3-1991

Duality in Logic Programming

Feng Yang

SU-CIS-91-10

Duality in Logic Programming

Feng Yang

March 1991

School of Computer and Information Science
Suite 4-116
Center for Science and Technology
Syracuse, New York 13244-4100

(315) 443-2368

Duality in Logic Programming

Feng Yang

School of Computer & Information Science
Syracuse University
Syracuse, N.Y. 13244-4100

March 14, 1991

Abstract

Various approximations of classic negation have been proposed for logic programming. But the semantics for those approximations are not entirely clear. In this paper a proof-theoretic operator, we call it *failure* operator, denoted as $F_{\mathbf{P}}$, is associated with each logic program to characterize the meaning of various *negations* in logic programming. It is shown that the *failure* operator $F_{\mathbf{P}}$ is a dual of the $T_{\mathbf{P}}$ immediate consequence operator developed by Van Emden and Kowalski and is downward continuous. It has the desirable properties entirely analogous to what $T_{\mathbf{P}}$ has such as continuity, having a unique least fixpoint and a unique greatest fixpoint. It provides natural proof theories for various version negations in logic programming. We prove that set complementation provides the isomorphism between the fixpoints of $F_{\mathbf{P}}$ and those of $T_{\mathbf{P}}$, which illustrates the duality of $F_{\mathbf{P}}$ and $T_{\mathbf{P}}$. The existing treatment of negation in logic programming can be given in a simple and elegant fixpoint characterization.

1 Introduction

Logic programs consist of finite sets of clauses. Each clause can be viewed as an inference rule consisting of two distinct parts: an antecedent and a consequent. Such a delineation provides a declarative meaning for a clause in that the consequent is *true* when the antecedent is *true*. Based on such a point of view Van Emden and Kowalski in [2] developed a closure operator, $T_{\mathbf{P}}$, for a logic program $\mathbf{P}$. The lattice $\mathbf{L_P}$ on which $T_{\mathbf{P}}$ operates, $\mathbf{L_P} =< 2^{B_{\mathbf{P}}}, \subseteq >$, is a complete lattice. The operator is the motivation behind the **SLD**-resolution proof procedure which allows positive information to be deduced from a logic program. They show that the operator $T_{\mathbf{P}}$ is upward continuous over $\mathbf{L_P}$. Hence, the operator always has a least fixed point, $T_{\mathbf{P}} \uparrow \omega$ which corresponds to the success set of the program. The success set is taken to be the intended meaning of the program. The least fixpoint of the program is a set of ground atoms and has a very important property: an ground atom is in the set if and only if it is a logical consequence of the program. This approach is very satisfactory when one just wants to extract positive information from the program. However, negation remains something of a problem when negation is involved. The solution to the problem has been to use default rules such *Closed World Assumption, Negation as Failure* and the *Herbrand Rule* to implicitly infer negative information from the program. Those default

rules are characterized by complements of fixpoints of $T_{\mathbf{P}}$ which are generally complements of recursively enumerable sets. The most natural notion of negation, complementation, is simply not available. It is desirable to provide some kind of proof procedures for deducting negative information embedded in the program. We found the $F_{\mathbf{P}}$ operator to be exactly a dual of $T_{\mathbf{P}}$. It provides a natural proof procedure to deduct negative information from a logic program.

We now present the intuition behind $F_{\mathbf{P}}$ operator. Contrary to van Emden and Kowalski's approach, we look at a logic program as a finite set of inference rules which provide us with negative information about a logic program instead. Let's look at the following clause:

$$A \leftarrow B_1, B_2, \ldots, B_n. \tag{1}$$

Informally, in van Emden and Kowalski's approach the clause is viewed to represent explicitly the positive information of A, in the sense that if truthhood of each B_i is established then truthhood of A can be established from (1). On the other hand, from an alternative point of view the clause can be interpreted to explicitly represent the negative information about A, in the sense that if one fails to establish any of B_is then A can not be established from (1). Suppose that the following clauses are precisely those clauses whose consequent is A in a logic program $\mathbf{P}$.

$$\begin{aligned}
&A \leftarrow B_{11}, B_{12}, \ldots, B_{1n_1}.\\
&A \leftarrow B_{21}, B_{22}, \ldots, B_{2n_2}.\\
&\vdots\\
&A \leftarrow B_{m1}, B_{m2}, \ldots, B_{mn_m}.
\end{aligned}$$

If one fails to to establish A from all the clause above then it is safe to claim the unprovability of A from $\mathbf{P}$ under *close world assumption.*

It is the motivation that is behind $F_{\mathbf{P}}$ operator.

In this paper we associate each logic program the *failure* operator, denoted as $F_{\mathbf{P}}$. $F_{\mathbf{P}}$ operates on a truth space $\overline{\mathbb{L}_{\mathbf{P}}}$ which is a dual of $\mathbb{L}_{\mathbf{P}}$ on which $T_{\mathbf{P}}$ operates. We prove a series of theorems which state that the *failure* operator $F_{\mathbf{P}}$ is actually a dual of the operator $T_{\mathbf{P}}$ in some sense. It shares almost all the attractive properties which $T_{\mathbf{P}}$ has, even including upward continuity (over $\overline{\mathbb{L}_{\mathbf{P}}}$). we can characterize most existing treatment of negation in logic programming and give a simple and elegant fixpoint characterization of negations. We prove that the complement of the least fixed point of $T_{\mathbf{P}}$ is the least fixed point of $F_{\mathbf{P}}$ which is reached in ω steps when negations are not allowed in clause bodies. We have hopes that result like Theorem 6.1 will help to shed more light on the meaning of *negation* in logic programming. We will show that $F_{\mathbf{P}}$ along with $T_{\mathbf{P}}$ provides a very natural framework for the general consideration of logic programming semantics in [3].

We begin with a background on lattice in section 2. In section 3 we briefly recall some definitions and notations for logic programming from [1]. Then in section 5 we introduce the notion of F-interpretation and F-model for a logic program. We give the definition of $F_{\mathbf{P}}$ and present some results concerning fixed-points of $F_{\mathbf{P}}$. Finally we present the relationship between $F_{\mathbf{P}}$ and $T_{\mathbf{P}}$.

2 Background From Lattice

This section introduces the requisite concepts and results regarding lattices, mappings and their fixpoints.

Definition 2.1 *Let S be a set. A **binary relation** R on S is a subset of $S \times S$. By the **converse** of a binary relation R is meant the relation R^{-1} such that $(x, y) \in R$ if and only if $(y, x) \in R^{-1}$.*

We denote the fact that $(x, y) \in R$ by xRy.

Definition 2.2 *A binary relation R on a set S is a **partial order** if it satisfied the following conditions:*

1. *for all $x \in S$, xRx.*
2. *for all $x, y \in S$, xRy and yRx imply $x = y$.*
3. *for all $x, y, z \in S$, xRy and yRz imply xRz.*

We use the standard notation and use $\preceq$ to denote a partial order.

Definition 2.3 *A **partial ordered system** P is a pair $< S, \preceq >$ where S is a set and $\preceq$ is a partial order on S. Let T be a operator $T : S \rightarrow S$. The partial ordered system $\overline{P} =< S, \preceq^{-1} >$ is called a **dual** of P. P is called a **complete self dual**, if and only if there is an isomorphism $h : S \rightarrow S$ such that the following conditions are satisfied:*

- *for all $x \in S$, $h(h(x)) = x$;*
- *for all $x, y \in S$, $x \preceq y \Longrightarrow h(y) \preceq h(x)$.*

Definition 2.4 *Let $P =< S, \preceq >$ be a complete self dual with an isomorphism $h : S \rightarrow S$ satisfying the conditions mentioned above and let $T : S \rightarrow S$ be a function. A function T^D is called a **dual** if the followings hold:*

- *for all $x \in S, T^D(x) = h(T(h(x)))$; and*
- *for all $x \in S, T(x) = h(T^D(h(x)))$.*

Definition 2.5 *A partial ordered set $\mathbf{L} =< \Delta, \preceq >$ is a complete lattice if for every subset $\mathbf{X}$ of Δ its greatest lower bound and least upper bound exist.*

3 Background From Logic programming

We assume that our language **L** for predicate logic with equality symbol $=$ is fixed in advance, and contains a countably infinite set of n-place function symbols and a countably infinite set of n-place predicate symbols for each natural number n. We refer the reader to [1] for terminology and notation concerning **terms, atoms, literals** and **substitutions** that are not otherwise presented in this paper.

Definition 3.1 *A **program clause** is of the form:*

$$A \leftarrow B_1, B_2, \ldots, B_n,$$

where $A, B_1, B_2, \ldots, B_n (0 \leq n)$ are atoms.

Definition 3.2 *A **goal** is of the form:*

$$\leftarrow B_1, B_2, \ldots, B_n$$

where $n \geq 1$ and each B_i is an atom.

Definition 3.3 *A **logic program** is a finite set of program clauses.*

In addition, we assume that any program clauses and goals do not contain any occurrence of the equality symbol.

By an *alphabet* $\Sigma_{\mathbf{P}}$ of a logic program $\mathbf{P}$ we mean the set of *constants, predicate and function symbols* which occur in the program. In addition, any alphabet is assumed to contain a countably infinite set of variable symbols, connective $(\neg, \vee, \wedge, \forall, \exists, \leftarrow)$ and the usual punctuation symbols.

The set of all ground terms of a logic program $\mathbf{P}$ is called the *Herbrand universe*, denoted as $U_{\mathbf{P}}$ and the set of all ground atoms is called the *Herbrand base*, denoted as $B_{\mathbf{P}}$. We use $\mathbf{P}^*$ to denote the ground instantiation of a logic program $\mathbf{P}$. An *Herbrand interpretation* I for a logic program P is a subset of the Herbrand Base of P. An *Herbrand model* of P is an Herbrand interpretation of P that makes all clauses in P true.

It is easy to prove that $\mathbb{L}_{\mathbf{P}} =< 2^{B_{\mathbf{P}}}, \subseteq >$ is a complete lattice. The greatest member of $\mathbb{L}_{\mathbf{P}}$ is $B_{\mathbf{P}}$ and the least member is $\emptyset$. Similarly one can prove that $\overline{\mathbb{L}_{\mathbf{P}}} =< 2^{B_{\mathbf{P}}}, \supseteq >$ is also a complete lattice.

4 Semantics

A Herbrand interpretation is a set of atoms which are precisely *true* in the interpretation. Since we are interested in its counterpart we define the notion of an F-interpretation for a logic program.

Definition 4.1 *Let $\mathbf{P}$ be a logic program and I be an Herbrand interpretation for $\mathbf{P}$ Then $B_{\mathbf{P}} - I$ is called an F-interpretation of $\mathbf{P}$.*

The class of Herbrand interpretations of a logic program with subset as the partial order forms the complete lattice $\mathbb{L}_{\mathbf{P}}$. From Definition 4.1 we can see that the class of F-interpretations of a logic program also forms the complete lattice $\overline{\mathbb{L}_{\mathbf{P}}}$ which is a dual of $\mathbb{L}_{\mathbf{P}}$. The following proposition states the relationship between $\mathbb{L}_{\mathbf{P}}$ and $\overline{\mathbb{L}_{\mathbf{P}}}$.

Proposition 4.1 *Let $\mathbf{P}$ be a logic program. Then $\overline{\mathbb{L}_{\mathbf{P}}}$ is a dual of $\mathbb{L}_{\mathbf{P}}$ and $\mathbb{L}_{\mathbf{P}}$ is a complete self dual.*

Proof: It is obvious that the set complement is the isomorphism which links $\mathbf{L_P}$ and $\overline{\mathbf{L_P}}$. □

The notion of Herbrand models is intended to capture the meaning of logical consequence of a logic program. Next we use the concept of F-model to capture the meaning of "immediate failure" in logic programming.

Definition 4.2 *An F-interpretation I of a logic program $\boldsymbol{P}$ is an F-**model** for $\boldsymbol{P}$ if for each ground atom $A \in I$ then for each ground clause of the form*

$$A \leftarrow B_1, B_2, \ldots, B_n \in P^*$$

$\{B_1, B_2, \ldots, B_n\} \cap I \neq \emptyset$.

It is straightforward to prove the following proposition.

Proposition 4.2 *Let I_1, I_2 be two F-models of logic program $\boldsymbol{P}$. Then their union is also an F-model of $\boldsymbol{P}$.*

Proposition 4.2 is a dualized property of *Herbrand models* of a logic program, which indicates that Herbrand models of a logic program are closed under set intersection.

5 Failure operator and its fixpoints

Van Emden and Kowalski obtain a deep characterization of the least Herbrand model of a logic program $\mathbf{P}$ by using $T_\mathbf{P}$ operator. $T_\mathbf{P}$ act like an *immediately logical consequence* operator. It transforms an Herbrand interpretation into a new Herbrand interpretations which contains all facts that can be deduced from the logic program by using the old interpretation. It provides the link between the declarative and procedural semantics of a logic program $\mathbf{P}$. The least Herbrand model is precisely the set of ground atoms which are logical consequences of the program.

We found that one can obtain a characterization of the least F-models of logic programs by using $F_\mathbf{P}$. To this end we associate every program with the complete lattice $\overline{\mathbf{L_P}}$.

Definition 5.1 *Let $\mathbf{P}$ be a logic program. The mapping $F_\mathbf{P}$ is defined as follows. Let I be a F-interpretation. Then $F_\mathbf{P} : 2^{B_\mathbf{P}} \rightarrow 2^{B_\mathbf{P}}$ is defined as follows:*

$$F_\mathbf{P}(I) = \{A | \forall A \leftarrow B_1, B_2, \ldots, B_n \in \mathbf{P}^* \{B_1, B_2, \ldots, B_n\} \cap I \neq \emptyset\}$$

A F-model of a logic program can be characterized by $F_\mathbf{P}$ operator. Its easy proof is omit here.

Theorem 5.1 *Let $\mathbf{P}$ be a logic program. Then I is an F-model of $\boldsymbol{P}$ if and only $I \subseteq F_\mathbf{P}(I)$.*

We next show that $F_\mathbf{P}$ is continuous and hence monotonic over $\overline{\mathbf{L_P}}$. For that propose we need the lemma stated below:

Lemma 5.1 *Let X be a directed subset of $\overline{\mathbf{L_P}}$. Then for all $I \in X$*

$$\{B_1, B_2, \ldots, B_n\} \cap lub(X) \neq \emptyset \Longleftrightarrow \{B_1, B_2, \ldots, B_n\} \cap I \neq \emptyset.$$

Proof: Let $S = \{B_1, B_2, \ldots, B_n\}$
($\Longrightarrow$): This direction is obvious since

$$lub(X) = \bigcap_{I \in X} I.$$

($\Longleftarrow$): Suppose that $S \cap I \neq \emptyset$ for all $I \in X$. Then we have that for all $I \in X$

$$m_I = |I \cap S| \geq 1.$$

Let

$$k = min\{m_I | I \in X\}.$$

From its definition we have that

$$1 \leq k \leq n.$$

Thus there is at least an $I_0 \in X$ such that

$$I_0 \cap S = \{B'_1, B'_2, \ldots, B'_k\}.$$

Now we show that for all $I \in X$

$$I_0 \cap S \subseteq I \cap S.$$

We prove it by contradiction. Suppose that there is an $J \in X$ such that

$$I_0 \cap S \not\subseteq J \cap S.$$

Thus it follows that

$$\exists A(A \in I_0 \cap S \wedge A \notin J \cap S).$$

Hence

$$(I_0 \cap S) \cap (J \cap S) = I_0 \cap J \cap S \subset I_0 \cap S.$$

Since X is a directed set and $I_0, J \in X$ there is an $J_0 \in X$ such that $J_0 \subseteq I_0 \cap J$. Now we know

$$|I_0 \cap J \cap S| < k.$$

Thus we have $|J_0 \cap S| \leq k-1$ which contradicts the definition of k. Therefore, for all $I \in X$

$$I_0 \cap S \subseteq I \cap S.$$

Hence

$$lub(X) \cap S = (\bigcap_{I \in X} I) \cap S \supseteq I_0 \cap S \neq \emptyset$$

which completes the proof. □

The following result shows that $F_{\mathbf{P}}$ shares the continuity with $T_{\mathbf{P}}$ but in opposite direction.

Theorem 5.2 *Let* $\mathbf{P}$ *be a program. Then the mapping* $F_{\mathbf{P}}$ *is continuous, and hence monotonic over* $\overline{\mathbb{L}_{\mathbf{P}}}$.

Proof: Let X be a directed subset of 2^{B_P}. In order to prove F_P is continuous, we have to show that

$$F_P(lub(X)) = lub(F_P).$$

Now we have that $A \in F_P(lub(X))$
iff for all clause of the form

$$A \leftarrow B_1, B_2, \ldots, B_n \in P^*$$

such that $lub(X) \cap \{B_1, B_2, \ldots, B_n\} \neq \emptyset$
iff for all clause of the form

$$A \leftarrow B_1, B_2, \ldots, B_n \in P^*$$

such that $I \cap \{B_1, B_2, \ldots, B_n\} \neq \emptyset$ for all $I \in X$ by Lemma 5.1
iff $A \in F_P(I)$ for all $I \in X$
iff $A \in lub(F_P(X))$. □

We now define the ordinal powers of F_P.

Definition 5.2 *Consider a logic program* **P**. *We define*

$$\begin{array}{llll} F_P \uparrow 0 & = B_P & F_P \downarrow 0 & = \emptyset \\ F_P \uparrow (n+1) & = F_P(F_P \uparrow n) & F_P \downarrow (n+1) & = F_P(F_P \downarrow n) \\ F_P \uparrow \alpha & = \bigcap_{n<\alpha} F_P \uparrow n & F_P \downarrow \alpha & = \bigcup_{n<\alpha} F_P \downarrow n \end{array}$$

where n is a successor ordinal, and α is a limit ordinal.

The next result is an analogue theorem for F_P. It is an application of Kleene's Theorem in [1].

Theorem 5.3 *Let* **P** *be a logic program. Then*

$$lfp(F_P) = F_P \uparrow \omega.$$

Let $\overline{M_P}$ be the union of all F-models of a logic program **P**. The folowing is an another analogue result for F_P parallel to T_P.

Theorem 5.4 *Let* **P** *be a logic program. Then*

$$lfp(F_P) = \overline{M_P}.$$

Proof:

$\overline{M_P} = glb\{I : I \text{ is a } F\text{-model for } \mathbf{P}\}$
$= glb\{I : I \subseteq F_P(I)\}$ by Theorem 5.1
$= lfp(F_P)$ Tarski Theorem.

6 Relationship between $T_{\mathbf{P}}$ and $F_{\mathbf{P}}$

Now we come to the first major result of the theory. This theorem states the relationship between $T_{\mathbf{P}}$ and $F_{\mathbf{P}}$.

Theorem 6.1 *Let* $\mathbf{P}$ *be a logic program. Then for any* $S \subseteq B_{\mathbf{P}}$

$$F_{\mathbf{P}}(B_{\mathbf{P}} - S) = B_{\mathbf{P}} - T_{\mathbf{P}}(S).$$

Proof:

$$\begin{aligned} F_{\mathbf{P}}(B_{\mathbf{P}} - S) &= \{A | \forall A \leftarrow B_1, B_2, \ldots, B_n \in \mathbf{P}^* \{B_1, B_2, \ldots, B_n\} \cap (B_{\mathbf{P}} - S) \neq \emptyset\} \\ &= \{A | \forall A \leftarrow B_1, B_2, \ldots, B_n \in \mathbf{P}^* ((\exists i : 1 \preceq i \preceq n) B_i \notin S)\} \\ &= B_{\mathbf{P}} - \{A | \exists A \leftarrow B_1, B_2, \ldots, B_n \in \mathbf{P}^* \neg((\exists i : 1 \preceq i \preceq n) B_i \notin S)\} \\ &= B_{\mathbf{P}} - \{A | \exists A \leftarrow B_1, B_2, \ldots, B_n \in \mathbf{P}^*, \{B_1, B_2, \ldots, B_n\} \subseteq S\} \\ &= B_{\mathbf{P}} - T_{\mathbf{P}}(S) \end{aligned}$$

□

Before we present results about relationship between $F_{\mathbf{P}}$ and $T_{\mathbf{P}}$ we recall a definition from [1].

Definition 6.1 (Lloyd 1988) *Consider a logic program* $\mathbf{P}$. *We define*

$$\begin{array}{llll} T_{\mathbf{P}} \uparrow 0 & = \emptyset & T_{\mathbf{P}} \downarrow 0 & = B_{\mathbf{P}} \\ T_{\mathbf{P}} \uparrow (n+1) & = T_{\mathbf{P}}(T_{\mathbf{P}} \uparrow n) & T_{\mathbf{P}} \downarrow (n+1) & = T_{\mathbf{P}}(T_{\mathbf{P}} \downarrow n) \\ T_{\mathbf{P}} \uparrow \alpha & = \bigcup_{n<\alpha} T_{\mathbf{P}} \uparrow n & T_{\mathbf{P}} \downarrow \alpha & = \bigcap_{n<\alpha} T_{\mathbf{P}} \downarrow n \end{array}$$

where n *is a successor ordinal, and* α *is a limit ordinal.*

Now we are in a proposition to present the duality between $F_{\mathbf{P}}$ and $T_{\mathbf{P}}$.

Theorem 6.2 *Let* $\mathbf{P}$ *be a logic program. Then we have*

1. $T_{\mathbf{P}} \uparrow \alpha = B_{\mathbf{P}} - F_{\mathbf{P}} \uparrow \alpha$.
2. $T_{\mathbf{P}} \downarrow \alpha = B_{\mathbf{P}} - F_{\mathbf{P}} \downarrow \alpha$.
3. $F_{\mathbf{P}} \uparrow \alpha = B_{\mathbf{P}} - T_{\mathbf{P}} \uparrow \alpha$.
4. $F_{\mathbf{P}} \downarrow \alpha = B_{\mathbf{P}} - T_{\mathbf{P}} \downarrow \alpha$.

Proof: We prove (1) by using transfinite induction.

- Induction base: $\alpha = 0$.

$$\begin{aligned} T_{\mathbf{P}} \uparrow 0 &= \emptyset \\ &= B_{\mathbf{P}} - B_{\mathbf{P}} \\ &= B_{\mathbf{P}} - F_{\mathbf{P}} \uparrow 0 \qquad \text{by definition of } F_{\mathbf{P}} \uparrow 0 \end{aligned}$$

- Induction hypothesis: the equation (1) holds for all ordinals $\beta \leq \alpha$.
- Induction step:

  Case a: $\alpha + 1$ is a successor ordinal. Then

$$\begin{aligned} T_{\mathbf{P}} \uparrow (\alpha+1) &= T_{\mathbf{P}}(T_{\mathbf{P}} \uparrow \alpha) && \text{by Definition 6.1} \\ &= B_{\mathbf{P}} - F_{\mathbf{P}}(B_{\mathbf{P}} - T_{\mathbf{P}} \uparrow \alpha) && \text{by Theorem 6.1} \\ &= B_{\mathbf{P}} - F_{\mathbf{P}}(F_{\mathbf{P}} \uparrow \alpha) && \text{by the induction hypothesis} \\ &= B_{\mathbf{P}} - F_{\mathbf{P}} \uparrow \alpha + 1 && \text{by Definition 5.2} \end{aligned}$$

  Case b: $\alpha + 1$ is a limited ordinal. Then

$$\begin{aligned} T_{\mathbf{P}} \uparrow (\alpha+1) &= \bigcup_{\beta \leq \alpha} T_{\mathbf{P}} \uparrow \beta && \text{by Definition 6.1} \\ &= \bigcup_{\beta \leq \alpha} (B_{\mathbf{P}} - F_{\mathbf{P}} \uparrow \beta) && \text{by the induction hypothesis} \\ &= B_{\mathbf{P}} - \bigcap_{\beta \leq \alpha} F_{\mathbf{P}} \uparrow \beta \\ &= B_{\mathbf{P}} - F_{\mathbf{P}} \uparrow (\alpha+1) && \text{by Definition 5.2} \end{aligned}$$

Similarly we can prove the equations (2), (3) and (4). □

The following result establishes the relationship between fixpoints of $F_{\mathbf{P}}$ and those of $T_{\mathbf{P}}$ which illustrates the duality of $F_{\mathbf{P}}$ and $T_{\mathbf{P}}$.

Corollary 6.1 *Let* **P** *be a logic program. Then*

- $lfp(F_{\mathbf{P}}) = F_{\mathbf{P}} \uparrow \omega = B_{\mathbf{P}} - T_{\mathbf{P}} \uparrow \omega$.
- $gfp(F_{\mathbf{P}}) = B_{\mathbf{P}} - gfp(T_{\mathbf{P}})$.

Proof: It follows immeditaely from the definitions of $F_{\mathbf{P}} \uparrow \omega$ and $gfp(F_{\mathbf{P}})$, and Theorem 6.2. □

The next result shows that the atoms in the least fixpoint of $F_{\mathbf{P}}$ are precisely those ground atoms that are not a logical consequence of the logic program **P**; the greatest fixed point of $F_{\mathbf{P}}$ are precisely those ground atoms that false in every Herbrand models. It also gives alternative characterization of *finite failure set* by using $F_{\mathbf{P}}$ operator.

Corollary 6.2 *Let* **P** *be a logic program. Then*

- $F_{\mathbf{P}} \uparrow \omega = \{A \in B_{\mathbf{P}} : A$ *is not a logical consequence of* $\mathbf{P}\}$.
- $F_{\mathbf{P}} \downarrow \omega = \{A \in B_{\mathbf{P}} : A$ *has a finite failed SLD-tree*$\}$.
- $gfp(F_{\mathbf{P}}) = \{A \in B_{\mathbf{P}} : A$ *is false in every Herbrand model of* $\mathbf{P}\}$.

Proof: It follows directly from Corollary 6.1 and the definition of $T_{\mathbf{P}}$. □

Acknowledgements

I wish to thank Dr. Howard Blair and Dr. Allen L. Brown for helpful discussions and comments.

References

[1] W.L. Lloyd. *Foundation of Logic programs.* Springer-Verlag, second edition, 1988.

[2] M.H. van Emden and R.A. Kowalski. The semantics of predicate logic as a programming language. *JACM*, 23(4):733–742, oct. 1976.

[3] Feng Yang. A unified framework for three-valued semantical treatments of logic programming. Submitted for publication, 1991.