ICOT Technical Report: TR-238

TR-238

On Semantics of Logic Programs with Uncertainties

by

Y. SAKAKIBARA
(FUJITSU Ltd.)

March, 1987

ICOT

Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan

(03) 456-3191~5
Telex ICOT J32964

Institute for New Generation Computer Technology

On Semantics of Logic Programs with Uncertainties

by

Yasubumi SAKAKIBARA*

* Research Associate, Fundamental Informatics Section, International Institute for Advanced Study of Social Information Science, FUJITSU LIMITED

140 Miyamoto, Numazu, Shizuoka 410-03, JAPAN

Abstract

A mechanism for dealing with uncertainties plays a central role in knowledge systems like expert systems. Especially, incorporating uncertainties into logic programs is a current study. On the other hand, there are few semantical considerations for it. In this paper, we will first attempt to give a semantics for logic programs with uncertainties. This is a further work of Shapiro's "Logic Programs with Uncertainties" which is the only paper so far, dealing with a semantics of logic programs with uncertainties. Our basic definitions of logic programs with uncertainties follow it. Secondly, we will define a proof procedure for logic programs with uncertainties which corresponds to an interpreter for it and present a sufficient condition for certainty functions of clauses to achieve the completeness of the proof procedure. Further, we study the finite-failure set with certainty threshold which is actually the implementation used to infer the negation as failure in logic programs with uncertainties. Then we will give an interesting characterization of it.

1. Introduction

A mechanism for dealing with uncertainties plays a central role in knowledge systems like expert systems. Various methods or theories of representing uncertainties have been discussed in literatures. Those approaches include methods of extending classical binary logic to many valued logics such as probability logic, fuzzy logic or infinite valued logic. Especially, a method of incorporating uncertainties into logic programs (e.g. Fuzzy-Prolog [1], [5], [6], [9]) is a current study. On the other hand, there are few semantical considerations for it.

In this paper, we will first attempt to give a semantics for logic programs with uncertainties. We choose Shapiro's approach to logic programs with uncertainties [9], because Shapiro's approach gives a general method for computing uncertainties. Because of this generality, all results concluded within this framework are proved for a whole class of quantitative schemes. This is a further work of Shapiro's approach in the sense that we will show several semantical results in a logic program with uncertainties P, while Shapiro's only semantical result is the model intersection property for P. Our basic definitions of logic programs with uncertainties follow it, where the certainty function is defined to compute the certainty of the conclusion of a clause from the multiset of certainties of solutions to goals in the condition of the clause. Secondly, we will define a proof procedure for logic programs with uncertainties which corresponds to an interpreter for it and present a sufficient condition for certainty functions of clauses to achieve the completeness of the proof procedure.

Further, we study the finite-failure set with certainty threshold which is actually the implementation used to infer the negation as failure in logic programs with uncertainties. The negation as failure rule introduced into logic programs with uncertainties in the natural way is the rule that if all derivations of $\leftarrow A$ cannot be successful with any certainty greater than or equal to certainty threshold c, then infer that $\neg A$ is successful with certainty $1-c$. Then we will give an interesting characterization of it. This is the first attempt to consider the negation as failure in logic programs with uncertainties from the semantical point of view.

This paper is organized as follows : Basic definitions of a logic program with uncertainties are given in Section 2. Section 3 defines an interpretation for logic programs with uncertainties and then gives a model semantics and a fixpoint semantics for it. In Section 4, a proof procedure for logic programs with uncertainties is defined and a sufficient condition for certainty functions of clauses to achieve the completeness of the proof procedure is presented. In Section 5, the negation as failure in logic programs with uncertainties is discussed from the semantical point of view. In Section 6, control strategies using certainties in logic programs with uncertainties are briefly mentioned, and followed by conclusions in Section 7.

2. Logic programs with uncertainties

Definition A *definite clause* is a clause of the form $A \leftarrow B$, where A is an atom and B is a conjunction of zero or more atoms. A *certainty factor* c is a real number greater than 0 and less than or equal to 1. A *certainty function* f is a function from multisets of certainty factors to certainty factors. A *logic program with uncertainties* P is a finite set of pairs $<A \leftarrow B, f>$, where $A \leftarrow B$ is a definite clause and f is a certainty function. (Especially in the case of $<A \leftarrow, f>$, we assume that $f(\emptyset)$ is defined and is a certainty factor.)

The certainty function computes the certainty of the conclusion of a clause from the multiset of certainties of solutions to goals in the condition of the clause. Two requirements of a certainty function f in [9] are that for every multisets S, $f(S \cup \{1\}) = f(S)$, and that f be monotonic increasing, which means that $S \leqq S'$ implies $f(S) \leqq f(S')$, where $\leq$ is the partial order over multisets, defined as follows. Let S and

$X=\{x_1,...,x_n\}$ ($n \geq 0$) be two multisets. Then $X \leq S$ iff there is a multiset $Y=\{y_1,...,y_n\}$ such that $S \subseteq Y$ and $x_i \leq y_i$ for $0 \leq i \leq n$. Notice that in this order the empty set $\varnothing$ ($=\{\}$) is the largest element. Thus this treatment is independent of particular certainty functions chosen, as long as they satisfy these two requirements.

3. Semantics for logic programs with uncertainties

Definition An *interpretation* I of a logic program with uncertainties P is a set of pairs $<A, c>$, where A is a ground atom and c is a certainty factor. An I contains at most one pair $<A, c>$ for any atom A. A ground atom A is *true* in I with certainty c iff there is a pair $<A, c'>$ in I such that $c \leq c'$. An atom A is *true* in I with certainty c iff for every ground instance A' of A, A' is true in I with certainty c. Let $A \leftarrow B_1,...,B_n$ ($n \geq 0$) be a ground definite clause, f be a certainty function and S be the multiset of certainties $\{c_1,...,c_n\}$ such that $<B_i, c_i>$ is in I for $1 \leq i \leq n$ (if no such pair exists for some atom B_i then S is considered undefined). Then $A \leftarrow B_1,...,B_n$ is *true* in I with respect to f iff either S is undefined or A is true in I with certainty f(S). A definite clause $A \leftarrow B$ is *true* in I with respect to f iff any ground instance $A' \leftarrow B'$ of it is true in I with respect to f. A logic program with uncertainties P is *true* in M iff for any pair $<A \leftarrow B, f>$ in P, $A \leftarrow B$ is true in M with respect to f. Such an interpretation M is called a *model* for P.

According to fuzzy set theory in the sense of Zadeh, we may regard such interpretations as fuzzy subsets which are collections of objects together with an indication of their grade of membership. In the theory of fuzzy sets, set inclusion, union and intersection operations are defined by using an order over grades of membership, a minimum function of them and a maximum function of them.

Now a partial order $\leq$, intersection $\cap$ and union $\cup$ on interpretations are defined in the natural way as follows.

Definition Let I and I' be interpretations. Then I≤I' iff for any pair <A, c> in I there is a pair <A, c'> in I' such that c≤c'.

Definition Let I and I' be interpretations. Then <A, ci> in I∩I' iff <A, c> in I and <A, c'> in I' and ci = min(c, c').

Definition Let I and I' be interpretations.

Then <A, cu> in I∪I' iff

cu = max(c, c') if <A, c> in I and <A, c'> in I'

and cu = c if <A, c> in I and there is no pair <A, c'> in I' for A

and cu = c' if <A, c'> in I' and there is no pair <A, c> in I for A.

Lemma 3.1 (model intersection property)

Let P be a logic program with uncertainties. Let M_1 and M_2 be two models for P. Then $M_1 \cap M_2$ is also a model for P.

(Proof) Since M_1 and M_2 are models for P, for any ground instance $A \leftarrow B_1,...,B_n$ and the certainty function f of a clause in P, $f(\{c^1{}_1,...,c^1{}_n\}) \leq c^1$ and $f(\{c^2{}_1,...,c^2{}_n\}) \leq c^2$ where $<A, c^1>$, $<B_1, c^1{}_1>$,..., $<B_n, c^1{}_n> \in M_1$ and $<A, c^2>$, $<B_1, c^2{}_1>$,..., $<B_n, c^2{}_n> \in M_2$. From the definition of intersection, $ci = \min(c^1, c^2)$ and $ci_j = \min(c^1{}_j, c^2{}_j)$ $(1 \leq j \leq n)$ for $<A, ci>$, $<B_1, ci_1>$,..., $<B_n, ci_n> \in M_1 \cap M_2$. Since f is monotonic increasing, $f(\{ci_1,...,ci_n\}) \leq \min(\ f(\{c^1{}_1,...,c^1{}_n\}),\ f(\{c^2{}_1,...,c^2{}_n\})\ )$. Thus $f(\{ci_1,...,ci_n\}) \leq ci$. Hence $M_1 \cap M_2$ is a model for P. □

By the above lemma, the least model for P which is the intersection of all models for P exists and we write ∩M(P).

We define a transformation Tc_p associated with a logic program with uncertainties P in the same way as standard logic programs.

Definition Let P be a logic program with uncertainties. Let I be an interpretation.

$<A, c> \in Tc_p(I)$ iff

$c = \sup\{f(S) \mid A \leftarrow B_1,\ldots,B_n$ is a ground instance and f is the certainty function of a clause in P such that $<B_i, c_i> \in I$ for $1 \leqq i \leqq n$ and $S = \{c_1,\ldots,c_n\}\}$.

Lemma 3.2 Let P be a logic program with uncertainties and I be an interpretation. Then I is a model for P iff $Tc_p(I) \leq I$.

(Proof) I is a model for P

iff for any ground instance $A \leftarrow B_1,\ldots,B_n$ and the certainty function f of a clause in P, $A \leftarrow B_1,\ldots,B_n$ is true in I with respect to f.

iff for $<A, c>, <B_1, c_1>,\ldots, <B_n, c_n> \in I$, $f(\{c_1,\ldots,c_n\}) \leqq c$.

iff $Tc_p(I) \leq I$. □

Lemma 3.3 Let P be a logic program with uncertainties. Tc_p is a monotonic increasing mapping in the sense that $I_1 \leq I_2$ implies that $Tc_p(I_1) \leq Tc_p(I_2)$, for any interpretations I_1 and I_2.

(Proof) Suppose that $I_1 \leq I_2$ for two interpretations I_1 and I_2. Suppose also that for any ground instance $A \leftarrow B_1,\ldots,B_n$ and the certainty function f of a clause in P, $<B_i, c^1{}_i> \in I_1$ and $<B_i, c^2{}_i> \in I_2$ for $1 \leqq i \leqq n$. Since $I_1 \leqq I_2$, $c^1{}_i \leqq c^2{}_i$ for $1 \leqq i \leqq n$, and since f is monotonic increasing, $f(\{c^1{}_1,\ldots,c^1{}_n\}) \leqq f(\{c^2{}_1,\ldots,c^2{}_n\})$. Then by the definition of Tc_p, for $<A, c^1> \in Tc_p(I_1)$ and $<A, c^2> \in Tc_p(I_2)$, $c^1 \leq c^2$. Therefore $Tc_p(I_1) \leq Tc_p(I_2)$. □

Then we have the following theorem.

Theorem 3.4 Let P be a logic program with uncertainties. The least model $\cap M(P)$ for P is equal to the least fixpoint of Tc_p ($lfp(Tc_p)$ for short).

(Proof) It is straightforward by lemma 3.1, 3.2, 3.3 and the Knaster-Tarski fixpoint theorem [2]. □

We consider $\cap M(P)$ and $lfp(Tc_p)$ as the semantics for a logic program with uncertainties P.

4. Proof procedure for logic programs with uncertainties

Definition Let P be a logic program with uncertainties and A be an atom. A *(P, F)-derivation* of a goal ←A is a sequence of quadruples $<G_i, C_i, \theta_i, F_i>$, $i=0,1,2,...$ such that:

① $G_0 = \leftarrow A$,

② G_i is of the form $\leftarrow(B_1,...,B_m)$ where $m \geqq 0$ and B_j is an atom ($1 \leqq j \leqq m$),

③ C_i is a list of m clauses $(A^{(1)} \leftarrow D^{(1)}{}_1,...,D^{(1)}{}_{n_1}, f^{(1)}),..., (A^{(m)} \leftarrow D^{(m)}{}_1,...,D^{(m)}{}_{n_m}, f^{(m)})$,

④ θ_i is a most general unifier of $(B_1,...,B_m)$ and $(A^{(1)},...,A^{(m)})$, and

⑤ G_{i+1} is $\leftarrow(D^{(1)}{}_1,...,D^{(1)}{}_{n_1},...,D^{(m)}{}_1,...,D^{(m)}{}_{n_m})\theta_i$

and F_i is $[f^{(1)}(\{X_1,...,X_{n_1}\}),...,f^{(m)}(\{X_1,...,X_{n_m}\})]$.

A (P, F)-derivation is *successful* if it is finite and its last goal (some G_i) is empty.

Next we define a computed certainty of a successful (P, F)-derivation.

Definition Let $<G_i, C_i, \theta_i, F_i>$ $(i=0,1,2,...)$ be a (P, F)-derivation. Let F_i be $[f_1(\{X_1,...,X_{n_1}\}),...,f_m(\{X_1,...,X_{n_m}\})]$ and F_{i+1} be $[T_{1,1},...,T_{1,n_1},...,T_{m,1},...,T_{m,m_n}]$. The *application* of F_{i+1} to F_i, denoted by $F_i(F_{i+1})$, is

$[f_1(\{T_{1,1},...,T_{1,n_1}\}),...,f_m(\{T_{m,1},...,T_{m,n_m}\})]$.

Definition Let $<G_i, C_i, \theta_i, F_i>$ $(i=0,1,2,...,n)$ be a successful (P, F)-derivation. Then the *computed certainty* of it is $F_0(F_1(...(F_{n-1})...))$.

We say a (P, F)-derivation is *successful with certainty* c if it is successful and the computed certainty is greater than or equal to c.

In preparation for following discussions, we define a power of Tc_p as $Tc_p{}^0(\varnothing)=\varnothing$ and $Tc_p{}^{m+1}(\varnothing)=Tc_p(Tc_p{}^m(\varnothing))$, and use $\cup_{i<\omega}Tc_p{}^i(\varnothing)$ to denote the infinite union of $Tc_p{}^i(\varnothing)$ for all natural numbers i. Then we have the following theorem which means the soundness of a successful (P, F)-derivation.

Theorem 4.1 Let P be a logic program with uncertainties. For a ground atom A, if $\leftarrow A$ has a successful (P, F)-derivation with certainty c, then A is true in $\cap M(P)$ with c.

(Proof) Suppose that, for a ground atom A, $\leftarrow A$ has a successful (P, F)-derivation of length n with certainty c : $<G_0=\leftarrow A, C_0, \theta_0, F_0>$, $<G_1, C_1, \theta_1, F_1>$,..., $<G_{n-1}, C_{n-1}, \theta_{n-1}, F_{n-1}>$, $<G_n=\leftarrow, , , >$ and $F_0(F_1(...(F_{n-1})...))\geq c$. Then we prove by induction on natural number n that $\{<A, c>\}\leq Tc_p{}^n(\varnothing)$.

Suppose first that n=1. This means that C_0 is a unit clause of the form $<A_1\leftarrow, f>$ and $A=A_1\theta_0$. By the above definition, the computed certainty F_0 is $f(\varnothing)$, and by the definition of Tc_p, $\{<A, c>\}\leq\{<A, f(\varnothing)>\}\leq Tc_p{}^1(\varnothing)$. Next suppose that the result holds for n−1. Suppose that C_0 is a clause of the form $<B_0\leftarrow B_1,...,B_m, g>$ in P such that $A=B_0\theta_0$, and $F_1(...(F_{n-1})...)$ is $[c_1,...,c_m]$. Let δ be any ground substitution. Then $\leftarrow B_i\theta_0...\theta_{n-1}\delta$ has a successful (P, F)-derivation of the same length of $\leftarrow B_i\theta_0$'s one such that the computed certainty is c_i $(1\leq i\leq m)$. By the induction hypothesis, $\{<B_1\theta_0...\theta_{n-1}\delta, c_1>,...,<B_m\theta_0...\theta_{n-1}\delta, c_m>\}\leq Tc_p{}^{n-1}(\varnothing)$. By the definition of Tc_p, $\{<A, c>\}\leq\{<A, g(c_1,...,c_m)>\}\leq Tc_p{}^n(\varnothing)$.

So A is true in $Tc_p{}^n(\varnothing)$ with c. On the other hand, $Tc_p{}^n(\varnothing) \leq \cup_{i<\omega}Tc_p{}^i(\varnothing) \leq lfp(Tc_p)$ because Tc_p is monotonic increasing. Thus A is true in $lfp(Tc_p)$ with c. Hence A is true in $\cap M(P)$ with c. □

In a standard logic program P, the least model for P, the least fixpoint of T_p, and the infinite union of the increasing sequence of sets $\varnothing\subseteq T_p(\varnothing)\subseteq T_p{}^2(\varnothing)\subseteq$... are the same where T_p is a transformation associated with P. Then as stated in [3], for an atom A in the least model for P, it follows immediately that there exists a natural number N such that $A\in T_p{}^N(\varnothing)$. Because T_p can be regarded as an operator adding one-step modus pones consequences to its argument set, we can show that a finite proof of A exists according to a given proof procedure and establish a completeness result for a proof procedure. However we do not generally have the completeness of a successful (P, F)-derivation in logic programs with uncertainties. For $<A, c>$ in

$\cup_{i<\omega} Tc_p{}^i(\varnothing)$, it does not follow that there is a natural number N such that <A, c> in $Tc_p{}^N(\varnothing)$.

Example Consider the logic program with uncertainties

$$P = \{<p \leftarrow q(X), f1(\{y\}) = y>,$$
$$<q(a) \leftarrow, f2(\{y\}) = 0.2>,$$
$$<q(s(X)) \leftarrow q(X), f3(\{y\}) = (2 \times y - y^2)^{\frac{1}{2}}\}.$$

Of course, f1, f2 and f3 are monotonic increasing. However, $<p, 0.2> \in Tc_p{}^2(\varnothing)$, $<p, 0.6> \in Tc_p{}^3(\varnothing)$, $<p, 0.916> \in Tc_p{}^4(\varnothing)$, $<p, 0.996> \in Tc_p{}^5(\varnothing)$,.... That is to say, the bigger the power of Tc_p is, the greater the certainty of p is. Thus $<p, 1> \in \cup_{i<\omega} Tc_p{}^i(\varnothing)$, however there is no natural number N such that $<p, 1> \in Tc_p{}^N(\varnothing)$.

With the restriction on a certainty function f such that $f(\{c_1,...,c_m\}) \leqq d \times \min(c_1,...,c_m)$ for some certainty factor d ($0<d<1$), the completeness can now be achieved.

Lemma 4.2 Let P be a logic program with uncertainties. Assume that for every certainty function f, $f(\{c_1,...,c_m\}) \leqq d \times \min(c_1,...,c_m)$ for some d ($0<d<1$). Then for <A, c> in $\cup_{i<\omega} Tc_p{}^i(\varnothing)$, there exists some natural number N such that <A, c> in $Tc_p{}^N(\varnothing)$.

(Proof) Because of the restriction on every certainty function f such that $f(\{c_1,...,c_m\}) \leqq d \times \min(c_1,...,c_m)$ for some d ($0<d<1$), the certainty c' of any atom A' such that <A', c'> in $Tc_p{}^h(\varnothing)$ is less than d^h. Then there exists some natural number $N \leqq m$ such that <A, c> in $Tc_p{}^N(\varnothing)$ where m is the smallest natural number for which $d^m < c$. □

Lemma 4.3 Let P be a logic program with uncertainties. Assume that for every certainty function f, $f(\{c_1,...,c_m\}) \leqq d \times \min(c_1,...,c_m)$ for some d ($0<d<1$). Then $\cup_{i<\omega} Tc_p{}^i(\varnothing) = lfp(Tc_p)$.

(Proof) By the Knaster-Tarski fixpoint theorem, it is sufficient to show that $Tc_p(\cup_{i<\omega}Tc_p{}^i(\varnothing)) \leq \cup_{i<\omega}Tc_p{}^i(\varnothing)$. Suppose that, for a ground atom A, $<A, c> \in Tc_p(\cup_{i<\omega}Tc_p{}^i(\varnothing))$. By the definition of Tc_p, $c = \sup\{f(\{c_1,...,c_n\}) \mid A \leftarrow B_1,...,B_n$ is a ground instance and f is the certainty function of a clause in P such that $<B_j, c_j> \in \cup_{i<\omega}Tc_p{}^i(\varnothing)$ for $1 \leq j \leq n\}$. By lemma 4.2, $c = \sup\{f(\{c_1,...,c_n\}) \mid A \leftarrow B_1,...,B_n$ is a ground instance and f is the certainty function of a clause in P such that $<B_j, c_j> \in Tc_p{}^N(\varnothing)$ for $1 \leq j \leq n\}$ for some natural number N. Then $<A, c> \in Tc_p{}^{N+1}(\varnothing) \leq \cup_{i<\omega}Tc_p{}^i(\varnothing)$. Hence $Tc_p(\cup_{i<\omega}Tc_p{}^i(\varnothing)) \leq \cup_{i<\omega}Tc_p{}^i(\varnothing)$ and so $\cup_{i<\omega}Tc_p{}^i(\varnothing) = lfp(Tc_p)$. □

Lemma 4.4 (lifting lemma)
Let P be a logic program with uncertainties, A be an atom and θ be a substitution. Suppose there exists a successful (P, F)-derivation of ←Aθ. Then there exists a successful (P, F)-derivation of ←A of the same length and the same computed certainty.

(Proof) Let $<G_i, C_i, \theta_i, F_i>$ $(i=0,1,2,...,n)$ be the successful (P, F)-derivation of ←A. We may assume θ does not act on any variables of the first input clause C_0. Now $\theta\theta_0$ is a unifier of the head of C_0 and the atom A. Then the result of deriving ←A and C_0 using $\theta\theta_0$ is exactly G_1. Thus we obtain a successful (P, F)-derivation $<G'_i, C'_i, \theta'_i, F'_i>$ $(i=0,1,2,...,n)$ of ←Aθ such that G'_i, C'_i, θ'_i, and F'_i are equal to G_i, C_i, θ_i, and F_i respectively for $i=0,1,2,...,n$ except that $G'0 = \leftarrow A\theta$ and $\theta'_0 = \theta\theta_0$. Clearly the computed certainty $F'_0(F'_1(...(F'_{n-1})...))$ is equal to the computed certainty $F_0(F_1(...(F_{n-1})...))$. □

Theorem 4.5 Let P be a logic program with uncertainties. Assume that for every certainty function f, $f(\{c_1,...,c_m\}) \leq d \times \min(c_1,...,c_m)$ for some d $(0<d<1)$. For a ground atom A, if A is true in ∩M(P) with c, then ←A has a successful (P, F)-derivation with certainty c.

(Proof) Suppose that, for a ground atom A, A is true in ∩M(P) with c. By theorem 3.4, A is true in $lfp(Tc_p)$ with c and, by lemma 4.3, A is true in $\cup_{i<\omega}Tc_p{}^i(\varnothing)$ with c.

Then there is a $<A, c'>$ in $\cup_{i<\omega}Tc_p{}^i(\emptyset)$ such that $c \leqq c'$. By lemma 4.2, for some natural number N, $<A, c'>$ in $Tc_p{}^N(\emptyset)$. Then we prove by induction on natural number N that $\leftarrow A$ has a successful (P, F)-derivation whose length is at most N and whose computed certainty is equal to c'.

Suppose first that $N=1$. Since P is finite, $Tc_p{}^1(\emptyset)$ is finite, and for $<A, c'>$ in $Tc_p{}^1(\emptyset)$, c' must be attained for some certainty function of a clause in P, say, $<A_1\leftarrow, f>$ where A and A_1 are unifiable and $f(\emptyset)=c'$. Then $\leftarrow A$ has a successful (P, F)-derivation of length 1 and the computed certainty is c' such that the first input clause of it is $<A_1\leftarrow, f>$. Next suppose that the result holds for $N-1$. Since P is finite and by the definition of Tc_p, there exists a ground instance of a clause $<B_0\leftarrow B_1,...,B_m, g>$ in P such that $A=B_0\theta$ and $\{<B_1\theta, c'_1>,...,<B_m\theta, c'_m>\} \subseteq Tc_p{}^{N-1}(\emptyset)$ and $c'=g(\{c'_1,...,c'_m\})$, for some ground substitution θ. By the induction hypothesis, $\leftarrow B_i\theta$ has a successful (P, F)-derivation whose length is at most $N-1$ and whose computed certainty is equal to c'_i ($1\leqq i\leqq m$). Let θ_0 be the most general unifier of A and B_0. Because $\theta=\theta_0\delta$ for some substitution δ and by the lifting lemma, $\leftarrow B_i\theta_0$ has a successful (P, F)-derivation whose length and computed certainty are equal to $\leftarrow B_i\theta$'s one ($1\leqq i\leqq m$). Then $\leftarrow A$ has a successful (P, F)-derivation whose length is at most N and whose computed certainty is equal to $g(\{c'_1,...,c'_m\})$, i.e. c', such that the first input clause of it is $<B_0\leftarrow B_1,...,B_m, g>$.

Hence $\leftarrow A$ has a successful (P, F)-derivation with the certainty c. □

5. Negation as failure in logic programs with uncertainties

In standard logic programs, the negation as failure rule is used to deduce negative information. This rule states that if all derivations of $\leftarrow A$ are finitely failed, then infer $\neg A$. For a standard logic program P, the finite-failure set of P is the set of all such ground atom A. Moreover the finite-failure set is the complement of $\cap_{i<\omega}T_p{}^i(HB)$ where T_p is a transformation associated with a standard logic program

P and HB stands for the Herbrand base for P which is the set of all ground atoms which can be formed out of predicates, functions and constants in P (see [8] for detail).

Now we define the finite-failure set FF(c) of a logic program with uncertainties P, which is actually the implementation of finite failure and can be used to infer the negation as failure. Later we will show other characterization of FF(c).

Before giving the definition of FF(c), we first define a proof procedure with certainty threshold c, augmented with a mechanism of pruning computations for which it is evident that any proof found along them will not meet this threshold c.

Definition Let P be a logic program with uncertainties, A be an atom and c be a certainty factor. A *(P, F, C)-derivation with certainty threshold* c of a goal ←A is a sequence of quadruples $<G_i, C_i, \theta_i, F_i>$, $i=0,1,2,\ldots$ such that:

① $G_0 = \leftarrow A$,

② G_i is of the form $\leftarrow(B_1,\ldots,B_m)$ where $m \geqq 0$ and B_j is an atom ($1 \leqq j \leqq m$),

③ C_i is a list of m clauses $(A^{(1)} \leftarrow D^{(1)}{}_1,\ldots,D^{(1)}{}_{n_1}, f^{(1)}),\ldots,(A^{(m)} \leftarrow D^{(m)}{}_1,\ldots,D^{(m)}{}_{n_m}, f^{(m)})$,

④ θ_i is a most general unifier of $(B_1,\ldots,B_m)$ and $(A^{(1)},\ldots,A^{(m)})$, and

⑤ G_{i+1} is $\leftarrow(D^{(1)}{}_1,\ldots,D^{(1)}{}_{n_1},\ldots,D^{(m)}{}_1,\ldots,D^{(m)}{}_{n_m})\theta_i$

and F_i is $[f^{(1)}(\{X_1,\ldots,X_{n_1}\}),\ldots,f^{(m)}(\{X_1,\ldots,X_{n_m}\})]$

and $F_0(F_1(\ldots(F_i(\varnothing))\ldots) \geqq c$ where the value of $F_i(\varnothing)$ is defined as $[f^{(1)}(\varnothing),\ldots,f^{(m)}(\varnothing)]$.

In the above derivation, the condition of $F_0(F_1(\ldots(F_i(\varnothing))\ldots) \geqq c$ is checked at each level i of $<G_i, C_i, \theta_i, F_i>$ and is the only difference of (P, F, C)-derivation from (P, F)-derivation. A successful (P, F ,C)-derivation with certainty threshold c is defined in the same way as the (P, F)-derivation in section 4. The semantics of a successful (P, F, C)-derivation with certainty threshold c of a goal ←A is "A is provable from P with certainty c', and c'≧c ", which is same as the semantics of "solve(A, c, c')"in [9].

Note that any (P, F, C)-derivation with certainty threshold c always becomes finite on the condition in section 4 that, for every certainty function f, $f(\{c_1,...,c_m\}) \leqq d \times \min(c_1,...,c_m)$ for some d ($0<d<1$). More precisely, as stated in [1] and [9], the length of any (P, F, C)-derivation with certainty threshold c would be no more than the smallest natural number n for which $d^n<c$.

Definition Let P be a logic program with uncertainties, A be an atom and C be a certainty factor. A (P, F, C)-derivation with certainty threshold c of a goal ←A is said to be *finitely failed* if the derivation is finite and ends with a goal G_i where the condition $F_0(F_1(...(F_{i-1}(\varnothing))...) \geqq c$ cannot be satisfied or an atom in the goal does not unify with the head of the clause of any pair in P.

The first part of the definition means that a goal ←A cannot be provable from P with any certainty greater than or equal to c. The second part is equal to the one of a standard logic program.

Definition Let c be a certainty factor. The *finite-failure set FF(c)* of a logic program with uncertainties P is the set of all pair $<A, 1-c>$ of a ground atom A and a certainty such that all (P, F, C)-derivations with certainty threshold c of ←A are finitely failed.

Because of the monotonicity of a certainty function, if all (P, F, C)-derivations with certainty threshold c of ←A are finitely failed, ←A never has any successful (P, F)-derivation with certainty c. Other form of this fact will be shown later.

As we stated before, in a standard logic program the finite-failure set is the complement of $\cap_{i<\omega}T_p{}^i(HB)$. We characterize FF(c) in the similar way. First of all, we define the Universe of a logic program with uncertainties P which corresponds to the Herbrand base of standard logic program, and define the complement of an interpretation.

Definition The *Universe* Uc of a logic program with uncertainties P is the set of all pairs <A, 1> where A is a ground atom which can be formed out of predicates, functions and constants in clauses of P.

Definition Let I and I' be interpretations. I' is the *complement* in Uc of I iff for each pair <A, 1> in Uc, <A, 1−c> in I' if there is a pair <A, c> in I and <A, 1> in I' otherwise. We write Uc−I to denote the complement in Uc of I.

Now we come to the major result of this section, which characterizes the finite-failure set FF(c). We define another power of Tc_p as $Tc_p{}^0(Uc)=Uc$ and $Tc_p{}^{m+1}(Uc) = Tc_p(Tc_p{}^m(Uc))$, and use $\cap_{i<\omega}Tc_p{}^i(Uc)$ to denote the infinite intersection of $Tc_p{}^i(Uc)$ for all natural numbers i. We will first prove the following lemma which is concerned with the maximum of certainties of (P, F, C)-derivations. Then we give a characterization of FF(c) as the main theorem.

Lemma 5.1 Let P be a logic program with uncertainties, A be a ground atom, c be a certainty factor and $<G_i, C_i, \theta_i, F_i>$ (i=0,1,2,...) be a (P, F, C)-derivation with certainty threshold c of ←A. Then for a pair <A, d> in $Tc_p{}^j(Uc)$, $F_0(F_1(...(F_{j-1}(\emptyset))...) \leq d$. Furthermore, the maximum of $F_0(F_1(...(F_{j-1}(\emptyset))...)$ of those derivations of ←A is equal to d.

(Proof) We prove it by induction on natural number j.

If j=1, then $F_0(\emptyset) = f(\emptyset)$ (where f is the certainty function of the input clause C_0)

$\leq \sup\{f(\emptyset) \mid A \leftarrow B_1,...,B_n$ is a ground instance and f is the certainty function of a clause in P$\}$

$= \sup\{f(\emptyset) \mid A \leftarrow B_1,...,B_n$ is a ground instance and f is the certainty function of a clause in P such that $<B_i, 1> \in Uc$ for $1 \leq i \leq n\}$

$= \sup\{f(S) \mid A \leftarrow B_1,...,B_n$ is a ground instance and f is the certainty function of a clause in P such that $<B_i, 1> \in Uc$ for $1 \leq i \leq n$ and $S = \{1,...,1\} = \emptyset\}$

$= \sup\{f(S) \mid A \leftarrow B_1,...,B_n$ is a ground instance and f is the certainty function of a clause in P such that $<B_i, c_i> \in Uc$ for $1 \leq i \leq n$ and $S = \{c_1,...,c_n\}\}$

$= d.$

Furthermore, since P is finite, the last set over which the supremum is taken is finite. Therefore the supremum must be attained for some clause in P, say, $<A' \leftarrow B'_1,...,B'_n, f>$. Hence the maximum of $F_0(\emptyset)$ of those derivations of $\leftarrow A$ is equal to d where the maximum is attained for the derivation of $\leftarrow A$ such that its C_0 is the clause $<A' \leftarrow B'_1,...,B'_n, f>$.

Now suppose that the result holds for $j-1$. Suppose that C_0 is a clause of the form $<B_0 \leftarrow B_1,...,B_n, g>$ in P such that $A = B_0\theta_0$, and $F_1(...(F_{j-1}(\emptyset))...)$ is $[c_1,...,c_n]$. Let δ be any ground substitution. Then $\leftarrow B_k\theta_0...\theta_{j-1}\delta$ has a derivation $<G^{(k)}_i, C^{(k)}_i, \theta^{(k)}_i, F^{(k)}_i>$ $(i = 0,1,2,...)$ such that $F^{(k)}_0(F^{(k)}_1(...(F^{(k)}_{j-2}(\emptyset))...)) = c_k$ $(1 \leq k \leq n)$. Then by the induction hypothesis, for $<B_1\theta_0...\theta_{j-1}\delta, d_1>,...,<B_n\theta_0...\theta_{j-1}\delta, d_n>$ in $Tc_P^{j-1}(Uc)$, c_k is less than or equal to d_k $(1 \leq k \leq n)$. Then

$(F_0(F_1(...(F_{j-1}(\emptyset))...)$ of the derivation of $\leftarrow A)$

$= g(\{c_1,...,c_n\})$

$\leq g(\{d_1,...,d_n\})$

$\leq \sup\{f(S) \mid A \leftarrow B_1,...,B_n$ is a ground instance and f is the certainty function of a clause in P such that $<B_i, c_i> \in Tc_P^{j-1}(Uc)$ for $1 \leq i \leq n$ and $S = \{c_1,...,c_n\}\}$

(by the induction hypothesis)

$= d.$

Furthermore, since d_k is the maximum of $F^{(k)}_0(F^{(k)}_1(...(F^{(k)}_{j-2}(\emptyset))...)$ of derivations of $\leftarrow B_k\theta_0...\theta_{j-1}\delta$ $(1 \leq k \leq n)$ by the induction hypothesis and P is finite, it is clear that the maximum of $(F_0(F_1(...(F_{j-1}(\emptyset))...)$ of those derivations of $\leftarrow A$ is equal to d. □

Theorem 5.2 Let P be a logic program with uncertainties, A be a ground atom and c be a certainty factor. If all (P, F, C)-derivations with certainty threshold c of $\leftarrow$A are finitely failed, then A is true in $Uc - \cap_{i<\omega} Tc_p^i(Uc)$ with certainty $1-c$.

(Proof) Let m be the length of any finitely failed (P, F, C)-derivations with certainty threshold c of $\leftarrow$A. If the derivation ends with a goal G_m where the condition $F_0(F_1(...(F_{m-1}(\emptyset))...) \geqq c$ cannot be satisfied, then $F_0(F_1(...(F_{m-1}(\emptyset))...) < c$, and then the maximum of $F_0(F_1(...(F_{m-1}(\emptyset))...)$ of those derivations, say max $F_0(F_1(...(F_{m-1}(\emptyset))...)$, is less than c. On the other hand, by lemma 5.1, max $F_0(F_1(...(F_{m-1}(\emptyset))...) =$ d where <A, d> in $Tc_p^m(Uc)$, and hence $d<c$. By the definition of complement, for <A, d'> in $Uc - \cap_{i<\omega} Tc_p^i(Uc)$, $d' \geqq 1-d > 1-c$. Therefore A is true in $Uc - \cap_{i<\omega} Tc_p^i(Uc)$ with certainty $1-c$.

If the derivation ends with a goal where an atom in the goal does not unify with the head of the clause of any pair in P, then there is no pair <A, c> in $\cap_{i<\omega} Tc_p^i(Uc)$, and by the definition of complement, <A, 1> in $Uc - \cap_{i<\omega} Tc_p^i(Uc)$. Thus A is true in $Uc - \cap_{i<\omega} Tc_p^i(Uc)$ with certainty $1-c$. □

Corollary 5.3 Let P be a logic program with uncertainties and c be a certainty factor. Then $FF(c) \leq Uc - \cap_{i<\omega} Tc_p^i(Uc)$.

By the above result, we can state that the negation as failure rule in logic programs with uncertainties is the rule that if all derivations of $\leftarrow$A cannot be successful with any certainty greater than or equal to certainty threshold c, then infer that $\neg$A is successful with certainty $1-c$.

Corollary 5.4 Let P be a logic program with uncertainties, A be a ground atom and c be a certainty factor. If A is true in FF(c) with certainty c', then A is not true in $\cap M(P)$ with certainty $1-c'$.

(Proof) It is straightforward by the fact that $\cup_{i<\omega} Tc_p^i(\emptyset) \leq lfp(Tc_p) \leq \cap_{i<\omega} Tc_p^i(Uc)$ and corollary 5.3 (see [8] for detail). □

The above result shows a *soundness* of our negation as failure rule in logic programs with uncertainties.

6. Certainties as Control

As Kowalski stated in [7], an algorithm can be regarded as consisting of a logic component, which specifies the knowledge to be used in solving problems, and a control component, which determines the problem-solving strategies by means of which that knowledge is used. Then certainties can be used for a control component. One of implementations for it is a best first search algorithm in logic programs with uncertainties [1], which is the algorithm to find the successful derivation with the best (largest) certainty. In standard PROLOG systems, the control is the top-down, left-to-right search and it is fixed so that we cannot change it. However, the efficiency of an algorithm can often be improved by improving the control component without changing the logic of the algorithm. Hence various search algorithms for a logic program with uncertainties give us various control components and efficiencies for it.

7. Conclusions

In this paper, we have first given model-theoretic semantics and fixpoint semantics for a logic program with uncertainties, and secondly defined a proof procedure for it and then presented a sufficient condition for certainty functions of clauses to achieve the completeness of the proof procedure (Theorem 4.5). Previously mentioned, all results concluded within the framework of Shapiro's approach can be proved for a whole class of quantitative schemes. Thus our results (except Theorem 4.5) in this paper can be proved for a whole class of quantitative schemes. (To prove Theorem 4.5, an extra condition is needed). For example, when the Bayesian inference rule is taken for a certainty function, our results can be proved for probabilistic reasoning, and when the fuzzy implication function is taken, our results can be proved for fuzzy reasoning. For remarks on related work, another paper dealing with a semantics is Emden's work [3]. Emden shows various results guided by a close analogy between the qualitative and the quantitative case of logic programs. However Emden's method for computing uncertainties is rather special and based on fuzzy theory. Thus Emden's results are proved only for one, but ours for a whole class of quantitative schemes.

Now what does Theorem 4.5 show ? We think that Theorem 4.5 provides a theoretical foundation to support the following Zadeh's argument in [4] : "Attempts to model human reasoning by formal systems of increasing precision will lead to decreasing validity and relevance. Most human reasoning is essentially shallow in nature and does not rely upon long chains of inference unsupported by intermediate data". In other words, this means that in human reasoning processes the longer chain of inferences decreases its certainty. The condition that $f(\{c_1,...,c_m\}) \leqq d \times \min(c_1,...,c_m)$ for some d $(0<d<1)$ provides this assumption. However this restriction seems too strong and therefore is clearly not the necessary condition to conclude the result of Theorem 4.5. We are now investigating other weaker conditions (like $f(\{c_1,...,c_m\}) \leqq \min(c_1,...,c_m)$ for every certainty function f) to achieve Theorem 4.5.

Further, we have introduced the negation as failure into logic programs with uncertainties, which is the rule that if all derivations of $\leftarrow$A cannot be successful with any certainty greater than or equal to certainty threshold c, then infer that $\neg$A is successful with certainty $1-c$. Then we have given an interesting characterization of it. This is the first attempt to consider the negation as failure from the semantical point of view, while [3] only remarks it and does not show any result. It is possible and interesting for us to investigate and characterize such various kinds of derivations .

Acknowledgements

The author would like to thank Dr. Tosio Kitagawa, the president of IIAS-SIS, Dr. Hajime Enomoto, the director of IIAS-SIS, for giving him the opportunity to pursue this work and helping him with it. He is deeply grateful to his colleagues, Hajime Sawamura, Takashi Yokomori, Toshiro Minami, for reading the draft and giving him many valuable comments.

This is part of the work in the major R&D of the Fifth Generation Computer Project, conducted under program set up by MITI.

References

[1] Akama,K. : "Best First Search Prolog", WGAI of IPSJ 47-8 (1986), 57-64 (in Japanese).

[2] Apt,K.R. and van Emden,M.H. : "Contributions to the theory of logic programs", *J. ACM* 29 (1982), 841-862.

[3] van Emden,M.H. : "Quantitative Deduction and its Fixpoint Theory", *J. Logic Programming* 1 (1986), 37-53.

[4] Gaines,B.R. : "Foundations of Fuzzy Reasoning", *Int. J. Man-Machine Studies* 8 (1976), 623-668.

[5] Hinde,C.J. : "Fuzzy Prolog", *Int. J. Man-Machine Studies* 24 (1986), 569-595.

[6] Ishizuka,M. and Kanai,N. : "Prolog-ELF Incorporating Fuzzy Logic", *Proc. 9th IJCAI* (1985), 701-703.

[7] Kowalski,R. : "Algorithm = Logic + Control", *Comm. ACM* 22 (1979), 424-436.

[8] Lloyd,J. : "Foundations of Logic Programming", Springer-Verlag, 1984.

[9] Shapiro,E.Y. : "Logic Programs With Uncertainties : A Tool for Implementing Rule-Based Systems", *Proc. 8th IJCAI* (1983), 529-532.