Advanced Data Modeling

Minimal Models

Steffen Staab

People and Knowledge Networks

Overview

- Logic as query language.
- Grounding.
- Minimal Herbrand models.
- Completion.

Given:

- first-order formula $A[x_1, \ldots, x_n]$
- Herbrand interpretation I

- This first-order formula can be considered as a definition of a relation R_A on T^n_Σ as follows:
  - $(t_1, \ldots , t_n) \in R_A \quad := \quad I \models A[t_1, \ldots , t_n]$

Clauses as definitions

We say that a clause

$p(t_1, \ldots, t_m) \text{ :- } L_1, \ldots, L_n$

defines the relation symbol p.

Let C be a set of clauses and p be a relation symbol. We call the definition of p in C the set of all clauses in C that define p.

Principles of semantics

A deductive database is a **set of clauses**.

This set of clauses is regarded as a **collection of definitions** of relations.

The **semantics** defines the meaning of this definitions by associating with them an **interpretation**, or a class of interpretations.

Query answering is based on the semantics.

Two key assumptions

- the **unique name assumption**:
  each name denotes a unique object.

- the **closed world assumption**:
  - a negative statement $\neg A$ holds if the corresponding positive one A does not hold.

Both assumptions are not supported by (Tarskian) models for first-order logics (**why not?**) → One solution: minimal models

Minimal Herbrand Models

Let I be a Herbrand model of a set of formulas S.

We call I a minimal Herbrand model of S if it is minimal w.r.t. the subset relation, i.e. for every Herbrand model I' of S of the same signature we have $I' \supseteq I$.

I is called the least Herbrand model of S if for every Herbrand model I' of S of the same signature we have $I \subseteq I'$.

Does every set of formulas S have a least Herbrand model?

Does every set of formulas S have a least Herbrand model?

Counterexample for normal clauses:

```
person(a).
man(X) :- person(X), not woman(X).
woman(X) :- person(X), not man(X).
```

Does every set of formulas S have a least Herbrand model?

What about definite clauses?

Instance and variant

Let E, E´ be a pair of terms or formulas.

E´ is an **instance** of E, denoted E > E´, if there exists a substitution θ such that Eθ = E´.

ground instance: instance that is ground,

E´ is a **variant** of E if E´ is an instance of E and E is an instance of E´.

Examples

- P(x,a) is instance of P(x,y) because of P(x,y)[y|a]
- P(b,a) is a ground instance
- P(x,y) and P(u,v) are variants of each other, because of
  - [x|u, y|v] and
  - [u|x, v|y]

Grounding

- Let C be a set of clauses and Σ be any signature containing all symbols used in C. The **grounding of C w.r.t. Σ,** denoted **C^*** is the set of all ground instances of the signature Σ of clauses in C.

- Lemma. Let I be a Herbrand interpretation and C be a set of clauses. Then $I \vDash C$ if and only if $I \vDash C^*$.

General proof scheme

$$\begin{tikzcd}[column sep=huge, row sep=huge]
\text{First order formula} \arrow[r, "\text{calculus}"] \arrow[d, "\text{(all possible ways of) grounding}"'] & \text{First order Consequences} \\
\text{Propositional formula} \arrow[r, "\text{calculus}"] & \text{Propositional Consequences} \arrow[u, "\text{lifting}"']
\end{tikzcd}$$

- Proof

Logic with equality

- Additional atomic formulas **$s = t$**, where s, t are terms.
- Abbreviation: $x \neq y \; := \; \neg(x = y)$.
- Unlike other relations, the semantics of $s = t$ is **predefined** in all Herbrand interpretations:
  $I \models s = t$ if s coincides with t.

Example valid formulas

- $f(x_1, \ldots, x_n) = f(y_1, \ldots y_n) \rightarrow x_1 = y_1 \wedge \ldots \wedge x_n = y_n$
- $f(x_1, \ldots, x_n) \neq g(y_1, \ldots, y_n)$
- $f(x_1, \ldots, x_n) \neq c$
- $d \neq c$
- $A[t] \leftrightarrow \forall x(x = t \rightarrow A[x])$

Consider a definition of a relation *r*

$$r(\bar{t}_1) :- G_1$$

$$\ldots$$

$$r(\bar{t}_m) :- G_m$$

What is the meaning of this definition?

- Is there a largest Herbrand model?

Especially in the light of negation as failure?

Idea of completion

man(hans).

man(adam).

person(eva).

woman(X) :- person(X), not man(X).

Is eva a woman?

She might be a man and we just don‘t know!

Minimal model says that she is not a woman!

Trick: Completion:

man(X) $\leftrightarrow$ (X=hans $\vee$ X=adam).

woman(X) $\leftrightarrow$ X=eva.

I.e. hans and adam are the only men.

Idea of Completion

man(hans).

man(X) :- lovesBeer(X,Y).

Completion:

man(X) $\leftrightarrow$ X=hans $\vee$ ($\exists$ Y: X=U $\wedge$ lovesBeer(U,Y))

A man is a man only if he is hans or if he loves some brand of beer.

Replace every clause by an equivalent one such that the arguments of r are $x_1, \ldots, x_n$:

Given:

$r(t_1, \ldots, t_n)$:- G

Replace by:

$r(x_1, \ldots, x_n)$:- $x_1 = t_1 \wedge \ldots \wedge x_n = t_n \wedge G$

Completion. Step 2.

If there are variables $y_1, \ldots, y_k$ occurring in a body but not in the head, apply $\exists$ to these variables, i.e.,

Given

$r(x_1, \ldots, x_n)$:- G

Modify to

$r(x_1, \ldots, x_n)$:- $\exists\, y_1 \ldots \exists\, y_k$ G

Completion. Step 3.

If there are several definitions, replace them by one

Given

$r(x_1, \ldots ,x_n)$:- G_1

…

$r(x_1, \ldots , x_n)$:- G_m

Replace by

$r(x_1, \ldots , x_n)$:- $G_1 \vee \ldots \vee G_m$

Completion. Step 4.

Replace :- by $\leftrightarrow$:

Given

$r(x_1, \ldots, x_n)$:- $G_1 \vee \ldots \vee G_m$

Replace by

$r(x_1, \ldots, x_n) \leftrightarrow G_1 \vee \ldots \vee G_m$

The formula

$r(x_1, \ldots, x_n) \leftrightarrow G_1 \vee \ldots \vee G_m$

is called the **completed definition** of the original set of clauses.

Example

$r(u,v) \text{:-} p(u,1,z).$

$r(v,u) \text{:-} p(2,u,z).$

$r(x,y) \text{ :- } x=u \wedge y=v \wedge p(u,1,z).$

$r(x,y) \text{ :- } x=v \wedge y=u \wedge p(2,v,z).$

$r(x,y) \text{ :- } \exists z,u,v\ x=u \wedge y=v \wedge p(u,1,z).$

$r(x,y) \text{ :- } \exists z,u,v\ x=u \wedge y=v \wedge p(2,v,z).$

$r(x,y) \leftrightarrow (\exists z,u,v\ x=u \wedge y=v \wedge p(u,1,z)) \vee (\exists z,u,v\ x=u \wedge y=v \wedge p(2,v,z))$

Properties

- All steps **preserve Herbrand models**, except for the last one.
  - Why?
- Gives a **unique semantics to non-recursive definitions**
  - What about recursive definitions?
- Logic programming **semantics and first-order semantics coincide for definite programs**

Recursive definitions

odd(1).

even(f(X)) :- odd(X).

odd(f(X)) :- even(X).

Completion:

$\text{odd}(X) \leftrightarrow X{=}1 \vee (X{=}f(Y) \wedge \text{even}(Y)).$

$\text{even}(X) \leftrightarrow X{=}f(Y) \wedge \text{odd}(Y).$

Recursive definitions

person(adam).
person(eva).
woman(X) :- person(X), not man(X).
man(X) :- person(X), not woman(X).

Completion:
woman(X) $\leftrightarrow$ (Y=X $\wedge$ person(Y) $\wedge$ not man(Y)).
man(X) $\leftrightarrow$ (Y=X $\wedge$ person(Y) $\wedge$ not woman(Y)).

Semantics not unique in logic programming:
Models are
I={woman(adam),woman(eva)}
I={man(adam),man(eva)}
I={woman(adam),man(eva)}
I={man(adam),woman(eva)}

What is the semantics in first order logics?

Recursive definitions

person(adam).
person(eva).
woman(X) :- person(X), not man(X).
man(X) :- person(X), not woman(X).

Completion:
woman(X) $\leftrightarrow$ (Y=X $\wedge$ person(Y) $\wedge$ not man(Y)).
man(X) $\leftrightarrow$ (Y=X $\wedge$ person(Y) $\wedge$ not woman(Y)).

Semantics not unique in logic programming:
Models are (add {person(adam),person(eva)} to each)
I={woman(adam),woman(eva)}
I={man(adam),man(eva)}
I={woman(adam),man(eva)}
I={man(adam),woman(eva)}
What is the semantics in first order logics?
Models are: $woman^I=\{\}=man^I$

Simple characterization of completion

Let C be a definition of r, I be a Herbrand model of the corresponding completed definition, and $r(t_1, \ldots, t_n)$ be a ground atom.

Then

$I \models r(t_1, \ldots, t_n) \Leftrightarrow$

$\exists\ (r(t_1, \ldots, t_n) \text{ :- } L_1, \ldots, L_m) \in C^* \ (I \models L_1 \wedge \ldots \wedge L_n)$:

Immediate consequence operator

$T_C(I) :=$ { A | there exists (A :- G) $\in C^*$
such that I $\models$ G }

Fixpoint: an interpretation such that $T_C(I) = I$.

Definite clauses have the least model

Let C be a set of definite clauses.

Define

$I_0 := \{\}$

$I_{n+1} := T_C(I_n)$, for all $n \geq 0$,

$I\omega := \cup^{\omega}{}_{i=0}\ I_i$

Then I_ω is the least fixpoint of T_C and also the least Herbrand model of C.

Let C be a non-recursive database and K be an arbitrary interpretation.

Define

$I_0 :=$ **K**

$I_{n+1} := T_C(I_n)$, for all $n \geq 0$,

$I\omega := \cup^{\omega}_{i=0} I_i$

Then I_ω is the only fixpoint of T_C. Moreover, for some n we have $I_\omega = I_n$.

Non-recursive sets of clauses

- Let C be a set of clauses.

- Its **dependency graph** consists of pairs $p \rightarrow r$ such that p occurs in the body of a clause which defines r in C.

- A set of clauses is **non-recursive** if the dependency graph contains no cycles.

Non-recursive sets of clauses

Let C be a set of clauses.

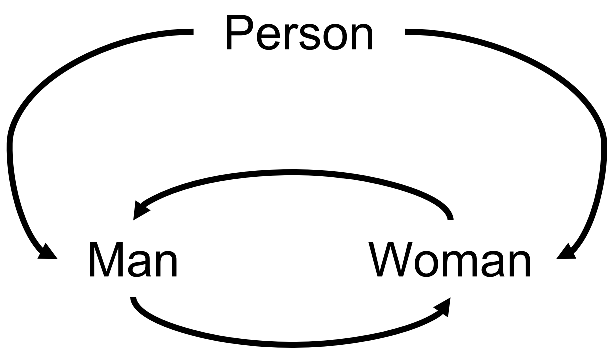
Its **dependency graph** consists of pairs $p \rightarrow r$ such that p occurs in the body of a clause which defines r in C.

A set of clauses is **non-recursive** if the dependency graph contains no cycles.

Person(a).

Woman(X) :- Person(X), Not Man(X).

Man(X) :- Person(X), Not Woman(X).



Non-recursive sets of clauses

Dependency graph of C consists of pairs $p \rightarrow r$ such that p occurs in the body of a clause which defines r in C.

C is non-recursive if the dependency graph contains no cycles.

A relation p depends on a relation q in C if there exists a path of length ≥ 1 from q to p in the dependency graph of C.

A set of clauses is non-recursive if and only if no relation depends on itself.