Inferenzmaschine

- 4-wertige Logik:
  w -- eingetragen oder abgeleitet,
  f -- negiert eingetragen oder negiertes Fakt ableitbar,
  widersprüchlich -- sowohl w als auch f eingetragen oder ableitbar,
  unbekannt -- weder w noch f eingetragen oder ableitbar
- explizite Negation
- Buchhaltung bei Fakten:
  Eingabe durch Benutzer, Ableitungspfad
- Vorwärtsinferenz mit Tiefenbeschränkung -- alle anwendbaren Regeln
- Rückwärtsinferenz mit Tiefenbeschränkung -- gezielter Beweis eines Faktes

Sorten

- Angabe von Sorten bei der Prädikatsdeklaration
  owner(<person>, <thing>)
- Angabe einer Sortentaxonomie mit Relation :<
  reflexiv, transitiv, antisymmetrisch
  woman:<person, female
  person:<living_thing
  NIL:=female, male (leere Schnittmenge)
- Zuordnung von Konstanten zu Sorten -- automatisch
  owner(morik, car3)
  ext(arg_sort(1, owner))={morik}
  ext(arg_sort(2, owner))={car3}

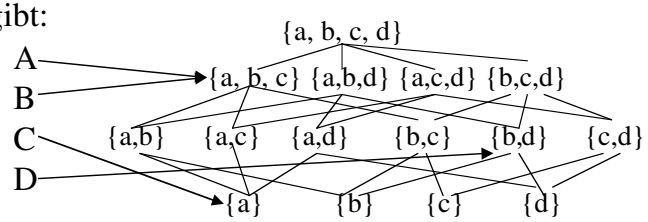
ext

ext: Sortennamen --> Potenzmenge der Konstanten

Fakten $p(t_1, ..., t_i, ... t_n)$, not $(p(t_1, ..., t_i, ... t_n))$

t_i in ext(arg_sort(i, p) gdw.
$p(t_1, ..., t_i, ... t_n)$ oder not $(p(t_1, ..., t_i, ... t_n))$ in Faktenbasis

Fakten: { p(a,b), p(b,c), not(p(c,a)), r(a,b), not(r(a,d)) }
Sortennamen:{A, B, C, D}
ergibt:



Äquivalenzklassen von Sorten: cext

- Zwei Sorten s1, s2 sind äquivalent, wenn ihre Extensionen gleich sind.
- Jede Äquivalenzklasse bekommt einen eindeutigen Namen.
- Alle Klassen haben mindestens eine Sorte, die zu der Klasse gehört.
- cext: Klassennamen --> Potenzmenge der Konstanten

Sortenkorrektheit

- Ein Fakt ist sortenkorrekt, wenn die Konstanten an Argumentposition kompatibel mit der Sortendeklaration des Prädikats sind.
- cext induziert eine partielle Ordnung über der Potenzmenge der Konstanten.
  $c1 \leq c2$ gdw. $cext(c2) \supseteq cext(c1)$
- Unterklasse: $c1 \leq c2$ und $\neg (c2 \leq c1)$
- subsort von c1: $\{c \mid c \leq c1 \text{ und } c \neq c1\}$
- supersort von c1: $\{c \mid c1 \leq c \text{ und } c \neq c1\}$
- Eine Konstante ist sortenkorrekt, wenn sie aus der angegebenen Sorte oder einer Untersorte stammt.

Verband der Äquivalenzklassen

Schnittklassen bilden

inters: Klassen --> Potenzmenge der Klassen

inters(c) = {c1 in Klassen | {c, c1} in IS}

IS = {{c1, c2} $\subseteq$ Klassen | $\neg$ (c1 $\leq$ c2), $\neg$ (c2$\leq$c1), $\neg$(cext(c1) $\cap$ cext(c2) = { }) }

Die Sorten zusammen mit den Schnittsorten bilden einen Verband. Die entsprechenden Klassen bilden wiederum einen Verband.

Automatisches Bilden des Verbands

- Berechnen der Extensionen an Argumentstellen
- Erstellen der Schnittsorten
- Berechnen des Verbands der Äquivalenzklassen.
- Der Platz von c im Verband wird durch supers(c), subs(c), inters(c) und disjoint(c) festgelegt.
- Bei neuen Eintragen in die Faktenbasis:
  Sorten erweitern oder
  Eintrag ablehnen, falls nicht sortenkorrekt.
- Entspricht unüberwachtem Lernen in Aussagenlogik.

Sortenkorrekte Inferenz

- Jeder Variablen in einer Regel ist eine Menge von Sorten zugeordnet.
- Substitutionen müssen sortenkorrekt sein, d.h. eine Variable kann nur durch eine Konstante aus der Extension der Schnittmenge der Sorten dieser Variablen ersetzt werden.
- var_sort: Variablen --> Potenzmenge der Sorten
- sort_correct($\{V_1/t_1, ..., V_n/t_n\}$) gdw.
  für alle $i_{1...n}$: $t_i \in$ cext(class(var_sort(V_i)))

Beispiel

contains (A, W) & affect (W, K) --> affect (A, K)

$A\sigma$ in ext(arg_sort(1, contains)) $\cap$ ext(arg_sort(1, affect))

$W\sigma$ in ext(arg_sort(2, contains)) $\cap$ ext(arg_sort(1, affect))

Unterschied zwischen *falsch* und *unsinnig*:

not(trinkt(paul, hustensaft1)) trinkt(<tier>, <fluessiges>)

trinkt(paul, stein1)
not(trinkt(paul,stein)) — gleichermaßen unsinnig

Vorteile von Sorten

Für das Beweisen bzw. Vorwärtsinferenz:

- weniger Regeln anwendbar,
- weniger Fakten heranzuziehen,
- weniger Substitutionsmöglichkeiten,
- nur sinnvolle Ableitungen.

Für das Modellieren:

- Prädikatsdeklaration als Merkliste fahrt(<start>, <ziel>,...)
- semantische Prüfung der Eingabe fahrt(auto1, person1)
- Strukturierung der Objekte

Vorteile der gelernten Klassen

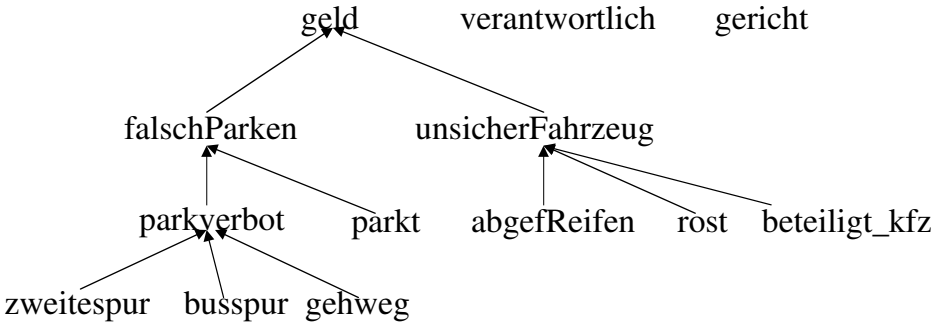
- noch weniger Unifikationsversuche (weniger anwendbare Regeln, verwendbare Fakten, Substitutionsmöglichkeiten)
- Betrachten des Klassenverbandes zeigt tatsächlich modellierte Struktur der Objekte -- stimmt sie mit der Absicht überein?
- Starke Einschränkung des Hypothesenraums für das Lernen von Regeln!

Regelstruktur

- Große Regelmengen sind unübersichtlich!
- Wie kann eine abstrakte Struktur wie heuristische Klassifikation in einer Regelmenge dargestellt werden?
- Kann für eine Regelmenge eine Struktur automatisch gefunden werden?

$\Rightarrow$Abstraktion über Regelmengen!

Regelgraphik

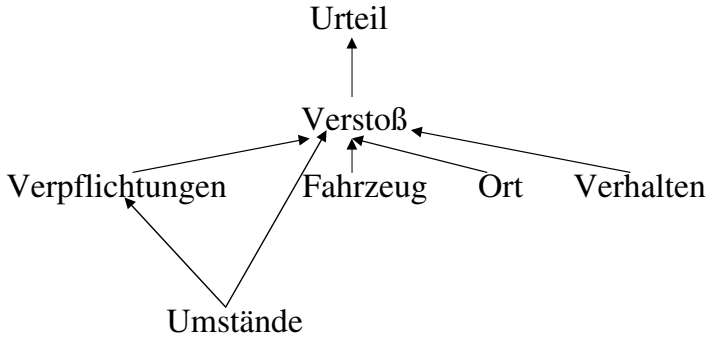


gehweg(X) --> parkverbot(X)
parkt(X, Y) & parkverbot(Y) --> falschParken(X)
beteiligt_ kfz(X,Y) & rost(Y) --> unsicherFahrzeug(X)
falschParken(X) --> geld(X)

Abstraktion

- Verschmelzen aller Zusammenhangskomponenten im Regelgraph
  - Zusammenhangskomponenten:
    zwischen allen Knoten i und j gibt es einen gerichteten Pfad von i nach j und zurück.
- Verschmelzen von Knoten mit identischer Menge von Vorgängern
- Verschmelzen von Knoten mit identischer Menge von Nachfolgern

Topologie



Ort: busspur, zweiteSpur,... Fahrzeug:abgefReifen, rost, ...
Umstände: dunkel, nebel,... Urteil:geld, gericht, punkte, ...
Verpflichtungen: lichtAn, parkverbot,...
Verstoß:unsicherFahrzeug, falschParken, ...

Verwendung der Topologie

- Inspektion der Regelmenge
- Entwurf der Regelmenge
- Fokussieren des Lernens
  - Prämissen einer gelernten Regel stammen aus demselben Topologieknoten wie die Konklusion oder aus einem Vorgängerknoten

RDT

LE: Grundfakten
LB: Grundfakten, die aus vorhandenen Regeln und Fakten mit Tiefenbeschränkung ableitbar sind
LH: Instanzen von Regelschemata, die bezüglich der Termsorten und Prädikatssorten kompatibel sind.

Benutzer gibt Akzeptanzkriterium ein.

- So muß eine Regel nicht 100%ig korrekt sein, wenn der Benutzer "Ausrutscher" in den Daten erwartet.
- So kann der Benutzer festlegen, ob er die closed-world assumption anwenden will oder lieber eine größere Vorhersagekraft der Regeln haben will.
- So kann der Benutzer festlegen, ob eine Regel vollständig sein muß.

RDT lernt alle nicht-redundanten Regeln, die dem Akzetanzkriterium entsprechen.

Hypothesenraum

Menge von Regelschemata RS

P (X) --> not (Q (X))
R (X,Y) & P (Y) --> Q (X) ...
sportscar/1 <car>, parked/2 <car, place>, ...

Menge von Prädikaten P

Prädikatssubstitution Σ

P/sportscar, Q/rusty
sportscar(X) --> not(rusty (X)),
parked(X,Y) & bus_lane (Y) --> parking_viol(X)

Menge der Hypothesen RS Σ

nach Allgemeinheit geordnete Regelschemata: RS $\geq_r$ **RS' gdw.** $\exists\sigma\ \exists\Sigma$ **RS** $\Sigma\sigma \subseteq$ **RS'**

Größe des Hypothesenraums: $(r\ p)^k$ **k: höchste Anzahl verschiedener Prädikatsvariablen in einem Regelschema**

RDT

Regelschemata

--> Q(X)
P(X) --> Q(X) R(X,Y)--> Q(X)
R(X,Y) & P(Y) --> Q(X)

Σ:
{R/r,P/p,Q/q}

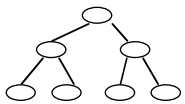
σ:
{X/a, Y/b}

Regeln

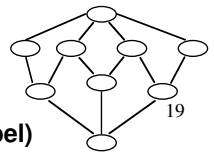
r(X,Y) & p(Y) --> q(X)

Fakten
r(a,b)
p (b)
q (a)

abstrahierter Regelgraph



Sortenverband



(Emde, Kietz, Klingspor, Morik, Wrobel)