

Logic Programs and Minimal Models

Marco Piastra

Logic Program

- An example of logic program:

$$\Pi \equiv \{ \{Human(x), \neg Philosopher(x)\}, \{Mortal(y), \neg Human(y)\}, \\ \{Philosopher(socrates)\}, \{Philosopher(plato)\}, \{Philosopher(aristotle)\} \}$$

$$\phi \equiv \exists x Mortal(x)$$

$$\neg\phi \equiv \neg\exists x Mortal(x)$$

$$\equiv \forall x \neg Mortal(x)$$

$$\equiv \{ \neg Mortal(x) \} \quad (\text{a goal, i.e. a Horn clause})$$

By applying resolution in an exhaustive way, we obtain:

$$\Sigma \equiv \{ [x/socrates], [x/plato], [x/aristotle] \}$$

Looks like a query on an *implicit* database ...

- **Answer Set**

It includes all complete substitutions of the variables in the *goal* corresponding to the closed branches (i.e. with an empty clause) in the SLD tree

Herbrand Universe, Herbrand Base

■ Herbrand **terms** and **atoms**

Given a signature Σ

A Herbrand **term** is a *ground term* (i.e. a term that contains no variables)

Examples:

$f(a), g(a,b), g(f(a),b), g(f(a),g(b,c)), g(f(a),g(f(b),c)), \dots$

A Herbrand **atom** is a *ground atom* (i.e. an atom that contains no variables)

Examples:

$P(f(a)), P(g(a,b)), Q(g(f(a),b), g(f(a),g(b,c))), \dots$

■ Herbrand **universe**

The set of all Herbrand *terms* from Σ

Example:

$U_H \equiv \{f(a), g(a,b), g(f(a),b), g(f(a),g(b,c)), g(f(a),g(f(b),c)), \dots\}$

■ Herbrand **base**

The set of all Herbrand *atoms* from Σ

Example:

$B_H \equiv \{P(f(a)), P(g(a,b)), Q(g(f(a),b), g(f(a),g(b,c))), \dots\}$

Herbrand models

■ Herbrand structure

A semantic structure $\langle \mathbf{U}_H, \Sigma, v_H \rangle$ such that

■ Herbrand interpretation v_H

For constants, $v_H(c) = c$

For ground terms, $v_H(t) = t$

For predicate symbols, $v_H \subseteq B_H$

i.e. a **subset** of the Herbrand base B_H

Example: $v_H \equiv \{P(a), P(f(b)), P(c), Q(a, g(b, c)), Q(b, c) \dots\}$

■ Herbrand model

$\varphi \in \text{Atom}(L_{PO}), \langle \mathbf{U}_H, \Sigma, v_H \rangle [s] \models \varphi$ iff $\varphi \in v_H$

$\varphi \in \text{Atom}(L_{PO}), \langle \mathbf{U}_H, \Sigma, v_H \rangle [s] \models \neg\varphi$ iff $\varphi \notin v_H$

$\langle \mathbf{U}_H, \Sigma, v_H \rangle [s] \models \neg\varphi$ iff $\langle \mathbf{U}_H, \Sigma, v_H \rangle [s] \not\models \varphi$

$\langle \mathbf{U}_H, \Sigma, v_H \rangle [s] \models \varphi \rightarrow \psi$ iff ($\langle \mathbf{U}_H, \Sigma, v_H \rangle [s] \not\models \varphi$ or $\langle \mathbf{U}_H, \Sigma, v_H \rangle [s] \models \psi$)

$\langle \mathbf{U}_H, \Sigma, v_H \rangle [s] \models \forall x \varphi$ iff for all $c \in \text{Cost}(L_{PO})$, $\langle \mathbf{U}_H, \Sigma, v_H \rangle [s](x:c) \models \varphi$

Horn clauses and Herbrand models

- **Herbrand Theorem**

Given a theory of universal sentences Φ ,
 $H(\Phi)$ has a model iff Φ has a model

- Corollary (for *Horn clauses*)

Given a set Φ of Horn clauses, the two following statements are equivalent:

- Φ is satisfiable
- Φ has an Herbrand model

This is not true in general: only if Φ is a set of Horn clauses

Clause di Horn e modelli di Herbrand

- Corollary to Herbrand theorem (for *Horn clauses*)

Given a set Φ of Horn clauses, the two following statements are equivalent:

- Φ is satisfiable
- Φ has an Herbrand model

This is not true in general: only if Φ is a set of Horn clauses

- Herbrand **minimal model**

The minimal model M_Φ for a set of Horn clauses Φ is:

$$M_\Phi \equiv \bigcap_{M_i} M_i \quad \text{where } M_i \text{ is a Herbrand model of } \Phi$$

- **Theorem**(van Emden e Kowalski, 1976)

Let Φ be a set of Horn clauses and φ a ground atom.

These three statements are equivalent:

- $\Phi \models \varphi$
- $\varphi \in M_\Phi$
- φ is derivable from Φ via resolution with refutation

Logic programming system and minimal model

- **Theorem** (Apt e van Emden, 1982)

Let Π be a **logical program** (i.e. a set of definite clauses).

The (finite) success set of Π with SLD-resolution (*fair*) coincides with M_Π

- A logic programming system (i.e. Prolog) can generate the *subset* of M_Π corresponding to a specific **goal**

A goal $\{\neg\alpha_1, \neg\alpha_2, \dots, \neg\alpha_m\}$ where the variables x_1, x_2, \dots, x_m occur is equivalent to the sentence $\forall x_1 \forall x_2 \dots \forall x_n (\neg\alpha_1 \vee \neg\alpha_2 \vee \dots \vee \neg\alpha_m)$ which is equivalent to $\neg\exists x_1 \exists x_2 \dots \exists x_n (\alpha_1 \wedge \alpha_2 \wedge \dots \wedge \alpha_m)$

A logic programming system can generate all possible **substitutions** $[x_1/t_1, x_2/t_2, \dots, x_n/t_n]$ such that $\Pi \cup \{\neg(\alpha_1 \wedge \alpha_2 \wedge \dots \wedge \alpha_m)[x_1/t_1, x_2/t_2, \dots, x_n/t_n]\}$ is *unsatisfiable*

(that implies $\Pi \models (\alpha_1 \wedge \alpha_2 \wedge \dots \wedge \alpha_m)[x_1/t_1, x_2/t_2, \dots, x_n/t_n]$)

(that implies $(\alpha_1 \wedge \alpha_2 \wedge \dots \wedge \alpha_m)[x_1/t_1, x_2/t_2, \dots, x_n/t_n] \in M_\Pi$)

Each goal act like a *filter*, i.e. defining the subset of M_Π

*NOTE: a logic programming system with a **fair** strategy can do so...*