

A Prolog implementation of a fuzzy algebraic structure for the uncertainty modelling

Antonio Gisolfi

Dipartimento di Informatica ed Applicazioni
Università di Salerno ITALY

Abstract

An appropriate algebraic fuzzy structure is introduced to cope with the problem of uncertainty management. Then the structure is implemented in Prolog and an example regarding medical diagnoses is illustrated. In this way we have got a solution to the problem of uncertainty modelling as regards the expert systems environments.

can have suitable mechanisms that allow fuzziness to spread appropriately during the reasoning process [1,2,3,5,8,9,10]. Thus such logic can provide a structure for the uncertainty modelling in expert systems. It includes both two-valued and multi-valued logics, it permits to represent both fuzzy statements and not, and moreover linguistic labels, whose values are words of the natural language, are allowed. Finally, one has fuzzy quantifiers which can be considered as fuzzy numbers which characterize the cardinality of the fuzzy sets [7].

Introduction

The design of expert systems represents today a major area of Artificial Intelligence. An expert system is a computer-based system that is able to solve complex problems in specific domains showing a competence level comparable with that of human experts. As an expert system should emulate the human reasoning, it should consider the uncertainty of knowledge and the problem of its representation. Thus it is of the uttermost importance to provide suitable logical tools that allow approximate reasoning.

The structure of an expert system include a knowledge base containing facts and rules encoded in an appropriate way. It is duty of the inference engine to activate the rules so that the solution of the problem can be achieved.

In particular we focus our attention on the knowledge base. If the knowledge base contains uncertainty, it will include imprecise facts and rules which cannot be represented by classical formalisms such as production rules, frames or semantic nets since the latter do not provide mechanisms for uncertainty modelling. We recall that the traditional methods devised to manage the uncertainty are based on probability theory or consider suitable certainty factors associated with each proposition. These uncertainty measures do not cope with the possible presence of fuzziness. We note that the fuzzy logic can adequately manage uncertainty modelling and moreover one

Prolog and the uncertainty management

Since its beginnings in the early seventies, the Prolog language (PROgramming in LOGic) has been used by many people for applications of symbolic computation and in many areas of Artificial Intelligence. Prolog is a descriptive language: the programmer describes known facts and relationships about a problem and it is duty of Prolog to carry out the computation by inferring new facts from the given ones. A Prolog program is a collection of clauses: each clause is either a *fact* about the given problem or a *rule* specifying the inference that can be performed from the given facts. Then we can ask questions and Prolog will answer the question by means of inferences from one fact to another.

However, we note that Prolog lacks a mechanism to handle fuzzy information. To cope with this problem one can use predicates which manage the uncertainty. Suitable variables, concerning the certainty factors, can be added to the clauses.

Alternatively, uncertainty can be immersed in the Prolog interpreter itself, so that uncertainty can be automatically handled [6]. In this way the language FProlog was developed and goals such as shorter code and understandable programs were achieved. This interpreter can be suitably modified in order to get an expert system that not only solves a problem but also provides an uncertainty measure of the solution.

Finally, a third methodology can be devised which is based on the fact that the human knowledge includes statements partially true and can be communicated through a language containing imprecise symbols. More precisely one has a system based on supported logic programming which allows handling uncertainty within the traditional logic programming. In such system a particular conclusion does not follow from some axioms but it is supported to some extent by means of the evidence. The support grades are not probability values. A supported logic program consists of a set of clauses of the form:

$$A \leftarrow B_1, B_2, \dots, B_n : [S_n, S_p]$$

where A is an atom and B_1, \dots, B_n are literals, i.e. positive or negative atoms. In fact, the above clause can be viewed as an ordinary Prolog clause plus the support couple $[S_n, S_p]$. This clause can be interpreted as follows: for each assignment to the variables present in the clause, if the literals are all true A is supported with grade S_n and NOT A is supported with grade $(1 - S_p)$ such that

$$S_n + (1 - S_p) \leq 1$$

When $S_n = S_p = 1$ one has the conventional Prolog clauses. A unit clause has the form $A : [S_n, S_p]$. In fact, the measure of uncertainty associated with the support of the couple of any rule is given by the difference $(S_p - S_n)$.

In the following we shall present an appropriate algebraic structure named *C-fcalculus* and we shall investigate the relationships between *C-fcalculus* and the theory of Approximate Reasoning [4].

Composite fuzzy sets and C-fcalculus

The basic elements of the structure are strings whose elements are fuzzy sets and whose ordering inside the string is not random but it is induced by an ordering relation that holds between the elements, each having a linguistic label.

More precisely, a C-fset is defined as follows:

let $\alpha_n, \alpha_{n-1}, \dots, \alpha_1$ be a family of fuzzy sets in X , then the corresponding *C-fset* is described by the string whose elements are the fuzzy numbers corresponding to each fuzzy sets.

Thus a C-fset is a string of fuzzy numbers associated with each fuzzy set and the ordering relation holds among the fuzzy numbers. Recalling that in fuzzy logic the truth values are

fuzzy numbers represented by linguistic labels, we can affirm that a C-fset consists of truth values which are fuzzy numbers. Determining the grade of membership of an element to one of the sets the string consists of, i.e. assigning a truth value, it is equivalent to assign a fuzzy set. Therefore the string can be considered as a fuzzy set of second type, whose values of the membership function are fuzzy numbers. The elements belonging to the same set are equivalent, i.e. to each of them the same linguistic label (fuzzy number) is assigned. Thus the sets are ordered according to the ordering relation induced by the fuzzy numbers

$$B \supseteq A \text{ iff } A \sqcap B = A \text{ iff } A \sqcup B = B$$

where \sqcup denotes the union of fuzzy numbers and \sqcap denotes their intersection.

As a consequence of the above discussion, a C-fset can be viewed as formed by two "parallel" C-fsets:

- the first, *primary*, consists of ordinary sets
 - the second, *secondary*, consists of fuzzy numbers.
- Then let us introduce some suitable operations that allow getting a new C-fset starting from the primary and the secondary ones.

The operations will be denoted by the symbols \oplus and \otimes as the formal structure of arithmetic addition and multiplication is preserved.

As regards the primary C-fset the operations \oplus and \otimes denote the set-theoretic union and intersection, respectively. As regards the secondary C-fset we have that the operation \sqcup and \sqcap denote the union and intersection of fuzzy numbers, respectively. However, the collection of all C-fsets constitutes a distributive lattice with respect to the operations so far defined.

Now we aim at showing how C-fcalculus can supply an inferential process and a knowledge representation such that it can be regarded as a possible alternative to the theory of AR. The statements object of study in the theory of AR will be represented by means of C-fsets and then the operations of the C-fcalculus will be utilized in order to obtain an appropriate inference rule.

In fact we note that nested fuzzy statements of the form $(u \text{ is } A) \text{ is } \tau$

where A is a unary fuzzy relation and τ is a truth value, can be represented by a C-fset with an appropriate ordering among the truth values. We can collect all the elements having the same value of τ , i.e. satisfying A to the same extent. If we suppose that the values are comparable, we get an ordering relation among the sets corresponding to each value τ and the resulting string of fuzzy sets represents the following C-fset:

$$C_1 = \alpha_n^{\tau_n} \alpha_{n-1}^{\tau_{n-1}} \dots \alpha_1^{\tau_1}$$

where $\alpha_n, \alpha_{n-1}, \dots, \alpha_1$ are the sets associated with the elements showing the same τ_i ($i=1,2,\dots,n$) and the τ_1, \dots, τ_n denote the grades with which the elements satisfy the relation A.

Let us consider another C-fset

$$C_2 = \beta_m \delta_m \beta_{m-1} \delta_{m-1} \dots \beta_1 \delta_1$$

obtained starting from the proposition

(x is A) is δ and then let us apply the operations

\otimes first to the primary C-fsets and then to the secondary ones.

Suppose that $n=3$ and $m=2$, then one gets

$$\begin{array}{cccc} \alpha_3 & \alpha_2 & \alpha_1 & \otimes \\ & \beta_2 & \beta_1 & = \\ \hline \beta_1 \cap \alpha_3 & \beta_1 \cap \alpha_2 & \beta_1 \cap \alpha_1 & \\ \beta_2 \cap \alpha_3 & \beta_2 \cap \alpha_2 & \beta_2 \cap \alpha_1 & \end{array}$$

$$\beta_2 \cap \alpha_3 [(\beta_1 \cap \alpha_3) \cup (\beta_2 \cap \alpha_2)] [(\beta_1 \cap \alpha_2) \cup (\beta_2 \cap \alpha_1)] \beta_1 \cap \alpha_1$$

Let us consider instead what happens to the associated fuzzy numbers

$$\begin{array}{cccc} \tau_3 & \tau_2 & \tau_1 & \otimes \\ & \delta_2 & \delta_1 & = \\ \hline \tau_1 \cap \delta_3 & \tau_1 \cap \delta_2 & \tau_1 \cap \delta_1 & \\ \tau_2 \cap \delta_3 & \tau_2 \cap \delta_2 & \tau_2 \cap \delta_1 & \end{array}$$

$$\tau_2 \cap \delta_3 [(\tau_1 \cap \delta_3) \cup (\tau_2 \cap \delta_2)] [(\tau_1 \cap \delta_2) \cup (\tau_2 \cap \delta_1)] \tau_1 \cap \delta_1$$

The resulting C-fset is

$$C_3 = \gamma_4 \sigma_4 \gamma_3 \sigma_3 \gamma_2 \sigma_2 \gamma_1 \sigma_1$$

where

$$\sigma_4 = LA[\tau_3 \cap \delta_1]$$

$$\sigma_3 = LA[(\tau_3 \cap \delta_1) \cup (\tau_2 \cap \delta_2)]$$

$$\sigma_2 = LA[(\tau_2 \cap \delta_1) \cup (\tau_1 \cap \delta_2)]$$

$$\sigma_1 = LA[\tau_1 \cap \delta_1]$$

(LA denotes the linguistic approximation).

We have got a finer description of the universe X.

The new proposition associated with the resulting C-fset will be denoted by $A \otimes B$ whose interpretation in linguistic terms will depend on the particular circumstances. One has:

$$(u \in A) \text{ is } \tau \quad \otimes$$

$$(u \in B) \text{ is } \delta$$

$$(u \text{ is } LA[A \otimes B]) \text{ is } LA[\tau \otimes \delta]$$

The Prolog implementation

In the following we shall discuss the main features of a Prolog prototype which implements the above sketched structure in order to cope with the problems of uncertainty modelling in expert systems. First of all we have to define the logical representation of the elements constituting the structure.

We have seen before that fuzzy propositions of the form (x is A) is τ can be represented by the notation

$$C = \alpha_n \tau_n \alpha_{n-1} \tau_{n-1} \dots \alpha_1 \tau_1$$

Correspondingly we can introduce the Prolog clauses:

$$\text{Cfset}(A, 1, \alpha_1, \tau_1).$$

$$\text{Cfset}(A, 2, \alpha_2, \tau_2).$$

$$\dots \dots \dots$$

$$\text{Cfset}(A, n, \alpha_n, \tau_n).$$

where A is the variable that represents the relation currently under examination. α_n consists of a list representing the set of elements satisfying the relation to the same extent, the variable τ_n is the value of the corresponding linguistic label, i.e. a fuzzy number. However, although τ_n is a linguistic label, it gets associated with the set of couples $\{ (x, \mu(x)) / x \in X, \mu(x) \in [0,1] \}$ which represent the value of the corresponding fuzzy set. Thus these values are to be adequately represented.

In turn, a fuzzy number τ_n is represented by the following clauses membership

$$\text{membership}(x, 1, \tau, \mu(x)).$$

$$\text{membership}(x, 2, \tau, \mu(y)).$$

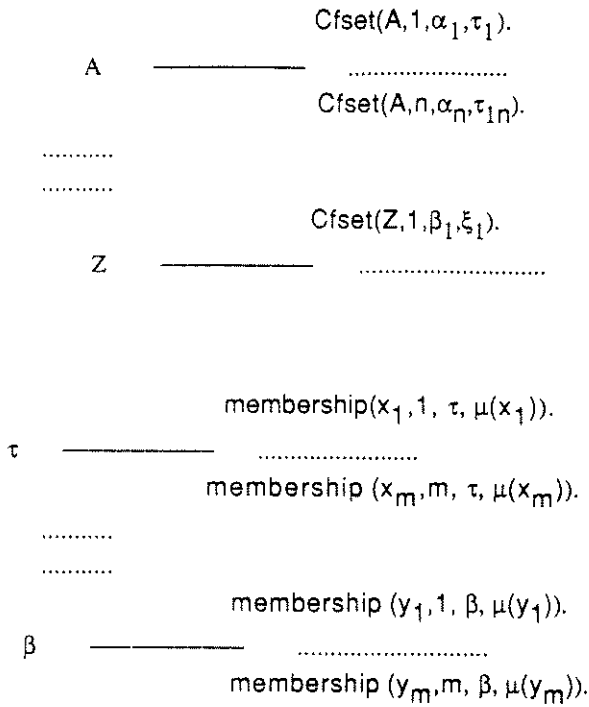
$$\dots \dots \dots$$

$$\text{membership}(x, n, \tau, \mu(z)).$$

where the first and last elements of the clause represent the items $x, \mu(x)$ of the couple $(x, \mu(x))$, the second is an index that will be useful for the implementation.

Now we tackle the problem of representing the C-fsets within the knowledge base. The starting point are hash tables that allow classifying the clause C-fsets(R, i, α, τ) associated with a particular relation R. Such representation is certainly adequate since we do not care that the sets represented by the clauses be ordered, but we are interested in efficient search time. Thus the elements are represented as follows, the first table is associated with the Set-Cfsets, the second one

with the Set-fuzzynumbers.



The clauses concerning each C-fset can be stored in the table via the pre-defined Prolog predicate recordh(table-name, key, item).

Consider now the algorithms which implement the operations \oplus, \otimes and complementation.

As regards the operation \oplus , one has ($n \geq m$)

$$\begin{array}{cccccccc} \alpha_n & \alpha_{n-1} & \alpha_m & \alpha_{m-1} & \alpha_2 & \alpha_1 & \oplus & \\ & & \beta_m & \beta_{m-1} & \beta_2 & \beta_1 & = & \end{array}$$

$$\alpha_n \cup \beta_n \quad \alpha_m \cup \beta_m \quad \alpha_{m-1} \cup \beta_{m-1} \quad \alpha_1 \cup \beta_1$$

From this scheme one obtains

$$\gamma_i = \beta_i \cup \alpha_i \text{ where } i=1, \dots, n, \quad \beta_j = \emptyset \text{ for } j = m+1, \dots, n$$

$$\sigma_i = \tau_i \cup \delta_i \text{ where } i=1, \dots, n, \quad \delta_j = \tau_j \text{ for } j = m+1, \dots, n$$

The procedure which implements this operation consists of a couple of routines concerning the primary C-fset and the secondary one, respectively.

In the following the generic formulae concerning the operation \otimes are shown:

$$\begin{array}{cccccccc} \alpha_n & \alpha_{n-1} & \dots & \alpha_2 & \alpha_1 & \oplus & & \\ \beta_m & \beta_{m-1} & \dots & \beta_2 & \beta_1 & = & & \\ \hline & \alpha_n \cap \beta_1 & \alpha_{n-1} \cap \beta_1 & \dots & \alpha_2 \cap \beta_1 & \alpha_1 \cap \beta_1 & & \\ & \alpha_n \cap \beta_2 & \alpha_{n-1} \cap \beta_2 & \dots & \alpha_2 \cap \beta_2 & \alpha_1 \cap \beta_2 & & \\ & \dots & \dots & \dots & \dots & \dots & & \\ & \alpha_n \cap \beta_n & \dots & \alpha_1 \cap \beta_n & & & & \\ \hline & \gamma_1 & \dots & \gamma_3 & \gamma_2 & \gamma_1 & & \end{array}$$

where $\gamma_i = \alpha_n \cap \beta_n$

$$\gamma_2 = (\alpha_2 \cap \beta_1) \cup (\alpha_1 \cap \beta_2)$$

$$\gamma_1 = \alpha_1 \cap \beta_1$$

The following relations are got, where $1=m+n-1$, and $n \geq m$

$$\gamma_i = \bigcup_{j=1, \dots, i} (\beta_j \cap \alpha_{i-j+1}) \quad 1 \leq i \leq m-1$$

$$\gamma_i = \bigcup_{j=i-m+1, \dots, n} (\alpha_j \cap \beta_{i-j+1}) \quad n+1 \leq i \leq 1$$

$$\gamma_i = \bigcup_{j=i-m+1, \dots, i} (\alpha_j \cap \beta_{i+1-j}) \quad m \leq i \leq n$$

In a similar way, as regards the fuzzy numbers, one gets:

$$\eta_i = \bigcup_{1 \leq j \leq i} (\tau_{i-j+1} \cap \xi_j) \quad 1 \leq i \leq m-1$$

$$\eta_i = \bigcup_{1-m+1 \leq j \leq n} (\tau_j \cap \xi_{i-j+1}) \quad n+1 \leq i \leq 1$$

$$\eta_i = \bigcup_{i-m+1 \leq j \leq i} (\tau_j \cap \xi_{i+1-j}) \quad m \leq i \leq n$$

We note that the computation of the quantities γ_i can be carried out in three different steps, each having a distinct subroutine which operates on both ordinary sets and fuzzy numbers. Of course, the correct management of the latter is by far more complex and additional subroutines are required.

Finally a suitable algorithm implements the operation "not" by complementing the fuzzy numbers corresponding to the sets present in the C-fset and then re-ordering the sets according to the values obtained.

In order that the system be truly operational and maintainable an appropriate user interface in pseudo-natural language has been developed so that fuzzy propositions are directly added to the knowledge base instead of the associated C-fsets.

It is duty of the system to translate the propositions into the equivalent C-fsets and this happens since we can translate the fuzzy proposition (x is A) is τ into the clause is(is (x, A) τ).

An example concerning medical diagnoses

We aim at studying in what measure a disease can be inferred as a likely consequence of some quoted symptoms. For instance, consider a population whose elements suffer, to different extents, from, say, hypertension, diabetes, cholesterol. Moreover, we have to consider also the social and psychological behavior of the individuals since these features are as important as the previous symptoms in order to give rise to the disease "coronaropathy". Such forerunners might be stress, social status, anxiety, etc. We construct as many C-fsets as are the distinguishing features. A C-fset contains the individuals that suffer from the symptom S and a fuzzy number, i.e. a linguistic label, is associated with each of them. In such way we get statements having the following structure (x has S) is τ (*)

Of course the labels have to be the same for all C-fsets so that the operations can be carried out coherently. For the sake of simplicity we consider only four grades for τ : very true, true, rather true, almost true. In the following we shall write, for short, vt, t, rt, at.

The propositions will be added like in (*) since the predicates "has" and "is" can be defined as operations $x \text{ f } x$, the first having a lower precedence.

We have developed a couple of procedures which implement the translation of the propositions into Prolog clauses. The output of the first procedure is the set of truth values, i.e. referred to our example the set {vt, t, rt, at}. The output of the second one is the set of symptoms, i.e. the set {cholesterol, stress, anxiety}.

Then we have the procedure that generates for each symptom the corresponding C-fset, i.e. it classifies the individuals according to the symptom they suffer from.

It is duty of a set of specific procedures to generate the hash table and to assign to the elements of each C-fset according to the values τ_i .

In such way the construction of the knowledge base is carried out and from fuzzy propositions the corresponding C-fsets are generated.

As regards the operations we note that \oplus is to be applied for symptoms viewed as similar, \otimes for symptoms inherently dishomogenous. It is duty of the user to decide about the equivalence of the symptoms.

Finally the C-fset associated with the disease "coronaropathy" is got. In this set the population is clustered and the individuals that share the

highest fuzzy numbers are the most likely candidates to the disease.

As the information is contained in the hash tables as C-fsets, a specific procedure outputs it as fuzzy proposition. Finally we note that the chain of logical steps leading to the result can be visualized by means of production rules having the form:

Rule: if x has S1 is τ and x has S2 is β and
then x has M is γ .

These rules become Prolog clauses thanks to a suitable definition of the operators.

Concluding remarks

We note that the basic difference between the C-fcalculus and the fuzzy logic is the inherent parallelism of our structure. This fact allows direct comparisons among the elements and moreover one gets remarkable data compression. The Prolog prototype so far illustrated provides, in our opinion, a convenient point of departure for dealing with some problems concerning uncertainty modelling and now we briefly discuss to what extent our approach differs from other aforementioned methodologies. Of course, the interpreter FProlog is deserving yet it is difficult to compare it with our interpreter since the former is very sophisticated as regards the implementation level and the software engineering techniques.

In turn, the methodologies based on certainty factors and supported logic programming use numerical values in order to deal with fuzzy predicates and this fact yields programs not easily maintainable. In our approach numerical values are not required: only linguistic labels are manipulated and as a consequence the user environment is friendly and no programming experience is necessary.

However, the usefulness of the prototype, in our opinion, is restricted to problems where fuzzy items derive from other fuzzy ones. In these situations "a priori" knowledge is available that connecting some properties leads to other properties and the prototype under consideration appears to be a particularly well-suited tool for uncertainty modelling. Applications of this nature are being deeply investigated and our current work is exploring further facets of reflection.

References

- [1] F. Dowlatshahi, L.J. Kohout, Intentional possibilistic approach to dealing with incompleteness, vagueness and uncertainty. *Fuzzy Sets and Systems*, 4(1988)
- [2] S. Fukami, M. Mizumoto, K. Tamaka, Some considerations on fuzzy conditional inference. *Fuzzy Sets and Systems*, 4(1980) 243-273
- [3] R. Gaines Brian, L.G. Shaw Mildred, Induction of inference rules for expert system. *Fuzzy Sets and Systems*, 18(1986)
- [4] A. Gisolfi, An algebraic fuzzy structure for the approximate reasoning. In press. *Fuzzy Sets and Systems*. (1990)
- [5] L. Lesmo, P. Torasso, Prototypical knowledge for interpreting fuzzy concepts and quantifiers. *Fuzzy Sets and Systems*, 23 (1987) 361-370
- [6] T.P. Martin, J.F. Baldwin, B.W. Pilsworth, The implementation of FProlog-a fuzzy prolog interpreter. *Fuzzy Sets and Systems*, 23(1987) 119-130
- [7] C.V. Negoita, *Expert Systems and Fuzzy Systems*. (The Benjamin/Cumming Publishing Company, Inc: 1989)
- [8] L.A. Zadeh, Fuzzy logic and approximate reasoning. *Synthese*, 30(1985)407-428
- [9] L.A. Zadeh, A theory of approximate reasoning. in: J.E. Hayes, D. Michie and L.I. Mikulich, Eds., *Machine Intelligence Vol.9* (John Wiley & Sons, New York, 1979) 149-194
- [10] L.A. Zadeh, The role of fuzzy logic in the management of uncertainty in E.S. *Fuzzy Sets and Systems*, 11(1983)199-227