen. Da $\iota'(a_0) = w$ und $w \Vdash \iota'(\mathrm{F}A)$ gilt, ist auch die erste Bedingung aus Definition 7.1.9 erfüllt.

Es folgt $\iota'(a_0) \Vdash \iota'(\sigma_Q(\mathrm{sform}(a_0)))$. Also ist der Startpfad unter σ durch ι' J4-erfüllbar. Wegen der vorangegangenen Lemmata folgt, daß zumindest einer der durch ordentliche Reduktionen erhaltenen atomaren Pfade ebenfalls J4-erfüllbar ist. Da aber alle atomaren Pfade nach Voraussetzung σ-komplementär sind, ist wegen Lemma 7.1.12 keiner von ihnen J4-erfüllbar. Aus diesem Widerspruch folgt die Behauptung. □

7.2 Vollständigkeit des Matrixkalküls

In diesem Abschnitt wird gezeigt werden, daß es für jede gültige Formel A eine Multiplizität μ und eine Substitution σ gibt, so daß der indizierte Formelbaum A^μ für FA eine Basis aus σ-komplementären Konnektionen besitzt.

7.2.1 J4-Hintikka-Mengen

Definition 7.2.1 *Es sei $(T_0, \leq_0, D_0, \bar{D}_0)$ ein J4-Kripke-Rahmen, und es sei $P_0 \subseteq T_0$ und $C_0 \subseteq D_0$. Eine* Präfixformel *über P_0 und C_0 ist ein Paar $p : B$, wobei $p \in P_0$ und B eine signierte Formel aus $L(C_0)$ ist.*

Eine Menge M von Präfixformeln über P_0 und C_0 heißt erfüllbar, *wenn es ein J4-Modell $(T, \leq, D, \bar{D}, \Vdash)$ und eine Interpretation ι von P_0 nach T und C_0 nach D gibt, so daß für jedes Paar $p : B \in M$ alle Konstanten von $\iota(B)$ in $\bar{D}(\iota(p))$ liegen und $\iota(p) \Vdash \iota(B)$.*

Definition 7.2.2 *Es sei $(T_0, \leq_0, D_0, \bar{D}_0)$ ein J4-Kripke-Rahmen, $P_0 \subseteq T_0$ und $C_0 \subseteq D_0$. Es sei*

$$\tilde{D}_0 : T_0 \longrightarrow \wp(D_0)$$

eine Partition von D_0, so daß gilt

$$D_0 = \bigcup_{w \in T_0} \tilde{D}_0(w)$$

Für eine Menge M von Präfixformeln über P_0 und C_0 sei

$$\tilde{C}_0(p) := C_0 \cap \tilde{D}_0(p)$$

Definition 7.2.3 *Es sei $(T_0, \leq_0, D_0, \bar{D}_0)$ ein J4-Kripke-Rahmen. Es gelte $P_0 \subseteq T_0$ und $C_0 \subseteq D_0$. Eine Menge H von Präfixformeln über P_0 und C_0 heißt* J4-Hintikka-Menge, *wenn*

1. *aus $p : A \in H$ folgt, daß es zu jeder Konstanten in A ein $q \leq_0 p$ gibt mit $c \in \tilde{C}_0(q)$;*
2. *es für keine Primformel Q aus $L(C_0)$ ein $p \in P_0$ gibt, so daß $p : \mathrm{T}Q \in H$ und $p : \mathrm{F}Q \in H$ gilt;*[6]
3. *aus $p : \alpha \in H$ folgt, daß $p : \alpha_1 \in H$ und $p : \alpha_2 \in H$;*
4. *aus $p : \beta \in H$ folgt, daß $p : \beta_1 \in H$ oder $p : \beta_2 \in H$;*

[6] Für J muß man fordern, daß aus $p : \mathrm{T}Q \in H$ auch $q : \mathrm{T}Q \in H$ für alle q mit $p \leq_0 q$ folgt. Für J4 ergibt sich das aus der noch folgenden ν-Bedingung.

5. *aus* $p:\gamma\in H$ *folgt, daß für alle* $c\in\tilde{C}_0(q)$ *mit* $q\leq_0 p$ *gilt* $p:\gamma_0(c)\in H$;

6. *aus* $p:\delta\in H$ *folgt, daß für mindestens ein* $c\in\tilde{C}_0(p)$ $p:\delta_0(c)\in H$ *gilt;*

7.
   - *für* $\forall$*-Formeln* ν *gilt:*
     aus $p:\nu\in H$ *folgt für alle* q *mit* $p\leq_0 q$, *alle* r *mit* $r\leq_0 q$ *und alle* $c\in\tilde{C}_0(r)$, *daß* $q:\nu_0(c)\in H$ *gilt;*
   - *für andere* ν*-Formeln(Primformeln, Negationen und Implikationen) aus* $p:\nu\in H$ *folgt, daß für alle* $q\in P_0$ *mit* $p\leq_0 q$ $q:\nu_1\in H$ *oder* $q:\nu_2\in H$ *gilt;*

8.
   - *für* $\forall$*-Formeln* π *gilt:*
     aus $p:\pi\in H$ *folgt, es gibt ein* q *mit* $p\leq_0 q$, *ein* r *mit* $r\leq_0 q$ *und ein* $c\in\tilde{C}_0(r)$ *mit* $q:\pi_0(c)\in H$;
   - *für andere* π*-Formeln aus* $p:\pi\in H$ *folgt, daß für mindestens ein* $q\in P_0$ *mit* $p\leq_0 q$ $q:\pi_1\in H$ *und* $q:\pi_2\in H$ *gilt.*

Satz 7.2.4 *Jede J4-Hintikka-Menge von Präfixformeln über* P_0 *und* C_0 *ist in einem Modell* $(P_0,\leq,C_0,\bar{C}_0,\Vdash)$ *mit der Identitätsfunktion als Interpretation* ι *erfüllbar, wobei für* $p\in P_0$ $\tilde{C}_0(p)\subseteq\bar{C}_0(p)$ *ist.*

Beweis: M sei eine J4-Hintikka-Menge von Präfixformeln über P_0 und C_0. P_0 sei die Menge der Präfixe, die in M auftreten. C_0 sei die Menge der Konstanten, die in M auftreten. Es sei $\bar{C}_0(p):=\bigcup_{q\leq_0 p}\tilde{C}_0(q)$. Für $p,q\in P_0$ sei $p\leq q$, falls $p\leq_0 q$.

Dann ist $(P_0,\leq_0,C_0,\bar{C}_0)$ ein J4-Kripke-Rahmen. ι sei die Identität auf C_0 und P_0. Für atomare signierte Formeln Q und Präfixe $p\in P_0$ sei $\Vdash$ wie folgt definiert:

$$p:Q\in M\Longrightarrow p\Vdash Q$$

Dadurch ist $\Vdash$ für alle Formeln eindeutig bestimmt, so daß

$$(P_0,\leq,C_0,\bar{C}_0,\Vdash)$$

ein J4-Kripke-Modell ist. Es bleibt zu zeigen: für jede signierte Formel A und jedes Präfix $p\in P_0$ gilt

$$p:A\in M\Longrightarrow p\Vdash A$$

Der Beweis läuft durch Induktion nach dem Grad von A.[7]

1. Für Primformeln folgt die Behauptung im Fall von J4 aus der ν-Bedingung in der Definition der Hintikka-Menge, und im Fall von J aus der dort in der Fußnote angegebenen Zusatzbedingung.

2. A sei vom Typ α.

$$\begin{aligned} p:\alpha\in M &\overset{\text{Def.}}{\Longrightarrow} p:\alpha_1\in M \text{ und } p:\alpha_2\in M\\ &\overset{\text{IV}}{\Longrightarrow} p\Vdash\alpha_1 \text{ und } p\Vdash\alpha_2\\ &\overset{*}{\Longrightarrow} p\Vdash\alpha \end{aligned}$$

3. A sei vom Typ β.

$$\begin{aligned} p:\beta\in M &\overset{\text{Def.}}{\Longrightarrow} p:\beta_1\in M \text{ oder } p:\beta_2\in M\\ &\overset{\text{IV}}{\Longrightarrow} p\Vdash\beta_1 \text{ oder } p\Vdash\beta_2\\ &\overset{*}{\Longrightarrow} p\Vdash\beta \end{aligned}$$

4. A sei vom Typ γ.

$$\begin{aligned} p:\gamma\in M &\overset{\text{Def.}}{\Longrightarrow} \forall c\in\tilde{C}_0(q) \text{ mit } q\le_0 p \text{ ist } p:\gamma_0(c)\in M\\ &\overset{\text{IV}}{\Longrightarrow} \forall c\in\bar{C}_0(p) \text{ gilt } p\Vdash\gamma_0(c)\\ &\overset{*}{\Longrightarrow} p\Vdash\gamma \end{aligned}$$

5. A sei vom Typ δ.

$$\begin{aligned} p:\delta\in M &\overset{\text{Def.}}{\Longrightarrow} \exists c\in\tilde{C}_0(p) \text{ mit } p:\delta_0(c)\in M\\ &\overset{\text{IV}}{\Longrightarrow} \exists c\in\bar{C}_0(p) \text{ mit } p\Vdash\delta_0(c)\\ &\overset{*}{\Longrightarrow} p\Vdash\delta \end{aligned}$$

6. A sei vom Typ π.

[7]Bei den mit “*” bezeichneten Zeilen geht die Modelleigenschaft ein (vgl. Definition 5.1.2).

(a) π sei eine $\forall$-Formel.

$$\begin{aligned} p:\pi \in M &\overset{\text{Def.}}{\Longrightarrow} \exists q \in P_0, p \leq_0 q\, \exists r \leq_0 q\, \exists c \in \tilde{C}_0(r) \\ &\quad\ \text{mit } q:\pi_0(c) \in M \\ &\overset{\text{IV}}{\Longrightarrow} \exists q \in P_0, p \leq q\, \exists c \in \bar{C}_0(q) \text{ mit } q \Vdash \pi_0(c) \\ &\overset{*}{\Longrightarrow} p \Vdash \pi \end{aligned}$$

(b) π sei keine $\forall$-Formel.

$$\begin{aligned} p:\pi \in M &\overset{\text{Def.}}{\Longrightarrow} \exists q \in P_0, p \leq_0 q \text{ mit } q:\pi_1 \in M \text{ und } q:\pi_2 \in M \\ &\overset{\text{IV}}{\Longrightarrow} \exists q \in P_0, p \leq q \text{ mit } q \Vdash \pi_1 \text{ und } q \Vdash \pi_2 \\ &\overset{*}{\Longrightarrow} p \Vdash \pi \end{aligned}$$

7. A sei vom Typ ν.

(a) ν sei eine $\forall$-Formel.

$$\begin{aligned} p:\nu \in M &\overset{\text{Def.}}{\Longrightarrow} \forall q, r \in P_0 \text{ mit } p \leq_0 q, r \leq_0 q, \forall c \in \tilde{C}_0(r) \text{ gilt} \\ &\quad\ q:\nu_0(c) \in M \\ &\overset{\text{IV}}{\Longrightarrow} \forall q \in P_0 \text{ mit } p \leq q, \forall c \in \bar{C}_0(q) \text{ gilt} \\ &\quad\ q \Vdash \nu_0(c) \\ &\overset{*}{\Longrightarrow} p \Vdash \nu \end{aligned}$$

(b) ν sei keine $\forall$-Formel.

$$\begin{aligned} p:\nu \in M &\overset{\text{Def.}}{\Longrightarrow} \forall q \in P_0 \text{ mit } p \leq_0 q \text{ gilt } q:\nu_1 \in M \text{ oder} \\ &\quad\ q:\nu_2 \in M \\ &\overset{\text{IV}}{\Longrightarrow} \forall q \in P_0 \text{ mit } p \leq q \text{ gilt } q \Vdash \nu_1 \text{ oder } q \Vdash \nu_2 \\ &\overset{*}{\Longrightarrow} p \Vdash \nu \qquad \square \end{aligned}$$

Sei μ wieder eine Multiplizität für eine signierte Formel A aus $\mathrm{L}(C^*)$ und σ eine zulässige Substitution für A^μ. Sei m ein Pfad durch A^μ. Wie gehabt bezeichnet $\mathcal{C}_\sigma(\mu, m)$ die Bilder der Konstanten in m unter σ_Q und $\mathcal{P}_\sigma(\mu, m)$ die Anfangsstücke der Bilder der Präfixe unter σ_M.

Definition 7.2.5 *Die zu einem Präfix p gehörige Menge von Konstanten* $\mathcal{C}_\sigma(\mu, m, p)$ *ist definiert durch*

$$\mathcal{C}_\sigma(\mu, m, p) := \{a \mid a \in \mathcal{C}_\sigma(\mu, m) \text{ und } \sigma_M(pre(a)) = p\}$$

Bemerkung 7.2.6 *Für den Fall, daß in A Konstanten aus* $L(C^*)$ *auftreten und* p_0 *das Präfix der Wurzelposition von* A^μ *ist, enthält* $\mathcal{C}_\sigma(\mu, m, p_0)$ *diese als zusätzliche Elemente.*

Im nächsten Abschnitt wird ein Komplettierungsverfahren für Pfade definiert werden. Die Punkte 7.2.7–7.2.11 werden für den Vollständigkeitssatz benötigt, falls das Verfahren terminiert.

Definition 7.2.7 *Ein atomarer Pfad m durch* A^μ *heißt* komplett unter σ, *wenn*

1. *es für alle Positionen* $\gamma^\kappa \in m$ *mit* $p = \sigma_M(pre(\gamma^\kappa))$ *und alle t mit* $t \in \mathcal{C}_\sigma(\mu, m, q)$ *für ein* $q \in \mathcal{P}_\sigma(\mu, m)$ *mit* $q \preceq p$ *ein* $i, 1 \leq i \leq \mu(\gamma_0)$, *gibt mit* $\sigma_Q(\gamma_0^{\kappa i}) = t$;
2.
   - *es für alle* $\forall$*-Formeln* ν^κ *mit* $p = \sigma_M(pre(\nu^\kappa))$, *alle* $q, r \in \mathcal{P}_\sigma(\mu, m)$ *mit* $p \preceq q$ *und* $r \preceq q$ *und alle t mit* $t \in \mathcal{C}_\sigma(\mu, m, r)$ *ein* $i, 1 \leq i \leq \mu(\nu_0)$, *gibt mit* $\sigma_Q(\nu_0^{\kappa i}) = t$ *und* $\sigma_M(pre(\nu_0^{\kappa i})) = q$.
   - *es für alle anderen* ν^κ *mit* $p = \sigma_M(pre(\nu^\kappa))$ *und alle* $q \in \mathcal{P}_\sigma(\mu, m)$ *mit* $p \preceq q$ *ein* $i, 1 \leq i \leq \mu(\nu_1)$, *gibt mit* $\sigma_M(pre(\nu_1^{\kappa i})) = q$ *oder ein* $i, 1 \leq i \leq \mu(\nu_2)$, *gibt mit* $\sigma_M(pre(\nu_2^{\kappa i})) = q$.

Definition 7.2.8 *Sei m ein Pfad in* A^μ. *Die Menge der Präfixformeln zur assoziierten Menge von m ist P-Formeln(m):*

$$\text{P-Formeln}(m) := \{p : B \mid p = \sigma_M(pre(a)), B = \sigma_Q(sform(a)), a \in \mathcal{A}(m)\}$$

P-Formeln(m) ist eine Menge von Präfixformeln über $\mathcal{P}_\sigma(\mu, m)$ *und* $\mathcal{C}_\sigma(\mu, m)$.

Diese beiden Definitionen ermöglichen die Formulierung des folgenden Lemmas. Es stellt eine Verbindung zwischen *kompletten Pfaden* und *Hintikka-Mengen* her.

Lemma 7.2.9 *Sei m ein nicht σ-komplementärer, atomarer Pfad in A^μ. Wenn m komplett ist, ist P-Formeln(m) eine J4-Hintikka-Menge.*

Beweis: Setze in den Definitionen 7.2.2 und 7.2.3

$$\begin{aligned} \leq_0 &:= \preceq \\ C_0 &:= \mathcal{C}_\sigma(\mu, m) \\ P_0 &:= \mathcal{P}_\sigma(\mu, m) \\ \tilde{C}_0(p) &:= \mathcal{C}_\sigma(\mu, m, p) \end{aligned}$$

Die Bedingung 1 für Hintikka-Mengen bezüglich der Konstanten ist erfüllt, weil σ J4-zulässig ist und wegen Bemerkung 7.2.6. Der Primformel-Fall ist erfüllt, weil m nicht σ-komplementär ist. Die übrigen Bedingungen aus der Definition der Hintikka-Mengen gelten ebenfalls, es wird hier nur die etwas kompliziertere ν-Bedingung gezeigt:

Sei $p : \nu \in$ P-Formeln(m).

1. ν sei eine $\forall$-Formel $\forall x A(x)$: Sei $\nu = \sigma_Q(\mathrm{sform}(a^\kappa))$ für eine Position $a^\kappa \in \mathcal{A}(m)$. Dann gibt es, weil m komplett ist, für alle $q, r \in \mathcal{P}_\sigma(\mu, m)$ mit $p \preceq q$, $r \preceq q$ und alle $t \in \mathcal{C}_\sigma(\mu, m, r)$ ein $i, 1 \leq i \leq \mu(a_0)$, so daß
$$\sigma_Q(a_0^{\kappa i}) = t$$
und
$$\sigma_M(\mathrm{pre}(a_0^{\kappa i})) = q$$
ist. Also ist
$$\sigma_Q(\mathrm{sform}(a_0^{\kappa i})) = \mathrm{T}A(t).$$
Es gilt also $q : \nu_0(t) \in$ P-Formeln(m), da m atomar ist.

2. ν sei keine $\forall$-Formel: Sei $\nu = \sigma_Q(\mathrm{sform}(a^\kappa))$ für eine Position $a^\kappa \in \mathcal{A}(m)$. Dann gibt es, weil m komplett ist, für alle $q \in \mathcal{P}_\sigma(\mu, m)$ mit $p \preceq q$ ein $i, 1 \leq i \leq \mu(\nu_1)$, so daß
$$\sigma_M(\mathrm{pre}(a_1^{\kappa i})) = q$$
oder ein $i, 1 \leq i \leq \mu(\nu_2)$, so daß
$$\sigma_M(\mathrm{pre}(a_2^{\kappa i})) = q.$$

Des weiteren gilt

$$\sigma_Q(a_1^{\kappa i})) = \nu_1,$$

$$\sigma_Q(a_2^{\kappa i})) = \nu_2.$$

Also ist $q : \nu_1 \in$ P-Formeln(m) oder $q : \nu_2 \in$ P-Formeln(m). $\Box$

Satz 7.2.10 *Sei m ein nicht σ-komplementärer Pfad in A^μ. Wenn m komplett unter σ ist, ist m unter σ J4-erfüllbar.*

Beweis: Wegen Lemma 7.2.9 ist P-Formeln(m) eine J4-Hintikka-Menge über $\mathcal{P}_\sigma(\mu, m)$ und $\mathcal{C}_\sigma(\mu, m)$.

Es sei $\bar{\mathcal{C}}_\sigma(\mu, m)$ eine Abbildung von $\mathcal{P}_\sigma(\mu, m)$ nach $\wp(\mathcal{C}_\sigma(\mu, m))$ mit

$$\bar{\mathcal{C}}_\sigma(\mu, m, p) := \bigcup_{q \preceq p} \mathcal{C}_\sigma(\mu, m, q)$$

Wegen Satz 7.2.4 ist diese Menge mit der Identitätsfunktion als Interpretation ι in $(\mathcal{P}_\sigma(\mu, m), \preceq, \mathcal{C}_\sigma(\mu, m), \bar{\mathcal{C}}_\sigma(\mu, m), \Vdash)$ erfüllbar, wobei $\mathcal{C}_\sigma(\mu, m, p) \subseteq \bar{\mathcal{C}}_\sigma(\mu, m, p) = \bigcup_{q \preceq p} \tilde{C}_\sigma(\mu, m, q)$ gilt.

Die beiden Bedingungen aus Definition 7.1.9 sind leicht gezeigt:

1. Sei $a \in \mathcal{A}(m), p = \sigma_M(\mathrm{pre}(a))$ und $B = \sigma_Q(\mathrm{sform}(a))$. Dann ist $p : B \in$ P-Formeln(m). Wegen Satz 7.2.4 ist P-Formeln(m) mit $\iota = \mathrm{id}$ erfüllbar, d.h.
   - alle Konstanten von $\iota(B)$ liegen in $\bar{\mathcal{C}}_\sigma(\mu, m, p)$ und
   - $\iota(p) \Vdash \iota(B)$.
2. Sei $v \in \mathcal{C}_\sigma(\mu, m)$ mit $q = \sigma_M(\mathrm{pre}(v))$. Dann ist nach Definition

   $$v \in \mathcal{C}_\sigma(\mu, m, q)) \subseteq \bar{\mathcal{C}}_\sigma(\mu, m, q),$$

   d.h.

   $$\iota(v) \in \bar{\mathcal{C}}_\sigma(\mu, m, \iota(q)).$$

   Also ist die Parameter-Bedingung erfüllt. $\Box$

Korollar 7.2.11 *Sei A eine Formel aus $L(C^*)$, μ eine Multiplizität, σ eine J4-zulässige Substitution und m ein atomarer Pfad durch A^μ, den indizierten Formelbaum für* FA. *Ist m komplett, so ist A nicht gültig.*

Beweis: Wegen Satz 7.2.10 ist der Pfad m unter σ erfüllbar in einem Modell $(T, \leq, D, \bar{D}, \Vdash)$. Die Startposition a_0 des Formelbaums für FA ist in $\mathcal{A}(m)$. Deshalb gilt, daß alle Konstanten von A in $\bar{D}(\iota(\sigma_M(\mathrm{pre}(a_0)))) = \bar{D}(\iota(a_0))$ enthalten sind und

$$\iota(a_0) \Vdash \iota(\mathrm{sform}(a_0)) = \iota(\mathrm{F}A).$$

Also gibt es ein Modell und eine Interpretation, so daß A nicht an allen Knoten erzwungen wird, d.h. A ist nicht gültig. □

7.2.2 Komplettierung von Pfaden

In diesem Abschnitt wird eine systematische Prozedur definiert, die, wenn sie terminiert, entweder einen der atomaren, nicht σ-komplementären Pfade durch einen Formelbaum für FA komplettiert, oder eine Multiplizität μ und eine zulässige Substitution σ liefert, für die eine Basis aus σ-komplementären Konnektionen existiert. Weil ein kompletter, nicht σ-komplementärer Pfad eine J4-Hintikka-Menge liefert, folgt aus der Existenz eines solchen Pfades, daß FA erfüllbar, also A nicht gültig ist. Wenn die Prozedur nicht terminiert, liefert sie eine erfüllbare Menge, die FA enthält. Für eine gültige Formel A muß die Prozedur also eine σ-komplementäre Basis liefern. Daraus folgt die Vollständigkeit.

Definition 7.2.12 *Sei $Q(\mu)$ die Menge der atomaren Pfade durch den indizierten Formelbaum A^μ für die Formel* FA. *Es wird eine Folge von Paaren aus Multiplizitäten und J4-zulässigen Substitutionen*

$$\langle \mu^0, \sigma^0 \rangle, \langle \mu^1, \sigma^1 \rangle, \ldots, \langle \mu^n, \sigma^n \rangle, \ldots$$

wie folgt definiert:

- $n = 0$. *μ^0 sei konstant 0 auf Γ_0 und $\boldsymbol{\nu}_i, i \in \{0, 1, 2\}$, und σ^0 sei die Identität auf $T_Q(\mu^0)$ und $T_M(\mu^0)$.*

σ^0 ist eine modale Substitution, da

$$\forall u, v \in T_M(\mu^0), u \in \sigma^0(v) \Longrightarrow \sigma^0(u) = u$$

gilt.

Da $\Gamma_0(\mu^0) = \boldsymbol{\nu}_0(\mu^0) = \emptyset$ ist, sind die Relationen $\rightsquigarrow_M, \rightsquigarrow_Q, \cong_M$ und $\cong_Q$ leer. Daraus folgt, daß σ^0 eine J4-zulässige Substitution gemäß Definition 6.4.9 ist.

- *$n \longrightarrow n+1$. Wenn es eine Basis für A^μ aus σ^n-komplementären Konnektionen gibt, bricht die Konstruktion ab.*

  Wenn es einen nicht σ^n-komplementären Pfad in $Q(\mu^n)$ gibt, der unter σ^n komplett ist, bricht die Konstruktion ab.

  Sonst: wähle aus den nicht σ^n-komplementären Pfaden in $Q(\mu^n)$ einen Pfad m mit einer minimalen assoziierten Menge $\mathcal{A}(m)$. Sei x^κ die erste[8] nicht-atomare Position in m. x^κ muß also vom Typ γ oder ν sein. Sei $p = \sigma^n(pre(x^\kappa))$ und x_0 bzw. $x_i, i \in \{0,1,2\}$ seien die Nachfolger von x im nicht-indizierten Formelbaum für A.

  1. *x^κ ist eine γ-Position. Dann ist $x_0 \in T_Q(\mu^n)$ (bzw. für $n = 0$ noch nicht im indizierten Formelbaum enthalten). Sei K die Menge der Konstanten c, für die es im Pfad m noch keine Position gibt, deren sform dieses $\gamma_0(c)$ ist, d.h.:*

     $$K := \{c \in \bigcup_{\substack{q \in \mathcal{P}_{\sigma^n}(\mu^n, m) \\ q \preceq p}} \mathcal{C}_{\sigma^n}(\mu^n, m, q) \mid \neg\exists j\, 1 \le j \le \mu^n(x_0) : \sigma^n_Q(x_0^{\kappa j}) = c\}$$

     Zwei Fälle sind zu unterscheiden:

     (a) *$K = \emptyset$. In diesem Fall ist fast nichts zu tun: setze $\mu^{n+1} := \mu^n$ und $\sigma^{n+1} := \sigma^n$.*

     (b) *$K = \{c_1, c_2, \ldots, c_l\}, l \ge 1$. Setze*

     $$\mu^{n+1}(b) := \begin{cases} \mu^n(b) + l, & \textit{falls } b = x_0 \\ \mu^n(b), & \textit{sonst} \end{cases}$$

[8] Es wird eine geeignete Ordnung der nicht-atomaren Positionen angenommen.

Setze

$$\sigma^{n+1}(b) := \begin{cases} c_i, & \textit{falls } b = x_0^{\kappa j}, \\ & \mu^n(x_0) < j \leq \mu^{n+1}(x_0), \\ & i = j - \mu^n(x_0) \\ \sigma^n(b), & \textit{sonst} \end{cases}$$

Der neue Formelbaum zur Multiplizität μ^{n+1} entsteht dadurch, daß man unterhalb von x^κ eine Kopie der Positionen unterhalb des x aus dem nicht-indizierten Formelbaum anhängt und entsprechend der neuen Multiplizität indiziert. Für die Positionen in diesem neuen Teil unterhalb von $x_0^{\kappa j}$ seien σ_Q und σ_M die Identität.
σ^{n+1} ist eine modale Substitution, weil σ^n und σ^{n+1} auf den Elementen von $T_M(\mu^n)$ übereinstimmen und weil σ^{n+1} auf den neuen Positionen ja die Identität ist.
Außerdem folgt, daß die Reduktionsordnung $\rhd$ in dem neuen Teil des Baumes mit der Baumordnung $\gg$ zusammenfällt, also nicht reflexiv ist.
Bei der Spitzenposition des neuen Baumabschnittes $x_0^{\kappa j}$, die ja vom Typ γ_0 ist, können bildlich gesprochen nur Pfeile ankommen, so daß insgesamt die Reduktionsordnung auf dem ganzen Baum irreflexiv bleibt. Es können also wieder unter Einhaltung der Reduktionsordnung alle atomaren Pfade gebildet werden.
Für die neuen Positionen gilt

$$\sigma^{n+1}(x_0^{\kappa j}) = c_i$$

und für alle c_i ist

$$\sigma^{n+1}(pre(c_i)) \preceq \sigma^{n+1}(pre(x_0^{\kappa j}))$$

nach Konstruktion. Deshalb ist σ^{n+1} eine zulässige Substitution.

2. *$x^\kappa \in \boldsymbol{\nu}(\mu^n)$ und $form(x^\kappa)$ ist keine $\forall$-Formel. Es sei P die Menge der Präfixe*

$$q \in \mathcal{P}_{\sigma^n}(\mu^n, m) \textit{ mit } p \preceq q,$$

für die es kein $j, 1 \leq j \leq \mu^n(x_d), d \in \{1,2\}$ *gibt mit*

$$p\sigma^n(x_d^{\kappa j}) = q.$$

(a) $P = \emptyset$. *In diesem Fall setze* $\mu^{n+1} := \mu^n$ *und* $\sigma^{n+1} := \sigma^n$. σ^{n+1} *ist dann nach Voraussetzung zulässig.*

(b) $P = \{q_1, q_2, \ldots, q_l\}, l \geq 1$. *Dann ist* μ^{n+1} *wie folgt definiert:*

$$\mu^{n+1}(b) := \begin{cases} \mu^n(b) + l, & \text{falls } b = x_d \\ \mu^n(b), & \text{sonst} \end{cases}$$

Seien $r_1, \ldots, r_l$ *Elemente von* $T_M(\mu^n)^\star$, *so daß*

$$q_i = pr_i, 1 \leq i \leq l$$

gilt. Dann sei σ^{n+1} *wie folgt definiert:*

$$\begin{array}{rcl} \sigma_M^{n+1}(b) & := & \begin{cases} r_i, & \text{falls } b = x_d^{\kappa j}, \\ & \mu^n(x_d) < j \leq \mu^{n+1}(x_d), \\ & i = j - \mu^n(x_d) \\ \sigma_M^n(x), & \text{sonst} \end{cases} \\ \sigma_Q^{n+1}(b) & := & \sigma_Q^n(b) \end{array}$$

Es ist zu zeigen, daß σ^{n+1} *eine zulässige Substitution ist:*

- $u \in \sigma^{n+1}(v) \Longrightarrow \sigma^{n+1}(u) = u$ *für* $u, v \in T_M(\mu^{n+1})$:
  Nur der Fall

  $$v = x_d^{\kappa j}, d \in \{1,2\}, \mu^n(x_d) < j \leq \mu^{n+1}(x_d)$$

  ist zu betrachten. Sei $u \in \sigma^{n+1}(x_d^{\kappa j}) = r_i$. *Dann ist*

  $$u \in q_i \in \mathcal{P}_{\sigma^n}(\mu^n, m).$$

  Also ist für ein $a \in \mathcal{A}(m)$ $u \in \sigma_M^n(pre(a))$. *Daraus folgt, weil* σ_M^n *eine modale Substitution ist, mit 6.4.3*

  $$\sigma_M^{n+1}(u) = \sigma_M^n(u) = u.$$

- $\triangleright$ *bleibt irreflexiv. Die Argumentation ist dieselbe wie oben: Die* $x_d^{\kappa j}$ *sind neu. Für den Graphen der* $\triangleright$*-Relation bedeutet das, daß die* $x_d^{\kappa j}$ *nur ankommende Pfeile besitzen. Sie können deshalb an keinem Zyklus beteiligt sein, der nicht schon vorher vorhanden gewesen wäre.*

– *Der quantorenlogische Teil der Substitution σ^n ist unverändert geblieben. Deshalb gilt die Bedingung 2 aus 6.4.9 nach wie vor.*

3. *$x^\kappa \in \boldsymbol{\mathcal{V}}(\mu^n)$ und $form(x^\kappa)$ ist eine $\forall$-Formel. Seien $q_1, q_2, \ldots, q_l$ die Präfixe aus $\mathcal{P}_{\sigma^n}(\mu^n, m)$ mit $p \preceq q_i$.*

   Seien $r_i \in T_M(\mu^n)^\star, 1 \leq i \leq l$, so daß $q_i = pr_i$. Für jedes q_i sei K_i die Menge der Konstanten $c \in \mathcal{C}_{\sigma^n}(\mu^n, m, q), q \preceq q_i$, für die kein $j, 1 \leq j \leq \mu^n(x_0)$, existiert, so daß $\sigma^n_Q(x_0^{\kappa j}) = c$ und $\sigma^n_M(x_0^{\kappa j}) = r_i$ gilt.

   Sei μ^{n+1} wie folgt definiert:

$$\mu^{n+1}(b) := \begin{cases} \mu^n(b) + \sum_{i=1}^{l} |K_i|, & \textit{falls } b = x_0 \\ \mu^n(b), & \textit{sonst} \end{cases}$$

   Sei $K_i = \{c_1^i, \ldots, c_{l_i}^i\}, 1 \leq i \leq l$, und K sei eine Aufzählung von $\bigcup_{i=1}^{l} K_i$. Dann sei σ^{n+1} wie folgt definiert:

$$\sigma_M^{n+1}(b) \quad := \quad \begin{cases} r_i, & \textit{falls } b = x_0^{\kappa s}, \\ & \mu^n(x_0) < s \leq \mu^{n+1}(x_0), \\ & \textit{und das s-te Element von } K \textit{ ist} \\ & c_h^i, 1 \leq h \leq l_i, 1 \leq i \leq l \\ \sigma_M^n(b), & \textit{sonst} \end{cases}$$

$$\sigma_Q^{n+1}(b) \quad := \quad \begin{cases} c_h^i, & \textit{falls } b = x_0^{\kappa s}, \\ & \mu^n(x_0) < s \leq \mu^{n+1}(x_0), \\ & \textit{und das s-te Element von } K \textit{ ist} \\ & c_h^i, 1 \leq h \leq l_i, 1 \leq i \leq l \\ \sigma_Q^n(b), & \textit{sonst} \end{cases}$$

   Es ist zu zeigen, daß σ^{n+1} eine zulässige Substitution ist:

   – *$u \in \sigma_M^{n+1}(v) \Longrightarrow \sigma_M^{n+1}(u) = u$ für $u, v \in T_M(\mu^{n+1})$ aus dem gleichen Grund wie oben.*
   – *$\rhd$ bleibt aus dem gleichen Grund wie oben irreflexiv.*
   – *für alle $u, v \in T_Q(\mu^{n+1})$ muß gelten, daß*

$$\sigma_Q^{n+1}(u) = v \Longrightarrow \sigma_M^{n+1}(pre(v)) \preceq \sigma_M^{n+1}(pre(u))$$

Für $u \in T_Q(\mu^n)$ ist das nach Voraussetzung erfüllt. Sei also $u = x_0^{\kappa j}, \mu^n(x_0) < j \leq \mu^{n+1}(x_0)$.

$$\begin{array}{rrcl} & \sigma_Q^{n+1}(x_0^{\kappa j}) & = & c_h^i \in \mathcal{C}_{\sigma^n}(\mu^n, m, q) \textit{ für } q \preceq q_i \\ \Longrightarrow & \sigma_M^n(pre(c_h^i)) & \preceq & q_i \\ \textit{und} & \sigma_M^{n+1}(pre(x_0^{\kappa j})) & = & pr_i = q_i \\ \Longrightarrow & \sigma_M^{n+1}(pre(c_h^i)) & \preceq & \sigma_M^{n+1}(pre(x_0^{\kappa j})) \end{array}$$

Wie beschrieben, kann man nach der Bearbeitung einer Position und der damit verbundenen Erweiterung des indizierten Formelbaums unter Einhaltung der Reduktionsordnung alle atomaren Pfade durch den neuen indizierten Formelbaum erzeugen.

x^κ war die erste nicht-atomare Position in m. In allen neuen Pfaden in $Q(\mu^{n+1})$, die durch die Erhöhung der Multiplizität irgendeiner Position entstanden sind, wird x^κ als letztes Element in der Ordnung der nicht-atomaren Positionen vermerkt, damit zyklisch alle nicht-atomaren Positionen bearbeitet werden.

Satz 7.2.13 (Vollständigkeitssatz) *Ist A eine gültige Formel aus $L(C^*)$, der Sprache von J4, dann gibt es eine Multiplizität μ, eine J4-zulässige Substitution σ und eine σ-komplementäre Basis für A^μ, dem indizierten Formelbaum für* F*A.*

Beweis: Führt man die systematische Komplettierung der Pfade durch A^μ aus Definition 7.2.12 durch, gibt es zwei Möglichkeiten:

1. Die Komplettierung stoppt, weil eine Multiplizität μ und eine zulässige Substitution σ gefunden wurden, so daß
   (a) eine σ-komplementäre Basis für A^μ existiert, oder
   (b) ein nicht σ-komplementärer Pfad existiert, der unter σ komplett ist. Wegen Korollar 7.2.11 ist das nicht möglich, da A eine J4-gültige Formel ist.
2. Die Komplettierung stoppt nicht. Wie in Definition 7.2.12 geschildert, erhält man bei jedem Komplettierungsschritt zu dem ausgewählten

Pfad endlich viele neue atomare Pfade die ihn enthalten. Gab es anfangs zu der Multiplizität μ^0 die n atomaren Pfade $m_1, \cdots, m_n$, so ergibt sich, mit dem leeren Pfad m^0 als Wurzelposition, ein Baum, dessen Knoten atomare Pfade sind und dessen Baumordnung die Enthaltenseinsrelation ($\subseteq$) ist. Bei jedem Komplettierungsschritt ergeben sich nur endlich viele neue atomare Pfade. Der Baum ist also endlich verzweigt. Da das Verfahren aber nicht terminiert, hat der Baum unendlich viele Knoten. Wegen Königs Lemma muß es also einen unendlich langen Faden aus atomaren Pfaden $m^0, m^1, m^2, \cdots$ in diesem Baum geben. Jetzt definiere:

$$\begin{aligned}
m &:= \bigcup_{i=0}^{\infty} m^i \\
\sigma_Q &:= \bigcup_{i=0}^{\infty} \sigma_Q^i \\
\sigma_M &:= \bigcup_{i=0}^{\infty} \sigma_M^i \\
\text{P-Formeln}(m) &:= \bigcup_{i=0}^{\infty} \text{P-Formeln}(m^i) \\
P_0 &:= \bigcup_{i=0}^{\infty} \mathcal{P}_\sigma(\mu, m_i) \\
C_0 &:= \bigcup_{i=0}^{\infty} \mathcal{C}_\sigma(\mu, m_i) \\
\leq_0 &:= \preceq
\end{aligned}$$

Es wird gezeigt werden, daß P-Formeln(m) eine J4-Hintikka-Menge über P_0 und C_0 ist:

Ist

$$p : B \in \text{P-Formeln}(m),$$

so gibt es ein $i \geq 0$ mit

$$p : B \in \text{P-Formeln}(m^i).$$

Das zugehörige σ^i ist nach Konstruktion J4-zulässig. Daraus folgt wegen Punkt 2 aus Definition 6.4.9, daß zu jeder Konstanten c in B ein $q \leq_0 p$ existiert, so daß

$$c \in \mathcal{C}_{\sigma^i}(\mu, m^i, q) \subseteq \mathcal{C}_\sigma(\mu, m, q)$$

ist. Also ist die erste Bedingung für Hintikka-Mengen erfüllt.

Wie bei Lemma 7.2.9 wird nur noch der ν-Fall gezeigt:

Sei $p : \nu \in$ P-Formeln(m) eine $\forall$-Formel. Angenommen, es gibt ein q mit $p \leq_0 q$, ein r, $r \leq_0 q$ und ein $c \in \mathcal{C}_\sigma(\mu, m, r)$, so daß

$$q : \nu_0(c) \notin \text{P-Formeln}(m)$$

ist. Auch nach der Einführung des c muß die ν-Position zwangsläufig von der systematischen Komplettierungsroutine bearbeitet worden sein. Deshalb muß (wie bei Lemma 7.2.9)

$$q : \nu_0(c) \in \text{P-Formeln}(m)$$

gelten. Es ergibt sich aus der Annahme also ein Widerspruch. Analog verfährt man, wenn ν keine $\forall$-Formel ist. Also folgt, daß P-Formeln(m) eine J4-Hintikka-Menge ist. Wegen Satz 7.2.4 gibt es also ein Kripke-Modell, in dem A nicht gilt. Das ergibt einen Widerspruch zur Voraussetzung, so daß dieser zweite Fall nicht eintreten kann.

Es muß also der Fall 1.a) eintreten. Damit endet der Beweis der Vollständigkeit. $\square$

Literaturverzeichnis

[Ada76] *J.B. Adams.* “A probability model of medical reasoning and the MYCIN model” in: Mathematical Biosciences 32, S. 177–186.

[AC84] *J.L. Alty* und *M.J. Coombs.* Expert Systems, NCC Publications, The National Computing Centre Ltd., Manchester, England, 1984.

[And81] *P.B. Andrews.* “Theorem Proving via general matings” in: Journal of the Association for Computing Machinery 28, 1981, S. 193–214.

[Ape87] *Apex.* Leveraging Financial Services Using Expert Systems, Informationsmaterial von Applied Expert Systems, Inc., Cambridge, Mass. 1986.

[Bac86] *J. Bachem.* “EVA — Ein Expertensystem zur Vermögensanlageberatung” in: Der GMD-Spiegel 2/86, Gesellschaft für Mathematik und Datenverarbeitung, Sankt Augustin.

[Bal85] *H. Baltes.* GABI — Ein wissensbasiertes Geldanlageberatungsprogramm, Memo Nr. 3, Sonderforschungsbereich 314, Universität des Saarlandes, Saarbrücken, November 1985.

[BF82] *A. Barr* und *E. Feigenbaum.* The Handbook of Artificial Intelligence, Vol. 1, William Kaufmann, Los Altos, Calif. 1982.

[Bib81] *W. Bibel.* “On matrices with connections” in: Journal of the Association for Computing Machinery 28, 1981, S. 633–645.

[Bib82a] *W. Bibel.* Automated Theorem Proving. Vieweg, Braunschweig 1982.

[Bib82b] *W. Bibel.* "A comparative study of several proof procedures" in: in: Artificial Intelligence 18, North-Holland, Amsterdam 1982, S. 269–293.

[BS83] *D. Bobrow* und *M. Stefik.* The LOOPS Manual, Technical Report KB-VLSI-81-13, XEROX Palo Alto Research Center 1983.

[BW77] *D. Bobrow* und *T. Winograd.* "An Overview of KRL" in: Cognitive Science 1 (1), 1977, S. 3–46, wiederveröffentlicht in: [BL85], S. 263–285.

[Bra79] *R.J. Brachman.* "On the Epistemological Status of Semantic Networks" in: [Fin79], S. 3–50, wiederveröffentlicht in: [BL85], S. 191–215.

[BL85] *R.J. Brachman, H.J. Levesque (Hrsg).* Readings in Knowledge Representation, Morgan Kaufman, Inc., Los Altos, Calif., 1985.

[CG88] *M.R.B. Clarke, D.M. Gabbay.* "Intuitionistic Bases for Non-Monotonic Reasoning" in: [SMDP88], S. 163–174.

[CM81] *W.F. Clocksin* und *C.S. Mellish.* Programming in Prolog, Springer, Berlin 1981.

[Dav80] *M. Davis.* "The Mathematics of Non-Monotonic Reasoning" in: Artificial Intelligence 13, North-Holland, Amsterdam 1980, S. 73–80.

[DFW81] *M. Deering, J. Faletti* und *R. Wilensky.* PEARL: A Package for Efficient Access to Representations in LISP, Proceedings IJCAI-81, Vancouver 1981, S. 930–932.

[Dil81] *J. Diller.* Intuitionismus I. Vorlesungsmitschrift Sommersemester 1981.

[Doy79] *J. Doyle.* "A Truth Maintenance System" in: Artificial Intelligence 12, North-Holland, Amsterdam 1979, S. 231–272.

[Dum77] *M. Dummett.* Elements of Intuitionism. Clarendon Press, Oxford 1977.

[Fin79] *N.V. Findler.* Associative Networks: Representation and Use of Knowledge by Computers, Academic Press, New York 1979.

[Fit69] *M. Fitting.* Intuitionistic Logic Model Theory and Forcing. North-Holland Publishing Company, Amsterdam, London 1969.

[Fit83] *M. Fitting.* Proof Methods for Modal and Intuitionistic Logics, D. Reidel Publishing Company, Dordrecht 1983.

[Gab82] *D.M. Gabbay.* "Intuitionistic basis for non-monotonic logic" *D.W. Loveland (Hrsg.)*, Proc. 6th Conf. on Automated Deduction, und in: Springer Lecture Notes in Computer Science, No. 138, S. 260–273.

[Gab85] *D.M. Gabbay.* "Theoretical foundations for non-monotonic reasoning in expert systems" in: *K. Apt (ed.)* Logics and Models of Concurrent Systems, Springer-Verlag, Berlin 1985, S. 439–457.

[Gal86] *J.H. Gallier.* Logic for Computer Science, Harper & Row, New York 1986.

[Gin87] *M.L. Ginsberg (Hrsg.).* Readings in Nonmonotonic Reasoning, Morgan Kaufman, Inc., Los Altos, Calif. 1987.

[GMD85] *Forschungsgruppe Expertensysteme.* BABYLON Benutzerhandbuch V 0.0/1 und BABYLON Referenzhandbuch V 0.0/1, Gesellschaft für Mathematik und Datenverarbeitung, Sankt Augustin 1985.

[GR83] *A. Goldberg* und *D. Robinson.* Smalltalk-80, The Language and and its Implementation, Addison-Wesley 1983.

[Hau81] *J. Haugeland (Hrsg.).* Mind Design, The MIT Press, Cambridge, Mass. 1981.

[Hay79] *P. Hayes.* "The Logic of Frames" in: *D. Metzing (Hrsg.)* Frame Conception and Text Understanding, S. 46–61, de Gruyter, Berlin 1979, wiederveröffentlicht in: [BL85], S. 287–295.

[Heu87] *S. Heuer.* Implementation einer Frame-basierten Wissensrepräsentation für Expertensysteme, Diplomarbeit, Institut für Numerische und instrumentelle Mathematik der Westfälischen Wilhelms-Universität, Münster 1987.

[HKC88] *S. Heuer*, *U. Koch* und *C.W. Cryer.* "INVEST: An Expert System for Financial Investments" in: IEEE Expert, Volume 3, Number 2, 1988, Los Alamitos, Calif., S. 60–68.

[HC68] *G.E. Hughes, M.J. Cresswell.* An Introduction to Modal Logic, Methuen and Co Ltd, London, 1968.

[HH88] *B. Humpert* und *P. Holley.* "Expert systems in finance planning" in: Expert Systems, Vol. 5, No. 2, 1988, Learned Information (Europe) Ltd., Oxford, S. 78–101.

[Isr80] *D.J. Israel.* "What's wrong with non-monotonic logic?" in: Proceedings of the First Annual National Conference on Artificial Intelligence, Stanford, Calif., S. 99–101, 1980, wiederveröffentlicht in: [Gin87], S. 53–55.

[Jac86] *P. Jackson.* Introduction to Expert Systems, Addison-Wesley 1986.

[Joh83] *P.E. Johnson.* "What kind of expert should a system be?" in: The Journal of Medicine and Philosophy, Vol. 8, S. 77–97, 1983.

[Koc88] *U. Koch.* Intuitionistische Logik und nicht-monotones Schließen, Preprints "Angewandte Mathematik und Informatik", 3/88 I, Institut für Numerische und instrumentelle Mathematik der Westfälischen Wilhelms-Universität, Münster 1988.

[Koc89] *U. Koch.* "INVEST: Ein Expertensystem zur Wertpapieranlageberatung" in: *J. Kuntner (Hrsg.).* Expertensysteme im Bankbereich, Schriftenreihe der Österreichischen Gesellschaft für Artificial Intelligence (ÖGAI), Band 2, Wien 1989.

[KH86] *U. Koch* und *S. Heuer.* Die Expertensystem-Shell Donald, Preprints "Angewandte Mathematik und Informatik", 8/86 I, Institut für Numerische und instrumentelle Mathematik der Westfälischen Wilhelms-Universität, Münster 1986.

[KH87] *U. Koch* und *S. Heuer*. "Die Expertensystem-Shell Donald" in: *A.B. Cremers, K.-H. Becks, (Hrsg.)*. Proc. Anwender-Forum Expertensysteme, Universität Dortmund, Lehrstuhl für Informationssysteme, 1987, S. 115–125.

[McD87] *D. McDermott*. "A Critique of Pure Reason" in: Computational Intelligence, 1987.

[McD80] *D. McDermott* und *J. Doyle*. "Non-Monotonic Logic I" in: Artificial Intelligence 13, North-Holland, Amsterdam 1980, S. 41–72.

[Min75] *M. Minsky*. "A Framework for Representing Knowledge" in: [Hau81], S. 95–128, wiederveröffentlicht in [BL85], S. 245–262, ursprünglich: Memo 306, Artificial Intelligence Laboratory, MIT, Cambridge, Mass., 1975.

[MPM82] *A.M. Miller, H.E. Pople* und *J.D. Myers*. "INTERNIST-I, an experimental computer-based diagnostic consultant for general internal medicine" in: New England Journal of Medicine, 307, 1982, S. 468–476.

[Ode87] *L.L. Odette*. Persönlicher Briefwechsel, Applied Expert Systems, Inc., Cambridge, Mass. 1987.

[Pop73] *H.E. Pople*. "On the mechanization of abductive logic" in: Proceedings of the 3rd International Joint Conference on Artificial Intelligence, 1973, S. 147–152.

[Pra60] *D. Prawitz*. "An improved proof procedure" in: *K. Marc-Wogau (Hrsg.)* Theoria XXVI, A Swedish Journal of Philosophy and Psychology, Lund 1960, S. 102–139.

[Pra70] *D. Prawitz*. "A proof procedure with matrix reduction" in: *M. Laudet (Hrsg.)* Symposium in Automatic Demonstration, Versailles 1968, Springer Lecture Notes in Mathematics, Berlin, New York, Heidelberg 1970, S. 207–214.

[Qui66] *M.R. Quillian*. "Word Concepts: A Theory and Simulation of Some Basic Semantic Capabilities" in: *J.G. Miller (Hrsg.)* Behavioral Science 12, 1967, S. 410–430, wiederveröffentlicht in: [BL85], S. 97–118.

[Rei87] *H. Reichgelt.* "A Review of McDermott's 'Critique of Pure Reason'" in: AICOM Vol. 0, No. 1, August 1987, S. 39–42.

[Rei80] *R. Reiter.* "A Logic for Default Reasoning" in: Artificial Intelligence 13, North-Holland, Amsterdam 1980, S. 81–132.

[Ric83] *E. Rich.* Artificial Intelligence, McGraw-Hill, Inc., 1983.

[RG77a] *R.B. Roberts* und *I.P. Goldstein.* The FRL Primer, Memo 408, Artificial Intelligence Laboratory, MIT, Cambridge, Mass., 1977.

[RG77b] *R.B. Roberts* und *I.P. Goldstein.* The FRL Manual, Memo 409, Artificial Intelligence Laboratory, MIT, Cambridge, Mass., 1977.

[Rob65] *J.A. Robinson.* "A Machine-Oriented Logic Based on the Resolution Principle" in: Journal of the ACM, Vol 12, 1965, S. 23–41.

[SR74] *R.C. Schank* und *C.J. Rieger.* "Inference and the Computer Understanding of Natural Language" in: Artificial Intelligence 5, 1974, S. 373–412, wiederveröffentlicht in: [BL85], S. 119–139.

[Sho76] *E.H. Shortliffe.* Computer-Based Medical Consultations: MYCIN. Elsevier, New York 1976.

[Sch68] *K. Schütte.* Vollständige Systeme modaler und intuitionistischer Logik. Springer-Verlag, Berlin 1968.

[SMDP88] *P. Smets, E.H. Mamdani, D. Dubois, H. Prade.* Non-Standard Logics for Automated Reasoning. Academic Press, London, San Diego 1988.

[Smu68] *R.M. Smullyan.* First-Order Logic, Springer-Verlag, Berlin 1968.

[SG87] *J.L. Stansfield* und *N.R. Greenfeld.* "PlanPower: A Comprehensive Financial Planner" in: IEEE Expert, Volume 2, Number 3, 1987, Los Alamitos, Calif., S. 51–60.

[Ste84] *G.L. Steele.* Common LISP: The Language, Digital Press 1984.

[Tan87] *S.L. Tanimoto.* The Elements of Artificial Intelligence, Computer Science Press, Rockville, Maryland 1987.

[Tou86] *D.S. Touretzky.* The Mathematics of Inheritance Systems, Pitman, London und Morgan Kaufmann, Los Altos, Calif. 1986.

[Tur84] *R. Turner.* Logics for Artificial Intelligence, Ellis Horwood, Chichester 1984.

[Wal87] *L.A. Wallen.* Automated Proof Search in Non-Classical Logics: Efficient Matrix Proof Methods for Modal and Intuitionistic Logics. University of Edinburgh, 1987.

[Wat86] *D.A. Waterman.* A Guide to Expert Systems, Addison-Wesley, Reading, Mass. 1986.

[Wil86] *R. Wilensky.* Common LISPcraft, W.W. Norton & Company, New York, London 1986.

[Win75] *T. Winograd.* "Frame Representation and the Declarative/Procedural Controversy" in: *D.G. Bobrow, A.M. Collins (Hrsg.)*, Representation and Understanding: Studies in Cognitive Science, S. 185–210, wiederveröffentlicht in: [BL85], S. 357–370.

[Win84] *P.H. Winston.* Artificial Intelligence, Addison-Wesley, Reading, Mass. 1984.

[WH84] *P.H. Winston* und *B.K. Horn.* LISP, Addison-Wesley, Reading, Mass. 1984.