AUTOMATED REASONING

Agostino Dovier

Università di Udine
CLPLAB

Udine, November 2016

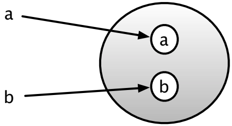
Semantics of a Logic Program

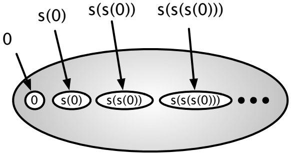
$P \models q(t_1, \ldots, t_n)$

An atom $q(t_1, \ldots, t_n)$ is a logical consequence of a program/theory P if $(\overline{t_1}, \ldots, \overline{t_n}) \in Q$ in all (interpretations that are) models of P.

Let us see how to compute it.

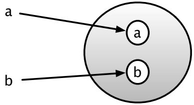
Let us consider the set of all ground terms that can be built with constant and function symbols in a program P.
This set can be used as a Universe for interpretations (the Herbrand Universe or H_P).
Ground terms are interpreted as themselves





Interpretations on the Herbrand Universe can be (or not) models (Herbrand models)

```
p(a).
q(b).
r(X) ← p(X).
```



Now, $\overline{a} = a$ and $\overline{b} = b$. Let us denote with P, Q, R the interpretations of the predicate symbols p, q, r.

1. $P = \{\overline{a}\}$, $Q = \{\overline{b}\}$, $R = \{\overline{a}, \overline{b}\}$ is a model.
2. $P = \{\overline{a}, \overline{b}\}$, $Q = \{\overline{b}\}$, $R = \{\overline{a}\}$ is NOT a model.

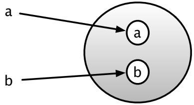
Herbrand interpretations and models can be represented uniquely by set of atoms:

1. $\{p(a), q(b), r(a), r(b)\}$ (model)
2. $\{p(a), p(b), q(b), r(a)\}$ (not a model)

Herbrand Models

Interpretations on the Herbrand Universe can be (or not) models (Herbrand models)

```
p(a).
q(b).
r(X) ← p(X).
```



Now, $\overline{a} = a$ and $\overline{b} = b$. Let us denote with P, Q, R the interpretations of the predicate symbols p, q, r.

1. $P = \{\overline{a}\}, Q = \{\overline{b}\}, R = \{\overline{a}, \overline{b}\}$ is a model.
2. $P = \{\overline{a}, \overline{b}\}, Q = \{\overline{b}\}, R = \{\overline{a}\}$ is NOT a model.

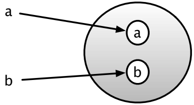
Herbrand interpretations and models can be represented uniquely by set of atoms:

1. $\{p(a), q(b), r(a), r(b)\}$ (model)
2. $\{p(a), p(b), q(b), r(a)\}$ (not a model)

Herbrand Models

Interpretations on the Herbrand Universe can be (or not) models (Herbrand models)

```
p(a).
q(b).
r(X) ← p(X).
```



Now, $\overline{a} = a$ and $\overline{b} = b$. Let us denote with P, Q, R the interpretations of the predicate symbols p, q, r.

1. $P = \{\overline{a}\}, Q = \{\overline{b}\}, R = \{\overline{a}, \overline{b}\}$ is a model.
2. $P = \{\overline{a}, \overline{b}\}, Q = \{\overline{b}\}, R = \{\overline{a}\}$ is NOT a model.

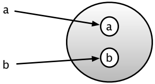
Herbrand interpretations and models can be represented uniquely by set of atoms:

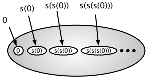
1. $\{p(a), q(b), r(a), r(b)\}$ (model)
2. $\{p(a), p(b), q(b), r(a)\}$ (not a model)

- Given a program P, the corresponding Herbrand Universe H_P is determined uniquely

```
p(a).        q(b).
r(X) ← p(X).

nat(0).
nat(s(X)) ← nat(X).
```





- Let $B_P = \{p(t_1, \ldots, t_n) : p$ is a predicate symbol in P, t_is are ground terms made with constant and function symbols in $P\}$ B_P is called the Herbrand base.
- Any subset of B_P uniquely determines an Herbrand Interpretation (some of them can be models)
- $(\wp(B_P), \subseteq)$ forms a complete lattice

A *clause* is a formula of the form $\forall \vec{X}(A_0 \vee \cdots \vee A_n)$ where A_is are positive or negative literals built on the variables $\vec{X}$.
Observe that $A_0 \vee \neg A_1 \vee \cdots \vee \neg A_n$ is $A_0 \leftarrow A_1 \wedge \cdots \wedge A_n$.
The notions given for “programs” in the previous slides apply to conjunction of clauses as well.
If T is a conjunction of clauses: H_T denotes the Herbrand Universe and B_T the Herbrand Base.

THEOREM

Let T be a conjunction of clauses. Then T has a model if and only if T has an Herbrand model.

THEOREM

Let T be a conjunction of clauses and $A \in B_T$ be a ground atom. Then $T \models A$ if and only if A is true in all Herbrand models of A.

The fundamental Theorem

A *clause* is a formula of the form $\forall \vec{X}(A_0 \vee \cdots \vee A_n)$ where A_is are positive or negative literals built on the variables $\vec{X}$.

Observe that $A_0 \vee \neg A_1 \vee \cdots \vee \neg A_n$ is $A_0 \leftarrow A_1 \wedge \cdots \wedge A_n$.

The notions given for "programs" in the previous slides apply to conjunction of clauses as well.

If T is a conjunction of clauses: H_T denotes the Herbrand Universe and B_T the Herbrand Base.

Theorem

Let T be a conjunction of clauses. Then T has a model if and only if T has an Herbrand model.

Theorem

Let T be a conjunction of clauses and $A \in B_T$ be a ground atom. Then $T \models A$ if and only if A is true in all Herbrand models of A.

Example

Let $T = p(a) \vee p(b)$. There are 4 Herbrand interpretations:

$$
\begin{array}{ccccc}
 & & \{p(a), p(b)\} & & \\
 & \nearrow & & \nwarrow & \\
\{p(a)\} & & & & \{p(b)\} \\
 & \nwarrow & & \nearrow & \\
 & & \emptyset & &
\end{array}
$$

3 of them are models. There is no A such that $T \models A$.

Example

Let T be
$(p(a) \vee p(b)) \wedge (\neg p(a) \vee p(b)) \wedge (p(a) \vee \neg p(b)) \wedge (\neg p(a) \vee \neg p(b))$.
Same 4 interpretations as above. No one of them is a model.

EXAMPLE

Let $T = p(a) \vee p(b)$. There are 4 Herbrand interpretations:

$$\{p(a), p(b)\}$$

$$\nearrow \qquad\qquad \nwarrow$$

$$\{p(a)\} \qquad\qquad\qquad \{p(b)\}$$

$$\nwarrow \qquad\qquad \nearrow$$

$$\emptyset$$

3 of them are models. There is no A such that $T \models A$.

EXAMPLE

Let T be
$(p(a) \vee p(b)) \wedge (\neg p(a) \vee p(b)) \wedge (p(a) \vee \neg p(b)) \wedge (\neg p(a) \vee \neg p(b))$.
Same 4 interpretations as above. No one of them is a model.

The fundamental Theorem (2)

Definite clauses have exactly one positive literals. The rule:

$$p(A) \leftarrow q(A, B), r(B).$$

is the clause

$$\forall A \forall B\, (p(A) \vee \neg q(A, B) \vee \neg r(B))$$

Programs are conjunctions of definite clauses.

Theorem

Let P be a (definite clause) program. Then P admits a (unique) minimum Herbrand model M_P (M_P is the semantics of P). (i.e., if I is a Herbrand model of P, then $M_P \subseteq I$).

Corollary

Let P be a (definite clause) program and $A \in B_P$ be a ground atom. Then $P \models A$ if and only if $A \in M_P$.

THE FUNDAMENTAL THEOREM (2)

Definite clauses have exactly one positive literals. The rule:

$$p(A) \leftarrow q(A, B), r(B).$$

is the clause

$$\forall A \forall B \, (p(A) \vee \neg q(A, B) \vee \neg r(B))$$

Programs are conjunctions of definite clauses.

THEOREM

Let P be a (definite clause) program. Then P admits a (unique) minimum Herbrand model M_P (M_P is the semantics of P). (i.e., if I is a Herbrand model of P, then $M_P \subseteq I$).

COROLLARY

Let P be a (definite clause) program and $A \in B_P$ be a ground atom. Then $P \models A$ if and only if $A \in M_P$.

COMPUTING M_P

1. Top-Down: using SLD resolution (Prolog).
   Query the SLD interpreter with the goal $:- A$
2. Bottom-Up: using the T_P (immediate consequence) operator (Datalog/ASP).

$$T_P(I) = \{a \,:\, a \leftarrow b_1, \ldots, b_n \in \text{ground}(P), \{b_1, \ldots, b_n\} \subseteq I\}$$

Example

Let P be the program:

```
r(a).
r(b).
p(a).
q(X) :- r(X),p(X).
```

Then:

$$
\begin{array}{rcl}
T_P(\emptyset) & = & \{r(a), r(b), p(a)\} \\
T_P(\{r(a), r(b), p(a)\}) & = & \{q(a), r(a), r(b), p(a)\} \\
T_P(\{q(a), r(a), r(b), p(a)\}) & = & \{q(a), r(a), r(b), p(a)\} \quad \Leftarrow \text{Fixpoint!}
\end{array}
$$

COMPUTING M_P

EXAMPLE

Let P be the program:

```
nat(0).
nat(s(X)) :- nat(X).
```

Then:

$$
\begin{array}{rcl}
T_P(\emptyset) & = & \{nat(0)\} \\
T_P(\{nat(0)\}) & = & \{nat(0), nat(s(0))\} \\
T_P(\{nat(0), nat(s(0))\}) & = & \{nat(0), nat(s(0)), nat(s(s(0)))\} \\
\vdots & \vdots & \vdots \\
T_P(\{nat(0), nat(s(0)), nat(s(s(0))), \dots\}) & = & \{nat(0), nat(s(0)), nat(s(s(0))), \dots\} \\
 & \Uparrow & \text{Fixpoint!}
\end{array}
$$

T_P is monotone: $I \subseteq J \rightarrow T_P(I) \subseteq T_P(J)$ and
upward continuous: if $I_0 \subseteq I_1 \subseteq I_2 \cdots$ then $T_P(\bigcup_{i\geq 0} I_i) = \bigcup_{i \geq 0} T_P(I_i)$

Let us define

$$\begin{array}{rcl} T_P \uparrow 0 & = & \emptyset \\ T_P \uparrow n+1 & = & T_P(T_P \uparrow n) \\ T_P \uparrow \omega & = & \bigcup_{n \geq 0} T_P \uparrow n \end{array}$$

THEOREM

If P is a definite clause program, then $T_P \uparrow \omega = M_P = T_P(T_P \uparrow \omega)$.

COMPUTING M_P

T_P is monotone: $I \subseteq J \rightarrow T_P(I) \subseteq T_P(J)$ and
upward continuous: if $I_0 \subseteq I_1 \subseteq I_2 \cdots$ then $T_P(\bigcup_{i \geq 0} I_i) = \bigcup_{i \geq 0} T_P(I_i)$

Let us define

$$\begin{array}{rcl} T_P \uparrow 0 & = & \emptyset \\ T_P \uparrow n+1 & = & T_P(T_P \uparrow n) \\ T_P \uparrow \omega & = & \bigcup_{n \geq 0} T_P \uparrow n \end{array}$$

THEOREM

If P is a definite clause program, then $T_P \uparrow \omega = M_P = T_P(T_P \uparrow \omega)$.

Let us consider P:

```
r(0) :- r(0).          p(0) :- q(X)
q(s(X)) :- q(X)
```

We have that: $T_P \uparrow \omega = \emptyset$. Let us compute instead

$$
\begin{array}{lclcl}
T_P \downarrow 0 & = & \mathcal{B}_P & = & \{r(0), r(s(0)), r(s(s(0))), r(s(s(s(0)))), \ldots \\
 & & & & p(0), p(s(0)), p(s(s(0))), p(s(s(s(0)))), \ldots \\
 & & & & q(0), q(s(0)), q(s(s(0)), q(s(s(s(0)))), \ldots \} \\
T_P \downarrow 1 & = & T_P(T_P \downarrow 0) & = & \{r(0), p(0), q(s(0)), q(s(s(0)), q(s(s(s(0)))), \ldots \} \\
T_P \downarrow 2 & = & T_P(T_P \downarrow 1) & = & \{r(0), p(0), q(s(s(0)), q(s(s(s(0)))), \ldots \} \\
T_P \downarrow 3 & = & T_P(T_P \downarrow 2) & = & \{r(0) p(0), q(s(s(s(0)))), \ldots \} \\
\vdots & \vdots & \vdots & \vdots & \vdots \\
T_P \downarrow \omega & = & \bigcap_{i \geq 0} T_P \downarrow i & = & \{r(0), p(0)\}
\end{array}
$$

The latter is not a fixpoint: $T_P(\{r(0), p(0)\}) = \{r(0)\}$. This is a fixpoint (the greatest fixpoint): a transfinite number of applications is needed.

SUMMARY

- The semantics of definite clause logic programming is based on the minimum Herbrand model M_P. It is the set of logical consequences. It is computable, i.e. it is a recursively enumerable set. You can compute it top down by SLD resolution or bottom up by $T_P \uparrow \omega$ (the least fixpoint). It is recursive (PTIME) if there are not function symbols in P. [J.W. Lloyd, Foundations of Logic Programming]
- Focusing on definite clause logic programming one can be interested in the greatest fixpoint of T_P for coinductive reasoning (Coinductive Logic Programming). This set is not computable (it is a productive set) but can be under approximated. [AD2015]

SUMMARY

- The semantics of definite clause logic programming is based on the minimum Herbrand model M_P. It is the set of logical consequences. It is computable, i.e. it is a recursively enumerable set. You can compute it top down by SLD resolution or bottom up by $T_P \uparrow \omega$ (the least fixpoint). It is recursive (PTIME) if there are not function symbols in P. [J.W. Lloyd, Foundations of Logic Programming]
- Focusing on definite clause logic programming one can be interested in the greatest fixpoint of T_P for coinductive reasoning (Coinductive Logic Programming). This set is not computable (it is a productive set) but can be under approximated. [AD2015]