UNIVERSIDADE DE SÃO PAULO

Herbrand Interpretation, Model and Least Model within the Framework of Logic Programming

MARIA DO CARMO NICOLETTI

MARIA CAROLINA MONARD

Nº 002

N O T A S

Instituto de Ciências Matemáticas de São Carlos

Instituto de Ciências Matemáticas de São Carlos

ISSN - 0103-2577

Herbrand Interpretation, Model and Least Model within the Framework of Logic Programming

MARIA DO CARMO NICOLETTI

MARIA CAROLINA MONARD

Nº 002

N O T A S D O I C M S C

Série Computação

São Carlos

jun. / 1993

Herbrand Interpretation, Model and Least Model within the Framework of Logic Programming

Maria do Carmo Nicoletti
Universidade Federal de São Carlos/DC
IFQSC/USP

Maria Carolina Monard [1]
Universidade de São Paulo/ILTC
Instituto de Ciências Matemáticas de São Carlos
Departamento de Ciências de Computação e Estatística

Abstract

Logic programming is a computational formalism which uses symbolic logic for representing problems and knowledge, together with logical inference, to search for a solution of those problems. Logic programming in its narrow sense is based on Horn clause and is often identified with Prolog.

Recently logic programming has been intensively used in Machine Learning — this area of research is known as *Inductive Logic Programming - ILP* — where the main idea is to combine logic programming with Machine Learning to induce first-order logic programs from examples. One of the problems of ILP is finding strategies to constrain the search space.
This work presents some concepts related to logic programming, namely Herbrand Universe, Model and Least Model that are basic for the understanding of logic programming itself, as well as for dealing with the intractability of search, in an ILP environment.

June 1993

[1] Work partially supported by National and State Research Council — CNPq and FAPESP.

Contents

1 Introduction

Recent inductive learning systems learn descriptions of relations in the form of logic programs and are named *Inductive Logic Programming — ILP — systems.* The task of empirical, single predicate learning in ILP can be formulated as follows [Lavrač 92] [Muggleton 91]:

Given:

- a set of training examples $\mathcal{E}$ described in a language $\mathcal{L}_{\mathcal{E}}$ and consisting of:
  - positive examples, $\mathcal{E}_+$
  - negative examples, $\mathcal{E}_-$

  of an unknown predicate p — target relation,
- a description language $\mathcal{L}_{\mathcal{C}}$, specifying syntatic restrictions on the definition of predicate p,
- background knowledge $\mathcal{B}$, described in language $\mathcal{L}_{\mathcal{B}}$, defining predicates q_i (other than p) which may be used in the definition of p and which provide additional information about the arguments of the examples of predicate p,
- a matching operator between $\mathcal{L}_{\mathcal{E}}$ and $\mathcal{L}_{\mathcal{C}}$ wrt[2] $\mathcal{L}_{\mathcal{B}}$ that determines whether an example is covered by a clause expressed in $\mathcal{L}_{\mathcal{C}}$.

Find:

- a definition $\mathcal{H}$ for p, expressed in $\mathcal{L}_{\mathcal{C}}$, such that:
  - $\mathcal{H}$ is complete, i.e., $\mathcal{B} \wedge \mathcal{H} \models \mathcal{E}_+$,
  - $\mathcal{H}$ is consistent , i.e., $\mathcal{B} \wedge \mathcal{H} \not\models \mathcal{E}_-$

  with respect to the examples.

Most of the concept-learning techniques available can be viewed as search-techniques [Mitchell 82]. As pointed out in [Muggleton 90], bottom-up algorithms based on inverting resolution have problems related to search strategies. In the framework of inverse resolution, clauses are constructed by progressively generalising examples with respect to a given background knowledge. Each step in this process inverts a step of a resolution derivation of the examples from the clause and background knowledge. Search problems arise mostly because:

- there may be many inverse resolvents at any stage,
- several inverse resolution steps may be necessary to construct the required clause.

[2] wrt — with respect to

If the background knowledge $\mathcal{B}$ consists of arbitrary clauses, the model $\mathcal{M}$ for $\mathcal{B}$ can be infinite, leading to possibly infinite rlgg clauses[3]. Among many strategies which aim to restrict the search space, are those related to restricting the background knowledge.

This work is concerned with the presentation and discussion of the necessary concepts and results of logic programming, which support a particular restricted form of background knowledge, known as h-easy model — to be investigated in future work — which is adopted by some ILP systems.

This work is organized as follows. Section 2 highlights how to prove unsatisfiability of a set of clauses. Section 3 and 4 provide the necessary concepts and notations for the theory to follow. Section 5 introduces some properties of partially ordered sets and monotonic mappings and their fixpoints, in order to use them in Section 6, where the concept of Least Herbrand Model is introduced and expressed in terms of fixpoint. Section 7 presents our conclusions and guidelines for future work.

2 Herbrand Proof Procedures

2.1 Initial Considerations

This section presents the necessary fundamentals for introducing a very important result to be used in refutation procedures, namely the Herbrand Theorem, which can be used as the basis of a computer program for showing that a formula is a logical consequence of a set of axioms. There are many ways of expressing and specializing the Herbrand Theorem.

The Herbrand's result ensures that it is possible to reduce the test for unsatisfiability of a given set of clauses S, to the problem of testing the existence of a finite and unsatisfiable set of formulas from the Propositional Calculus, systematicaly generated from S.

To establish that S is unsatisfiable using Herbrand Theorem consists in sistematicaly enumerate all finite sets of basic instances of clauses in S, seeking for unsatisfiability. It is important to note that sistematicaly enumerating the basic instances sets of clauses of S can be a time consuming and, most of the time, an extremely inefficient procedure.

2.2 Proving Unsatisfiability of a Set of Clauses

Let S be a given set of wffs and suppose that G is a logical consequence[4] of S (or that G logically follows from S).

From the definition of logical consequence, if I is a model[5] for S, I is a model for G

[3] rlgg — relative least general generalization. See reference [Nicoletti 93].

[4] A formula G is a *logical consequence* of a formula S iff for every interpretation I whenever S is true in I, G is also true in I.

[5] An interpretion I which gives the value **true** to a sentence S is said to *satisfy* that sentence and

(in other words, every interpretation satisfying S also satisfies G). None of the models for S can be a model for $\neg\ G$ and, therefore, no model can satisfy $S\ \wedge\ \{\neg\ G\}$; so $S\ \wedge\ \{\neg\ G\}$ is unsatisfiable. Thus, if G is a logical consequence of S, $S\ \wedge\ \{\neg\ G\}$ is unsatisfiable.

It is known that

> *A set of clauses is unsatisfiable (or inconsistent) iff it is false under all interpretations over all domains.*

Since it is not feasible to consider all interpretations over all domains, a particular domain H will be defined, such that S is unsatisfiable iff S is false under all interpretations over H. This particular domain is called the *Herbrand Universe of* S and the interpretations, *Herbrand Interpretations.*

All the Herbrand Interpretations share the same universe and in order to prove unsatisfiability all it is necessary is to consider instances made by substituing elements of the Herbrand Universe for variables. It will be shown that the class of Herbrand Interpretations can stand for all the others interpretations.

3 Herbrand Universe and Herbrand Base

Definition 3.1 *Let S be a set of clauses and H_0 be the set of constants appearing in S.*

- *if no constant appears in S, then let e.g. $H_0 = \{a\}$,*
- *for $i = 0, 1, \ldots$, let*

$$H_{i+1} = H_i \bigcup T$$

where T is the set of all terms of the form $f^n(t_1, \ldots, t_n)$ for all n-ary functions f^n occurring in S, where t_j, for $j = 1, \ldots, n$ are members of H_i.

Each H_i is called the *i-level constant set of* S and H_∞ (or simply H) is called the *Herbrand Universe of* S.

Thus the Herbrand Universe coincides with the set of ground terms which are generated from the constants and functional symbols appearing in S. Three situations may arise:

1. if S contains both constants and functions, then the Herbrand Universe will always be countably infinite.
2. if S contains only constants the Herbrand Universe will be the finite set of those constants.

---

such an interpretation I is called a *model* for S.

3. if S contains no constants, one constant, say a, is arbitrarily chosen. Depending on the presence or absence of functional symbols in S, the Herbrand Universe will be infinite or not, respectively.

Example 3.1 *Let*

$$S = \{\neg p(X,Y) \vee q(X,f(X)), \neg p(X,Y) \vee q(Y,g(Y))\}$$

Since there is no constant in S, let then $H_0 = \{a\}$. So

$$\begin{aligned}
H_1 &= H_0 \bigcup \{f(a), g(a)\} \\
H_2 &= H_1 \bigcup \{f(f(a)), f(g(a)), g(f(a)), g(g(a))\} \\
H_3 &= H_2 \bigcup \{f(f(f(a))), f(f(g(a))), f(g(f(a))), f(g(g(a))), g(f(f(a))), g(f(g(a))), \\
&\quad g(g(f(a))), g(g(g(a)))\} \\
&\cdots\cdots \\
H_\infty &= \{a, f(a), g(a), f(f(a)), f(g(a)), g(f(a)), g(g(a)), f(f(f(a))), f(f(g(a))), \\
&\quad f(g(f(a))), f(g(g(a))), g(f(f(a))), g(f(g(a))), g(g(f(a))), g(g(g(a))), \ldots\}
\end{aligned}$$

Example 3.2 *Let*

$$S = \{p(X) \vee q(X), r(Z), \neg\, s(Y) \vee t(W)\}$$

Since there is no constant in S, let $H_0 = \{a\}$.

Since there is no function symbol in S, $H_0 = H_1 = \ldots = \{a\}$.

Example 3.3 *Let*

$$S = \{\neg p(f(X), g(X), h(Y)) \vee q(X), r(Y)\}$$

Again, since there is no constant in S, let $H_0 = \{a\}$. So

$$\begin{aligned}
H_1 &= H_0 \bigcup \{f(a), g(a), h(a)\} \\
H_2 &= H_1 \bigcup \{f(f(a)), f(g(a)), f(h(a)), g(f(a)), g(g(a)), g(h(a)), \\
&\quad h(f(a)), h(g(a)), h(h(a))\} \\
H_3 &= H_2 \bigcup \{f(f(f(a))), f(f(g(a))), f(f(h(a))), g(f(f(a))), g(f(g(a))), g(f(h(a))), \\
&\quad h(f(f(a))), h(f(g(a))), h(f(h(a))), f(g(f(a))), f(g(g(a))), f(g(h(a))), \\
&\quad g(g(f(a))), g(g(g(a))), g(g(h(a))), h(g(f(a))), h(g(g(a))), h(g(h(a))), \\
&\quad f(h(f(a))), f(h(g(a))), f(h(h(a))), g(h(f(a))), g(h(g(a))), g(h(h(a))), \\
&\quad h(h(f(a))), h(h(g(a))), h(h(h(a)))\} \\
&\cdots\cdots \\
H_\infty &= \{a, f(a), g(a), h(a), f(f(a)), f(g(a)), f(h(a)), \ldots\}
\end{aligned}$$

Example 3.4 *Let*

$$S = \{p(a, b, f(X), g(Y), c)\}$$

Since there are three constants — a, b and c — in S, then

$$
\begin{aligned}
H_0 &= \{a, b, c\} \\
H_1 &= H_0 \bigcup \{f(a), g(a), f(b), g(b), f(c), g(c)\} \\
H_2 &= H_1 \bigcup \{f(f(a)), f(g(a)), f(f(b)), f(g(b)), f(f(c)), f(g(c)), \\
&\quad g(f(a)), g(g(a)), g(f(b)), g(g(b)), g(f(c)), g(g(c))\} \\
&\cdots\cdots \\
H_\infty &= \{a, b, c, f(a), g(a), f(b), g(b), f(c), g(c), f(f(a)), \ldots\}
\end{aligned}
$$

Definition 3.2 *Let*

S be a set of clauses and
H be the Herbrand Universe of S

The set of ground atoms of the form $p^n(t_1, \ldots, t_n)$ for all n-ary elements of H is called the Herbrand Base or the atom set of S, and it is noted by B_S.

Example 3.5 *Let*

$$S = \{p(X) \vee q(X), r(f(Y))\}$$

then

$$B_S = \{p(a), q(a), r(a), p(f(a)), q(f(a)), r(f(a)), \ldots\}$$

Example 3.6 *Let*

$$S = \{p(X), \neg p(X) \vee q(f(X)), \neg q(f(a))\}$$

then

$$B_S = \{p(a), q(a), p(f(a)), q(f(a)), \ldots\}$$

Example 3.7 *Let*

$$S = \{\neg p(X, a, Y) \vee \neg p(b, X, Z) \vee q(Y, Z)\}$$

Since S has no functional symbol, the Herbrand Universe of S is $H_\infty = \{a, b\}$.

The Herbrand Base of S is:

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

Definition 3.3 *A ground instance of a clause C of a set S of clauses is a clause obtained by replacing variables in C by members of the Herbrand Universe of S.*

Example 3.8 *Let*

$$S = \{p(X), q(f(Y)) \vee r(Y)\}$$
$$H_\infty = \{a, f(a), f(f(a)), \ldots\}$$

Some of the ground instances for the two clauses — C_1 and C_2 — in S are shown in the next table.

$C_1 : p(X)$	$C_2 : q(f(Y)) \vee r(Y)$
$p(a)$	$q(f(a)) \vee r(a)$
$p(f(a))$	$q(f(f(a))) \vee r(f(a))$
$p(f(f(a)))$	$q(f(f(f(a)))) \vee r(f(f(a)))$
...	...

Definition 3.4 *Let S be a set of clauses. A ground instantiation of S is the set of all the ground instances of all its clauses and is denoted by G_S.*

Example 3.9 *Let P be the Prolog program*[6]

f(a,X):- f(X,b).
f(c,b).

Taking $H = \{a, b, c\}$, the ground instantiation G_P[7] of P is the ground program:

f(a,a):- f(a,b).
f(a,b):- f(b,b).
f(a,c):- f(c,b).
f(c,b).

Note that G_P is essentially propositional.

Next it will be defined a special interpretation over the Herbrand Universe of S called Herbrand Interpretation of S.

4 Herbrand Interpretation and Herbrand Model

Definition 4.1 *Let*

S *be a set of clauses,*
H *be the Herbrand Universe of S,*
I *be an interpretation of S over H.*

I is said to be a Herbrand interpretation of S if it satisfies the following conditions:

[6] Which can be rewritten as the set of clauses $\{\neg f(X,b) \vee f(a,X), f(c,b)\}$.

[7] Which can be rewritten as the set $G_S = \{\neg f(a,b) \vee f(a,a), \neg f(b,b) \vee f(a,b), \neg f(c,b) \vee f(a,c), f(c,b)\}$.

1. *I maps all constants in S to themselves.*

2. *let f be an n-ary function symbol and* h_1, h_2,...,h_n *be elements of H. In I, f is assigned a function that maps* $(h_1, h_2, \ldots, h_n)$ *— an element of* H^n *— to* $f(h_1, h_2, \ldots, h_n)$ *— an element of H.*

There is no restriction on the assignment to each n-ary predicate symbol in S, so that different Herbrand Interpretations arise by taking different such assignments. Since for Herbrand Interpretations the assignment of constants and functions is fixed, it is possible to identify a Herbrand Interpretation with a subset of the Herbrand Base. So, from now on,

> *a Herbrand Interpretation I of a set of clauses S is any subset of* B_S*, the Herbrand Base of S.*

understanding that, under this interpretation I, any atom q occuring in B_S is assigned *true* if $q \in I$ but is otherwise assigned *false.*

In order to evaluate a given set of clauses with relation to a given Herbrand Interpretation, the following rules should be applied [Van Emden 76]:

1. a ground atomic formula A is *true* in a Herbrand Interpretation I iff $A \in I$.

2. a ground negative literal $\neg A$ is *true* in I iff $A \notin I$.

3. a ground clause $L_1 \vee L_2 \vee \ldots \vee L_n$ is *true* in I iff at least one literal L_i is *true* in I.

4. in general a clause C is *true* in I iff every ground instance $C\sigma$ of C is *true* in I ($C\sigma$ is obtained by replacing every occurrence of a variable in C by a term in H. Different occurrences of the same variable are replaced by the same term).

5. a set of clauses S is *true* in I iff each clause in S is *true* in I.

A literal, clause or set of clauses is *false* in I iff it is not *true.*

A *Herbrand Interpretation* is just a a free assignment of truth-values (*true* or *false*) to all the atoms in B_S. Such an interpretation must therefore also assign either *true* of *false* to all the atoms in G_P; thus the truth-values of these atoms determine truth-values for all the clauses in G_P.

Definition 4.2 *Let*

S	*be a set of clauses,*
H	*be the Herbrand Universe of S,*
I	*be a Herbrand Interpretation of S over H.*

If S is true in I, then I is a Herbrand Model of S.

Example 4.1 *Let*

$$S = \{p(X) \vee \neg q(X)\}$$

be the set of clauses.

Since S has $B_S = \{p(a),q(a)\}$, *the possible* $2^{|B_S|}$ *Herbrand Interpretations for S are:*

$$I_0 : \emptyset$$
$$I_1 : \{p(a)\}$$
$$I_2 : \{q(a)\}$$
$$I_3 : \{p(a),q(a)\}$$

According to the evaluation rules stated before, the interpretations I_0, I_1 *and* I_3 *are Herbrand Models of S and* I_2 *is simply a Herbrand Interpretation of S.*

Example 4.2 *Let*

$$S = \{p(X) \vee \neg q(X), m(Y) \vee \neg g(a), q(Z)\}$$

be a set of clauses and

$$B_S = \{p(a), q(a), m(a), g(a)\}$$

Next table shows all the 16 possible Herbrand Interpretations for the set of clauses S, and indicates with a $\star$ *which among them are Herbrand Models.*

interpretation	model	interpretation	model
$\emptyset$		$\{q(a), m(a)\}$	
$\{p(a)\}$		$\{q(a), g(a)\}$	
$\{q(a)\}$		$\{m(a), g(a)\}$	
$\{m(a)\}$		$\{p(a), q(a), m(a)\}$	$\star$
$\{g(a)\}$		$\{p(a), q(a), g(a)\}$	
$\{p(a), q(a)\}$	$\star$	$\{p(a), m(a), g(a)\}$	
$\{p(a), m(a)\}$		$\{q(a), m(a), g(a)\}$	
$\{p(a), g(a)\}$		$\{p(a), q(a), m(a), g(a)\}$	$\star$

Example 4.3 *([Hogger 90]) Let a clause set P be:*

likes(chris,X) **if** *likes(X,logic)*
likes(bob,logic)

The Herbrand Universe is the set {*chris, bob, logic*} and the ground instantiation of P, namely G_P is

likes(chris,chris) **if** *likes(chris,logic)*
likes(chris,bob) **if** *likes(bob,logic)*
likes(chris,logic) **if** *likes(logic,logic)*
likes(bob,logic)

which can be rewritten, for facility, as:

CC **if** CL
CB **if** BL
CL **if** LL
BL

The Herbrand Base is $B_P = \{CC, CL, CB, LC, LL, LB, BC, BL, BB\}$

The set $\{BL, CB\} \subset B_P$ is thus an Herbrand Interpretation. For this interpretation G_P is reproduced below with the associated truth-values written in place of its atoms:

false **if** *false*
true **if** *true*
false **if** *false*
true

With this assignment, all the clauses of G_P are true. This interpretation is therefore an Herbrand Model for G_P and hence an Herbrand Model for P.

Proposition 4.1 *Let S be a set of clauses and suppose S has a model. Then S has a Herbrand Model.*

Proof: Let I be an interpretation of S. The Herbrand Interpretation I' of S can be defined as follows:

$$I' = \{p(t_1, \ldots, t_n) \in B_S : p(t_1, \ldots, t_n) \text{ is true wrt } I\}$$

It is straightforward to show that if I is a model, I' is also a model. The next two examples show the construction of the Herbrand Models corresponding to models.

Example 4.4 *Let*

$$S = \{\neg p(X) \vee q(f(X), a)\}$$

be the clausal form of the formula $F : \forall(X).(p(X) \rightarrow q(f(X), a))$ and let I be the following model[8]:

- *Domain:*

[8] Note that it is a model since $\{\neg p(1) \vee q(f(1), 1)\}$ and $\{\neg p(2) \vee q(f(2), 1)\}$ are *true*.

{1,2}

- Assignment for constant:

a
1

- Assignment for function symbols:

$f(1)$	$f(2)$
2	1

- Assignment for predicate symbols:

$p(1)$	$p(2)$	$q(1,1)$	$q(1,2)$	$q(2,1)$	$q(2,2)$
$false$	$true$	$true$	$true$	$false$	$true$

For the definition of the corresponding Herbrand Model, it is necessary:

- to find the Herbrand Base B_S of S and
- to evaluate each member of B_S, using the given interpretation I

in order to select as elements of the Herbrand Model those elements of the B_S which are evaluated as *true* by the model I.

For the set S considered

$$B_S = \{p(a), q(a,a), p(f(a)), q(a,f(a)), q(f(a),a), q(f(a),f(a)), \ldots\}$$

and the evaluation of each element of B_S using the given model I is

$$\begin{array}{lcll} p(a) & = & p(1) & = & false \\ q(a,a) & = & q(1,1) & = & true \\ p(f(a)) & = & p(2) & = & true \\ q(a,f(a)) & = & q(1,2) & = & true \\ q(f(a),a) & = & q(2,1) & = & false \\ \ldots\ldots & & & & \end{array}$$

Therefore the Hebrand model I' corresponding to I is

$$I' = \{q(a,a), f(p(a)), q(a,f(a)), \ldots\}$$

In case there is no constant in S, the arbitrarily chosen constant a used to initiate the Herbrand Universe of S, can be mapped into any element of the domain D. In such a case, when there is more than one element in D, then there is more than one Herbrand Model corresponding to I, as shown in the next example.

Example 4.5 *Let*

$$S = \{\neg p(X) \vee q(f(X), Y)\}$$

with the interpretation I as the one shown in the previous example, except that there is no assignment to constant.

The Herbrand Base of S is the same as the previous example and the evaluation of each element of B_S using the given model I is:

$$\begin{array}{lllll} \textit{for } a = 1 & p(a) & = & p(1) & = \ \textit{false} \\ & q(a,a) & = & q(1,1) & = \ \textit{true} \\ & p(f(a)) & = & p(2) & = \ \textit{true} \end{array}$$

.....

Therefore $I' = \{p(a), q(a,a), p(f(a)), \ldots\}$

$$\begin{array}{lllll} \textit{for } a = 2 & p(a) & = & p(1) & = \ \textit{false} \\ & q(a,a) & = & q(1,1) & = \ \textit{true} \\ & p(f(a)) & = & p(2) & = \ \textit{true} \\ & q(a,f(a)) & = & q(1,2) & = \ \textit{true} \\ & q(f(a),a) & = & q(2,1) & = \ \textit{false} \end{array}$$

.....

Therefore, $I'' = \{q(a,a), f(p(a)), q(a,f(a)), \ldots\}$

To test whether a set S of clauses is unsatisfiable, it is only necessary to consider interpretations over the Herbrand Universe S. If S is false under all interpretations over the Herbrand Universe of S, then S is unsatisfiable.

Since there are usually many, possibly an infinite number of these interpretations, they should be organized in some systematic way. This can be done by using a semantic tree [Chang 73].

There are many ways of expressing and specializing the Herbrand Theorem, but perhaps the specialization stated in Theorem 4.1 is the one best suited to the context of logic programming.

Theorem 4.1 *A contradiction can be derived from a set of clauses $S \bigcup \{\neg G\}$ iff it can be derived from some finite subset of the ground instantiation $G_{(S \bigcup \{\neg G\})}$.*

The word *finite* ensures the existence of an effective procedure (algorithm) by which one can demonstrate a contradiction and hence, indirectly, that G is a consequence of

S. In other works, this theorem has been restated in terms of unsatisfiability instead of derivability and becomes:

Theorem 4.2 *A set of clauses $S \bigcup \{\neg G\}$ is unsatisfiable iff some finite subset of $G_{(S \bigcup \{\neg G\})}$ is unsatisfiable.*

Example 4.6 *Let*

$$S = \{p(X), \neg p(f(a))\}$$

S is unsatisfiable. Hence, by Herbrand Theorem, there is a finite unsatisfiable set S' of ground instances of clauses in S. It can be seen that one of these sets is

$$S' = \{p(f(a)), \neg p(f(a))\}$$

Example 4.7 *Let*

$$S = \{\neg p(X) \vee q(f(X), X), p(g(b)), \neg(q(Y, Z))\}$$

S is unsatisfiable. One of the unsatisfiable sets of ground instances of clauses in S is

$$S' = \{\neg p(g(b)) \vee q(f(g(b)), g(b)), p(g(b)), \neg q(f(g(b)), g(b))\}$$

It is important to notice that generally it is not very easy to find by hand a finite unsatisfiable set S' of ground instances of clauses in S. One way to find such a set is to generate a closed semantic tree T' for S. The set S' of all the ground instances falsified at all the failure nodes of T' is such a desired set. A detailed description of this procedure can be seen in [Chang 73].

One of the most important aspects of Herbrand Interpretations, is that it is not longer necessary to consider the universe of every possible interpretation and use the objects of each universe in all possible ways, since Herbrand Interpretations can stand for all the others.

All the Herbrand Interpretations share the same universe — the Herbrand Universe. This means that it is only necessary to consider instances made by substituting elements of the Herbrand Universe for variables.

With a little care it is possible to be sure that none of the possible conjunctions of instances of clauses is omitted from the testing. In this case, the testing eventually will find out if a goal G is a logical consequence of a set of clauses S. However, the search through possible instances could go on forever. In fact the following cases can arise:

1. G is a logical consequence of S. In this case the search will eventually terminate with success, i.e. a contradiction will be found.

2. G is not a logical consequence of S. There are two possible subcases:

   - all possible intances have been generated without having found a contradiction. In this case G is not a logical consequence of S.
   - instances are generated forever and a contradiction is never found. In this case there is no certainty about G being a logical consequence of S or not — could be either case 1. or 2.

5 Monotonic Mappings and their Fixpoints

This section introduces some of the general properties of partially ordered sets as well as some results related to monotonic mappings and their fixpoints. These concepts are important for the characterization of the Least Herbrand Model, considered in the next section.

The relationships between the various Herbrand Models of a definite program can be investigated in the framework of the theory of lattices — a particular kind of partially ordered set. It will be seen, as well, that associated with every program there is a monotonic mapping which plays a very important role in the theory of logic programming.

The proofs of some results are omitted; they can be found in [Sanderson 73], [Lloyd 84] and [Hogger 90].

Definition 5.1 *Let S be a set. A relation R on S is a subset of $S \times S$. The fact that $(x, y) \in R$ will be denoted by xRy.*

Definition 5.2 *A relation R on a set S is a partial order if the following conditions are satisfied:*

1. *reflexivity — xRx, for all $x \in S$*
2. *antisymmetry — xRy and $yRx \rightarrow x = y$, for all $x, y \in S$*
3. *transitivity — xRy and $yRz \rightarrow xRz$, for all $x, y, z \in S$*

The relation R will be noted by $\leq$.

The structure $[S, \leq]$[9] is called a *partially-ordered set.* Figures 1, 2 and 3 show examples of partially ordered sets. It is important to notice that there is no requirement that every pair of elements in a partially-ordered set be related by $\leq$. If it is the case that $x \not\leq y$ and $y \not\leq x$ then x and y are said to be *incomparable.*

Unless otherwise stated, S denotes a partially ordered set. Small letters are used to denote elements and capital letters subsets of S. In order to denote functions small letters are used as well.

[9] Some of the figures, which show partially ordered sets have the extra arrows that would be induced by the reflexivity and transitivity of $\leq$ been omitted, in order to reduce clutter.

Figure 1: The integers ordered by the relation less_equal

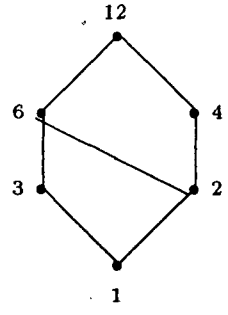

Figure 2: The factors of 12 ordered by divisibility

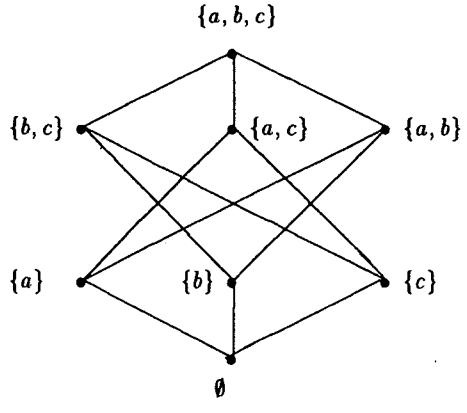

Figure 3: Subsets of the set $\{a, b, c\}$ ordered by $\subseteq$

The relation $\leq$ orders the members s_i of S into a collection of zero or more (directed) paths. For example, one of the path shown in Figure 2 signifies that $1 \leq 2$, $2 \leq 4$ and $4 \leq 12$.

The relation between elements and sets will be noted as follows:

$$a \leq X \text{ means } a \leq x \ \forall x \ x \in X$$

$$X \leq a \text{ means } x \leq a \ \forall \text{x} \ x \in X$$

Definition 5.3 *Let S be a set with partial order $\leq$ and X any subset of S*

1. *$a \in S$ is an upper bound of X if $x \leq a$, for all $x \in X$.*
2. *$b \in S$ is a lower bound of X if $b \leq x$, for all $x \in X$.*
3. *$a \in X$ is the greatest element of X if $x \leq a$ for all $x \in X$.*
4. *$b \in X$ is the least element of X if $b \leq x$ for all $x \in X$.*
5. *$a \in S$ is the least upper bound of X — $a = lub(X)$ — if a is an upper bound of X and, for all upper bounds a' of X, $a \leq a'$. That is to say that a is the least element of the set of all upper bounds for X.*
6. *$b \in S$ is the greatest lower bound of X — $b = glb(X)$ — if b is an lower bound of X and, for all lower bounds b' of X, $b' \leq b$. That is to say that b is the greatest element of the set of all lower bounds for X.*
7. *$a \in X$ is a maximal element of X if $a \leq x$ for no $x \in X, x \neq a$.*
8. *$b \in X$ is a minimal element of X if $x \leq b$ for no $x \in X, x \neq b$.*

Sometimes a set has no upper bounds, while others have several (or even infinitely many). It is important to notice that not all $X \subseteq S$ need to have least upper bound.

Example 5.1 *Let S be the partially ordered set shown in Figure 3 and let $X = \{\{a,c\},\{a,b\}\}$ be its subset. The set $\{a\} \in S$ is a lower bound of X, as well as the empty set $\emptyset$.*

Theorem 5.1 *For any $X \subseteq S$ the lub of S is unique if it exists. Similarly, the glb of S is unique if it exists.*

Example 5.2 *Considering the partially ordered set shown in Figure 3 and its subset $X = \{\{a\},\{c\},\{a,c\}\}$, it can be seen that $lub(X) = \{a,c\}$. The set $\{a,b,c\}$ is an upper bound of X.*

Example 5.3 *Consider the partially ordered set shown in Figure 4.*

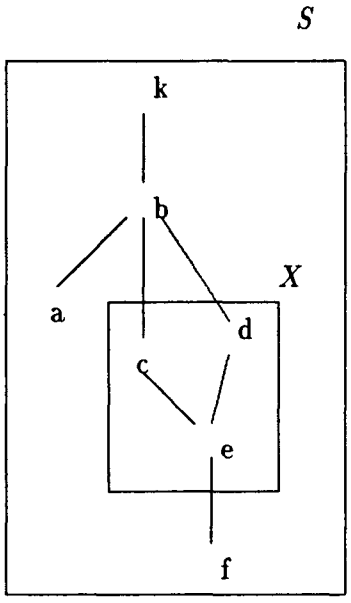

Figure 4: A lattice

It can be seen that:

	S	X
upper bound	k	k,b
lower bound	-	f,e
greatest element	k	-
least element	-	e
least upper bound	k	b
greatest lower bound	-	e
minimal element	a,f	e
maximal element	k	c,d

There are kinds of partially ordered sets which are of special interest in logic programming. Suppose that $[S, \leq]$ is such that every subset X of S has a least upper bound. In this case

1. since $S \subseteq S$, it follows that S has a lub which, being a member of S, must be the (unique) greatest element of S;

2. since $\emptyset \subseteq S$, it follows that $\emptyset$ has a lub and, because all the upper bounds of $\emptyset$ are exactly all the members of S, the least of them must be the (unique) least element of S.

The existence of least and greatest elements determines that every member of S must lie on some path in $[S, \leq]$, which begins at the greatest element of S and ends at the

least element of S. This structure is called a complete lattice and is defined formally next.

Definition 5.4 *A partially ordered set S is a complete lattice*[10] *if $lub(X)$ and $glb(X)$ exist, for every subset X of S. In a complete lattice S, the symbols:*

- $\top$ *denote the top element $lub(S)$,*
- $\bot$ *denote the bottom element $glb(S)$*

In a complete lattice the elements of most interest are the top and the bottom. Every other element lies on some path directed from the top element down to the bottom element.

Example 5.4 *The integers shown in Figure 1, pg. 14, form a lattice. If the elements $+\infty$ and $-\infty$ are adjoined respectively at its top and bottom, it becomes a complete lattice. The lattice shown in Figure 2, pg. 14, is complete as well.*

Example 5.5 *Let 2^S denote the powerset of the set S. The set 2^S under $\subseteq$ is a complete lattice. In fact, the lub of a collection of subsets of S is their union and the glb is their intersection. The top element is S and the bottom element is $\emptyset$. Figure 3, pg. 14, shows a complete lattice where $S = \{a, b, c\}$.*

Example 5.6 *Figure 5 gives an idea of what a complete lattice might look like (the arrows induced by reflexivity and transitivity are omitted).*

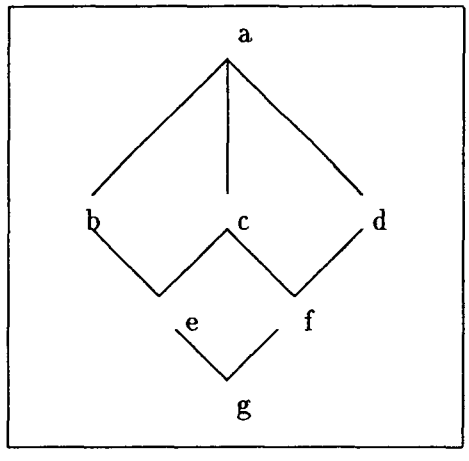

Figure 5: Complete lattice

[10] A complete lattice is more restricted than a lattice. In a lattice the $lub(X)$ and the $glb(X)$ exist only for every non-empty finite subset $X \subseteq S$. That is, not all subsets of S need to have bounds.

A lattice can be thought as an ordering of the value space according to some *information content* of the elements. If functions are to be considered, it is reasonable to assume that if more information about the argument of the function is known, more information about its result should be known. That is to say, functions must be monotonic.

Definition 5.5 *Let S be a complete lattice and $T : S \Rightarrow S$ be a mapping. The mapping T is monotonic if $T(x) \leq T(y)$ whenever $x \leq y$.*

Definition 5.6 *Let S be a complete lattice and $X \subseteq S$. The set X is directed if every finite subset of X has an upper bound in X, i.e. the upper bound of X is also an element of X.*

Example 5.7 *Next table shows some examples of subsets $X \subseteq S$ which are directed and some which are not, considering the lattice shown in Figure 3, pg. 14.*

Directed	Upper Bound	Not Directed	Upper Bound
$\{\{a\},\{a,c\}\}$	$\{a,c\}^*\{a,b,c\}$	$\{\{a\},\{b,c\},\{b,c\}\}$	$\{a,b,c\}$
$\{\{a\},\{b\},\{a,b\}\}$	$\{a,b\}^*\{a,b,c\}$	$\{\{b\},\{c\}\}$	$\{b,c\},\{a,b,c\}$
		$\{\{a\},\{b\},\{c\}\}$	$\{a,b,c\}$

$*$ note that this upper bound belongs to X.

Definition 5.7 *Let S be a complete lattice and $T : S \Rightarrow S$ be a mapping. The mapping T is continuous if $T(lub(X)) = lub(T(X))$, for every directed subset X of S.*

As commented in [Hogger 90], monotonicity can be regarded as a structure-preserving property: the structure of a partially-ordered set is preserved in the set obtained by applying a given monotonic function to its members.

Continuity can be viewed as a limit-preserving property: if a directed subset S has a least upper bound $lub(S)$, then the set obtained by applying a continuous function f to the members of S has $f(lub(S))$ as its least upper bound.

Definition 5.8 *Let S be a complete lattice and $T : S \Rightarrow S$ be a mapping.*

The element $a \in S$ is the least fixpoint of T if it is a fixpoint ($T(a) = a$) and for all fixpoints b of T, $a \leq b$.

The element $a \in S$ is the greatest fixpoint of T if it is a fixpoint ($T(a) = a$) and for all fixpoints b of T, $b \leq a$.

Definition 5.9 *Let S be a complete lattice and $T : S \Rightarrow S$ be a mapping. An element $u \in S$ is a pre-fixpoint of T iff $T(u) \leq u$.*

Theorem 5.2 *Knaster - Tarski Theorem*[11]

Let S be a complete lattice and $T : S \Rightarrow S$ be monotonic. Then T has a least fixpoint — $lfp(T)$ — which is also the least pre-fixpoint in S.

This theorem says about the existence of a least fixpoint but does not show how to construct it. The Kleene's First Recursion Theorem provides a constructive definition of least fixpoint.

Proposition 5.1 *Let S be a complete lattice and $T : S \Rightarrow S$ be monotonic. Suppose $a \in S$ and $a \leq T(a)$. Then there exists a fixpoint a' of T such that $a \leq a'$. Similarly, if $b \in S$ and $T(b) \leq b$, then there exists a fixpoint b' of T such that $b' \leq b$.*

Since it will be necessary unions and intersections of powers of T and since the exponents in those powers may have to go beyond the natural numbers, the following ordinal powers of T will be defined:

$$
\begin{array}{lll}
T \uparrow 0 = & \cap S & \\
T \uparrow n = & T(T \uparrow (n-1)) & \text{if } n \text{ is a successor ordinal,} \\
 & \cup\{T \uparrow k : \ k < n\} & \text{if } n \text{ is a limit ordinal;} \\
T \downarrow 0 = & \cup S & \\
T \downarrow n = & T(T \downarrow (n-1)) & \text{if } n \text{ is a successor ordinal,} \\
 & \cap\{T \downarrow k : \ k < n\} & \text{if } n \text{ is a limit ordinal.}
\end{array}
$$

Proposition 5.2 *Let S be a complete lattice and $T : S \Rightarrow S$ be monotonic. Then, for any ordinal α*

- $T \uparrow \alpha \leq lfp(T)$
- $T \downarrow \alpha \geq gfp(T)$

and there exist ordinals β_1 and β_2 such that

- $\gamma_1 \geq \beta_1 \rightarrow T \uparrow \gamma_1 = lfp(T)$
- $\gamma_2 \geq \beta_2 \rightarrow T \downarrow \gamma_2 = gfp(T)$

The least α such that $T \uparrow \alpha = lfp(T)$ is called closure ordinal of T.

Theorem 5.3 *Kleene's First Recursion Theorem*

Let N denote the set of natural numbers $\{0, 1, 2, 3, \ldots\}$ and let ω denote its cardinality (that is, the first limit ordinal).

Then every continuous function T on a complete lattice $[S, \leq]$ whose bottom (least) element is $\perp$ has a least fixpoint in S which is

[11] Adapted for the case of complete lattices.

$$T \uparrow \omega = lub\{T^k(\bot) | k \in N\}$$

Moreover, when $\leq$ *is the partial order* $\subseteq$ *with bottom element* $\emptyset$,

$$T \uparrow \omega = lub\{T^k(\emptyset) | k \in N\} = \bigcup_{k \in N} T^k(\emptyset)$$

The expression $T \uparrow \omega$ assumes particular relevance when T is the function T_P (see next section) on the lattice of interpretations for a definite program P — it turns out to be exactly the minimal model for P.

6 Least Herbrand Model

In this section it is shown that the Least Herbrand Model is the set of ground atoms which are logical consequences of the program[12]. It is also presented an expression of the Least Herbrand Model in terms of fixpoints.

The main interest is in the question of whether a given model makes more atoms *true* than it needs to, in order to satisfy a given definite program — equivalently, whether that model includes any smaller models. Thus, the interesting relation between models is that of *set-inclusion* $\subseteq$.

For a given definite program P, the set of all models of P forms a complete lattice under the partial order $\subseteq$. So, there must always be a unique minimal model for P. For the ground instantiation G_P given in Example 4.3, pg. 8, namely:

CC **if** CL
CB **if** BL
CL **if** LL
BL

the entire Herbrand Base, $B_P = \{CC, CL, CB, LC, LL, LB, BC, BL, BB\}$ is a model, for sure. The Herbrand Base B_P is the unique maximal model for G_P and hence for P.

It is important to notice that many of the atoms in B_P do not even occur in G_P; consequently their truth values are irrelevant. Among all the atoms of B_P, those that must be part of any model are BL and CB. So, the set of these atoms,

$$\{likes(bob, logic), likes(chris, bob)\}$$

is the minimal model of P and is noted by M_P.

Proposition 6.1 *(Model Intersection Property) Let* P *be a program and* $\{M_i\}_{i \in I}$ *be a non-empty set of Herbrand Models for* P. *Then* $\cap_{i \in I} M_i$ *is a Herbrand Model for* P

[12] Meaning a definite program.

Proof: Let

P — definite program and

$\mathcal{I}$ — intersection of several Herbrand Models for P,

$$\mathcal{I} = M_1 \cap M_2 \cap \ldots \cap M_n$$

Suppose $\mathcal{I}$ is not a model for P. Then,

- some clause q **if** *body* of G_P is *false* in $\mathcal{I}$, then
- $q \notin \mathcal{I}$ and *body* is *true* in $\mathcal{I}$ then
- $q \notin M_i$ and *body* is *true* in M_i for some i, then
- M_i is not a model for q **if** *body* then
- M_i is not a model for P, which contradicts the initial assumption.

It is important to notice that:

- $\mathcal{I}$ does not need to be the intersection of **all** models of P;
- the result does not generally hold if P is indefinite.

Proposition 6.2 *Let P be a program. The intersection $\mathcal{I}^*$ of all the Herbrand models of P is the minimal model M_P.*

Proof: Suppose $\mathcal{I}^*$ is not the minimal model, then

- there must exist some model M_j such that $M_j \subset \mathcal{I}^*$ then
- there must exist some atom q such that $q \notin M_j$ and $q \in \mathcal{I}^*$, but
- $q \in \mathcal{I}^*$ implies that $q \in M_i$ for all i, contradicting $q \notin M_j$, therefore

the initial assumption is false.

Example 6.1 *Let the set S of the Example 4.2, pg. 8, which has as models: $M_1 = \{p(a), q(a)\}$, $M_2 = \{p(a), q(a), m(a)\}$, $M_3 = \{p(a), q(a), m(a), g(a)\}$. Its Least Herbrand Model is the intersection of M_1, M_2, M_3, which is $M = \{p(a), q(a)\}$.*

There is a relation among any definite program P, its Least Herbrand Model M_P and each of its ground atomic logic consequence q, as stated next.

Theorem 6.1 *Let P be a program. Then*

$$M_P = \{q \in B_P : q \textit{ is a logical consequence of } P\} \textit{ or}$$

$$P \models q \textit{ iff } q \in M_P$$

Let P be a program. Then 2^{B_P}, which is the power set of all Herbrand Interpretations of P, is a complete lattice under the partial order of set inclusion $\subseteq$. The top element of this lattice is B_P and the bottom element is $\emptyset$. The least upper bound of any set of Herbrand Interpretations is the Herbrand Interpretation which is the union of all the Herbrand Interpretations in the set. The greatest lower bound is the intersection.

As pointed out in [Hogger 90], other properties of M_P arise from the truth of the Skolem-Herbrand-Godel Theorem, which states that a clause-set S is unsatisfiable iff there exists some finite subset of G_S which has no Herbrand models. From this, if P is any definite program, it can be concluded that:

$q \in M_P$
iff $P \models q$
iff $P \bigcup \{\neg q\}$ is unsatisfiable
iff some finite subset of $G_{(P \bigcup \{\neg q\})}$ has no Herbrand models
iff some finite subset g of G_P exists such that $g \bigcup \{\neg q\}$ has no Herbrand models
iff some finite subset of G_P implies q.

In short, even when H, B_P and G_P are infinite, every atom q in M_P can be sought as a consequence of some finite subset of G_P. This fact is closely related to the refutation-completeness of resolution, which postulates that every solvable query $\neg q$ can be refuted using P after some **finite** number of resolution steps, thus utilizing only some finite subset of P.

Definition 6.1 *Let P be a program. The mapping $T_P : 2^{B_P} \Rightarrow 2^{B_P}$ is defined as follows. Let I be a Herbrand interpretation. Then $T_P(I) = \{A \in B_P : A \leftarrow A_1, A_2, \ldots, A_n$ is a ground instance of a clause in P $(A \leftarrow A_1, A_2, \ldots, A_n \in G_P)$ and $\{A_1, \ldots, A_n\} \subseteq I\}$.*

The function T_P, which is a total mapping from the powerset B_P to itself, is specific to the given program P. This definition states how one Herbrand Interpretation from the powerset of B_P is obtained from another.

Proposition 6.3 *Let P be a program. The mapping $T_P : 2^{B_P} \Rightarrow 2^{B_P}$ is continuous.*

Proposition 6.4 *Let P be a program and I be a Herbrand interpretation of P. Then I is a model for P iff $T_P(I) \subseteq I$.*

Theorem 6.2 *Let P be a program. Then $M_P = lfp(T_P) = T_P \uparrow \omega$.*

Theorem 6.3 *Let P be a program and G a goal $\leftarrow A_1, \ldots, A_k$. Suppose θ is an answer substitution for $P \cup \{G\}$ such that $(A_1 \wedge \ldots \wedge A_k)\theta$ is ground. Then the following are equivalent:*

- θ *is correct.*
- $(A_1 \wedge \ldots \wedge A_k)\theta$ *is true wrt every Herbrand model of* P.
- $(A_1 \wedge \ldots \wedge A_k)\theta$ *is true wrt the least Herbrand model of* P.

7 The Lattice of Herbrand Interpretations

Example 7.1 *Let Prog1 be the following Prolog program*

```
p(X):- q(X).
m(Y):- g(a).
```

which has as clausal representation the set

$$S = \{p(X) \vee \neg q(X), m(Y) \vee \neg g(a)\}$$

and as Herbrand Universe and Base, the following sets respectively:

$$H = \{a\}$$

$$B_{Prog1} = \{p(a), q(a), m(a), g(a)\}$$

As the Herbrand Base for $Prog1$ has four elements, the set of all possible interpretation has 16 elements.

Let $T_{Prog1} : 2^{B_{Prog1}} \rightleftharpoons 2^{B_{Prog1}}$ be the mapping as defined in Definition 6.1, pg. 22. As commented earlier, the function T_P is specific to the given program P, and states how one Herbrand Interpretation from the powerset of B_P is obtained from another.

In Table 1, the column $T_{Prog1}(id)/id'$ presents the interpretation — identified by id' — which results from applying T_{Prog1} to the interpretation identified by id. The column $T_{Prog1}(id')/id''$ presents the interpretation — identified by id'' — which results from applying T_{Prog1} to the interpretation identified by id'.

Each line in the table represents one interpretation. For each line, and for columns labeled $p(a)$, $q(a)$, $m(a)$ and $g(a)$, the presence of a $\star$ means that the atom which labels the correspondent column is part of the interpretation. For example, the line identified by the id value 7, represents an interpretation which has the associated sample 0111, meaning that the atoms $q(a)$, $m(a)$ and $g(a)$ belong to the interpretation.

Figure 6 shows the hypercube which graphically represents the space of all possible Herbrand Interpretations for $Prog1$. The function T_{Prog1} maps the 16 interpretations from $2^{B_{Prog1}}$ to interpretations in $2^{B_{Prog1}}$, according to Table 1.

As shown in Table 2, 0 is the only interpretation which satisfies the definition of being a fixpoint of T_{Prog1}. In Table 2 line 1, for example, is read as: the interpretations which have $id's$ $0, 2, 4, 8$ are noted in Figure 6 as $\bullet$ and are mapped by T_{Prog1} in interpretation 0, noted in Figure 7 as $\bullet$, which afterwards is mapped by T_{Prog1} in interpretation 0.

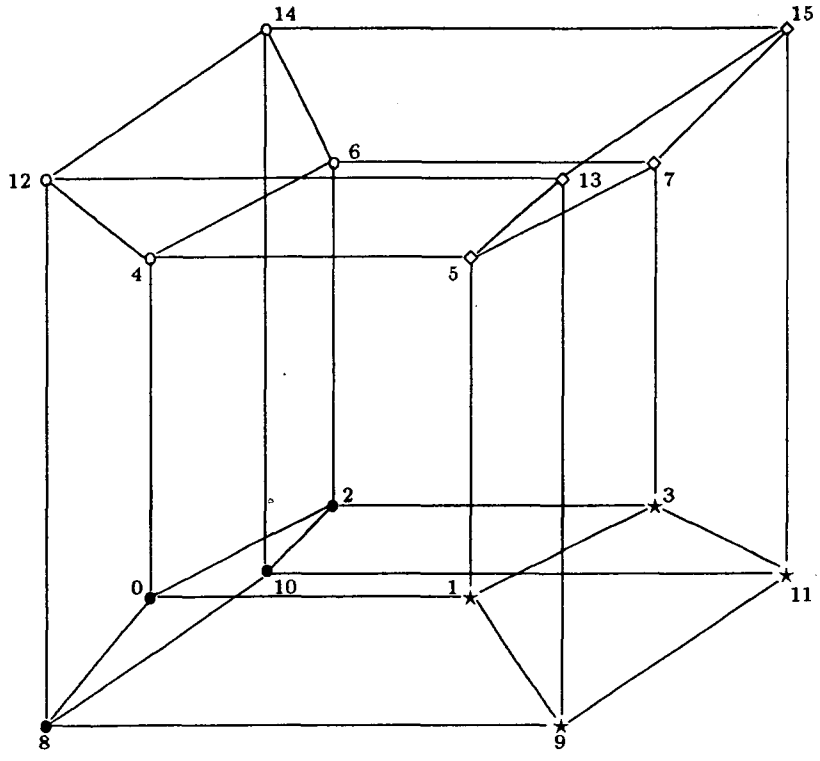

Figure 6: Lattice representing all the Herbrand Interpretation for *Prog*1

id	sample	$p(a)$	$q(a)$	$m(a)$	$g(a)$	models	$T_{Prog1}(id)/id'$	$T_{Prog1}(id')/id''$
0	0000					$\star$	$\emptyset/0$	$T(0)/0$
1	0001				$\star$		$\{m(a)\}/2$	$T(2)/0$
2	0010			$\star$		$\star$	$\emptyset/0$	$T(0)/0$
3	0011			$\star$	$\star$	$\star$	$\{m(a)\}/2$	$T(2)/0$
4	0100		$\star$				$\{p(a)\}/8$	$T(8)/0$
5	0101		$\star$		$\star$		$\{p(a), m(a)\}/10$	$T(10)/0$
6	0110		$\star$	$\star$			$\{p(a)\}/8$	$T(8)/0$
7	0111		$\star$	$\star$	$\star$		$\{p(a), m(a)\}/10$	$T(10)/0$
8	1000	$\star$				$\star$	$\emptyset/0$	$T(0)/0$
9	1001	$\star$			$\star$		$\{m(a)\}/2$	$T(2)/0$
10	1010	$\star$		$\star$		$\star$	$\emptyset/0$	$T(0)/0$
11	1011	$\star$		$\star$	$\star$	$\star$	$\{m(a)\}/2$	$T(2)/0$
12	1100	$\star$	$\star$			$\star$	$\{p(a)\}/8$	$T(8)/0$
13	1101	$\star$	$\star$		$\star$		$\{p(a), m(a)\}/10$	$T(10)/0$
14	1110	$\star$	$\star$	$\star$		$\star$	$\{p(a)\}/8$	$T(8)/0$
15	1111	$\star$	$\star$	$\star$	$\star$	$\star$	$\{p(a), m(a)\}/10$	$T(10)/0$

Table 1: Finding the fixpoint of T_{Prog1}

id	noted in Figure 6 as	id'	noted in Figure 7 as	id''
0, 2, 4, 8	$\bullet$	0	$\bullet$	0
1, 3, 9, 11	$\star$	2	$\star$	0
4, 6, 12, 14	$\circ$	8	$\circ$	0
5, 7, 13, 15	$\diamond$	10	$\diamond$	0

Table 2: Fixpoint of $T_{Prog1} = 0$

The resulting interpretations — identified by $\bullet$, $\star$, $\circ$ and $\diamond$ — from the application of T_{Prog1} to the lattice of Herbrand Interpretations shown in Figure 6 is graphically shown in Figure 7.

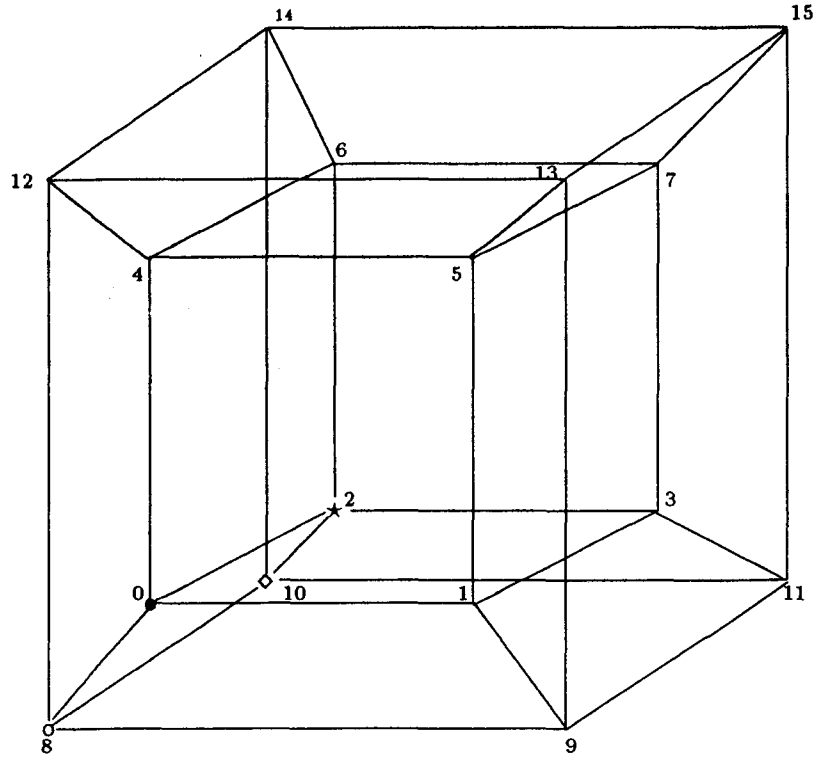

Figure 7: Interpretations — •, ⋆, ∘ and ◇ — obtained by applying the function T_{Prog1} to the lattice shown in previous figure

Two more similar examples are presented next.

Example 7.2 *Let Prog2 be the following Prolog program*

p(X):- q(X).
m(Y):- g(a).
q(Z).

which has as clausal representation the set

$$S = \{p(X) \vee \neg q(X), m(Y) \vee \neg g(a), q(Z)\}$$

and as Herbrand Universe and Base, the following sets respectively:

$$H = \{a\}$$

$$B_{Prog2} = \{p(a), q(a), m(a), g(a)\}$$

For $Prog2$, the results of applying T_{Prog2} to $2^{|B_{Prog2}|}$ are shown in Table 3.

id	sample	$p(a)$	$q(a)$	$m(a)$	$g(a)$	models	$T_{Prog2}(id)/id'$	$T_{Prog2}(id')/id''$
0	0000						$\{q(a)\}/4$	$T(4)/12$
1	0001				$\star$		$\{q(a), m(a)\}/6$	$T(6)/12$
2	0010			$\star$			$\{q(a)\}/4$	$T(4)/12$
3	0011			$\star$	$\star$		$\{q(a), m(a)\}/6$	$T(6)/12$
4	0100		$\star$				$\{p(a), q(a)\}/12$	$T(12)/12$
5	0101		$\star$		$\star$		$\{p(a), q(a), m(a)\}/14$	$T(14)/12$
6	0110		$\star$	$\star$			$\{p(a), q(a)\}/12$	$T(12)/12$
7	0111		$\star$	$\star$	$\star$		$\{p(a), q(a), m(a)\}/14$	$T(14)/12$
8	1000	$\star$					$\{q(a)\}/4$	$T(4)/12$
9	1001	$\star$			$\star$		$\{q(a), m(a)\}/6$	$T(6)/12$
10	1010	$\star$		$\star$			$\{q(a)\}/4$	$T(4)/12$
11	1011	$\star$		$\star$	$\star$		$\{q(a), m(a)\}/6$	$T(6)/12$
12	1100	$\star$	$\star$			$\star$	$\{p(a), q(a)\}/12$	$T(12)/12$
13	1101	$\star$	$\star$		$\star$		$\{p(a), q(a), m(a)\}/14$	$T(14)/12$
14	1110	$\star$	$\star$	$\star$		$\star$	$\{p(a), q(a)\}/12$	$T(12)/12$
15	1111	$\star$	$\star$	$\star$	$\star$	$\star$	$\{p(a), q(a), m(a)\}/14$	$T(14)/12$

Table 3: Finding the fixpoint of T_{Prog2}

As shown in Table 4, 12 is the only interpretation that satisfies the definition of being a fixpoint of T_{Prog2}.

id	id'	id''
0, 2, 8, 10	4	12
1, 3, 9, 11	6	12
4, 6, 12, 14	12	12
5, 7, 13, 15	14	12

Table 4: Fixpoint of $T_{Prog2} = 12$

Example 7.3 *Let Prog3 be the following Prolog program*

p(X).
q(Y).
m(Z).
g(W).

which has as clausal representation the set

$$S = \{p(X), q(Y), m(Z), g(W)\}$$

and as Herbrand Universe and Base, the following sets respectively:

$$H = \{a\}$$

$$B_{Prog3} = \{p(a), q(a), m(a), g(a)\}$$

For $Prog3$, the results of applying T_{Prog3} to $2^{|B_{Prog3}|}$ are shown in Table 5.

As shown in Table 6, 15 is the only interpretation that satisfies the definition of being a fixpoint of T_{Prog3}.

In the last three examples, the whole lattice of Herbrand Interpretations was considered, namely $2^{|B_{Prog}|}$ for applying the function T_{Prog}. This has been done in order to identify

id	sample	$p(a)$	$q(a)$	$m(a)$	$g(a)$	models	$T_{Prog3}(id)/id'$	$T_{Prog3}(id')/id''$
0	0000						$\{p(a), q(a), m(a), g(a)\}/15$	$T(15)/15$
1	0001				$\star$		$\{p(a), q(a), m(a), g(a)\}/15$	$T(15)/15$
2	0010			$\star$			$\{p(a), q(a), m(a), g(a)\}/15$	$T(15)/15$
3	0011			$\star$	$\star$		$\{p(a), q(a), m(a), g(a)\}/15$	$T(15)/15$
4	0100		$\star$				$\{p(a), q(a), m(a), g(a)\}/15$	$T(15)/15$
5	0101		$\star$		$\star$		$\{p(a), q(a), m(a), g(a)\}/15$	$T(15)/15$
6	0110		$\star$	$\star$			$\{p(a), q(a), m(a), g(a)\}/15$	$T(15)/15$
7	0111		$\star$	$\star$	$\star$		$\{p(a), q(a), m(a), g(a)\}/15$	$T(15)/15$
8	1000	$\star$					$\{p(a), q(a), m(a), g(a)\}/15$	$T(15)/15$
9	1001	$\star$			$\star$		$\{p(a), q(a), m(a), g(a)\}/15$	$T(15)/15$
10	1010	$\star$		$\star$			$\{p(a), q(a), m(a), g(a)\}/15$	$T(15)/15$
11	1011	$\star$		$\star$	$\star$		$\{p(a), q(a), m(a), g(a)\}/15$	$T(15)/15$
12	1100	$\star$	$\star$				$\{p(a), q(a), m(a), g(a)\}/15$	$T(15)/15$
13	1101	$\star$	$\star$		$\star$		$\{p(a), q(a), m(a), g(a)\}/15$	$T(15)/15$
14	1110	$\star$	$\star$	$\star$			$\{p(a), q(a), m(a), g(a)\}/15$	$T(15)/15$
15	1111	$\star$	$\star$	$\star$	$\star$	$\star$	$\{p(a), q(a), m(a), g(a)\}/15$	$T(15)/15$

Table 5: Finding the fixpoint of T_{Prog3}

id	id'	id''
0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15	15	15

Table 6: Fixpoint of $T_{Prog3} = 15$

the fixpoints of T_{Prog} aiming to find the Minimal Herbrand Model. For the examples considered, all the interpretations which are models have been identified. In such case, the Minimal Herbrand Model can be identified as well using the result stated in Proposition 6.2, pg. 21, i.e. for a program P, the intersection of all the Herbrand Models of P is the Minimal Model M_P.

However this procedure can be simplified by taking into account only one initial interpretation, instead of the whole lattice of interpretations, and incrementally constructs the minimal model of *Prog*, by repeatedly applying T_{Prog}.

There is no guarantee that this method will stop — it will not stop, for instance, when the minimal model is infinite. Nervertheless, the more steps are performed, the better is the obtained approximation.

Considering the problem of finding the Minimal Herbrand Model for program *Prog*2, as defined in Example 7.2, the incremental method is applied as follows:

For *Prog*2, G_{Prog2} is given by

p(a):- q(a).
m(a):- g(a).
q(a).

Let the initial interpretation chosen be *id* 7, namely, $\{q(a), m(a), g(a)\}$. Now consider any clause in G_{Prog2}

head:- body.

If the chosen interpretation assigns *true* to all the atoms in *body*, then an immediate consequence is that *head* should also be made *true* since, if that is not the case, the clause would be made *false*.

In the next iteration, the interpretation taken into account consists of the set of those

heads made *true* by this argument, and the process is repeated.

In the example considered, the set of those *heads* is $\{p(a), m(a), q(a)\}$, which corresponds to interpretation *id* 14 — see Table 3. Applying again the method using now interpretation 14, the set of *heads* obtained is $\{p(a), q(a)\}$, which corresponds to the interpretation *id* 12. It can be observed that if the method is applied again, the same set of *heads*, namely $\{p(a), q(a)\}$ is replicate. In this particular example the minimal model has been obtained, although this is not always the case.

It should be noted that the initial chosen interpretation is not a model for *Prog*2. In a general case, if the initial chosen interpretation is not a model, the iterative process does not necessarily converges upon the minimal model, although it may do so in some cases, as in the example just shown. Furthermore if that is the case, eventually the method could not find any models at all.

On the other side, if the initial chosen interpretation is a model, or if the method reaches an interpretation which is a model, from that point on, the method will only obtain interpretations which are models and must converge upon some model, although it is not necessarily the minimal Herbrand model.

8 Conclusions

Concept learning from examples can be viewed as search of the space of possible concept description. Within the Inductive Logic Programming framework — which induces first order logic programs from examples — the learning task is to examine the search space under constraints imposed by training instances, background knowledge and selected description language.

Although standard first order logic can be used as a representation language, in particular, most researchers are using the Prolog programming language — restricted subset of first order logic — to a represent concept. However, many difficulties need to be overcome in the ILP framework and, among them, the intractability of the search space.

In this work are presented important concepts related to logic programming, namely Herbrand Universe, Model and Least Model that are basic for the understanding of logic programming itself, as well as for dealing with the intractability of search in an ILP environment.

Several examples are presented and discussed in detail in order to improve the understandability and grasping of those concepts.

Acknowledgments: to Carmen Lucia Pagadigorria for the assistance in preparing the Figures contained in this Notes.

References

[Chang 73] Chang, C.L.; Lee, R.C.T. *Symbolic Logic and Mechanical Theorem Proving.* Academic Press, New York, 1973.

[Hogger 90] Hogger, C. J. *Essentials of Logic Programming.* Clarendon Press, Oxford, 1990.

[Lavrač 92] Lavrač, N., Džeroski, S. *Background Knowledge and Declarative Bias in Inductive Concept Learning.* Lectures Notes on Artificial Intelligence 462, pp. 51-71, 1992.

[Lloyd 84] Lloyd J.W. *Foundations of Logic Programming.* Springer-Verlag, 1984.

[Mendelson 79] Mendelson, E. *Introduction to Mathematical Logic.* (2nd Ed.) Van Nostrand, Princeton, 1979.

[Mitchell 82] Mitchell, T.M. *Generalization as Search.* Artificial Intelligence 18, pp 203-226, 1982.

[Muggleton 90] Muggleton, S.H.; Feng, C. *Efficient Induction of Logic Programs.* In Proceedings First Conference on Algorithmic Learning Theory, Tokyo, pp 368-381, 1990.

[Muggleton 91] Muggleton, S.H. *Inductive Logic Programming.* New Generation Computing, Vol. 8, pp 295-318, 1991.

[Nicoletti 93] Nicoletti, M.C.; Monard, M.C. *Inverting Resolution: Plotkin's Least General Generalization.* Notas ICMSC/USP, Nº130, 30pg., Março, 1993.

[Sanderson 73] Sanderson J.G. *The Lambda Calculus, Lattice Theory and Reflexive Domains.* Mathematical Institute Lecture Notes, University of Oxford, 1973.

[Van Emden 76] Van Emden M.H.; Kowalski R.A. *The Semantics of Predicate Logic as a Programming Language.* Journal of the Association for Computing Machinery, Vol. 23, Nº4, pp. 733-742, october 1976.