Advanced Data Modeling

Minimal Models

Steffen Staab
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 $T^n_S$ as follows:

- $(t_1, \ldots, t_n) \in R_A := I \models A[t_1, \ldots, t_n]$
We say that a clause
\[ p(t_1, \ldots, t_m) :- L_1, \ldots, L_n \]
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 \( p \).
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
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 $I' \subseteq I$.

$I$ 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?
Let $E, E'$ be a pair of terms or formulas.

$E'$ is an **instance** of $E$, denoted $E > E'$, if there exists a substitution $\mu$ such that $E\mu = E'$.

**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 $\mathcal{S}$ be any signature containing all symbols used in $C$. The **grounding of $C$ w.r.t. $\mathcal{S}$**, denoted $C^*$, is the set of all ground instances of the signature $\mathcal{S}$ of clauses in $C$.

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

First order formula \(\xrightarrow{\text{calculus}}\) First order Consequences

(all possible ways of) grounding

Propositional formula \(\xrightarrow{\text{calculus}}\) Propositional Consequences

lifting
• Proof
Logic with equality

- Additional atomic formulas \( s = t \), where \( s, t \) are terms.

- Abbreviation: \( x \neq y \) := \( \lnot(x = y) \).

- Unlike other relations, the semantics of \( s = t \) is **predefined** in all Herbrand interpretations: \( I^s = t \) if \( s \) coincides with \( t \).
Example valid formulas

- \( f(x_1, \ldots, x_n) = f(y_1, \ldots, y_n) \) ! \( x_1 = y_1 \land \ldots \land x_n = y_n \)
- \( f(x_1, \ldots, x_n) \neq g(y_1, \ldots, y_n) \)
- \( f(x_1, \ldots, x_n) \neq c \)
- \( d \neq c \)
- \( \forall x(x = t \rightarrow A[x]) \)
Consider a definition of a relation $r$

\[ r(\bar{t}_1) : \neg G_1 \]

\[ \ldots \]

\[ r(\bar{t}_m) : \neg G_m \]

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) $ (X=hans \lor X=adam).
  woman(X) $ X=eva.

I.e. hans and adam are the only men.
man(hans).
man(X) :- lovesBeer(X,Y).

Completion:
man(X) $ X=hans Ç (9 Y: X=U Æ lovesBeer(U,Y))

A man is a man only if he is hans or if he loves some brand of beer.
Completion. Step 1.

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:
$r(x_1, \ldots, x_n) :- x_1 = t_1 \land \ldots \land x_n = t_n \land G$
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
\[ r(x_1, \ldots, x_n) :- G_1 \]
\[ \ldots \]
\[ r(x_1, \ldots, x_n) :- G_m \]

Replace by
\[ r(x_1, \ldots, x_n) :- G_1 \bigcap \ldots \bigcap G_m \]
Completion. Step 4.

Replace :- by $:

Given
\[ r(x_1, \ldots, x_n) :- G_1 \lor \ldots \lor G_m \]

Replace by
\[ r(x_1, \ldots, x_n) \quad G_1 \lor \ldots \lor G_m \]

The formula
\[ r(x_1, \ldots, x_n) \quad G_1 \lor \ldots \lor G_m \]

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

\[ r(u,v) :- p(u,1,z). \]
\[ r(v,u) :- p(2,u,z). \]

\[ r(x,y) :- x = u \land y = v \land p(u,1,z). \]
\[ r(x,y) :- x = v \land y = u \land p(2,v,z). \]

\[ r(x,y) :- \exists z, u, v \quad x = u \land y = v \land p(u,1,z). \]
\[ r(x,y) :- \exists z, u, v \quad x = u \land y = v \land p(2,v,z). \]

\[ r(x,y) \supset (\exists z, u, v \quad x = u \land y = v \land p(u,1,z)) \land (\exists z, u, v \quad x = u \land y = v \land p(2,v,z)) \]
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

\[
\begin{align*}
\text{odd}(1).
\text{even}(f(X)) & : - \text{odd}(X).
\text{odd}(f(X)) & : - \text{even}(X).
\end{align*}
\]

Completion:

\[
\begin{align*}
\text{odd}(X) & \iff X=1 \lor (X=f(Y) \land \text{even}(Y)).
\text{even}(X) & \iff X=f(Y) \land \text{odd}(Y).
\end{align*}
\]
Recursive definitions

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

Completion:
woman(X) $ (Y=X \land \neg \text{person}(Y) \land \neg \text{man}(Y)).
man(X) $ (Y=X \land \neg \text{person}(Y) \land \neg \text{woman}(Y)).

Semantics not unique in logic programming:
Models are
I=\{\text{woman}(adam),\text{woman}(eva)\}
I=\{\text{man}(adam),\text{man}(eva)\}
I=\{\text{woman}(adam),\text{man}(eva)\}
I=\{\text{man}(adam),\text{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) $ (Y=X \land person(Y) \land not man(Y)).
man(X) $ (Y=X \land person(Y) \land 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: \( woman^I = \emptyset = man^I \)
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

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

$$9 (r(t_1, \ldots, t_n) :- L_1, \ldots, L_m) \models^* (I \models L_1 \land \ldots \land L_m):$$
Immediate consequence operator

\[ T_C(I) := \{ A \mid \text{there exists } (A :\, G) \text{ in } C^* \text{ such that } I \subseteq G \} \]

Fixpoint: an interpretation such that \( T_C(I) = I \).
Definite clauses have the least model

Let C be a set of definite clauses.

Define

\[ I_0 := \{\} \]
\[ I_{n+1} := T_C(I_n), \text{ for all } n \geq 0, \]
\[ I! := \bigcup_{i=0}^n I_i \]

Then \( I! \) is the least fixpoint of \( T_C \) and also the least Herbrand model of C.
Non-recursive databases

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

Define

\[ I_0 := K \]
\[ I_{n+1} := T_C(I_n), \text{ for all } n \geq 0, \]
\[ I! := \left[ i=0 \right] I_i \]

Then $I!$ is the only fixpoint of $T_C$. Moreover, for some $n$ we have $I! = I_n$. 
Non-recursive sets of clauses

- Let C be a set of clauses.

- Its dependency graph consists of pairs $p \nrightarrow r$ 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 $p ! r$ 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 $\geq 1$ 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.