# PROLOG-ELF INCORPORATING FUZZY LOGIC

Mitsuru ISHIZUKA and Naoki KANAI

Institute of Industrial Science, University of Tokyo
7-22-1, Roppongi, Minato-ku, Tokyo, 106, Japan

ABSTRACT: Prolog-BLF incorporating fuzzy logic and some useful functions into Prolog has been implemented as a basic language for building knowlegde systems with uncertainty or fuzziness. Prolog-EUF inherits all the preferable basic features of Prolog. In addition to assersions with a truth-value between 1.0 and 0.5 (0 for an exceptional case), fuzzy sets can be manipulated very easily to a certain extent. An application to a fuzzy logical database is illustrated.

## 1. INTRODUCTION

As a basic language for building knowledge systems, Prolog has many preferable features, such as pattern matching (unification), automatic backtracking and relatinal database. It has been chosen as the kernel language of Japanese fifth generation computer. Many Prolog-based knowledge systems have become to appear recently.

Prolog, which is based on first-order predicate logic (more precisely, Horn logic), deals with only a two-valued logic. Knowledge sometimes manifests uncertainty in real-world problems. Uncertain knowledge has played important role in many expert systems, such as Mycin[1], Prospector[2], Casnet[3], Speri1[4,5], etc.. When dealing with uncertain knowledge in Prolog, some special programming techniques are required [6], which are annoying for us to build a large system. Therefore, there exists a need for a basic language capable of dealing with uncertain knowledge.

In fuzzy logic, the uncertainty of a fact or a rule is expressed with a truth-value between 1 and 0. There are of course several other methods, such as Mycin's certainty factor[1], subjective Bayesian method[2], and Dempster-Shafer theory[5]. However, unlike these methods, the fuzzy logic has a certain kind of logical feature. This logical feature can be extended to a fist-order predicate logic, which provids us a rich expressive power.

This paper presents a language called Prolog-ELF [7] incorporating fuzzy logic into Prolog. The Prolog-ELF is different from existing fuzzy languages such as described in [8,9], because the Prolog-ELF is a general-purpose language which inherits all the preferable basic features of Prolog. Moreover, several useful predicates are imbedded in the Prolog-ELF. Using these predicates, we can manipulate fuzzy sets very easily to a certain extent.

The Prolog-ELF is implemented on Pascal and completely operational on VAX-11. This paper describes some features and considerations on the Prolog-ELF and its application to a fuzzy logical database.

## 2. TOWARD FUZZY PROLOG

Although fuzzy sets are considered *in* general fuzzy logic [10], we will at first consider logical formulae expressed with symbols in ordinaly sense. The certainty of the logical formula is expressed with a truth-value T, Where 1>T>0. in the fuzzy logic. (A different kind of fuzzy logic in which linguistic truth-values are used is presented in [11].)

Let T(P) be the truth-value of a logical formula P. The following treatments of the truth-value are foundamental. (¬ denotes NOT.)

If P=A and A is an atomic formula: T(P)=T(A)
If P=¬Q : T(P)=1-T(Q)
If P=Q AND R : T(P)=min(T(Q),T(R))
If P=Q OR R : T(P)=max(T(Q),T(R)).

For example, if
T(P)=0.6, T(Q)=0.9, T(R)=0.8 and
S=(P OR Q) AND ¬R
then,
T(S)=min(max(T(P),T(Q)), 1-T(R))
=min(max(0.6,0.9), 1-0.8)
=0.2 .

Most of logical laws, such as commutative, associative, distributive and DeMorgan's laws, are hold in fuzzy logic, except excluded-middle law; that is, T(P AND -P) and T(P OR *-P)* are not always 0 and 1, respectively.

Lee[12] and Mukaidono[13] considered the resolution principle as a mechanical inference of fuzzy logic. If we interprete P->Q as ("P OR Q) in fuzzy logic, the well-known inference rules, i.e., Modus ponens, Modus tollens and syllogism, become the special cases of the resolution principle. By introducing arguments attached to the logical symbol (predicate) and unification, the fuzzy logic can be extended to fuzzy first-order predicate logic.

Lee[12] has proved that, if all the truth-values of parent clauses are greater than 0.5, then a resolvent clause derived by the resolution principle is always meaningful and has a truth-value between the maximum and minimum of those of the parent clauses. Mukaidono[13] showed an interpretation that, even if a truth-value of the parent clause is less than 0.5, a resolvent clause is meaningful in the sense of reducing ambiguity.

Expressions in Prolog are restricted to Horn clauses which have at most one positive literal, so that the efficient linear input resolution can work as a complete resolution. If we define a Horn clause witrh a truth-value less than 0.5 in fuzzy Horn logic, it is equivalent to the negation of the clause which goes beyond the scope of Horn clause. Therefore, in order to realize a fuzzy-Prolog, we basically restrict our logical expression to the Horn clause with a truth-value greater than 0.5. The result of the resolution in this case is guaranteed to be meaningful based on the Lee's paper[12].

It is a problem if the truth-value of a clause including variables changes when the variables are instantiated by the unification. We take however a position that the truth-value of the clause will not change due to the unification.

One major difference in the inference process of fuzzy-Prolog from that of ordinary Prolog is that exhaustive search is often required at an OR-branch, since the maximum truth-value of a literal is looked for at this branch.

## 3. PROLOG-ELF

Prolog-ELF has been implemented based on the above-sentioned principle and by adding some useful functions for a basic language of knowledge systems.

The clauses in Prolog-ELF are expressed in the following forms.

```
        Definition clause: +P-Q・・・-R.   or   +P.
        Goal clause        -Q・・・-R.
```

The truth-value of the clause can be assigned as;

```
        0.7.+P-Q.    or    -assert(0.7:+P-Q.).
```

When the definition of the truth-value is omitted, then the default truth-value is 1. If all the clauses have the truth-value 1, the behavier of the Prolog-ELF is essentially the same as that of ordinary Prolog.

The reason why we have adopted above notation for a clause, i.e., +P-Q-R., instead of a popular notation such as P:-Q,R., is to avoid a question regarding the interpretation of implication. In fuzzy logic, ^PV Q is not only the interpretation of P->X); there are several other interpretations [10,14]. Using our notation, we can make it clear that our Prolog-ELF works starting from axioms defined in clause forms, some of which can be interpreted, if appropriate, into implication forms.

A variable in Prolog-ELF is denoted as a character string headed by *. (Single * can be also a variale.) The result of the execution of a goal clause, if its truth-value is larger than a predetermined threshold (described later), is displayed in the form of, for example;

```
        0.75:-P(apple).
```

Some of the system predicates characterizing Prolog-ELF are as follows.

(1 )THRESH : This predicate sets a value given as its argument to a threshold. The clause with a truth-value less than this threshold is ignored or regarded as false, which initiates the backtracking during the execution process. The default of this threshold is 0.5. A threshold below 0.5 is permitted only for exceptional cases, which suffer a risk of going out of the scope of Horn logic. (See a comment regarding a predicate NOT.)

(2)Mode setting predicates There are three operational modes in Prolog-ELF. The following predicates perform as a switch to change the mode.

(2-1 )NOQUERY : The system searches and displays a first-encountered answer regardless of its truth-value. This is the default mode and the behavior is equivalent to that of ordinary Prolog.

(2-2)QUERY All the answers are searched and displayed.

(2-3)BEST(n) : The system displays top n answers according to their truth-values. In order to improve search efficiency, a dynamic thresholding strategy is adopted in this mode. The threshold is set dynamically to the value of the n-th truth-value from the top of already obtained answers.

(3)USEVALDE This pridicate assigns a truth-value given as its argument to the associated clause. Using this predicate, we *can* represent fuzzy sets very easily as exemplified in Fig.l.

(4)VALUE : This is a deterministic predicate which returns a truth-value of a clause given as its first argument to its second argument. For example, suppose that

```
            0.8:+P(a).
            0.6:+P(b).
```

have been defined. Then, when

```
        -value(-P(+x)., *val).
```

is executed, *x and 8val are at first unified to a and 0.8, respectively. Once the backtrack occurs, *x and *val are then unified to b and 0.6, respectively. Fig.2 shows an interaction exemplifying a usage of the predicate VALUE.

(5)CHVALUE : This predicate updates the truth-value of an associated clause to a new value.

(6)MAXCL, DMAXCL : An exhaustive search for the goal clause with the maximum truth-value is essential at OR-branch in fuzzy logic. In general, this is achieved in the BEST(1) mode in Prolog-ELF. The use of predicates MAXCL and DMAXCL is another way to specify this search. They are also used in sub-goal node for the following reason. For example, if

```
        0.9:+R(*x)-P(*x)-Q(*x).
        0.9:+P(a).
        0.7:+P(b).
        0.6:+Q(a).
        0.8:+Q(b).
```

are defined, then

```
        T(R(a))=min(0.9, min(0.9.0.6))=0.6
        T(R(b))=min(0.9, min(0.7.0.8))=0.7.
```

This means that, if we take an instance of R(*x) with the maximum truth-value, i.e., R(b) in this case, this instantiation does not necessarily correspond to that of P(*x) with the maximum truth-value. There exists a case where we want to fix a sub-goal literal to the unification yielding its maximum truth-value. For example, if we add the following clause to the above definition,

```
        0.9:+S(*x)-maxcl(P(*x).)-Q(*x).
```

then the system operates as follows.

```
        >-maxcl(S(*x).).
         0.6:maxcl(S(a).).
```

MAXCL is a non-deterministic predicate, whereas DMAXCL is a deterministic predicate. That is, if more than one answers with the same maximum truth-value are obtained, an alternative answer is used upon the backtracking in the case of MAXCL, whereas the search fails upon the backtracking in the case of DMAXCEL.

```
+old(*age)-gt(*age,70)-/-usevalue(1).
+old(*age)-gt(*age,50)-/-minus(*age,50,*x)
  -times(*x,0.015,*y)-plus(*y,0.7,*z)-usevalue(*z).
+old(*age)-gt(*age,30)-/-minus(*age,30,*x)
  -times(*x,0.01,*y)-plus(*y,0.5,*z)-usevalue(*z).
```

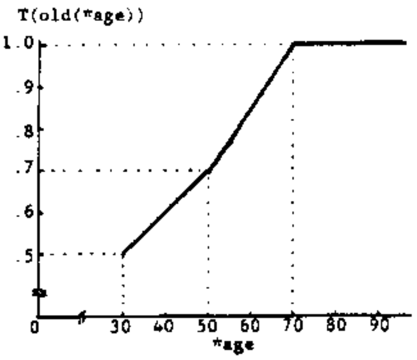

Fig.1 Definition of a fuzzy set — old — using USEVALUE (/ denotes the cut operator) and its graphical representation.

```
>-load(demo_).
>+very(*cl)-value(*cl,*x)-sqr(*x,*y)-usevalue(*y).
success
>-old(60).
 0.8500:-old(60).
>-very(-old(60).).
 0.7225:-very(-old(60).).
```

Fig.2 An example of the usage of VALUE.
(> is a system prompt.)
The definition of Fig.1 is assumed.

(7)NOT : As in ordinary Prolog, NOT fails in the normal threshold setting of Prolog-ELF if there exists at least one answer. However, if the threshold setting is less than 0.5. the result of unification during the execution of NOT may remain in Prolog-ELF. For example, suppose that the threshold is 0.2 and the answer of dmaxcKP(x). ) is 0.7:dmaxcKP(a).), then not(P:>x). ) will succeed with the truth-value of 0.3(-1-0.7) and with the unification that +x is a.

## 4. APPLICATION TO A *FUZZY* LOGICAL DATABASE

As an application of Prolog-ELF, a fuzzy logical database including fuzzy rules and fuzzy set concepts is illustrated in Fig.3, where a person who may possibly have a second house is retrieved from stored facts and heuristic rules. Several other applications including a production system have been written by the authors.

## 5. CONCLUDING REMARKS

Prolog-ELF incorporating fuzzy logic and related some useful functions has been implemented as a basic language of knowledge systems with uncertainty or fuzziness.

One problem of the Prolog-ELF is its slow speed, since the exhaustive search is often required at the OR node. One way to avoid this problem is to set the threshold to an appropriatly high value. Some mechanisms to improve search efficiency are planned to implement in a future version; but it seems that ultimate solution may be a parallel processors like Japanese fifth-generation computer. A knowledge management mechanism using meta-predicates is scheduled to be added to Prolog-ELF.

ACKNOWLEDGMENT: The Prolog-ELF has been implemented with reference to the program of Dr. H.Nakashima at Electrotechmcal Lab., who is the author of Prolog/KR. The authors are grateful to him.

This work was supported by Grant-in-aid tor Special Project Research, Ministry of Education, Japan.

## REFERENCES

[I] E.H.Shortliffe. "Computer-Based Medical Consultation: Mycin,* American Elsevier, New York, 1976
[2] R.O.Duda, P.Hart. N.J.Nilson, "Subjective Bayesian Methods for Rule-Based Inference Systems," NCC, 1S76
[3] S.M.Weiss, C.A.Kulikowski et al., "A Model-Based Method for Computer-Aided Medical Decision-Making," Artificial Intell., 11, pp.145-172, 1978
[4] M.Ishizuka, K.S.Fu, J.T.P.Yao. "Rule-Based Damage Assessment System for Existing Structures," Solid Mechanics Archives, 8, pp.99-118, 1983
[5] M.Ishizuka, "Inference Methods Based on Extended Dempster & Shafer's Theory for Problems with Uncertainty/Fuzziness, New Generation Computing. 1. pp.159 168, 1983
[6] E.Y.Shapiro, "Logic Programs with Uncertainties: A Tool for Implementing Rule-Based Systems," 8th IJCAI, 1983
[7] N.Kanai, M.Ishizuka, "Prolog-ELF incorporations Fuzzy Logic," Report of Research Group on KE and AI, Info. Soc. of Japan, 34-4, 1984
[8] L.A.Zadeh. "PRUF–A Meaning Representation Language for Natural Language," Int'l J. Man-Machine Studies, 10, pp.395-460. 1978
[9] M.Umano, M.Mizumoto, K.Tanaka, "FSTDS System. A Fuzzy-Set Manipulation System," Info. Sci., 14, pp.115 159, 1978
[10] D.Dubois. H.Prade. "Fuzzy Set and Systems: Theory and Applications." Academic Press, 1980
[II] L.A.Zadeh. "Fuzzy Logic and Approximate Reasoning," Synthese. 30, pp.407-428. 1978
[12] R.C.T.Lee, "Fuzzy Logic and the Resolution Principle," J.ACM, 19. pp.109-119, 1972
[13] M.Mukaidono. "Fuzzy Inference of Resolution Style," in Fuzzy Set and Possibility Theory (R.R.Yager Ed.), Pergamon Press, 1982
[14j T.Whalen, B.Schott, "Issues in Fuzzy Production Systems." Int'l J. Man-Machine Studies. 19, pp.57-71, 1983

```
; Have a second house ;
+have_a_second_house(Kato).
0.9:+have_a_second_house(*who)-rich(*who).
0.7:+have_a_second_house(*who)-president(*who).

; Rich ;
+rich(*who)-income(*who,*yen)-rich2(*yen).
+rich2(*yen)-gt(*yen,30000000)-/.
+rich2(*yen)-gt(*yen,15000000)-/
        -minus(*yen,1.5e7,*x)-rdiv(*x,1e8,*y)
        -times(3,*y,*z)-plus(0.5,*z,*w)
        -usevalue(*w).

; Income ;
+income(Yamada,20000000).
+income(Sato,25000000).
+income(Nomura,8000000).
+income(Kobayashi,40000000).

; President ;
+president(Suzuki).
+president(Kobayashi).
```

(a) database

```
%elf
Prolog-ELF version 2.1a                    6/12/84
    written by N. Kanai,  Ishizuka lab.
    type "-help." for help, "-exit." for exit.
>-load(house_).
>-have_a_second_house(*who).
 1.0000:-have_a_second_house(Kato).
>-best(5).
 1.0000:-best(5).
>-have_a_second_house(*who).
 1.0000:-have_a_second_house(Kato).
 0.9000:-have_a_second_house(Kobayashi).
 0.8000:-have_a_second_house(Sato).
 0.7000:-have_a_second_house(Suzuki).
 0.6500:-have_a_second_house(Yamada).
```

(b) execution

Fig.3 An application to a logical database.