Wissensrepräsentation

Zur Problemlösung benötigt man immer eine Repräsentation.

Wir haben kennen gelernt:

Operatoren zur Problemlösung als Suche

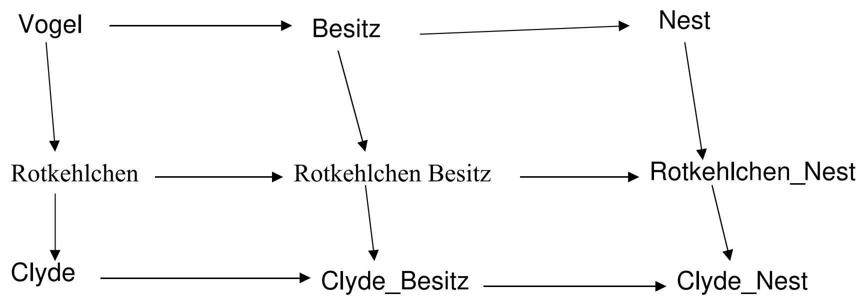
Produktionen zur Problemlösung als Suche in UND-ODER Bäumen

Formeln zur Problemlösung als Erfüllbarkeitsbeweis

Wissenrepräsentations-

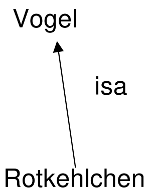
formalismus	entspricht Kalkül
sprache	entspricht Signatur
system	unterstützt jede Signatur im Kalkül
	und bietet bestimmte Kontrollstrukturen

Semantische Netze

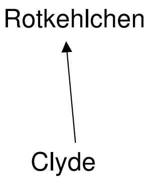


Zugriff auf zusammenhängende Begriffe
spreading activation: gemeinsam erreichbar über Pfad, der von zwei Knoten ausgeht.
Das Gemeinsame von Vogel und Nest ist Rotkehlchen Nest oder Clyde Nest

Unterscheidung von Kantentypen



Oberbegriff - Unterbegriff
Vererbung der Eigenschaften des Oberbegriffs
auf den Unterbegriff
Klassifizierung des Unterbegriffs aufgrund der
Eigenschaften

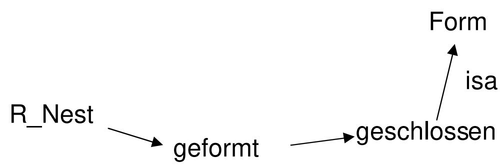


Instanz - Begriff
Realisierung der Instanz als Mitglied der Klasse,
die durch den Begriff angegeben ist, anhand der
Eigenschaften

Eigenschaften

Relation zwischen Vogel und Nest
Eigenschaft des Vogels, dass das,
was er besitzt, ein Nest ist.

Vererbung mit Beschränkung
Ein Rotkehlchen ist ein Vogel,
dessen Besitz eine bestimmte Art
von einem Nest ist.



Das R_Nest ist geschlossen
(nicht struppig oder ausgefranst)

Implementation

Repräsentationssprache:

isa(rotkehlchen,vogel). isa(r_nest,nest). isa(geschlossen,form).
isa(vogel,tier). isa(nest,behausung).
role(besitzt,tier,behausung). role(geformt,r_nest,geschlossen).
role(besitzt,vogel,nest).
role(besitzt,rotkehlchen,r_nest).

Repräsentationssystem:

```
ask_isa(isa(X,Y)):-isa(X,Y).          ask_role(role(R,C,VC)):-role(R,C,VC).
ask_isa(isa(X,Y)):-isaX,Z),           ask_role(role(R,C,VC)):-isa(C,X),
        ask_isa(Z,Y)),!.                                 role(R,X,VC).
```

Verwendung:

Ist das R. einTier?	?-ask_isa(isa(rotkehlchen,tier)).	yes
Was besitzt das R.?	?-ask_role(role(besitzt,rotkehlchen,X)).	X=r_nest

etwas komplizierter

Welcher Vogel ist das,
der etwas besitzt,
was eine geschlossene Form aufweist?

```
?- ask_complex(and_complex
                    (isa(X,vogel),
                     and_complex(role(besitzut,X,Y),
                                 role(geformt,Y,geschlossen)))).
X = rotkehlchen
Y = r_nest
ask_complex(Form):- Form = isa(X,Y), ask_isa(isa(X,Y)).
ask_complex(Form):- Form = role(X,Y,Z), ask_role(role(X,Y,Z)).
ask_complex(Form):- Form = and_complex(A,B), ask_complex(A),
                                              ask_complex(B).
```

Unterschied zu Logik

∃ X | isa(X,vogel) & besitzt(X,Y) & geformt(Y,geschlossen)

Für die Signatur eines neuen Sachbereich muss jedes Prädikat definiert werden.

Hier gibt es nun vordefinierte Prädikate, mit deren Hilfe Relationen für einen neuen Sachbereich angegeben werden: isa, role.

Statt die Relation selbst als Prädikat zu wählen (besitzt, geformt), wird der Relationsname als erstes Argument des allgemeinen Prädikats role angegeben.

Somit ist die Wissensrepräsentation bereits für bestimmte Anwendungen eingeschränkt bzw. vorbereitet.

Ebenen der Wissensrepräsentation

Implementation:
- LISP oder PROLOG- Prozeduren, die den Zugriff erlauben
  - ask_isa, ask_role, ask_complex

Logik:
- Erfüllbarkeit über allen Strukturen zu einer Signatur entscheiden

Begriffe:
- bestimmte Dinge eines Sachbereichs mit ihren Eigenschaften
  - tier, vogel, rotkehlchen, nest , form, ...

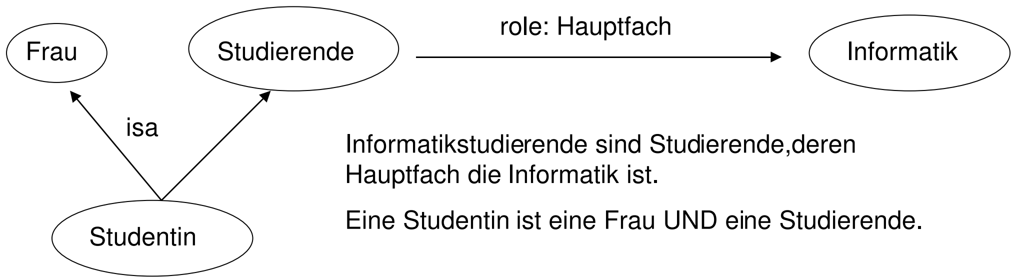
Sprache:
- Wörter, Sätze zur Beschreibung bestimmter Sachverhalte
  - Ein Rotkehlchen ist ein Vogel, dessen Nest geschlossen ist.

Epistemik (Wissensebene):
- Primitive zum Definieren von Begriffen
  - Oberbegriff - Unterbegriff (isa), Eigenschaft (role)

Termsubsumtionsformalismen (KL-ONE)

In epistemischen Primitiven Begriffe definieren. Teile der Signatur und der Interpretation sind vorgegeben. Einige Schlussfolgerungen sind vorbereitet.

≠ logische Ebene!

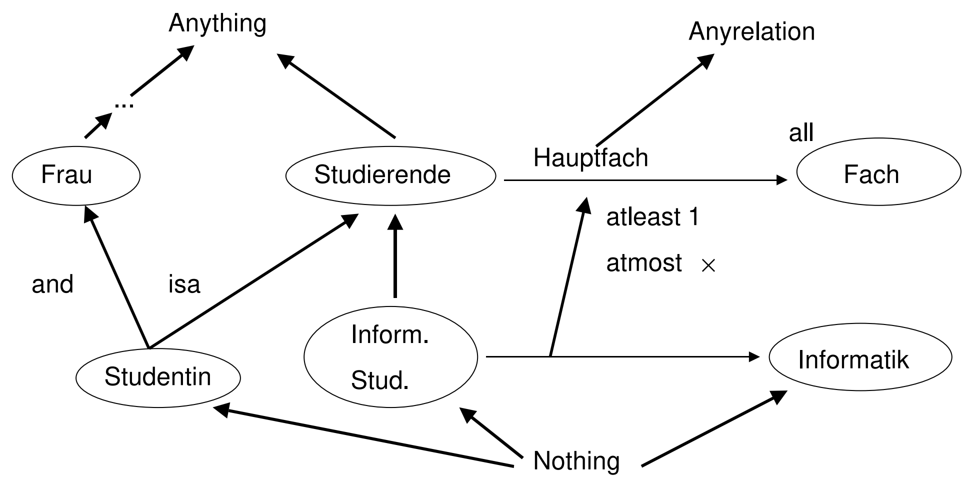


Informatikstudierende sind Studierende,deren Hauptfach die Informatik ist.

Eine Studentin ist eine Frau UND eine Studierende.

T-Box

für terminologisches Schließen



Studentin .=. (and Frau Studierende)

Informatikstudierende .=. (and Studierende (all Hauptfach Informatik))

Syntax von Termsubsumtionsformalismen

<terminology> ::= {<term-introduction> | <restriction> }*

<term-introduction> ::= <concept-introduction> | <role-introduction>

<concept-introduction> ::= <atomic-concept> .=. <concept> |
<atomic-concept> ≤ <concept> |
<atomic-concept> ≤ **ANYTHING**

<role-introduction> ::= <atomic-role> .=. <role> |
<atomic-role> ≤ <role> |
<atomic-role> ≤ **ANYRELATION**

<concept> ::= <atomic-concept> |
(and <concept>+) |
(all <role> <concept>) |
(atleast <number> <role>) |
(atmost <number> <role>)

<role> ::= <atomic-role> |
(androle <role>+)

<restriction> ::= (disjoint <atomic-concept> <atomic-concept>)

<number> ::= <non-negative-integer>

<atomic-role> ::= <identifier>

<atomic-concept> ::= <identifier>

Wohlgeformte, aus epistemischen Primitiven zusammengesetzte Ausdrücke

Semantik epistemischer Primitive und zusammengesetzter Ausdrücke

Knoten: Entscheidung über Begriffszugehörigkeit über notwendige Eigenschaften
d.h. ∀x x∈Begriff | ....

ext

Objekte: a1, a2, a3, ... **D**

Relationen (a1,b1),(a1,b2),(a1,b3),(a2,b1),... **D** x **D**

ext

für alle a ≤ t sei ext(t) ⊇ ext(a)

für alle a.=. t sei ext(t) = ext(a)

ext(**ANYTHING**) = **D**

ext(**ANYRELATION**) = **D** x **D**

ext((and c1 ... cn)) = ∩ ext(ci) wobei i von 1 bis n

ext((all r c)) = {x ∈ **D** | ∀ y : (x,y) ∈ ext(r) --> y ∈ ext(c) }

ext((atleast n r)) = { x ∈ **D** | card ({y ∈ **D** | (x,y) ∈ ext(r)}) ≥ n }

ext((atmost n r)) = { x ∈ **D** | card({y ∈ **D** | (x,y) ∈ ext(r)}) ≤ n }

ext((androle r1 ... rn)) = ∩ ext(ri) wobei i von 1 bis n

für alle (disjoint c1 c2) sei ext(c1) ∩ ext(c2) = {}