THEORY AND APPLICATION OF MULTIPLE-VALUED LOGICS FOR KNOWLEDGE-BASED SYSTEMS

José Antonio Reyes Francesc Esteva Josep Puyol-Gruart
Artificial Intelligence Research Institute, IIIA
Spanish Council for Scientific Research, CSIC
Campus UAB. 08193 Bellaterra, Catalonia, Spain
E-mail: {reyes, esteva, puyol}@iiia.csic.es

Abstract

Multiple-valued logics are useful for dealing with uncertainty and imprecision in knowledge-based systems. In this paper, we present the foundations of **QMORPH**: a tool that assists users in the declaration of such logics and in the declaration of the communication mechanism between two of these different logics by preserving inference.

Keywords: uncertain reasoning, multiple-valued logics, knowledge-based systems.

1 DEFINING MV-LOGICS

Multiple-valued logics (MV-Logics) have been proved to be a useful way to manage uncertainty and imprecision [3; 5; 6; 10].

We consider a restricted family of finite MV-logics expressive enough to model the uncertain reasoning used in many rule-based systems [1]. Each logic is determined by a particular algebra of truth-values from this family.

An **Algebra of truth values** is a finite algebra $A_T^n = \{A_n, \mathbf{0}, \mathbf{1}, N_n, T, I_T\}$ such that:

1. The ordered set of **truth-values** A_n is a chain of n linguistic terms:

   $$\mathbf{0} = a_0 < a_1 < \ldots < a_{n-1} = \mathbf{1}$$

   where $\mathbf{0}$ and $\mathbf{1}$ are the Boolean *False* and *True* respectively.

2. The **negation** operator N_n is the unary operation defined as:

   $$N_n(a_i) = a_{n-1-i}$$

   the only one that fulfills the following properties:

   N1: If $a < b$ then $N_n(a) > N_n(b)$, $\forall a, b \in A_n$
   N2: $N_n^2 = Id$

3. The **conjunction** operation T is any binary operation such that the following properties hold $\forall a, b, c \in A_n$:

   T1: $T(a,b) = T(b,a)$
   T2: $T(a,T(b,c)) = T(T(a,b),c)$
   T3: $T(\mathbf{0},a) = \mathbf{0}$
   T4: $T(\mathbf{1},a) = a$
   T5: If $a \leq b$ then $T(a,c) \leq T(b,c)$ for all c

4. The **implication** operation I_T is defined by residuation with respect to T, i.e.

   $$I_T(a,b) = Max\{c \in A_n \mid T(a,c) \leq b\}$$

Therefore, to establish an algebra of truth-values it is only necessary to determine the set of truth-values A_n more adequate for the concerning problem, and define a conjunction operator T.

The **sentences** of the language are pairs (p, V), where: p is a wff, obtained in the usual way from a denumerable set of propositional symbols and the connectives *'not'* ($\neg$), *'and'* ($\wedge$) and *'detachment'* ($\rightarrow$); and V is an interval of truth-values from A_n.

Likewise, semantic is evaluated in an algebra of truth-values. The semantic interpretation is given by the following signification:

- **Models** are defined by valuations; i.e. mappings ρ from the propositional symbols to A_n terms provided that:

  $\rho(\neg p) = N_n(\rho(p))$
  $\rho(p_1 \wedge p_2) = T(\rho(p_1), \rho(p_2))$
  $\rho(p \rightarrow q) = I_T(\rho(p), \rho(q))$

- The **Satisfaction Relation** between models and sentences is defined by:

  $M_\rho \models (p, V)$ if, and only if, $\rho(p) \in V$,

  where M_ρ stands for the model defined by a valuation ρ.

The inference rule of these logics is *Modus Ponens*, which gives from $(p \rightarrow q, V)$ and (p, V') a sentence (q, V''). In fact, modus ponens evaluates V'' from V and V' (See [9] for a theoretical study of modus ponens, and [6] for its application to expert systems).

1.1 INTERVALS OF TRUTH-VALUES

We extend the previous algebra to an algebra of intervals, [4]. We have three motivations to do this. The first one is imprecision. The second reason is related with the modus ponens operator, which requires the use of intervals to

chain rules. And finally, intervals are needed to make possible mappings between different logics.

Given an algebra of truth-values $A = < A_n , N_n , T >$, we will consider the set of intervals of A_n as:

$$Int(A_n) = \{[a,b] \mid a,b \in A_n\}$$

being $[a,b] = \{x \in A_n \mid a \leq x \leq b\}$.

The extensions to intervals of the above operators are: $N_n^*([a,b]) = [N_n(b), N_n(a)]$, $T^*([a,b], [c,d]) = [T(a,c), T(b,d)]$.

1.2 CONJUNCTION GENERATION

Given a set of linguistic terms, we need to know all the possible conjunction operators in order to choose one of them.

The mentioned algebra properties (**T1-T5**) are requirements that a conjunction operator must fit. We have modelled this task as a classic CSP problem where these properties act as constraints over the set of possible solutions. We represent the conjunction operator as a matrix, where each element is a variable $V_{i,j}$, $0 \leq i, j \leq n-1$. Satisfaction of each property or constraint, causes the following guidelines which influence the conjunction search generation problem:

- Commutativity allows us to consider only the set of variables $V = \{V_{i,j}, i \geq j\}$.
- The absorbent and neutral elements fix the values for variables $V_{0,j}$ and $V_{j,n-1}$.
- Monotonicity implies non-decreasing rows and columns.
- Associativity test is expensive in time and memory, but considering that $T(a_i,a_j) \leq min(a_i,a_j)$, each sub-matrix where $i = j$ is closed with respect to the conjunction operator. In this way, the matrix will be associative if all its sub-matrices are so. This property lets us check associativity in an incremental way every time a value is assigned to a variable $V_{i,i}$.

Applying these constraints, the number of operators generated is exponential with respect to n. We can consider two options to reduce this number: introducing some new desirable constraints, as well as fixing several values of the matrix to reduce the search space to be explored.

Two types of additional, and optional, constraints are discussed:

T6 Strictness: $T(a_i ,a_j) \neq \mathbf{0}$, for all $i, j \neq \mathbf{0}$

Given two truth-values different than $\mathbf{0}$ *(false)*, it does not seem reasonable that their conjunction may be $\mathbf{0}$.

T7 α-Smoothness: given $\alpha \in \mathbb{N}$, T is said to satisfy α-smoothness property if $T(a_i,a_j) = a_k$ and $T(a_{i-1},a_j) = a_p$, then $k - p \leq \alpha$

We also want that the result of the conjunction of near values has to be not very distant.

Satisfaction of these new properties (**T6**, **T7**) has the following consequences in the conjunction search generation problem:

- Strictness: the generation of strict matrices of dimension n is equivalent to generate non-strict matrices of dimension $n-1$.
- α-Smoothness: it reduces the set of possible values for a variable taking into account column and row adjacency.

2 COMBINING LOGICS

We can declare different logics by varying the set of truth-values (linguistic terms) and the conjunction operator. That depends of how the expert will deal with uncertainty in each problem.

When two different logics need to exchange information, it is necessary some mechanisms of translation to make the communication between these logics compatible. We want to preserve the inference properties of each logic but, as usual, it is not possible to transmit information without loss of precision. Hence we map values of an algebra into intervals of the other.

Let's take a look to which requirements we need in order to map the language of a logic into another one when they require to exchange information [2].

2.1 MAPPINGS BETWEEN MV-LOGICS

Let $(L ,\vdash)$ and $(L',\vdash')$ be two logics, L and L' standing for the languages, $\vdash$ and $\vdash'$ for the entailment relations defined on L and L' respectively. To establish a correspondence between both logics, a *mapping* $H: L \longrightarrow L'$ is needed. Now, we will analyze some natural requirements for the mapping H with respect to the entailment systems $\vdash$ and $\vdash'$. We propose that at least one of the following three requirements should be fulfilled by the mapping H in order to ensure a consistent communication. Henceforth Γ and e will denote a set of sentences and a sentence of L respectively. A map H is said to be a *forward conservative* map when,

RQ.1. If $\Gamma \vdash e$, then $H(\Gamma)\vdash' H(e)$

For every sentence e, deducible from a set of sentences Γ, its corresponding sentence, $H(e)$, will also be deducible from the corresponding sentences of $H(\Gamma)$.

A map fulfilling this second requirement is said to be a *backward conservative* map:

RQ.2. If $H(\Gamma)\vdash' H(e)$, then $\Gamma \vdash e$

This is the inverse requirement of **RQ.1**. Hence, if a fact is not deducible from Γ, then its corresponding fact from $H(\Gamma)$ won't be deducible either. Nevertheless $\Gamma \vdash e$ does not imply $H(\Gamma)\vdash' H(e)$.

Conditions **RQ.1** and **RQ.2**, which are very strong, can sometimes be weakened in the uncertain reasoning framework. Formally, this can be expressed by the third and last requirement:

RQ.3. If $H(\Gamma)\vdash' e'$, then there exists e such that $\Gamma \vdash e$ and $H(e)\vdash' e'$

This requirement assures that every sentence deducible from $H(\Gamma)$ must be in agreement with what can be deduced from Γ. This requirement is slightly different from **RQ.2**, in the sense that it is not necessary e' being an exact translation of a deducible sentence e from Γ, but only something deducible from such a translation. In the framework of logics for uncertainty management, e' is interpreted as a *weaker* form of e, i.e. a sentence expressing more uncertainty than e. We will call it a *weak conservative* map.

Now we will consider the problem of finding inference preserving correspondences between two logics $A = <A_n , N_n , T >$ and $B = <B_m , N_m , T'>$. We are interested in mapping the entailment system $(L_A,\vdash_A)$ into the entailment system $(L_B,\vdash_B)$, by means of renaming functions between the corresponding linguistic term sets. This means that we will only consider those mappings translating sentences from L_A to L_B that just involve translations of truth-values; i.e. any mapping $H: L_A \longrightarrow L_B$ will be defined as $H(e,V) = (e,h(V))$, where h translates subsets of values of A_n into subsets of values of B_m, and e (a wff) remains invariant.

2.2 ALGEBRA MORPHISMS

To establish conservative communications, it is necessary to consider what kind of relation between the uncertainty logics is required. In [2], we can find the necessary and/or sufficient conditions for a mapping $h: A_n \longrightarrow Int(B_m)$ to satisfy these requirements. From this analysis, we deduce the following relationships between the conditions needed to satisfy every requirement (see Figure 1):

$$\textbf{RQ.1} \Leftrightarrow \left\{\begin{array}{c} h(T(V1,V2)) \supseteq T'(h(V1),h(V2)) \\ h(N(V)) = N_m(h(V)) \\ h(V1 \cap V2) = h(V1) \cap h(V2) \end{array}\right\}$$

$$\textbf{RQ.2} \Leftarrow \left\{\begin{array}{c} h(T(V1,V2)) \subseteq T'(h(V1),h(V2)) \\ h(N(V)) = N_m(h(V)) \\ h(V1) \supseteq h(V2) \Rightarrow V1 \supseteq V2 \end{array}\right\}$$

$$\Downarrow$$

$$\textbf{RQ.3} \Leftrightarrow \left\{\begin{array}{c} h(T(V1,V2)) \subseteq T'(h(V1),h(V2)) \\ h(N(V)) = N_m(h(V)) \end{array}\right\}$$

Figure 1: Requirements conditions.

We define mappings from elements of A_n into intervals of B_m, but sometimes it is possible to find mappings that translate an element of A_n into an element of B_m. In this case, requirement **RQ.3** is satisfied and the mapping is a *morphism* between the corresponding algebras. As a particular case, if a map fulfill **RQ.1** and **RQ.2**, we have not only the morphism conditions, but also a one-to-one application, that is, an injective function called *monomorphism*. But we can not always find these kinds of mappings, so in the case of a map involving intervals of truth-values, we call it a *quasi-morphism*.

We are mainly interested in *monomorphisms* because they embed A_n into B_m and because they are order preserving mappings (it is equivalent to a communication without any loose of information). The second preference is *morphisms*, which accomplish the algebra operations (it is a transmission of information fulfilling the required properties). Finally, because of the strong conditions *morphisms* and *monomorphisms* must satisfy, it is not always possible to find these kind of renaming them, so *quasi-morphisms* can be useful thanks to the additional freedom of map truth-values of an algebra into intervals of the other (we allow certain loose of information).

2.3 RENAMING GENERATION

For a given set of truth-values A_n, there exists only one negation operator N_n. Then, we can make the following partition:

- the set of negative elements $\boldsymbol{N_n} = \{x \mid x < N_n(x)\}$
- the set of fixed elements $\boldsymbol{F_n} = \{x \mid x = N_n(x)\}$
- the set of positive elements $\boldsymbol{P_n} = \{x \mid x > N_n(x)\}$

Besides, we can do the same with a set B_m, obtaining $\boldsymbol{N_m^*}$, $\boldsymbol{F_m^*}$ and $\boldsymbol{P_m^*}$ for the case of working with intervals. Then, the renaming algorithm consists in generating all the maps $h_1: \boldsymbol{N_n} \cup \boldsymbol{F_n} \longrightarrow \boldsymbol{N_m^*} \cup \boldsymbol{F_m^*}$ such that:

a) $h_1(0) = 0$, (i.e. $h_1(a_0) = b_0$).

b) $h_1(\boldsymbol{F_n}) \in \boldsymbol{F_m^*}$.

c) $x \le y$ implies $h_1(x) \not> h_1(y)$, where $x, y \in A_n$ and $h_1(x), h_1(y) \in Int(B_m)$.

and extending each of these mapping h_1 with respect to the negation operation defining the morphism h as:

$$h(x) = \begin{cases} h_1(x), & \text{if } x \in \boldsymbol{N_n} \cup \boldsymbol{F_n}. \\ N_m^*(h_1(N_n(x))), & \text{if } x \in \boldsymbol{P_n}. \end{cases}$$

Next, we check which ones are compatible with the conjunction operators T and T', and finally, we check which ones are monomorphisms, morphisms or quasi-morphism. The resulting maps are presented to the user in this order.

Due to the strong condition monomorphisms and morphisms must fit, it is not always possible to find them. However, it is very possible that the renaming generation produces a large list of quasi-morphisms, so the possibility of giving an ordered list of quasi-

morphisms to aid users in their selection may be considered. Note that in the case of morphisms, all the renaming mappings have the same evaluation, hence the selection is left to the user's criteria.

For the purpose of producing an ordered list of quasi-morphisms, we consider a weigh among the cardinal c of the set of terms which have an atomic image (that is, an interval with the form $[a_i , a_i]$) and the length L of the remainder of intervals (number of truth-values included in this interval).

Given two chains A_n and B_m, we will generate mappings between A_n and the set of intervals $Int(B_m)$. In order to obtain an evaluation for every map, we can use the following empirical function:

$$\eta = \frac{\sum_{i=1}^{n} L_i}{\sum_{i=1}^{n} c_i} , \; 1 \le L_i \le m , \; 0 \le c_i \le 1 \qquad (1)$$

where L_i is the length of the interval i and c_i is the number of points which have an atomic image.

User is also able to establish the behavior that can take the generated maps defining a partial map between both algebras. This definition must satisfy the set of requirements suggested previously in order to be a negation morphism. When we have defined our partial map (or total, or none), a list of renaming functions fulfilling these defined characteristics will be generated.

3 THE QMORPH TOOL

QMORPH [8] is a tool that allows users to define the more adequated logic for a concrete problem, as well as to decide a consistent communication between two different MV-logics. This tool makes automatic these problems and assists users along the process. Two different interfaces have been developed: a graphical interface for the UNIX operative system using *Tcl/Tk* packages, running under the X-Windows environment and developed on Sun machines; and a generic text mode interface performed in Common Lisp.

In the present, this tool is attached to **Milord II** [7] - a specific expert system building environment - as an aid tool to support experts when developing modular applications, although it can be used in a more extensive framework.

4 CONCLUSIONS

In this paper we have defined which are the theoretical bases that allow dealing with uncertainty by means of multiple-valued logics. We have settled how to declare a suitable MV-logic from a parametric family of algebra of truth-values, and also, which are the necessary and/or sufficient requirements when we need to establish a communication between two of these logics. This problem arises in large knowledge-based systems in which different tasks need to cooperate using uncertain reasoning, as well as in distributed systems.

QMORPH has been designed with the aim of automating these problems, as well as assisting users and offering different alternative possibilities. Beyond its particular use within **Milord II**, it can be deployed by any other system using MV-logics.

This work focuses on the unidirectional interaction between two logics, but the more general problem of communicating various uncertain reasoning systems is far more complex. This work along with further research will make possible this goal.

Acknowledgments

Partially supported by project MODELOGOS funded by CICYT (TIC 97-0579-C02-01) and through SMASH by CICYT (TIC 96-1038-C04-01).

References

[1] Agustí, J.; Esteva, F.; Garcia, P.; Godo, L.; Sierra, C.; Combining multiple-valued logics in modular expert systems, *Proceedings of 7th Conference on Uncertainty in AI*, Bruce d-Ambrosio et al. (eds), (1991), pp. 17-25.

[2] Agustí, J.; Esteva, F.; Garcia, P.; Godo, L.; López de Mántaras, R.; Local multi-valued logics in modular expert systems, *Journal of Experimental & Theoretical AI (JETAI)*, vol. 6 n. 3 (1993), pp. 303-321.

[3] Bonissone, P.; Gans, S.; Decker, K.; Rum: A layered architecture for reasoning with uncertainty, in *IJCAI'87* (1987), pp. 891-898.

[4] Esteva, F.; Garcia-Calves, P.; Godo, L.; Enriched interval bilattices: An approach to deal with uncertainty and imprecision, *International Journal on Uncertainty, Fuzzyness and Knowledge-Based Systems*, 1-2 (1994).

[5] Godo, L.; López de Mántaras, R; Sierra, C.; Verdaguer, A.; Milord : The architecture and management of linguistically expressed uncertainty, *International Journal of Intelligent Systems*, Vol. 4 n. 4 (1989), pp. 471-501.

[6] López de Mántaras, R.; *Approximate Reasoning Models*, Ellis Horwood Series in Artificial Intelligence, 1990.

[7] Puyol, J.; *MILORD II: A language for knowledge-Based Systems*, Vol. 1 of Monografies del IIIA, IIIA-CSIC, 1996.

[8] Reyes, J. A.; **QMORPH**: A tool to define and combine local logics in Milord II, *Mst. Thesis, Universitat Autònoma de Barcelona*, (1997).

[9] Trillas, E.; Valverde, L.; On mode and implication in approximate reasoning, in Gupa et al. (eds.): *Approximate Reasoning in expert systems*, (1985), pp. 157-166.

[10] Turner, R.; *Logics for Artificial Intelligence*, Ellis Horwood Series in Artificial Intelligence, 1984.