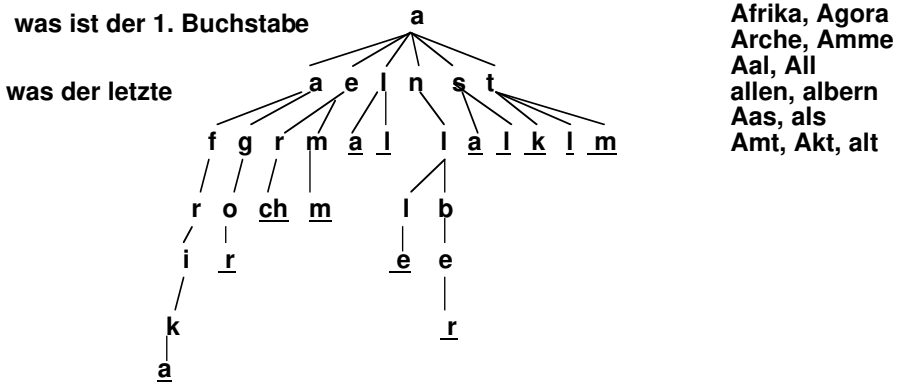
Gedächtnis

Kurzzeitgedächtnis: 7 ± 2 kognitive Einheiten (chunks)
Langzeitgedächtnis: Index und Inhalt
Elementary Perceiver And Memorizer - EPAM (Feigenbaum, Simon 1963)



Produktionensysteme - Ziele

Trennung von Kontrolle
- unveränderlich, sachbereichsunabhängig, aufgabenspezifisch
und "Wissen"
- veränderlich, sachbereichsspezifisch, aufgabenunabhängig
Verständlichkeit
leichtere Beherrschbarkeit eines Systems durch den Benutzer
leichtere Wartung der Regelmenge
Kleine, unabhängige Einheiten
Lernen von Regeln durch ein System

Produktionensysteme (Post 1943)

- A Alphabet, bestehend aus den disjunkten Mengen terminaler und nichtterminaler Symbole
- Axiom, ein ausgezeichnetes Wort, mit dem die Ableitungen beginnen
- Produktionen, W --> V mit W,V aus A* -- das Teilwort W wird durch das Teilwort V ersetzt
- Das Ergebnis einer Produktion wird einem Speicher hinzugefügt, in dem anfangs nur das Axiom ist.

Produktionensystem

IF <Bedingung1, ..., Bedingung n> THEN Handlung
Interpreter:
Abgleich von Bedingungsteil (Vorwärtsinferenz) oder
Handlungsteil (Rückwärtsinferenz) der Regeln mit Kontext
Konfliktauflösung zwischen allen anwendbaren Regeln
Anwendung der Handlung durch Modifizieren des Kontexts

Strategien:
Abgeglichene Bedingungen oder Handlungen werden aus dem Kontext gelöscht - zerstörerisch (irrevocable)
Aufbewahren des Kontextes mit Angabe der Regelauswahl bei diesem Kontext - tentativ

Beispiel 1

Kontext

ein (lampe1). tutnicht (lampe1). verb (lampe1, sicherung).
ein (lampe2). tutnicht (lampe2). verb (lampe2, sicherung).

Regeln

R1: IF ok(Geraet) & verbunden (Geraet, Sicherung)
THEN intakt (Sicherung).
R2: IF ein (Geraet) & arbeitet (Geraet)
THEN ok (Geraet).
R3: IF ein (Geraet) & tutnicht (Geraet)
THEN gestoert (Geraet).
R4: IF verb (Geraet1, Sicherung) & verb (Geraet2, Sicherung)
THEN gleiche_sicherung (Geraet1, Geraet2, Sicherung).
R5: IF gestoert (Geraet) & verb (Geraet, Sicherung)
THEN verdacht (Geraet, Sicherung).
R6: IF gleiche_sicherung (G1, G2, S) & gestoert (G1) & gestoert (G2)
THEN defekt (S).

Beispiel 2

Kontext

gestoert (lampe1). verb (lampe1, sicherung).
ein (lampe2). tutnicht (lampe2). verb (lampe2, sicherung).

Regeln

R1: IF ok(Geraet) & verbunden (Geraet, Sicherung)
THEN intakt (Sicherung).
R2: IF ein (Geraet) & arbeitet (Geraet)
THEN ok (Geraet).
R3: IF ein (Geraet) & tutnicht (Geraet)
THEN gestoert (Geraet).
R4: IF verb (Geraet1, Sicherung) & verb (Geraet2, Sicherung)
THEN gleiche_sicherung (Geraet1, Geraet2, Sicherung).
R5: IF gestoert (Geraet) & verb (Geraet, Sicherung)
THEN verdacht (Geraet, Sicherung).
R6: IF gleiche_sicherung (G1, G2, S) & gestoert (G1) & gestoert (G2)
THEN defekt (S).

Beispiel 3

Kontext

gestoert (lampe1). verb (lampe1, sicherung).
gestoert (lampe2). verb (lampe2, sicherung).

Regeln

R1: IF ok(Geraet) & verbunden (Geraet, Sicherung)
THEN intakt (Sicherung).
R2: IF ein (Geraet) & arbeitet (Geraet)
THEN ok (Geraet).
R3: IF ein (Geraet) & tutnicht (Geraet)
THEN gestoert (Geraet).
R4: IF verb (Geraet1, Sicherung) & verb (Geraet2, Sicherung)
THEN gleiche_sicherung (Geraet1, Geraet2, Sicherung).
R5: IF gestoert (Geraet) & verb (Geraet, Sicherung)
THEN verdacht (Geraet, Sicherung).
R6: IF gleiche_sicherung (G1, G2, S) & gestoert (G1) & gestoert (G2)
THEN defekt (S).

Beispiel 4

Kontext

gestoert (lampe1). gestoert (lampe2).
gleiche_sicherung (lampe1, lampe2, sicherung)

Regeln

R1: IF ok(Geraet) & verbunden (Geraet, Sicherung)
THEN intakt (Sicherung).
R2: IF ein (Geraet) & arbeitet (Geraet)
THEN ok (Geraet).
R3: IF ein (Geraet) & tutnicht (Geraet)
THEN gestoert (Geraet).
R4: IF verb (Geraet1, Sicherung) & verb (Geraet2, Sicherung)
THEN gleiche_sicherung (Geraet1, Geraet2, Sicherung).
R5: IF gestoert (Geraet) & verb (Geraet, Sicherung)
THEN verdacht (Geraet, Sicherung).
R6: IF gleiche_sicherung (G1, G2, S) & gestoert (G1) & gestoert (G2)
THEN defekt (S).

Beispiel 5

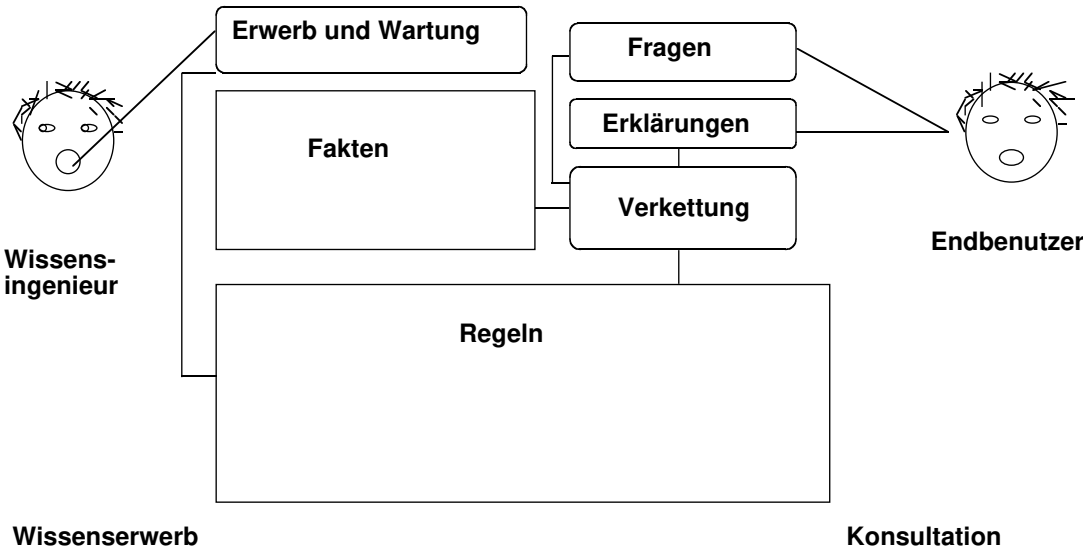
Kontext

defekt (sicherung).

Regeln

R1: IF ok(Geraet) & verbunden (Geraet, Sicherung)
THEN intakt (Sicherung).
R2: IF ein (Geraet) & arbeitet (Geraet)
THEN ok (Geraet).
R3: IF ein (Geraet) & tutnicht (Geraet)
THEN gestoert (Geraet).
R4: IF verb (Geraet1, Sicherung) & verb (Geraet2, Sicherung)
THEN gleiche_sicherung (Geraet1, Geraet2, Sicherung).
R5: IF gestoert (Geraet) & verb (Geraet, Sicherung)
THEN verdacht (Geraet, Sicherung).
R6: IF gleiche_sicherung (G1, G2, S) & gestoert (G1) & gestoert (G2)
THEN defekt (S).

Expertensysteme



Einfache Fragekomponente

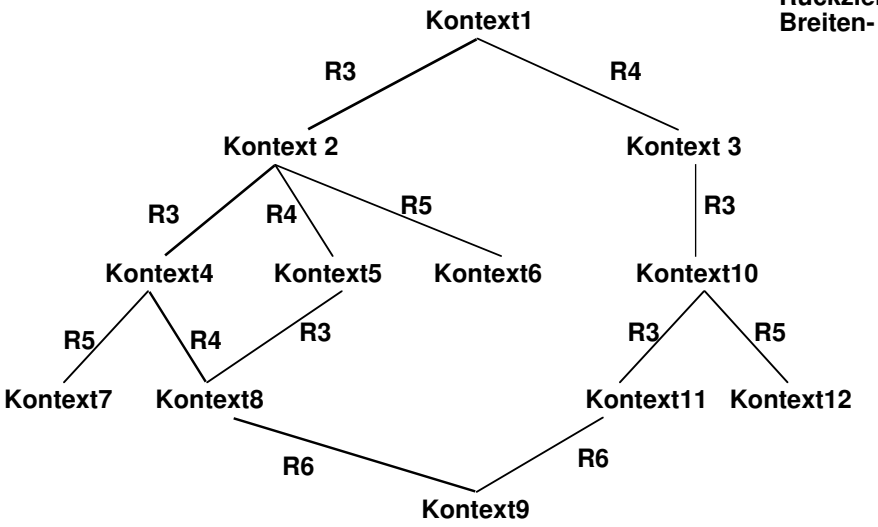
Rückwärtsverkettung
Wenn es keine Regel mit dem aktuellen Ziel im Handlungsteil und keinen Kontextinhalt gibt, der das aktuelle Ziel wahr macht, dann frage den Benutzer nach dem aktuellen Ziel.

Einfache Erklärungskomponente

Zu jedem (Teil-) Ziel wird die Regel und die Bedingungen aufgesammelt; ist eine Bedingung hergeleitet, so wird diese Herleitung vor die bisherige Spur gehängt.

Regelabarbeitung als Suche

Rückziehverfahren
Breiten- /Tiefensuche



Produktionensystem in Prolog

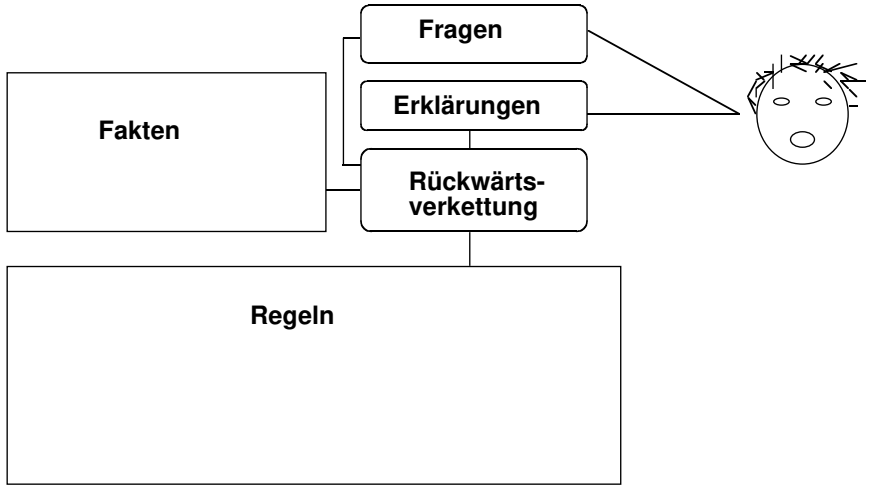
Rückwärtsverkettung
Kontext als Menge von Fakten
Zum Finden einer Antwort A auf eine Frage F:

- **Ist F ein Fakt im Kontext, dann ist A "F ist wahr".**
- **Gibt es eine Regel der Form IF B THEN F,**
  erkunde B, um die Antwort A zu finden.
- **Ist B von der Form F1 UND F2, dann untersuche F1.**
  Ist F1 falsch, dann ist A "F ist falsch".
  Ist F1 wahr, dann erkunde F2 und kombiniere die Antworten.
- **Ist B von der Form F1 ODER F2, dann untersuche F1.**
  Ist F1 wahr, dann ist A "F ist wahr".
  Ist F1 falsch, untersuche F2 und kombiniere die Antworten.
- **Ist F1 (F2) eine fragbare Frage,**
  dann frage Benutzer nach F1 (F2).

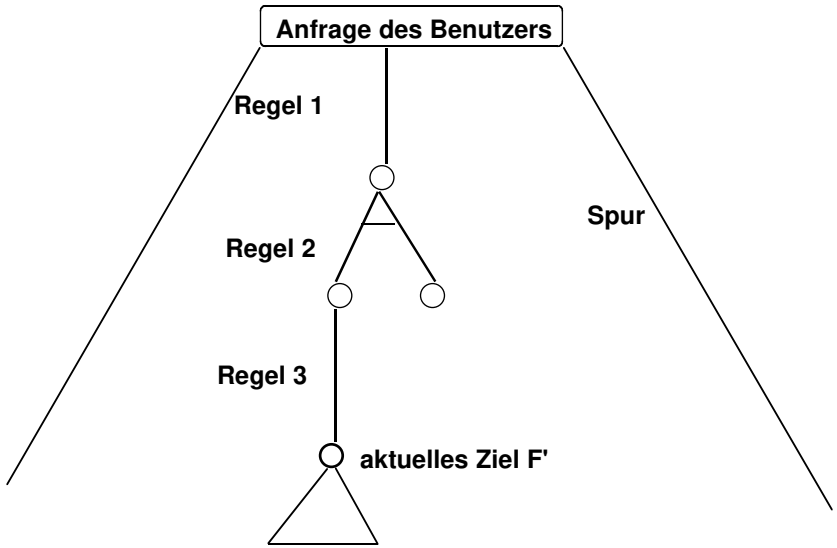
erforsche

hole_antwort

einfache Expertensystemhülle



Suche im UND/ODER Baum



Regeln

```
regel1: wenn   hat(Tier, fell)  oder  saeugt(Tler, jungen)
          dann   saeugetier (Tier).
:-  op( 900, xfx, :).
:-  op( 870, fx, wenn).
:-  op( 880, fx, dann).
:-  op( 550, xfy, oder).
:-  op( 540, xfy, und).
```

Hauptprozedur

erforsche (Ziel, Spur, Antwort)
gibt zur Antwort auch die Spur aus, die zur Antwort geführt hat.
ruft auf: hole_antwort, um vom Benutzer einfache Merkmale zu erfragen.

Verhalten des einfachen Systems

BEN: erforsche (saeugetier(wal), Spur, Antwort).
SYS: Ist die Aussage wahr: hat(wal,fell) ?
BEN: nein
SYS: Ist die Aussage wahr: saeugt(wal,junge) ?
BEN: warum?
SYS: regel1 leitet_ab saeugetier(wal) wenn saeugt(wal,junge).
Ist die Aussage wahr: saeugt(wal,junge) ?
BEN: ja
SYS: saeugetier(wal) ist wahr
BEN: warum?
SYS: regel1 leitet_ab saeugetier(wal), wenn saeugt(wal,junge).

Was ist zu tun?

Wissensbasis:
Regeln für einen Sachbereich
Fakten für einen Sachbereich

Interpreter:
Fragen und Antworten an den Benutzer definieren
Rückwärtsverkettung implementieren (erforsche)
Erklärungen für Systemantworten und für Fragen an Benutzer vorbereiten
(in erforsche die Spur bisheriger Regeln und Fakten aufsammeln)
Prozedur zur Interaktion mit Benutzer implementieren (hole_antwort)

Tips

Wie bastelt man Spur zusammen?
Rekursiver Aufruf mit "Bastelanleitung":
erforsche (Ziel, Spur, Antwort):-
Regel: wenn Bedingung dann Ziel,
erforsche (Bedingung, [Regel leitet_ab Ziel wenn Bedingung | Spur], Antwort).
leitet_ab und wenn auch als Operatoren definieren!

Wie bastelt man die Lösungen der Bedingungen zusammen?
UND/ODER Problemzerlegung - s. einfaches Produktionensystem
erforsche (Ziel1 und Ziel2, Spur, Antwort):-
erforsche (Ziel1, Spur, Antwort1),
fahre_fort (Antwort1, Ziel1 und Ziel2, Spur, Antwort).
und und oder sind auch als Operatoren definiert.

GUIDON - NEOMYCIN - HERACLES

William J. Clancey: From GUIDON to NEOMYCIN and HERACLES in Twenty Short Lessons: ORN Final Report 1979 - 1985
THE AI Magazine, August 1986, 40 - 60

MYCIN Heuristic Programming Project Stanford University 1973 - 1978
Regeln zur heuristischen Klassifikation von Bakterieninfektionen
EMYCIN Interpreter von MYCIN mit HCI
NEOMYCIN kontrastive Diagnostik, Diagnostk von medizinischem Wissen abgegrenzt
HERACLES Hülle für NEOMYCIN
GUIDON Lehrsystem (tutoring system) für EMYCIN / HERACLES

MYCIN

Produktionen mit Evidenzwerten (Shortliffe)
Eine Bedingung ist in estimmtem Maße erfüllt.
Evidenzen von Bedingungen werden an den Handlungsteil weitergegeben.
Unterschiedliche Strategien:
minimale, maximale, verrechnete
IF B_1 ev_1 & ... & B_m ev_m
THEN H_n ev_n & ... & H_w ev_w

minimale Evidenz von allen Bedingungen $ev_n = \min(ev_1, ..., ev_m)$
Durchschnitt der Evidenzen aller Bedingungen $ev_w = ø(ev_1, ..., ev_m)$

IF Infektion ist Meningitis (0,8) & Meningitistyp ist bakteriell (0,3)
THEN Diplococcus (0,55)

Mehrfachableitungen

Evidenzverstärkung bei mehrfacher Herleitung derselben Aussage durch voneinander unabhängige Regeln:
n Herleitungen
ev i Evidenzwert der i-ten Herleitung $i \leq n$

1 -

$$\prod_{i=1}^{n} (1 - ev_i)$$

Diplococcus 0,2 **1- ((1- 0,2) * (1 - 0,3) * (1 - 0,8)) = 1 - 0,112 = 0,888**
Diplococcus 0,3
Diplococcus 0,8

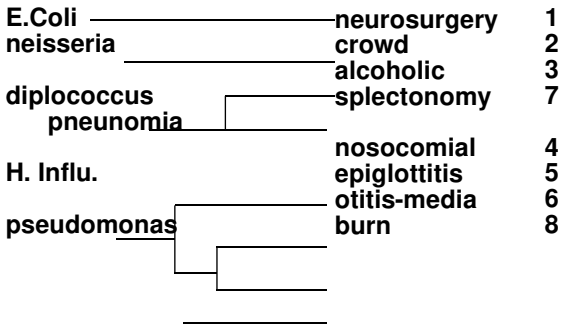
Wie kann die Unabhämngigkeit garantiert werden?
Können Experten soche Zahlen angeben?

Problem der Fragereihenfolge

Does Smith have a history of neurosurgery? - NO
Does Smith live in a crowded environment? - NO
Do you suspect recent alcoholic history in Smith? - NO
Is meningitis a hospital-acquired infection? - YES
Is Smith's clinical history consistent with epiglottitis? - NO
Is Smith's clinical history consistent with otitis-media? - NO
Has Smith undergone splenectomy? - NO
Is Smith a burn patient? - YES

MYCIN

Hypothesen- Fragen



Strategie bei MYCIN:
Reihenfolge der Regeln; nur ein Prädikat für das Ziel (cover_for);
Gleichbehandlung aller Prämissen: wenn nicht bekannt, dann fragen.

Problem der Vermischung von Wissensarten

IF Typ der Infektion ist Meningitis &
keine Labordaten vorhanden &
Typ der Meningitis ist bakteriell &
Patient ist älter als 17 Jahre &
Patient ist Alkoholiker
THEN Evidenz für E. Coli 0,2 und Diplococcus 0,3

Anwendbarkeit der Regel (Meningitis, Meningitistyp, fehlende Labordaten)
Sachbasierte Bedingung (Alkohol begünstigt E.Coli und Diplococcus)
Konsultationsbasierte Bedingung (frage keine Kinder, ob sie Alkoholiker sind)

Verschiedene Erklärungstypen

Warum wird jetzt diese Regel angewandt?
- Weil die Infektion Meningitis vom Typ bakteriell ist und es keine Labordaten gibt.

Warum soll jetzt festgestellt werden, ob der Patient Alkoholiker ist?
- Weil das für E.Coli und Diplococcus spricht.

Warum soll jetzt festgestellt werden, ob der Patient über 17 Jahre alt ist?
- Weil festgestellt werden soll, ob der Patient Alkoholiker ist und Kinder keine Alkoholiker sind.

Verschiedene Wissensarten

strategisches Wissen:
Beachte Kontraindikationen und Dosis,
bevor du etwas verschreibst!

strukturelles Wissen:

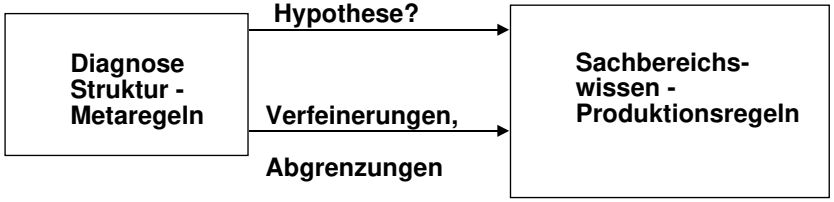
Kontraindikationen:
Patientendaten
infektionsspezifische Daten

heuristische Regel:
age < 7 --> no tetracycline

Rechtfertigungswissen:
Verfärbung wachsender Zähne, ...

Wenn es also keine Alternative zu Tetracyclinen gibt und die Krankheit ernst ist, dann kann man doch Tetracycline verschreiben!

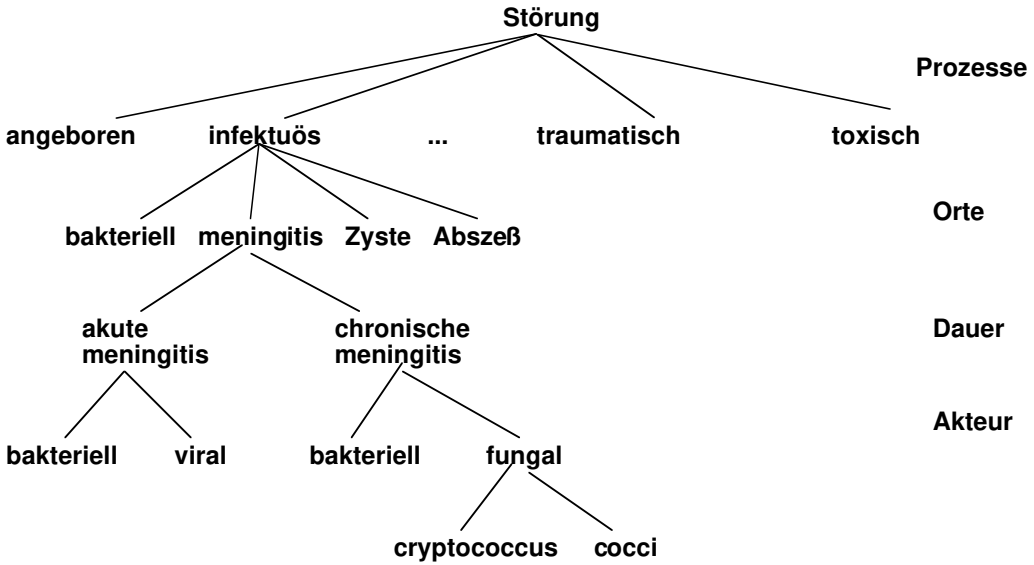
NEOMYCIN



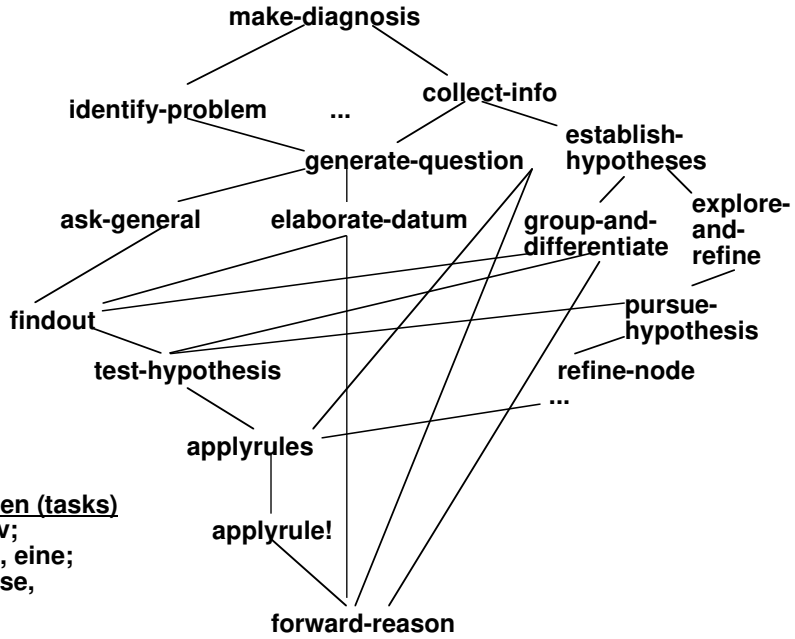
Beschwerden stoßen initiale Hypothese an;
group-and-differentiate: übergeordnete Diagnosen, Alternativen
explore-and-refine: Verfeinerungen zur Therapie

Prozesse: angeboren, infektuös, traumatisch, toxisch, ...
Orte: Meningitis, Zyste, Gehirnabszeß,...
Dauer: akute, chronische Meningitis
Patient:bakteriell, fungal, ...
cryptococcen, cocci, ...

Sachbereichsklassen



Struktur



Metaregeln für Aufgaben (tasks)
control: einmal, iterativ;
alle Lösungen, eine;
focus: Regel, Hypothese,
Befund
Abbruchbedingung

Deklarative Repräsentation der Metaregeln

```
task: process-finding
focus: $Finding
Metarule:
IF (AND ( OR ( findingtype $Finding redflag)
             (not (diffexplained $Finding)))
        (makeset (Triggers? $Finding $Rule)
    Rulelst)
THEN (task applyrule Rulelst)
```

Wenn ein Befund auf jeden Fall erklärt werden muß (redflag) oder noch nicht differentiell erklärt ist,
dann stoße Hypothesen an, die ihn erklären.

Eine von 6 Metaregeln für die Aufgabe process-finding.
Triggers? bindet eine anstoßende (triggering) Regel für einen Befund.
$Finding und $Rule werden durch Einheiten der Objektebene gebunden: Patientendaten (Befund), Sachbereichsregel

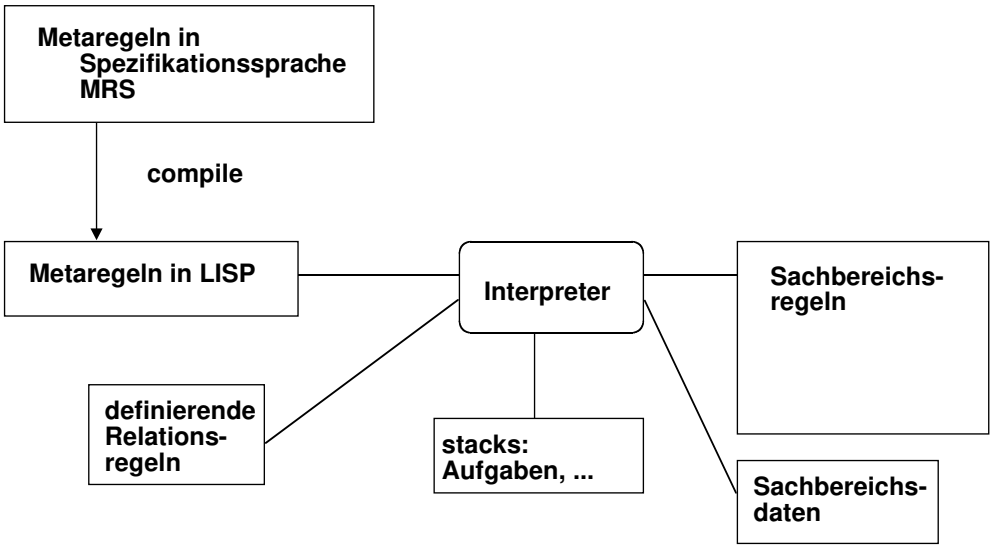
diffexplained wird durch eine Regel vom Typ definitional relation rule definiert.

Erklärungen

Die Strukturierung der Regelmenge durch die Metaregeln ermöglicht bessere Erklärungen.
MYCIN: alle Schritte des backward chaining werden als Erklärung ausgegeben.
NEOMYCIN:Aufgaben werden ausgegeben, wenn ein neuer Fokus etabliert wird (eine Hypothese durch group-and-differentiate oder eine Sachbereichsregel durch test-hypothesis).
Die Erklärungsprozedur gibt immer noch einfach die Schritte der Problemlösung aus, aber die Problemlösung ist jetzt gegliedert.

Ähnlich wird auch das Studentenmodell und das Debugging durch die Strukturierung unterstützt.

NEOMYCIN



Benutzerschnittstelle

KONSULTATION

What is S.'s temperature?
105.8 Fahrenheit
Has S. experienced seizures recently?

ERKLÄRUNG

We are trying to decide whether S. has meningitis.
Recent seizures are strongly related with meningitis.

SACHBEREICHSREGEL

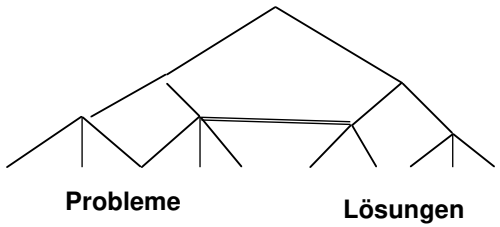
r060:
IF patient has experienced seizures recently, and
patient's anterior fontanel is abnormally tense
or bulging
THEN there is strong evidence (.8) that
the patient has meningitis.

AUFGABEN:

mr628
make-diagnosis()
mr384
collect-info()
mr062
establish-hypotheses()
mr585
group-and-differentiate()
mr400
test-hypothesis(meningitis)
mr411
applyrules (r060, r060, r323)
mr094
applyrule! (r060)
mr095
findout (seizures)

Heuristische Klassifikation

Sachbereich:
Klassifikationsnetzwerk,
Klassen von Problemen,
Klassen von Lösungen
explizite Kontrollstruktur:
Relationen zwischen Falldaten
und bekannten Lösungen



Krankheiten als Stereotype
Krankheitsklassen und Beziehungen
zwischen den Klassen

Patienten-Krankheiten,
Mahlzeiten-Weine,
Stereotype-Bücher, ...

Überdeckende Diagnose (cover & differentiate)

- Mögliche Ursachen für einen diagnostizierten Zustand.
- Die Ursachen werden ihrerseits als Zustand aufgefasst, der durch Ursachen erklärt wird.
- Regeln: Diagnose --> Symptom
- cover: mögliche Ursachen für Zustände
- differentiate: wenig Ursachen, die viele Zustände erklären!

Beschreibungen

Ziel war: erklärbare, wartbare, erweiterbare Problemlösung,
bei der das Fachvokabular des Sachbereichs vom Rechner so verwendet wird, wie es explizit und formal definiert ist - und nicht als eingefrorener Text, der dem Rechner nur eine Folge von ASCII-Zeichen ist.
Also ist das Vokabular, das wir dem Rechner zur Verfügung stellen, wichtig!
Wenn wir nur Produktionen haben, gibt es auch nur Bedingungen und Handlungen.
Was der Repräsentationsformalismus nicht unterscheiden kann, kann auch nicht unterschiedlich behandelt werden!

Beschreibungsebenen

70er **Vokabular:**
Bedingungen, Handlungen, Evidenz
Vorwärts-/ Rückwärtsverkettung, Verstärkung

80er **Anwendbarkeit, Kausalität, Dialog**
heuristische Klassifikation, Planverfeinerung, cover and differentiate

90er **Sachbereichsstrukturen (Ontologien)**
Problemlösungsstrategien

Methodologie praktisch:

1. **Wir bauen ein System, das eine Aufgabe erfüllt. Dafür entwerfen wir eine Repräsentationssprache.**
   MYCIN: Befund, Hypothese als Parameter repräsentiert,
   Kausalität, Ober-/Unterklasse als Folge von Prämissen in Regeln oder als Wert zu einem Attribut repräsentiert.
2. **Wir abstrahieren von dem konkreten Sachbereich der ersten Anwendung. Dabei finden wir allgemeinere, in verschiedenen Anwendungsbereichen auftretende Muster. Für diese entwerfen wir eine abstraktere Repräsentationssprache.**
   NEOMYCIN: Hypothesen als Sachbereichsklassen, werden mit group-and-differentiate und explore-and-refine bearbeitet.
3. **Wir analysieren die abstraktere Repräsentationssprache (Störung, Prozeß, Taxonomie)**
   und entwerfen eine neue Repräsentationssprache der unteren Ebene, die besser zur abstrakten Repräsentationssprache paßt.

...