4. Wissensbasierte Anwendungssysteme

Künstliche Intelligenz: Eine Systematisierung

A) Systems that **think** like **humans**	B) Systems that **think rationally**
"The exciting new effort to make computers think ... *machines with minds*, in the full and literal sense." (Haugeland, 1985) "[The automation of] activities, that we associate with human thinking, activities such as decision-making, problem solving, learning ..." (Bellman 1978)	"The study of mental faculties through the use of computational models." (Charniak and McDermott 1985) "The study of the computations that make it possible to perceive, reason, and act." (Winston 1992)
C) Systems that act like humans	**D) Systems that act rationally**
"The art of creating machines that perform functions that require intelligence when performed by people." (Kurzweil 1990) "The study of how to make computers do things at which, at the moment, people are better." (Rich and Knight 1991)	"Computational Intelligence is the study of the design of intelligent agents." (Poole et al. 1998) "AI ... is concerned with intelligent behavior in artifacts." (Nilsson 1998)

Quelle: Russell / Norvig 2003

A) Menschliches Denken: Der Ansatz der kognitiven Modellierung

B) Vernünftiges (rationales) Denken: „Gesetze des Denkens" (Logik)

C) Menschliches Handeln: Der Ansatz des Turing-Tests

D) Vernünftiges (rationales) Handeln: Der Ansatz des rationalen Agenten

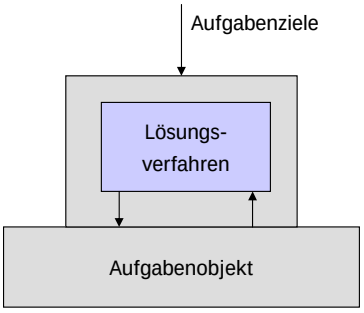
Wissensbasierte Aufgabenmodelle

Klassifikation betrieblicher Aufgaben

Modelltyp der Aufgabe:
- Analytische Modelle
- Wissensbasierte Modelle
- Konnektionistische Modelle
- Natürlichsprachige Beschreibungen

Typ des Lösungsverfahrens:
- Exakte Verfahren
- Approximierende Verfahren
- Heuristische Verfahren
- Lernende Verfahren

Beschreibungsmittel für Aufgaben:
- Imperative Sprachen
- Deklarative Sprachen



Klassifikation betrieblicher Aufgaben

Modelltyp der Aufgabe

Modellierung eines Ausschnitts der Diskurswelt in Form einer Aufgabe:

a) Analytische Modelle:

•Formulierung der Aufgabe als mathematisches Modell auf der Grundlage einer zugehörigen Theorie.

•Prüfung des Modells auf Vollständigkeit und Widerspruchsfreiheit anhand der dem mathematischen Modell zugrundeliegenden Theorie.

•Modellierung bezieht sich in der Regel auf die Außensicht der Aufgabe, d.h. auf Aufgabenobjekt und Aufgabenziele. Die Verwendbarkeit des Modells setzt somit die Existenz eines Lösungsverfahrens voraus.

•Beispiele: Modelle des Operations Research.

b) Wissensbasierte Modelle:

•Formulierung der Aufgabe auf der Grundlage von Erfahrungswissen, das auf Beobachtungen, Erfahrungen oder logischen Ableitungen beruht; ohne zugrundeliegende, geschlossene Theorie.

•Wissensquellen sind Diskursweltexperten.

•Prüfung des Modells auf Vollständigkeit und Widerspruchsfreiheit ist nur eingeschränkt möglich.

•Formale Modellierung ist auch unter abgeschwächten Voraussetzungen möglich, Lösungsverfahren sind im Allgemeinen verfügbar.

•Beispiele: Expertensysteme, unter Verwendung von Methodenwissen der Künstlichen Intelligenz.

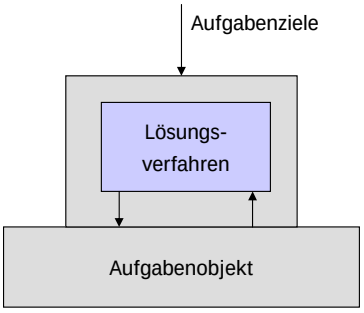
Wissensbasierte Aufgabenmodelle

Klassifikation betrieblicher Aufgaben

Modelltyp der Aufgabe:
- Analytische Modelle
- Wissensbasierte Modelle
- Konnektionistische Modelle
- Natürlichsprachige Beschreibungen

Typ des Lösungsverfahrens:
- Exakte Verfahren
- Approximierende Verfahren
- Heuristische Verfahren
- Lernende Verfahren

Beschreibungsmittel für Aufgaben:
- Imperative Sprachen
- Deklarative Sprachen



c) Konnektionistische Modelle:

•Während (a) und (b) Struktur und Verhalten einer Aufgabe modellieren, beschränken sich konnektionistische Modelle auf die Modellierung des Verhaltens.

•Modellierung von Aufgaben als Black Box ohne interne Zustände.

•Anwendung, wenn die Struktur der Diskurswelt nicht bekannt ist, oder Hypothesen über die Struktur der Diskurswelt nicht validierbar sind.

•Beispiel: Künstliche neuronale Netze als Teilgebiet der Kognitionswissenschaften.

•Anwendungsbeispiele: Aktienprognose, Motorenprüfung

d) Natürlichsprachige Beschreibungen:

•Anzuwenden, wenn nur Teilaspekte der Diskurswelt bekannt sind, oder die Komplexität der Diskurswelt eine formale Modellierung verhindert.

•Prüfung des Modells auf Vollständigkeit und Widerspruchsfreiheit sind nur sehr beschränkt möglich.

•Aufgaben in natürlichsprachiger Beschreibung sind nicht automatisierbar.

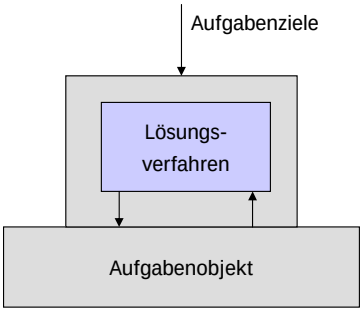
Wissensbasierte Aufgabenmodelle

Klassifikation betrieblicher Aufgaben

Modelltyp der Aufgabe:
- Analytische Modelle
- Wissensbasierte Modelle
- Konnektionistische Modelle
- Natürlichsprachige Beschreibungen

Typ des Lösungsverfahrens:
- Exakte Verfahren
- Approximierende Verfahren
- Heuristische Verfahren
- Lernende Verfahren

Beschreibungsmittel für Aufgaben:
- Imperative Sprachen
- Deklarative Sprachen



Typ des Lösungsverfahrens

Lösungsverfahren realisieren die Sach- und Formalziele einer Aufgabe. Abhängig vom Zielerreichungsgrad werden unterschieden:

a) Exakte Verfahren

ermitteln in endlicher Zeit im Lösungsraum der Aufgabe exakte Lösungen, die ein Optimum bezüglich der Sach- und Formalziele darstellen.

Beispiele:

•Bestimmung eines Maximums oder Minimums durch Differentialrechnung,

•Simplexalgorithmus der linearen Programmierung

•Enumerationsverfahren (Branch & Bound)

•Dekompositionsverfahren zur Zerlegung von Lösungsräumen

b) Approximierende Verfahren

nähern sich einer exakten Lösung mit zunehmender Schrittzahl an, der Abstand zur exakten Lösung kann bei jedem Schritt abgeschätzt werden.

Beispiel: Interpolations- und Extrapolationsverfahren.

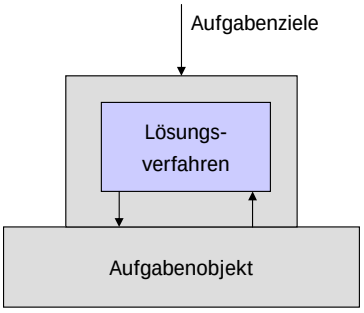
Wissensbasierte Aufgabenmodelle

Klassifikation betrieblicher Aufgaben

Modelltyp der Aufgabe:
- Analytische Modelle
- Wissensbasierte Modelle
- Konnektionistische Modelle
- Natürlichsprachige Beschreibungen

Typ des Lösungsverfahrens:
- Exakte Verfahren
- Approximierende Verfahren
- Heuristische Verfahren
- Lernende Verfahren

Beschreibungsmittel für Aufgaben:
- Imperative Sprachen
- Deklarative Sprachen



c) Heuristische Verfahren

zielen auf eine schrittweise Annäherung an eine exakte Lösung, der Abstand zur exakten Lösung kann in der Regel nicht abgeschätzt werden. Sie dienen zum Auffinden „guter“ Lösungen.

Beispiele:

•Heuristische Tourenplanungsverfahren

•Layout-Verfahren für Graphen

d) Lernende Verfahren

Voraussetzung ist die „Erkundung“ des Aufgabenobjekts und der Aufgabenziele durch das lernende Verfahren, keine Vorkenntnisse über Aufgabenmerkmale erforderlich. Einsatz, wenn (a), (b) und (c) nicht anwendbar.

Beispiel: Lösungsverfahren für konnektionistische Modelle (Künstliche Neuronale Netze)

Analytische und wissensbasierte Modelle verwenden alle vier Lösungsverfahren, konnektionistische Modelle verwenden ausschließlich lernende Verfahren.

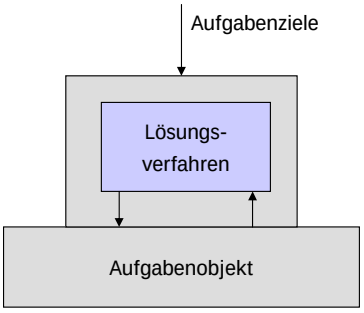
Wissensbasierte Aufgabenmodelle

Klassifikation betrieblicher Aufgaben

Modelltyp der Aufgabe:
- Analytische Modelle
- Wissensbasierte Modelle
- Konnektionistische Modelle
- Natürlichsprachige Beschreibungen

Typ des Lösungsverfahrens:
- Exakte Verfahren
- Approximierende Verfahren
- Heuristische Verfahren
- Lernende Verfahren

Beschreibungsmittel für Aufgaben:
- Imperative Sprachen
- Deklarative Sprachen



Beschreibungsmittel für Aufgaben

Die Beschreibung der Außensicht und der Innensicht einer Aufgabe erfolgt abhängig vom Modelltyp und vom Typ des Lösungsverfahrens in formalisierter oder natürlichsprachiger Form.

Varianten für die formalisierte Beschreibung sind:

a) Imperative Sprachen

•Die Beschreibung des Lösungsverfahrens besteht aus Anweisungen sowie aus Ablaufstrukturen für die sequentielle und parallele Ausführung von Anweisungen.

•Jede Anweisung besteht aus Operator(en) und Operand(en).

•Die Anweisungen werden in übersetzter Form ausgeführt (Hardware-Prozessor) oder sie werden interpretiert (Software-Interpreter).

•Die Aufgabenaußensicht wird getrennt, z.B. natürlichsprachig dokumentiert. Problem: Aufgabenänderungen!

•Beispiele: Pascal, C, Cobol usw.

b) Deklarative Sprachen

•Die Aufgabenaußensicht wird mit Hilfe eines Systems von Relationen beschrieben.

•Diese Relationen beschreiben gleichzeitig den Lösungsraum der Aufgabe.

•Lösungen werden in dem Lösungsraum durch einheitliche Lösungsverfahren ermittelt (Interpreter, Inferenzmaschine), mit auf die Beschreibungsform abgestimmt sind.

•Die Aktualisierung von Aufgaben beschränkt sich auf die Außensicht, das Lösungsverfahren bleibt unverändert.

•Beispiele: SQL, Prolog

Wissensbasierte Systeme und Expertensysteme

Begriffsbestimmung

Wissensbasiertes System:
- System, welches auf wissensbasierten (Aufgaben-) Modellen beruht.
- Architekturprinzip, bei dem Wissen und Wissensverarbeitung getrennt sind.

Expertensystem (XPS):
- System, welches es gestattet, Expertenwissen „geeignet“ darzustellen und
- mithilfe einer Problemlösungskomponente Probleme einer bestimmten Domäne zu lösen

XPS sind in der Regel wissensbasierte Systeme, aber auch als konventionelle Programme denkbar

Ein wissensbasiertes System muss nicht notwendig ein XPS sein.

Anforderungen an XPS

Ein XPS soll bestimmte, üblicherweise menschlichen Experten vorbehaltene Merkmale aufweisen:

•ein Problem „verstehen“ und lösen,

•die Lösung und den Lösungsweg erklären,

•Wissen erwerben und strukturieren,

•seine Kompetenz einschätzen und

•Randbereiche „überblicken“.

Konventionelle Programme vs. Wissensbasierte Systeme

Konventionelles Programm P	Expertensystem XPS (als wissens-basiertes System)
Algorithmus	Bereichsunabhängige Problemlösungskomponente (Inferenzmaschine) Bereichsspezifisches Expertenwissen
Daten	Fallspezifisches Wissen

Paradigmenwechsel von

•Programm = Algorithmus + Daten

•Wissensbasiertes System = Wissensbasis + Inferenzmaschine

a) Konventionelles Programm P

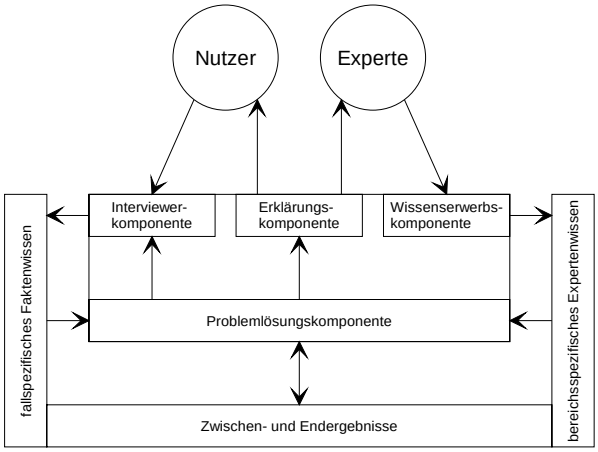
- •Modelltyp: analytisches Modell
- •Lösungsverfahren: exakt, approximierend, heuristisch, lernend
- •Beschreibungsform: imperativ
- •Modellwissen und Lösungsverfahren sind im Algorithmus kombiniert („vermischt“). Eignung für **wohlstrukturierte Probleme**.

b) Expertensystem XPS (als wissensbasiertes System)

- •Modelltyp: wissensbasiertes Modell
- •Lösungsverfahren: exakt, approximierend, heuristisch, lernend
- •Beschreibungsform: deklarativ
- •Modellwissen und Lösungsverfahren sind getrennt. Eignung für **schlechtstrukturierte Probleme.**

Funktionsweise von Expertensystemen

Funktionsmodell (Architektur) nach Puppe:



Hauptkomponenten

• Bereichsunabhängiges **Steuersystem** (Shell)

• Bereichsabhängige und anwendungsspezifische **Wissensbasis**

Wissensbasis

• bereichsspezifisches **Expertenwissen** (ändert sich i.d.R. während einer Konsultation nicht)

• fallspezifisches **Faktenwissen** (wird vom Nutzer des XPS während einer Konsultation eingegeben)

• **Zwischen- und Endergebnisse** (vom XPS während einer Konsultation hergeleitet)

Steuersystem (Shell)

• **Wissenserwerbskomponente** (Eingabe und Abänderung des Expertenwissens; i.d.R. nicht während einer Konsultation)

• **Interviewerkomponente** führt Dialog mit dem Nutzer und/oder liest Problemdaten ein (Kopplung mit Messstationen)

• **Problemlösungskomponente** wendet Expertenwissen auf fallspezifisches Wissen (Problemdaten) an, um eine Problemlösung zu generieren (Inferenzmaschine). Dabei werden i.d.R. weitere Daten angefordert.

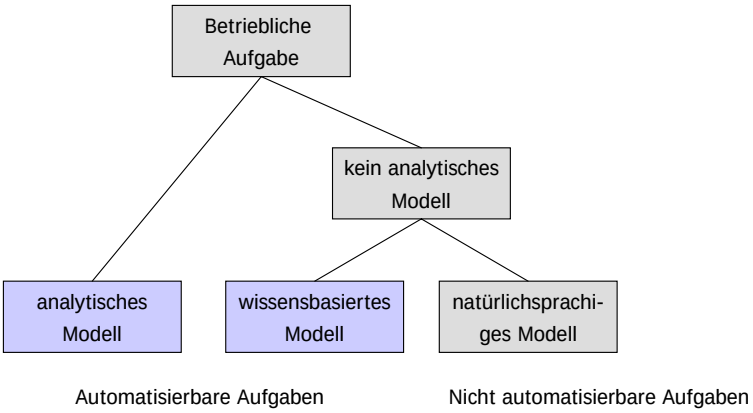
• **Erklärungskomponente** macht die Vorgehensweise des XPS bei der Lösungssuche nachvollziehbar

a) für Nutzer (Lösungsweg, Plausibilitätskontrolle der Lösung)

b) für Experten (Fehleranalyse, Test)

Die Erklärung muss dabei auf die Nutzermodelle von (a) und (b) abgestimmt sein.

Wissensbasierte Systeme im IS



Charakterisierung

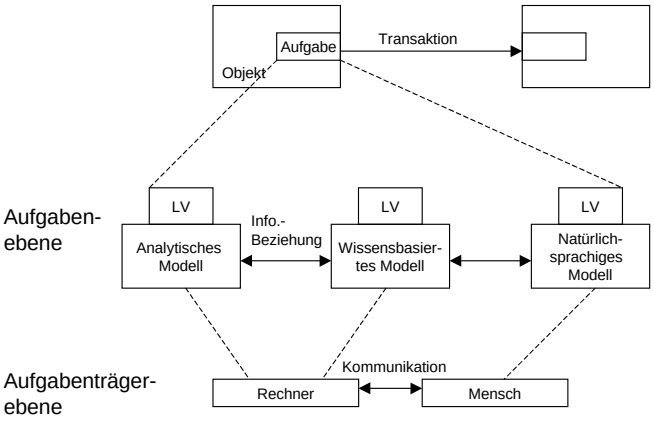
•Wissensbasierte Systeme stellen eine Alternative bei der Automatisierung betrieblicher Aufgaben dar.

•Durch wissensbasierte Systeme werden bisher nicht automatisierbare Aufgaben einer Automatisierung zugänglich.

•Durch die Einbeziehung wissensbasierter Systeme kann der Automatisierungsgrad eines betrieblichen Systems erhöht werden.

Wissensbasierte Systeme im IS

Kooperation zwischen unterschiedlich automatisierten Teilaufgaben einer betrieblichen Aufgabe



Eine betriebliche Aufgabe kann

- auf der Basis eines analytischen Modells automatisiert,
- auf der Basis eines wissensbasierten Modells automatisiert und/oder
- nicht automatisiert sein.

Teilaufgaben von unterschiedlichem Automatisierungsgrad und unterschiedlicher Automatisierungsform kooperieren untereinander. Die Kooperation der automatisierten Teilaufgaben kann auf dem Prinzip

- der engen Kopplung (Vereinigung der Aufgabenobjekte, shared memory) oder
- der losen Kopplung (Nachrichtenkopplung)

erfolgen.

Anwendungsbereiche und Nutzungsformen für XPS

Problemklasse	Input	Output
Klassifikation / Diagnostik	Befunde, Messwerte	Wiedererkennung bekannter Muster
Design	Anforderungen	Objekte, die den Anforderungen genügen
Planung	Ausgangs- und Zielzustand (bzw. Zielfunktion	Sequenz von Aktionen, die Ausgangszustand in Zielzustand überführt
Simulation	Ausgangszustand	Entwicklung von Folgezuständen

Beispiele:

•**Klassifikation / Diagnostik:** Medizinische Diagnose (MYCIN), Fehlerdiagnose, Warnsysteme

•**Design:** Konfiguration von Rechenanlagen (XCON), Erstellen von Stundenplänen, CAD, VLSI-Design

•**Planung:** Produktionsplanung

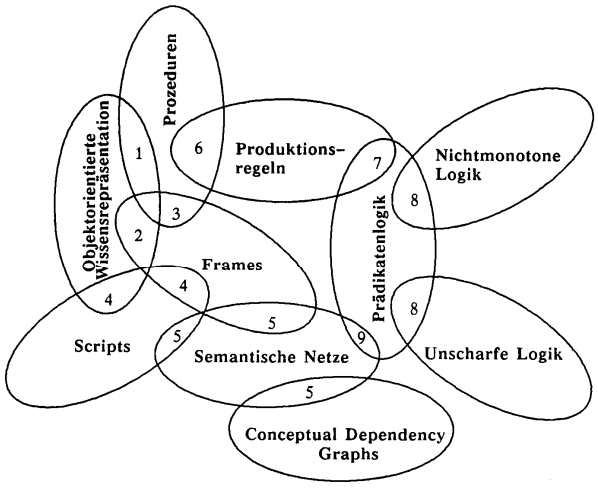
•**Simulation:** Wettervorhersage usw.

Grundformen der Wissensrepräsentation

Im Folgenden werden zwei Grundformen der Wissensrepräsentation unterschieden:

- **Relationsorientierte Wissensrepräsentation**
  - Logik
  - Produktionsregeln
- **Objektorientierte Wissensrepräsentation**
  - Semantische Netze
  - Frames

Überlappungen zwischen Formen der Wissensrepräsentation



Beschreibung der Überlappungen von Wissensrepräsentationsformen:

1. Die Objekte objektorientierter Sprachen enthalten Prozeduren (sog. Methoden).
2. Frames sind objektorientierte Datenstrukturen, jedoch einfacher und passiver als die Objekte objektorientierter Sprachen.
3. Frames enthalten Prozeduren.
4. Scripts sind für die Repräsentation von Aktivitäten weiterentwickelte Frames.
5. Frames, Scripts und Conceptual Dependency Graphs sind aus Semantischen Netzen entstanden und werden wie diese meist als Slot-and-Filler-Struktur implementiert.
6. Die Aktionsteile der Regeln sind Prozeduren.
7. Regeln sind prozedural interpretierte prädikatenlogische Implikationen.
8. Unscharfe und nichtmonotone Logik basieren auf der Prädikatenlogik.
9. Prädikatenlogische Formeln lassen sich als Semantische Netze veranschaulichen.

Logiken

Grundlage der relationsorientierten Wissensdarstellung sind die formalen Systeme

- Aussagenlogik und
- Prädikatenlogik.

Man unterscheidet weiter

- zweiwertige Logiken (wahr, falsch) und
- mehrwertige Logiken (wahr, falsch, unbekannt)

sowie

- monotone Logiken (hergeleitetes Wissen ist endgültig) und
- nichtmonotone Logiken (hergeleitetes Wissen ist vorläufig und kann durch zusätzliches Wissen wieder revidiert werden).

Logiken

Klassische Logiken:

Die klassischen Logiken sind **zweiwertig** und **monoton**

Merkmale	Aussagenlogik	Einstufige Prädikatenlogik (1. Ordnung)	Zweistufige Prädikatenlogik (2. Ordnung)
Operatoren	Junktoren	Junktoren Quantoren	Junktoren Quantoren
Variablen		Objektvariablen	Objektvariablen Prädikatvariablen

Aussagenlogik

Aussagen:

Aussagen (Formeln) sind sprachliche Konstrukte, denen ein Wahrheitswert zugeordnet werden kann. Beispiele:

- A: „es regnet“
- B: „die Straße ist nass“

Aussagen (Formel) werden im Folgenden mit A, B, C, ... bezeichnet.

Junktoren (Verknüpfungen):

$\neg$	Negation, logisches Nicht (einstelliger Junktor):	$\neg$ A
$\wedge$	Konjunktion, logisches Und (zweistelliger Junktor):	A $\wedge$ B
$\vee$	Disjunktion, logisches Oder (zweistelliger Junktor):	A $\vee$ B
$\rightarrow$	Implikation, logische Folgerung (zweistelliger Junktor):	A $\rightarrow$ B

Interpretation der Implikation:
- A impliziert B
- Aus A folgt B
- A ist hinreichend für B
- B ist notwendig für A

Aussagenlogik

Interpretation von Aussagen:

Wahre Aussage A:	Falsche Aussage A:
• „A ist richtig“ • „A gilt“ • „A ist erfüllt“	• „A ist nicht richtig“ • „A gilt nicht“ • „A ist nicht erfüllt“

Bei der Negation von Aussagen bitte aufpassen, wie folgendes Beispiel verdeutlicht. Ausspruch des Bürgermeisters im Zorn:

A: „Die Hälfte der Gemeinderäte sind Idioten“.

Am nächsten Tag widerruft er die Aussage:

Nicht A: „ Die Hälfte der Gemeinderäte sind keine Idioten“.

Richtig wäre:

Nicht A: „Es ist nicht richtig, dass die Hälfte der Gemeinderäte Idioten sind“.

Aussagenlogik

Wahrheitswerte verknüpfter Aussagen:

Die einzelnen Junktoren werden anhand von Wahrheitstafeln definiert:

A	$\neg$A
f	w
w	f

A	B	A $\vee$ B	A $\wedge$ B	A $\rightarrow$ B	$\neg$ A $\vee$ B
f	f	f	f	w	w
f	w	w	f	w	w
w	f	w	f	f	f
w	w	w	w	w	w

(Schluss-) Regeln des Aussagenkalküls

Eine Schlussregel ordnet einer Folge von Aussagen $A_1, A_2, ..., A_n$ eine andere Aussage B zu, welche die logische Konsequenz aus $A_1, A_2, ..., A_n$ darstellt. Die $A_1, A_2, ..., A_n$ heißen Prämissen, B heißt Konklusion oder Schlussfolgerung.

Folgende Schlussregeln gehören u.a. zur Aussagenlogik:

Modus ponens

$$\begin{array}{l} A \rightarrow B \\ A \end{array} \Big\} \; B$$

Modus tollens

$$\begin{array}{l} A \rightarrow B \\ \neg B \end{array} \Big\} \; \neg A$$

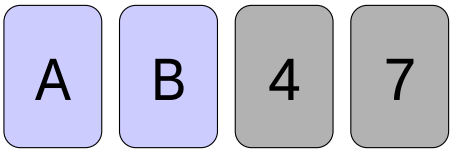
Syllogismus (Transitivität)

$$\begin{array}{l} A \rightarrow B \\ B \rightarrow C \end{array} \Big\} \; A \rightarrow C$$

(Schluss-) Regeln des Aussagenkalküls

Beispiel:

Gegeben seien 4 zweiseitig beschriftete Spielkarten. Es wird folgende Aussage A behauptet: „Wenn auf der Vorderseite ein Vokal steht, steht auf der Rückseite eine gerade Zahl“.



Welche der dargestellten Karten sind umzudrehen, um die Gültigkeit der Aussage zu überprüfen?

Hinweis:

•Aussage a: „Auf der Vorderseite steht ein Vokal“.

•Aussage b: „Auf der Rückseite steht eine gerade Zahl“.

Verwenden Sie die vorher eingeführten Schlussregeln!

Gesetze (Tautologien) des Aussagenkalküls

Konvertierung für Implikation und Alternative

$(A \rightarrow B) \Leftrightarrow (\neg A \vee B)$

de Morgan´sche Gesetze

$\neg(A \wedge B) \Leftrightarrow (\neg A \vee \neg B)$

$\neg(A \vee B) \Leftrightarrow (\neg A \wedge \neg B)$

Gesetz der doppelten Verneinung

$\neg\neg A \Leftrightarrow A$

Idempotenzgesetz

$A \vee A \Leftrightarrow A$

$A \wedge A \Leftrightarrow A$

Begriffe:

• *Tautologie:* Verknüpfte Aussage, die unabhängig von den Wahrheitswerten der Einzelaussagen stets wahr ist (mithilfe der Wahrheitstafeln oder der Schlussregeln beweisbar).

• *Kontradiktion:* Verknüpfte Aussage, die unabhängig von den Wahrheitswerten der Einzelaussagen stets falsch ist.

Gesetze (Tautologien) des Aussagenkalküls

tertium non datur

$A \vee \neg A$

Widerspruch

$\neg(A \wedge \neg A)$

Kontraposition

$A \rightarrow B \Leftrightarrow \neg B \rightarrow \neg A$

Kommutativgesetze

$A \wedge B \Leftrightarrow B \wedge A$

$A \vee B \Leftrightarrow B \vee A$

Assoziativgesetze

$A \wedge (B \wedge C) \Leftrightarrow (A \wedge B) \wedge C$

$A \vee (B \vee C) \Leftrightarrow (A \vee B) \vee C$

Distributivgesetze

$A \vee (B \wedge C) \Leftrightarrow (A \vee B) \wedge (A \vee C)$

$A \wedge (B \vee C) \Leftrightarrow (A \wedge B) \vee (A \wedge C)$

Gesetze und Schlussregeln bilden gemeinsam die Boole´sche Algebra.

(Schluss-) Regeln des Aussagenkalküls

Beispiel:

Studienvertiefung Wirtschaftsinformatik für Studierende der Betriebswirtschaftslehre an der Universität Bamberg (*Hinweis:* „EbIS-1“ bedeutet „Wahl des Teilgebiets EbIS-1“):

Definition der Fächer:

MobIS $\wedge$ EBM $\wedge$ IM $\rightarrow$ aWI
Ebis-1 $\wedge$ EbIS-2 $\wedge$ EbIS-3 $\rightarrow$ Seda
IS-DL-1 $\wedge$ IS-DL-2 $\wedge$ IS-DL-3 $\rightarrow$ IS-DL
WII-1 $\wedge$ WII-2 $\wedge$ WII-3 $\rightarrow$ IAWS

Definition der Fächerkombinationen:

aWI $\wedge$ (Seda $\vee$ IS-DL $\vee$ IAWS) $\wedge$ WI-DA
$\rightarrow$ „Studienvertiefung WI“
aWI $\wedge$ ((Seda $\wedge$ IS-DL) $\vee$ (Seda $\wedge$ IAWS) $\vee$ (IS-DL $\wedge$ IAWS))
$\rightarrow$ „Studienvertiefung WI“
aWI $\wedge$ ((Seda $\wedge$ IS-DL) $\vee$ (Seda $\wedge$ IAWS) $\vee$ (IS-DL $\wedge$ IAWS)) $\wedge$ WI-DA
$\rightarrow$ „Studienvertiefung WI“

Prädikatenlogik

Symbole der Prädikatenlogik:

Freie Variablen: x, y, z, ...

Quantoren:

- Der Allquantor $\forall x$: P(x) liefert den Wert wahr, wenn das Prädikat P(x) für alle x wahr ist.
- Der Existenzquantor $\exists x$: P(x) liefert den Wert wahr, wenn das Prädikat P(x) für mindestens ein x wahr ist.

Regeln und Gesetze des Prädikatenkalküls (Auswahl):

- **Distributivgesetze**

  $\forall x$: [P(x) $\wedge$ Q(x)] $\Leftrightarrow$ [$\forall x$: P(x) $\wedge$ $\forall x$: Q(x)]

  $\exists x$: [P(x) $\vee$ Q(x)] $\Leftrightarrow$ [$\exists x$: P(x) $\vee$ $\exists x$: Q(x)]

- **de Morgan´sche Gesetze**

  $\neg$ $\exists x$: P(x) $\Leftrightarrow$ $\forall x$: [$\neg$P(x)]

  $\neg$ $\forall x$: P(x) $\Leftrightarrow$ $\exists x$: [$\neg$P(x)]

Die Programmiersprache Prolog

Prolog (PROgramming in LOGic) ist eine deklarative Programmiersprache. Sie beruht auf dem Prädikatenkalkül 1. Ordnung. Die in Prolog zur Lösungssuche verwendete Inferenzmaschine erwartet Formeln in einer bestimmten Normalform:

- **Konjunktive Normalform (Klausel)**

  $((L_1 \vee ... \vee L_n) \wedge ... \wedge (K_1 \vee ... \vee K_n))$

- **Hornklausel: Konjunktive Normalform mit *einem* positiven Literal**

  $(\neg L_1 \vee \neg L_2 \vee ... \vee \neg L_n \vee L_{n+1}) \wedge ... \wedge (...)$

- **Hornklausel-Schreibweise**

  $(\neg L_1, \neg L_2,..., \neg L_n, L_{n+1}), ... (...)$

- **Negation ausgeklammert**

  $(\neg(L_1 \wedge L_2 \wedge ... \wedge L_n) \vee L_{n+1}), ... (...)$

- **Implikation**

  $(L_1 \wedge L_2 \wedge ... \wedge L_n \rightarrow L_{n+1}), ... (...)$

- **Prolog-Notation**

  L_{n+1} :- L_1, L_2, ... L_n

Semantische Netze

Semantische Netze (SN) stellen die allgemeinste und älteste Form der Wissensrepräsentation dar:

- Ein Semantisches Netz umfasst eine Menge von Knoten (nodes).
- Knoten sind durch Bögen (arcs) oder Glieder (links) verbunden.
- Knoten und Glieder sind mit Namen bezeichnet.

Konventionen zur Bildung semantischer Netze (1):

- Namenseindeutigkeit von Knoten: In einem SN werden zwei Knoten mit unterschiedlichen Namen als unterschiedlich angesehen.
- Einmaligkeit von Knoten: In einem SN treten keine zwei Knoten mit gleichem Namen auf.
- Implizite Konjunktion: Alle in einem SN repräsentierten Aussagen sind konjunktiv verknüpft.

Semantische Netze

Konventionen zur Bildung semantischer Netze (2)

- Knoten repräsentieren Konzepte (bzw. Objekte) sowie Eigenschaften von Konzepten.
- Glieder verbinden Konzepte untereinander sowie Konzepte mit Eigenschaften.

Typische Relationen von Gliedern sind:

- Ist-ein: Relation zwischen Klasse und Instanz sowie zwischen Klassen unterschiedlichen Abstraktionsgrades.
- Hat-ein: Relation zwischen Objekten und Teilobjekten.

Darüber hinaus können beliebige Relationen verwendet werden, wie z.B. die definierende Relation trägt.

Semantische Netze

Beispiel

„Von den offensichtlichen Tatsachen einmal abgesehen, dass Mr. Wilson irgendwann manuelle Arbeit geleistet hat, dass er in China gewesen ist und dass er kürzlich in größerem Umfang Schreibarbeiten erledigt hat, kann ich keinen anderen Schluss ziehen …“

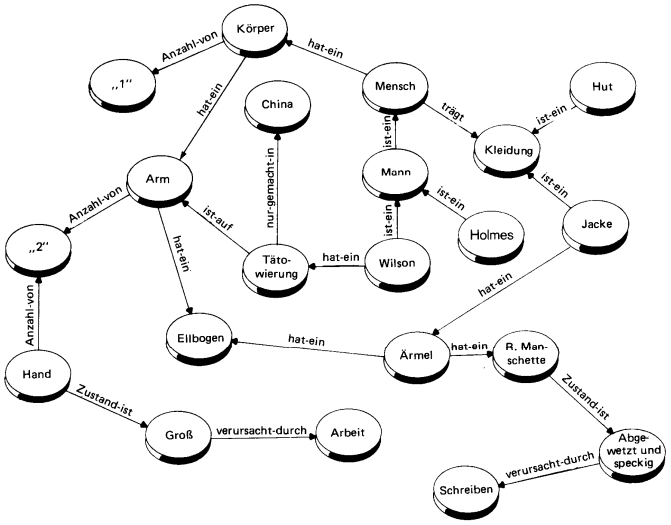
[„Aber wie …“]

„Ihre Hände, mein lieber Herr! Die Rechte ist ein ganzes Stück größer als die Linke. Sie müssen damit gearbeitet haben, zumal die Muskeln stärker entwickelt sind … Die Fischtätowierung direkt über dem Handgelenk kann nur aus China stammen. Diese Art, einen Fisch mit zartrosa Schuppen darzustellen, ist ziemlich typisch für China. [und] Was sonst sollte man aus dieser rechten Manschette schließen, die fünf Zoll breit so speckig aussieht, und links aus dem abgewetzten Fleck am Ellbogen, dort wo Sie ihn auf dem Pult aufstützen …“

(frei nach Sherlock Holmes, *The Red-Headed League*)

Semantische Netze

Beispiel:

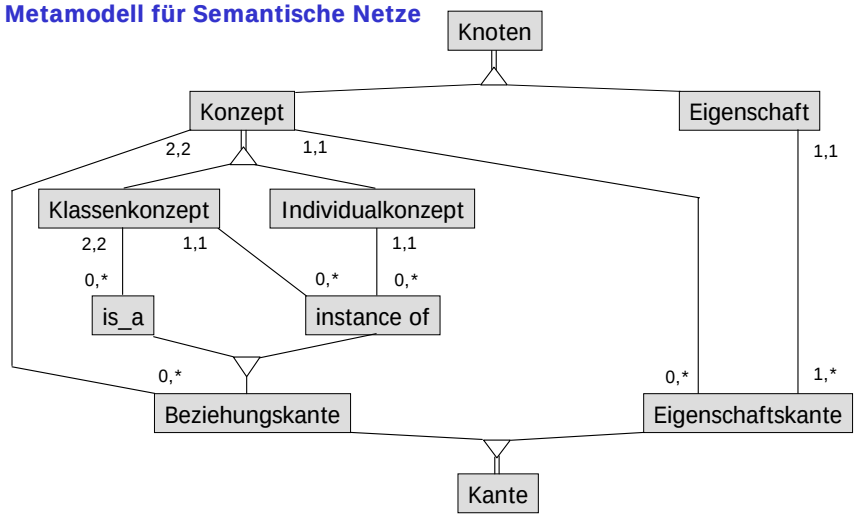


(Quelle: Harmon/King)

Anhand der Ist-ein-Relation werden in Semantischen Netzen Eigenschaften vererbt. Speziellere Knoten erben die Eigenschaften allgemeinerer Knoten.

Semantische Netze

Metamodell für Semantische Netze



Beispiele:

- Konzepte: Körper, Mensch
- Eigenschaften: „1“, „2“, „abgewetzt und speckig“
- Klassenkonzepte: Mensch, Mann
- Individualkonzepte: Wilson, China
- Beziehungskante: ist_ein(Mensch, Mann)
- Eigenschaftskante: Anzahl_von(Arm,2)

Frames

Einordnung

- Frames haben ihren Ursprung im Schema-Begriff der Kognitionspsychologie, wo sie als Modell des menschlichen Gedächtnisses verwendet werden.
- Schema: Modell von Gedächtnisstrukturen, welches stereotypische Erinnerungsmuster berücksichtigt.
- Im Gegensatz dazu berücksichtigen Semantische Netze Assoziationen zwischen Begriffen.
- Beim Menschen werden Konzeptklassen durch ihr stereotypisches Klassenelement (Prototyp) repräsentiert. Dieses Element ist dadurch bestimmt, dass es möglichst viele Eigenschaften mit anderen Klassenelementen gemeinsam hat und von den Elementen anderer Klassen möglichst verschieden ist.
- Die Eigenschaften des Prototyps sind mit Default-Werten besetzt. Sie stellen die erwarteten Eigenschaften dar, die im Fall konkreter Instanzen mehr oder weniger abweichen können.
- Minsky (1975): Umsetzung des Schema-Begriffs in ein Repräsentationsformat.

Beispiel: Gedächtnisschema „Hochgebirge“

- Aktivierung durch Erinnerung an eine Wanderung oder an ein Foto.
- Durch die Aktivierung entsteht eine mentale Präsenz einer (stereo)typischen Hochgebirgslandschaft.
- Gleichzeitig entsteht eine gewisse Erwartungshaltung, die eine kontextabhängige Interpretation von Objekten und Ereignissen bewirkt.

Frames

Begriffsbestimmung

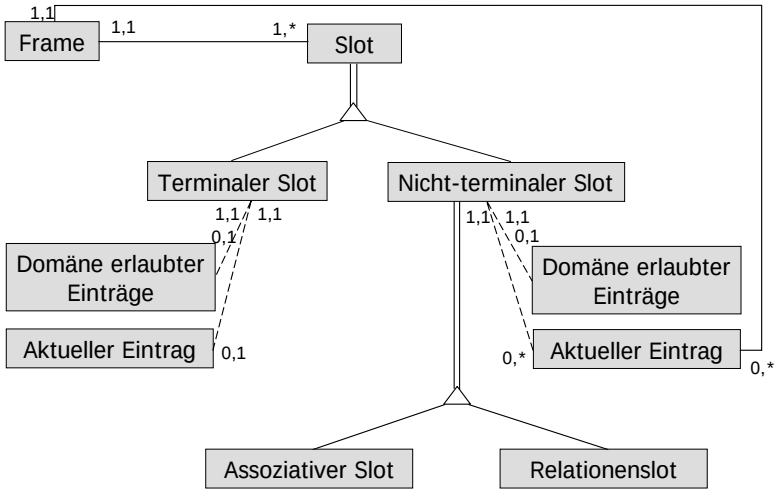
- Frames stellen Konzepte dar.
- Ein Frame besteht aus mehreren Slots, welche die Eigenschaften oder Rollen von Frames beschreiben.
- Slots besitzen Sloteinträge, welche die Eigenschaftsausprägungen darstellen.
- Sloteinträge können sein

  (1) bei terminalen Slots konkrete Werte oder

  (2) bei nicht-terminalen Slots wiederum Frames. Dadurch können Beziehungen zwischen Frames aufgebaut werden.

Frames

Metamodell für Frames



Frames

Modellierung von Eigenschaften (1)

- Eigenschaften von Konzepten werden durch Slots dargestellt.
- Eigenschaftswerte werden durch Sloteinträge dargestellt.
- Terminale Slots enthalten als Werte Zeichenfolgen.

Notebook	Größe	Gewicht	Verwendungsform
	klein	leicht	mobil

Desktop-PC	Größe	Gewicht	Verwendungsform
	mittel	schwer	stationär

Die Eigenschaftswerte terminaler Slots sind grundsätzlich einwertig (im Sinne von 1NF). Dadurch sind Wiederholungswerte, wie z.B. (mobil, stationär), ausgeschlossen.

Frames

Modellierung von Eigenschaften (2)

- Die Beschreibung eines Slot kann mit einer Angabe erlaubter Einträge versehen sein.
- Die konkreten Werte werden als aktuelle Einträge bezeichnet.

Fujitsu-Siemens Scaleo 800Si P24051	...	Arbeitsspeicher	Festplatte	...
	...	**erlaubt**: {512 MB, 1 GB, 1,5 GB} **aktuell**: {512 MB}	**erlaubt**: {20 GB, 40 GB, 80 GB} **aktuell**: {40 GB}	...

Erlaubte Einträge sind analog zum Domänenkonzept der Datenmodellierung.

Prolog

- Ein Prolog-Programm besteht aus einer Menge von Klauseln (Horn-Klauseln, Prädikatenlogik 1. Ordnung).
- Jede Klausel ist entweder ein Fakt oder eine Regel.

a) Fakten

```
likes(john,mary).
```

Beziehung:	likes
Objekte:	john, mary (Literal beginnt mit einem Kleinbuchstaben: Konstante; die Reihenfolge der Objekte muss für alle Fakten gleich sein)
Namen der Objekte:	Argumente
Name der Beziehung:	Prädikat
Datenbasis:	Menge von Fakten und Regeln (= Prolog-Programm):
Anzahl der Objekte:	Stelligkeit des Prädikats

Prolog

b) Fragen an Prolog (Goals)

```
?- likes(john,mary).
=> yes
?- likes(john,jane).
=> no
```

Lösungssuche erfolgt durch pattern matching. Ein matching fact ist gefunden, wenn

- gleiches Prädikat und
- gleiche, korrespondierende Argumente vorliegen.

„no“ bedeutet:

- Es wurde kein matching fact in der Datenbasis gefunden, d.h. „das Prolog-Programm weiß darüber nichts“.
- In der Realität kann die Aussage durchaus erfüllt sein.

Prolog

c) Variable

```
?- likes(john,X).
X = mary
```

Literal beginnt mit einem Großbuchstaben: Variable.

Variable können

- instantiiert (an ein Objekt gebunden) oder
- nicht instantiiert (nicht an ein Objekt gebunden)

sein. Die Instantiierung kann nicht durch Zuweisung geändert werden (keine Von-Neumann-Variable, keine computational history).

Prolog

c) Variable

Bei der Lösungssuche wird eine Variable mit dem ersten korrespondierenden Objekt instantiiert, das in der Datenbasis gefunden wird.

```
likes(john,mary).
likes(john,wine).
likes(john,dogs).
likes(jack,mary).
likes(john,hotdogs).

?- likes(john,X).
```

```
X = mary;
X = wine;
X = dogs;
X = hotdogs
```

Fundstellen werden „markiert“. Durch Eingabe von „;“ (bei manchen Prolog-Implementationen automatisch) wird das goal erneut zu erfüllen versucht (re-satisfy). Dabei wird die Variable neu instantiiert.

Prolog

d) Konjunktionen

```
likes(mary,food).
likes(mary,wine).
likes(john,wine).
likes(john,mary).
```

```
?- likes(john,mary), likes(mary,john).
=> no
```

Und-Verknüpfung: „ , “
Oder-Verknüpfung: „ ; “

Prolog

d) Konjunktionen

```
likes(mary,food).
likes(mary,wine).
likes(john,wine).
likes(john,mary).
```

```
?- likes(mary,X), likes(john,X).
     (1)          (2)
```

likes(mary,food).	(1) X = food	(1)		
likes(mary,wine).			(1) X = wine	
likes(john,wine).				(2) yes
likes(john,mary).				
Anmerkung	X instantiiert	goal (2) fails	Backtracking mit goal (1); X wird neu instantiiert	X = wine

Bei der Lösungssuche wird für jedes goal bzw. subgoal eine eigene Markierung verwendet: (1), (2)

Prolog

e) Regeln

Beispiel:

„X ist ein Vogel wenn: X ist ein Tier und X hat Federn.“
Eine Regel ist eine generelle Aussage über Objekte und Beziehungen.

```
likes(john,X) :- likes(X,wine).
```

„john likes X if X likes wine“

Prolog

e) Regeln

Eine Regel besteht aus

- Kopf (head) => goal
- Körper (body) => subgoals

verbunden durch „if“ (:-). Der Körper beschreibt die Konjunktion derjenigen goals, die den Kopf erfüllen.

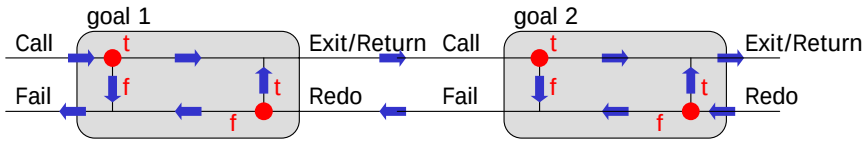
```
likes(john,X) :- likes(X,wine), likes(X,food).
```

Ein bestimmtes Prädikat kann gleichzeitig zur Bildung von Fakten und von Regeln genutzt werden:

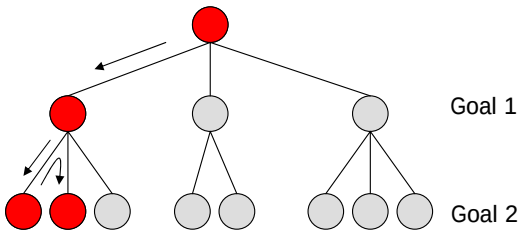
```
likes(mary,food).
likes(mary,wine).
likes(john,X) :- likes(X,wine).
```

Ausführung von Prolog-Programmen

1. Ausführung nichthierarchischer und nichtrekursiver goals (Boxenmodell)



Die Lösungssuche erfolgt nach dem Prinzip des Leftmost-Depth-First



Ausführung von Prolog-Programmen

2. **Suche nach weiteren Lösungen**
   Eingabe von „;“
   Dabei wird nach goal 2 EXIT mit REDO verknüpft.

3. **Der CUT !**
   Auftrennen der Verbindung zwischen REDO und EXIT.

4 . **Hierarchische und rekursive goals**
   Durch Auswerten der Subgoal-Liste wird die „Weichenstellung“ {true, false} ermittelt.

5. **Backtracking**
   a) Prolog-Programm (Datenbasis) wird von oben nach unten durchsucht.
   b) Subgoals werden entlang der Subgoal-Liste von links nach rechts geprüft.
   c) Redefinitionen eines Prädikats werden untersucht, wenn Vorgängerdefinition scheitert.

Ausführung von Prolog-Programmen

Prolog-Beispiel (1): Verwandtschaftsbeziehungen

Deklarative Beschreibung von Verwandtschaftsbeziehungen in Prolog

- **Prädikate zur Bildung von Fakten**

```
elter(X,Y).        „X ist Elternteil von Y“
geschl(X,Y).       „Geschlecht von X ist Y“
ehemann(X,Y).      „X ist Ehemann von Y“
```

- **Prädikate zur Bildung von Regeln**

```
ehefrau(X,Y) :- ehemann(Y,X)
„X ist Ehefrau von Y, wenn Y Ehemann von X ist“
```

Beschreibung aller weiteren Verwandtschaftsbeziehungen ebenfalls in Form von Regeln.

→ Prolog-Beispiel1.pdf

Ausführung von Prolog-Programmen

Prolog-Beispiel (2): gcd

Berechnung des größten gemeinsamen Teilers zweier natürlicher Zahlen

```
PREDICATES
gcd(INTEGER,INTEGER,INTEGER)

CLAUSES
gcd(0,X,X) :- !.
gcd(X,0,X) :- !.
gcd(X,X,X) :- !.
gcd(X,Y,E) :- Y > X, Y1 = Y - X, gcd(X,Y1,E).
gcd(X,Y,E) :- X > Y, X1 = X - Y, gcd(X1,Y,E).
```

Beispiel:

gcd(49,35) = 7 oder

gcd(49,35,7)

Rechenschema:

49	14	7
35	21	7

Erklärung:

7*7 – 5*7 = 2*7

5*7 – 2*7 = 3*7

3*7 – 2*7 = 1*7

2*7 – 1*7 = 1*7

1*7 – 1*7 = 0*7

Ausführung von Prolog-Programmen

Prolog-Beispiel (3): Logelei

Gerade ist der deutsche Botschafter in Südknabusien eingetroffen, da schickt er auch schon ein Fernschreiben nach Bonn:
„Erhalte soeben Kleidervorschrift für Antrittsbesuch beim Präsidenten. Sie lautet:

- Falls Sie kein gelbes Jackett tragen, müssen Sie eine blaue Hose und braune Schuhe tragen.
- Falls Sie braune Schuhe tragen, sind lange Hose und grünes Hemd vorgeschrieben.
- Es ist verboten, ein gelbes Jackett und dazu einen roten Schlips, jedoch keine blaue Hose zu tragen.
- Falls Sie keinen roten Schlips tragen, dürfen Sie weder lange Hosen, noch ein grünes Hemd tragen.
- Es ist verboten, kurze Hosen und dazu einen roten Schlips, jedoch keine braunen Schuhe zu tragen.
- Es ist verboten, eine lange blaue oder eine kurze nichtblaue Hose zu tragen.

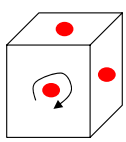
Was soll ich bloß anziehen?“
(aus ZEIT-Magazin)

→ Prolog-Beispiel3.pdf

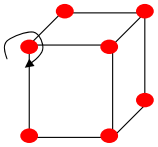
Ausführung von Prolog-Programmen

Prolog-Beispiel (4): Würfelspiel

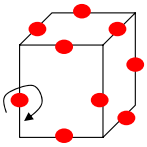
3 Möglichkeiten zur Modellierung eines Würfels



6 Flächen je
4 Positionen



8 Ecken je
3 Positionen



12 Kanten je
2 Positionen

→ Prolog-Beispiel4.pdf

Die ausgewogenste Struktur des Problemraums (Suchraums) entsteht durch die erste Modellierung: 6 Flächen je 4 Positionen.