Possibilistic Frames and Semantic Nets in a Customizable Shell Generator Tool

Sandra SANDRI Guilherme BITTENCOURT

Instituto Nacional de Pesquisas Espaciais - INPE
Caixa Postal 515 - São José dos Campos - SP - Brasil
Tel.: 55 (123) 41 8977 r. 650 Fax: 55 (123) 21 8743
E-mail: sandri@lac.inpe.br, gb@lac.inpe.br

ABSTRACT

In this paper, we propose a theoretical framework in which possibilistic logic can be uniformly used to treat uncertainty associated with production rules, when knowledge is represented in either of the formalisms: logic, frames and semantic nets. This model is being implemented in the uncertainty module of a tool designed to allow the construction of expert system shells. The tool is capable of handling the above three knowledge representation formalisms, forward and backward control strategies, and several conflict resolution methods. These features are collected together to compose expert system shells adapted to the characteristics and complexity level of the target problem.

1. Introduction

Expert Systems technology was the first commercial achievement of Artificial Intelligence. In a few years, expert systems made their way from research laboratories to industry, stock market, hospitals, public administration and nowadays there are expert system implementations in almost all knowledge fields.

To facilitate the development of an expert system it is important to have available an expert system shell adapted to the problem to be solved. An expert system shell is an "empty" expert system, i.e. a knowledge representation language and an inference engine. The degree of adaptation of such a shell to a given problem depends on the type of knowledge representation language available: logic, frames, semantic nets, or other similar formalisms.

After chosing the suitable shell, the next step is to code the problem domain knowledge into the shell's knowledge representation language. This task, known as *knowledge acquisition*, is the most important bottleneck in the expert system development activity. Besides the inherent difficulties of extracting the expertise hiden inside expert brains and representing it in an artificial formalism, it should be taken into account that this expertise, in many cases, is not a sound, complete and coherent body of knowledge, but is usually pervaded with noise, uncertainty and inconsistency. To represent this "less than certain" knowledge, it is necessary that, embedded into the knowledge representation language, some uncertainty formalism is available.

Several approaches have been proposed to deal with uncertain knowledge in the last years. Some are purely symbolic, others make use of numbers to quantify uncertainty; some are more formal, others are less formalized. From the expert system development point of view, it is important that the adopted uncertainty model be easy to understand by domain experts, in such a way that the informations provided by them correspond as closely as possible to their intuitions. One of such models is the possibilistic framework, in which numerical quantification is used as a way of ordering competing hypothesis. Its main operations are the minimum and the maximum, which makes it fairly easy to be dealt with from the developer point of view, and whose comprehension do not require any special mathematical skills from the average user.

In this paper, we present the initial configuration of the uncertainty management module of a tool designed to allow the construction of customized expert system shells according to the level of complexity of the target problem. The tool, called Fase [Mar 92] [BM 92], is capable of handling three knowledge representation formalisms — logic, frames and semantic nets. The main contribution of the paper is the proposal of an environment where possibilistic logic PL1 can be uniformly used to treat uncertainty associated with production rules and knowledge represented in each of the knowledge representation formalisms provided by the tool.

This paper is organized as follows. In Section 2 we present system Fase, briefly describing its main features. In Section 3 we give a short description of the knowledge representation paradigms used in the tool. Section 4 constitutes the main subject of this paper, exposing the treatment of uncertainty in the tool. It briefly reviews the main concepts in possibility theory and possibilistic logic PL1, and then shows how the possibilistic paradigm is used in our tool when the knowledge in an application is represented in logic, frames and semantic nets. Finally, Section 5 brings the conclusion.

2. General Description of the Tool

The tool architecture includes two management modules - the tool interface and the shell generator - and three package libraries - the kernel, the knowledge acquisition interface, and the expert system interface. Each package library consists of a set of Lisp function packages that can be used as building blocks to implement an expert system shell. This architecture is presented in Fig. 1.

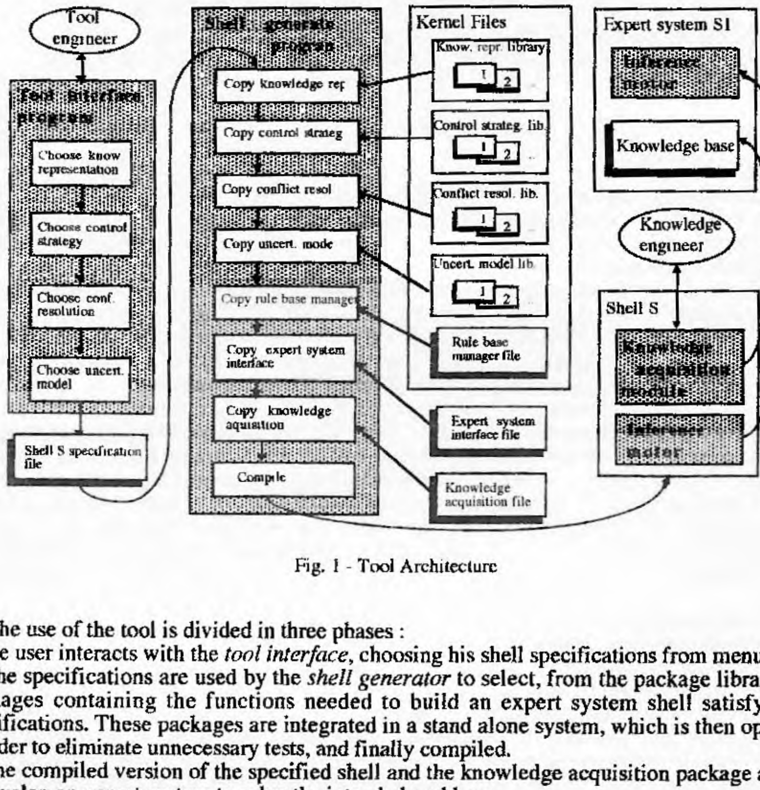


Fig. 1 - Tool Architecture

The use of the tool is divided in three phases :

i) the user interacts with the *tool interface*, choosing his shell specifications from menus.

ii) the specifications are used by the *shell generator* to select, from the package libraries, the packages containing the functions needed to build an expert system shell satisfying the specifications. These packages are integrated in a stand alone system, which is then optimized in order to eliminate unnecessary tests, and finally compiled.

iii) the compiled version of the specified shell and the knowledge acquisition package are used to develop an expert system to solve the intended problem.

An expert system shell generated by the tool consist of the following modules: knowledge representation, uncertainty management, control strategy, conflict resolution, rule base manager, expert system interface, and knowledge acquisition interface (see Fig. 1). All these items, except the knowledge acquisition interface, compose the inference motor of the expert systems that can be generated using that shell.

To build a shell, the user chooses as many knowledge representation models as he wishes. He then chooses an uncertainty model, a control strategy, and a conflict resolution mechanism. Based on these choices the tool itself specifies the remaining items and builds the shell thereupon. An expert system generated by this shell consists of the inference motor and a knowledge base generated by the interaction between a knowledge engineer and the knowledge acquisition interface of the shell (see Fig. 1).

3. Knowledge Representation in the Tool

All the knowledge contained in the knowledge base of an expert system generated by the tool is encoded in rules of the type *if* <expression> *then* <expression>. Each expression is a conjuction of items, each of which may be either a graphical comand or a term representing a piece of knowledge, encoded in a knowledge representation model (logic, frames, etc...). How the terms are classified and interpreted depends on the knowledge representation formalism used and on the control strategy adopted. A typical rule of an image classification expert system is *rule-1* :

If the texture of region x is of type t1 then x is a bear soil region.

In a cycle, when forward-chaining is used, the left hand side of the rule is matched against the facts known to the system and a list of valid substitutions is returned. Each substitution is applied to the right-hand side of the rule, generating a new fact.

Each of the knowledge representation formalisms dealt with in the tool is defined by a formal language, a reasoning mechanism, and an interface between the formalism and the rule bases. In its present version the tool deals with 3 paradigms : logic, frames [Min 75] and semantic nets [Qui 68]. In this work we concentrate on the first two paradigms.

The *logical formalism* consists of a collection of independent facts and is thus adequate to domains where the knowledge is largely unstructured. The main drawback of the formalism is the inefficiency of the inference method: automatic deduction. The tool manipulates rules written in both propositional and first-order logic. The facts however can only be propositional or grounded formulae. In this formalism rule-1 is represented by :

If (Texture(x, t1)) then (Bear-soil(x)),

where *x* is a variable, *t1* is a constant and *Texture* and *Bear-soil* are predicates.

The *frame formalism* [Min 75] consists of a hierarchy of data structures called frames. Each frame is composed of a set of slots to which values can be associated. These values can be either any relevant information about the concept represented by the frame, or another frame. Efficient inference mechanisms are integrated into the formalism: inheritance through the frame hierarchy, facets that control the type of value adequate for each slot, default values that can be used when no information is available, procedural attachment to allow the execution of external functions. The formalism is adequate to taxonomically structured domains where the inheritance mechanism can be efficiently explored. Here rule-1 is represented by :

If (x (texture t1)) then (x (class bear-soil)),

where *x* is a variable to be matched with a frame,*texture* and *class* are slots of *x*, with values *t1* and *bear-soil* respectively.

A *semantic net* [Qui 68] is a model for the human associative memory, consisting of a set of nodes connected by edges. The formalism does not dispose of a general agreed-on semantics: the nodes can be used to represent concepts, predicates, objects, etc..., and the edges are associated to arbitrary binary relations between the nodes. The main inference mechanism in semantic nets is inheritance through the net and network matching. We represent rule-1 in this formalism by :

If (x is-a texture-t1) then (x is-a region-bear-soil).

where *x*, *texture-t1* and *region-bear-soil* are nodes, and is-a are links in a semantic net.

Differently from frame inheritance, semantic nets allows for explicit exception links resulting in much more complex inheritance algorithms (e.g. the skeptical inheritance algorithm [TRT 87]). This is usually done through the introduction of negative links. For instance, we could have a rule of the form :

If (x is-not-a texture-t1) then (x is-not-a region-bear-soil).
where is-not-a is a negative link.

The semantic nets formalism is also adequate for taxonomically structured domains, but its expressivity is higher because of the fact that the inheritance hierarchy is represented by a graph, allowing multiple inheritance, and the possibility of representing exceptions.

4. The Treatement of Uncertainty in the Tool

All the pieces of knowledge given to the system, - the rules extracted from experts and the facts forwarded by the user -, may be inherently pervaded with uncertainty. On the other hand, several knowledge representation models may coexist in a single application. The uncertainty models furnished by the tool should then be such as to deal with different knowledge representation models in a uniform manner. Moreover, these models should be of fairly easy comprehension by the average experts, knowledge engineers and expert system users. Possibility theory is such a model. In the following we present the possibilistic framework and how it is used in our tool.

4.1. The Possibilistic Paradigm

Possibility theory [Zad 78] [DP 88] is one of the most significant paradigms used in the treatment of uncertainty. In this framework, what is known about the value of variables or about the existing relations between variables is represented by means of possibility distributions. Namely, if X is a variable taking its values in a domain $\mathcal{X}$, then a possibility distribution π_X attached to X, is a mapping from $\mathcal{X}$ to the interval [0,1]. For any given value x in $\mathcal{X}$, $\pi_X(x)$ reflects to what extent it is possible that X = x. The normalization condition which is usually applied to π_X is $\sup_{x\in\mathcal{X}} \pi_X(x) = 1$. This condition expresses that at least one value in $\mathcal{X}$ is considered as completely possible for X (exhaustiveness of the domain), and it allows that distinct values in $\mathcal{X}$ be simultaneously regarded as completely possible.

From a possibility distribution π_X on $\mathcal{X}$ we derive a possibility measure Π and a necessity measure N, which are mappings from the subsets of $\mathcal{X}$ to [0,1] [DPL 91]. A possibility measure is defined as $\forall A \subset \mathcal{X}, \Pi(A) = \sup \{\pi(w) / w \in A\}$, and expresses to what extend there is a value $u \in A$ that may stand as a value of x. A necessity measure is the dual set function of Π, and is defined by $\forall A \subset \mathcal{X}, N(A) = 1 - \Pi(\neg A) = \inf\{1 - \pi(w) / w \in \neg A\}$, where $\neg A$ stands for the complement of A in $\mathcal{X}$. N(A) (respec. $\Pi(A)$) quantifies how much the available evidence supports (respec. does not contradict) the hypothesis that A contains the real value of X.

A possibility distribution $\pi_{X_1,\dots,X_n}$, defined on a Cartesian product $X_1 \times \dots \times X_n$, expresses a dependency relation between the variables $X_1, \dots, X_n$. For instance, a distribution π_{X_1,X_2} could be used to represent the dependencies between two variables X and Y, expressed in the rule of thumb *if X = A then Y = B*, where A and B are fuzzy sets*. Here, $\pi_{X_1,X_2}(x1, \cdot)$ is the fuzzy set of the more or less possible values of X_2, when $X_1 = x_1$.

We can deduce information from a knowledge base about the value of a variable of interest, using a general procedure known as the conjunction/projection method [Zad 79], which jointly constrains a set of variables. Namely, let $\pi_{X_1,\dots,X_n}$ be the possibility distributions representing the possible values of the variables $X_1, \dots, X_n$; if nothing is known about the value of X_i, then $\pi_{X_i}(x_i) = 1, \forall x_i \in \mathcal{X}_i$. Let $R_1,\dots, R_m$ be possibility distributions defined on universes

---

* The main difference between a "classical" set and a fuzzy set [Zad 78], in mathematical terms, is that in a "classical" set we make use of a function μ: $\mathcal{X}$ -> {0, 1},as its characteristic function, whereas a fuzzy set is defined through a function μ: $\mathcal{X}$ -> [0, 1]. Informally, an element of a universe of discourse either belongs or not to a "classical" set, but it can "more or less" belong to a fuzzy set. We very often define a possibility distribution from a fuzzy set. For instance, if all we know about John's height is that he is tall, we can use the fuzzy set "tall", as the possibility distribution of John's height, i.e. we make $\pi_{height} = \mu_{tall}$.

of arity larger than 1, represented by the (fuzzy) relations stated in the knowledge base between variables. Each R_j is thus defined on the Cartesian product of the domains $\mathcal{X}_k$'s of the variables X_k involved in the relationship represented by R_j. Then the conjunction/projection method consists in :

i) performing the combination

$$\pi^*{X_1,\ldots,X_n} = \min(\min_{i=1,n} \pi X_i, \min_{j=1,m} \pi R_j)$$

which is the least restrictive possibility distribution for the tuple $(X_1, \ldots, X_n)$ compatible with all the constraints ;

ii) projecting the result $\pi^*{X_1,\ldots,X_n}$ on the domain(s) of the variable(s) we are interested in ; for instance we get for X_i

$$\forall x_i \in \mathcal{X}_i, \pi X_i(x_i) = \sup_{x_j, j=1,n, j\neq i} \pi^*{X_1,\ldots,X_n}(x_1, \ldots, x_n)$$

For example, let us suppose we have in the knowledge base a possibility distribution on the cartesian product of X_1 and X_2, represented by the fuzzy relation R_j. Relation R_j may have been generated by a production rule of the type *If* $X_1 = A_1$ *then* $X_2 = A_2$, where A_1 and A_2 are fuzzy sets on the possible values of X_1 and X_2. Let us also suppose that we have a possibility distribution πX_1 representing our knowledge about variable X_1. Then the conjunction/projection method will yield a possibility distribution πX_2, which is a function of πX_1 and R_j. The successive use of the propagation process on the relations in the knowledge base will yield a result that takes into account all the information in the knowledge base.

The development of Possibility Theory gave rise to a non-standard logic, called necessity-valued logic [DPL 91], or PL1. In this logic, to each first-order formula φ, representing a statement in a knowledge base, we associate a constraint $N(\varphi) \geq \alpha$, where N is a necessity measure. The constraint $N(\varphi) \geq \alpha$ in PL1 is represented by the pair $(\varphi\ \alpha)$, called a *necessity-valued formula* (nvf). The quantity α is called the *valuation* of formula φ. In necessity-valued logic we make extensive use of the following properties :

$$N(\varphi \vee \neg\varphi) = 1$$
$$N(\varphi \wedge \psi) = \min(N(\varphi), N(\psi))$$
$$N(\varphi \vee \psi) \geq \max(N(\varphi), N(\psi))$$
$$\text{if } \varphi \models \psi \text{ then } N(\psi) \geq N(\varphi)$$

In PL1, the *graded modus ponens* [DPL 91], introduced in [DP 88], replaces the classical modus ponens rule:

$$(\varphi\ \alpha), (\varphi \rightarrow \psi\ \beta) \vdash (\psi\ \min(\alpha, \beta))$$

where $\rightarrow$ denotes the classical logical implication.

4.2 Logic PL1 and the production rule formalism

The use of logic PL1 in the rules formalism of the tool is quite straighforward, independently of the knowledge representation model. Let us suppose that we have a set of n facts φ_i and a rule R = *If* φ_1 *and* φ_2 *and ... and* φ_n *then* ψ_1 *and ... and* ψ_k, where both the facts and the rule are pervaded with uncertainty, represented in the possibilistic paradigm. Let us suppose that for each fact φ_i we have $N(\varphi_i) \geq \alpha_i$. The facts can then represented by n nvf's $(\varphi_i\ \alpha_i)$. Let us suppose that for rule R we have $N(R) \geq \beta$. Rule R stands in fact for k formulae $(\varphi_1 \wedge \ldots \wedge \varphi_n \rightarrow \psi_i\ \beta)$, $1 \leq i \leq k$, one for each conclusion ψ_i.

Using the n facts referenced in the premise, we obtain a global nvf $(\varphi_1 \wedge \ldots \wedge \varphi_n\ \alpha)$, where

$\alpha = \inf_{1 \le j \le n} \alpha_j$. Since these facts match the left hand side of the rules generated from R we obtain as result k nvf's (ψ_i min(α, β)), $1 \le i \le k$,. If the use of several rules yields m nvf's (ψ δ_l), $1 \le l \le m$, for a given conclusion ψ, the final global nvf relative to ψ will be (ψ δ), where $\delta = \sup_{1 \le l \le m} \delta_l$.

In what it refers to facts, the treatment of uncertainty using PL1 yields no problem when the facts are represented in logic. Indeed, we only have to represent each fact by a nvf (φ α), and build the knowledge base with only the meaningful ones, i.e. those where $\alpha > 0$. In knowledge representation models in which the concept of inheritance is present, such as frames and semantic nets, the treatment of facts is not so simple.

4.3 The Use of Possibilistic Logic in Frames

We show below how the logic PL1 was used in the frame formalism, in our system. We consider that uncertainty in this formalism has three ways of manifestation.

The first manisfestation of uncertainty to be dealt with is how much we believe in the value attached to a given slot in a given frame. In other words, it concerns how much we believe that an objet has a certain property. For instance, we have to find a way of stating that although all birds are oviparous, not all of them are capable of flying.

Here, we propose to simply attach a pair (c α) to a slot s in a frame f, where c is the value of c in f, and α is a valuation in logic PL1. This is represented by the nvf (φ(f,s,c) α), meaning that there is a necessity of at least α that φ(f,s,c) = "the value of slot s of frame f is c" is true. For instance, for the bird problem presented above, we could attach the nvf (oviparous 1) to the slot "reproduction" in frame "bird", and a nvf (True .9) for the slot "flies".

The second manifestation of uncertainty refers to how much we believe that a frame g inherits all the properties of its father frame f, which are not explicitly specified in g itself. In other words, it refers to how much a class is a true sub-class of another one. For instance, when constructing a frame structure concerning elephants, we may not be completely sure that Clyde is really a gray elephant, although it is our best guess. On the other hand, we cannot state that all circus elephants are royal elephants, although most of them are so. Note that these two cases characterize distinct kinds of uncertainty (the first one is more a matter of personal belief, the second one is derived from statistics). Here, however, we give them an uniform treatment.

We deal with the second manifestation of uncertainty by attaching a valuation to each link between two frames. This is done in the following manner. In each frame g with father f, we create a slot s_0 = "father" which represents in fact the link between frames f and g. To slot s_0 in g we then attach the pair (f α), where α is the valuation on the link between f and g. Formally, this is represented by the nvf (φ(g,s_0,f) α_0), meaning that there is a necessity of at least α_0 that φ(g,s_0,f) = "frame g inherits the attributs of frame f" is true.

Finally, the third manifestation of uncertainty refers to the rules in the knowledge base. We will see further on that the valuation attached to a rule will influence the valuations in the slots referenced in the conclusion of the rule.

4.3.1 Search of the Value of a Slot

We now discuss how we obtain the value and valuation concerning a given slot and frame. For that, we need some more notation. Let U be the set of all the frames that exist in a given moment of the processing of an application. Let g be a frame in U. Then

. S(g) is the set of slots in g.
. F(g) is the family of g defined by :
 i) $g \in F(g)$,
 ii) if $h \in F(g)$, and $k \in U$ is the father of h, then $k \in F(g)$.

First of all, let us recall the basic mechanism of inheritance in the frame paradigm, when no uncertainty is involved. A slot s concerning a particular frame f will not be created on a frame g descending from f, unless the user so specifies explicitly. If g does not have slot s, the value for s in g is the same as the value for s in f, i.e. the value in g is "inherited" from f.

When logic PL1 is used in the frame paradigm, we adopt the following solution for the problem of inheritance. Three situations are possible, when we search the pair (value valuation) of a slot s on a frame g :

i) s is a slot of g, i.e. $s \in S(g)$.

In this case the system simply yields the pair (c α) attached to s in g. For instance, in Fig. 2, the system yields the value "forest" and valuation .9 as the answer to the question "what is the type of area-1 ?" .

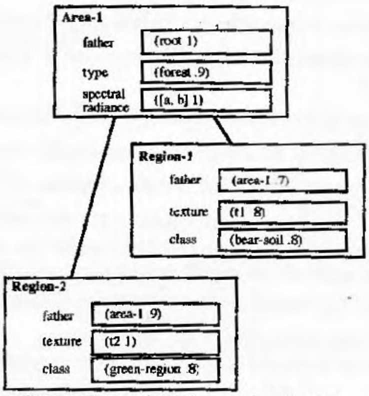


Fig. 2 - Frames in an image classification application

ii) s is not a slot of g, but s belongs to a frame h in the family of g, i.e. $s \in S(h)$, $h \in F(g)$, $h \neq g$.

Let (c α) be the pair attached to s in h. In this case the system yields c as the value, but the valuation will take into account not only α but also all the valuations on the links between g and h, represented by the slot s_0 = "father". Let (k_{0i} γ_{0i}) be the pair attached to slot s_0 in frame i. Formally, the system yields the pair (c $\min(\alpha, \gamma)$), where $\gamma = \min_{i \in F(g) - F(h)} \gamma_{0i}$. For instance, in Fig. 2 the system yields the value "forest" and valuation .7 as the answer to the question "what is the type of region-1 ?" .

iii) s is not present in any frame in the family of f. In this case the system disregards the query.

4.3.2 Creation and Modification of frames

A frame f can be created before the execution of the expert system. In this case, the user determines the nvf's (c_i α_i) to be attached to each slot s_i in S(f). The system makes sure that slot s_0, corresponding to the link, is created. It also verifies that all valuations α_i take their values in [0, 1].

Frame f can also be created or modified during run time, by performing the actions contained in the right-hand side of a rule R under execution (here we only consider the case in which inference is done using the forward-chaining mechanism [DK 77]).

The creation of frame f occurs if the right-hand side of rule R contains a clause referencing the (yet inexistant) frame f and slot s_0. Otherwise the system considers the action as a modification of frame : if the slot already exists the new value completely overrides the older one ; otherwise the new slot is simply created with the new value. In either case (creation or modification), the valuation to be attached to a slot is calculated using the valuation from the rule, the specific valuation for the slot (given in the conclusion of the rule) and the valuation calculated for the premise of the rule.

The treatment given to the creation and the modification of a frame f by a rule is uniform. Let R be a rule R = *If* φ_1 *and* φ_2 *and ... and* φ_n *then* ψ_1 *and ... and* ψ_k, where both the φ_i's and the ψ_i's reference slots in frames. Rule R stands in fact for k formulae ($\varphi_1 \wedge ... \wedge \varphi_n \rightarrow \psi_i\ \beta$), $1 \leq i \leq k$, where β is the valuation of rule R. The φ_j's are given by clauses of the type (f (s c)), and the ψ_i's are given by clauses of the type (f (s (c α))), where f is a frame, s is a slot, c a value and α a valuation, ((c α) is thus an nvf attached to s in f).

Let (c α_i) be the pair (value valuation) given to slot s_i by the right-hand side of R, and δ be the valuation obtained by the system for the left-hand side of R. Then the pair (c min(δ, β, α_i)) will be attached to slot s_i in f.

Note that valuations α_i and β are known to the system before the execution process starts. Therefore, it seems unnecessary to have the rule valuation β, since it could be incorporated directly in the slots valuations. In other words, instead of ($c_i\ \delta_i$) we could attach (c_i min(δ_i, β)) to slot s in f. We have chosen to keep the rule valuation β in our framework because it can help the conflict resolution module to order the rules. For instance, if in an application we want that a single rule be fired per cycle, we can choose the rule R_k which has the highest valuation min(δ_k, β_k), where δ_k stands for the valuation calculated for the left-hand side of R_k, and β_k stands for the rule valuation of R_k.

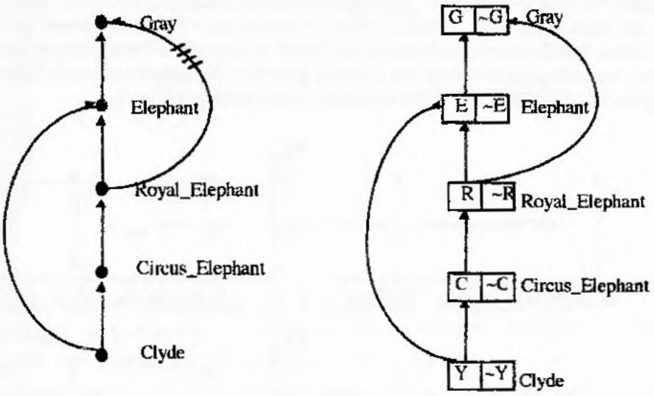
As an example of the use of rules, let us suppose that in our base we have a frame "region-1" having the slot "father" with nvf (area-1 .7) and the slot "texture" with nvf (t1 .8). Let R = *If (x (texture t1)) then (x (class (bear-soil .9))*, be a rule in the system with rule valuation $\beta = 1$. Substituting x as "region-1" in rule R yields the nvf (t1 .8) in the left-hand side of the rule. Following the notation given above we have $\beta = 1$, $\delta = .8$, and $\alpha = .9$. The application of the rule will then produce nvf (bear-soil min(.8, 1, .9)), which thus gives nvf (bear-soil .8) as the final result (see Fig. 2).

4.4 The Use of Possibilistic Logic in Semantic Nets

In the frame formalism show above, each slot in a frame may have only one value ; including the slot "father", which stands for the link between a frame and the frame directly above it in the frame hierarchy. The values in the slots may be pervaded with uncertainty, but only a single value per slot is allowed ; we are not interested in the other possible values, or the negation of the value. That is not the case in the semantic nets paradigm. In this formalism the negation of a hypothesis is present, even if implictly, by the negative link (see Fig. 3.a).

To make this duality character clearer, a semantic net with positive and negative links will here be represented using only *positive links* but where the nodes are split in two nodes : a *positive* and a *negative node*. In our new semantic nets formalism, a positive link to a node k in the former formalism, is now represented by a link to a (positive) node k. In this case, nothing really changes. However, a former negative link to a node k is now represented by a link to a (negative) node ¬k. In other words, we have transferred the negation on a link to the node at the end of the link. Furthermore, in this new formalism, only positive nodes can propagate information (see Fig. 3.b).

A first approach to incorporate the possibilistic framework in the semantic nets paradigm would be to assign a valuation ß to each link (f, g), where f is a positive node and g either a positive or a negative node. That is, this information would be represented as a pair (f -> g ß), meaning that N(f -> g) ≥ ß. To determine the final pair (value valuation) on a node of interest k in the former formalism, we would propagate information on the links in the new formalism and evaluate the results in nodes k and ¬k. The direct implementation of this approach leads to partial inconsistency, since in the possibilistic paradigm if we have a hypothesis h such that N(h) > 0, then we should have N(¬h) = 0 if the normalization condition is taken into account. In other words, the valuations on nodes k and ¬k could not be strictly positive at the same time, if the system is to be formally sound.



a) b)

Fig.3 : Semantic net in: a) formalism with negative links, b) formalism with positive and negative nodes

We propose here to make a short detour on the basic PL1 formalism in order to avoid partial inconsistency : we use possibility distributions to deal with part of the treatment of uncertainty in semantic nets, instead of using exclusively necessity measures as in the frame formalism. Of course, once we have a possibility distribution we can derive a necessity measure. What we propose here is in fact to use the mathematics of possibility theory to perform calculations in the semantic nets formalism, and only when a final resulting possibility distribution is obtained, to turn to the more synthetic logical possibilistic paradigm.

4.4.1 Possibility Distributions and Semantic Nets

We now discuss the general framework envolving possibility distributions and semantic nets. We will explain our approach using the example depicted in Fig. 3.a, taken from [Tou 84]. Fig. 3.b shows the same example using our formalism, where only one type of link is available. Every pair of positive/negative nodes in the new formalism represents the non-negated/negated propositions that represent the values of a variable. We have the following variables and values :

Variable	Positive Value	Negative Value
Elephant	E	¬E
Royal_Elephant	R	¬R
Circus_Elephant	C	¬C
Gray	G	¬G
Clyde	Y	¬Y

Each link is seen as a logical implication. For instance, the link in Fig.3.a between Clyde and Circus_Elephant is represented by Y -> C, and the negative link between Royal_Elephant and Gray is represented by E -> ¬G. The relations in our example are :

R_1 : R -> E	R_4 : C -> R
R_2 : E -> G	R_5 : Y -> C
R_3 : R -> ¬G	R_6 : Y -> E

These relations (i.e. the links) may convey information considered to be certain, like "Royal elephants are elephants" and "No royal elephant is gray". They may, on the other hand convey information pervaded with uncertainty, like "Most circus elephants are royal elephants", or "Most elephants are gray", based on statistics, and "There are reasons to think that Clyde is a circus elephant but one is not sure", based on personal intuition. Here we will represent the uncertainty on each link by a possibility distribution on a bi-dimensional space. In a real application, these distributions can be obtained from an expert in a very simple manner. At this point however we will suppose they are directly given to the system as such. The distributions representing the knowledge base of our example are depicted in Fig. 4.

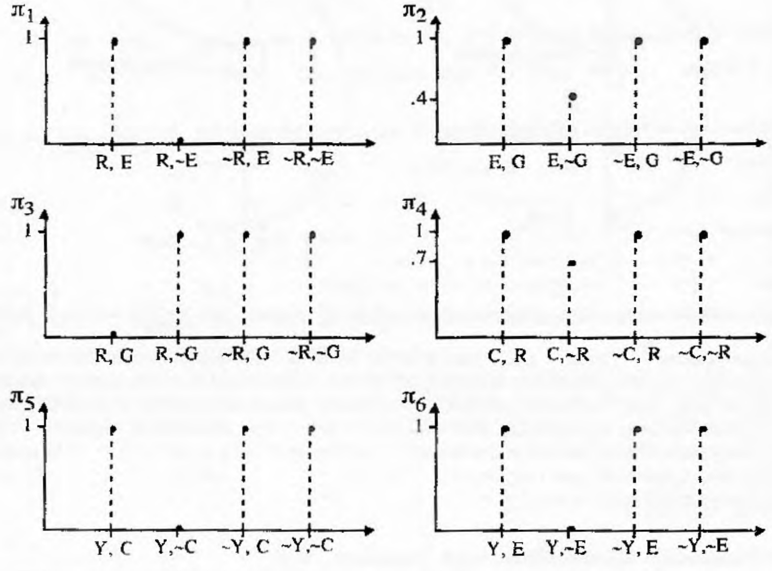


Fig 4. Possibility distributions representing the uncertainty on the links in a semantic net

4.4.2 Inference on Possibilistic Semantic Nets

Inference in a classical semantic net consists in propagating information from a start node and verifying if the property represented on the node of interest is satisfied or not, e.g. we propagate information from node Clyde to find out if it is gray or not. In our formalism, inference is done using the conjunction/projection process shown in Section 4.1.

In our example, we would insert on the knowldge base the possibility distribution π_7 for variable Clyde, given by $\pi_7(Y) = 1$, and $\pi_7(\neg Y) = 0$, and propagate information until variable Gray. In the first cycle of the application of the conjunction/projection procedure, distributions π_7 (representing the information on variable Clyde), and distribution π_5 (the relation between variables Clyde and Circus_Elephant, representing the statement "Clyde is surely a circus elephant"), are combined yielding distribution π_8 on variable Circus_Elephant (see Fig. 5).

Also in the first cycle, distributions π_6 and π_7 are combined yielding a distribution regarding variable Elephant. After applying this procedure successively on all nodes in the net, we obtain as the final result concerning variable Gray the distribution π given by $\pi(G) = .7$, and $\pi(\neg G) = .4$. We normalize this result and obtain $\pi(G) = 1$, and $\pi(\neg G) = .57$. The necessity that Clyde is a gray elephant is thus $N(G) = 1 - .57 = .43$. This result bring us finally back to the PL1 logical formalism : (G . 43) is the corresponding necessity-valued formula.

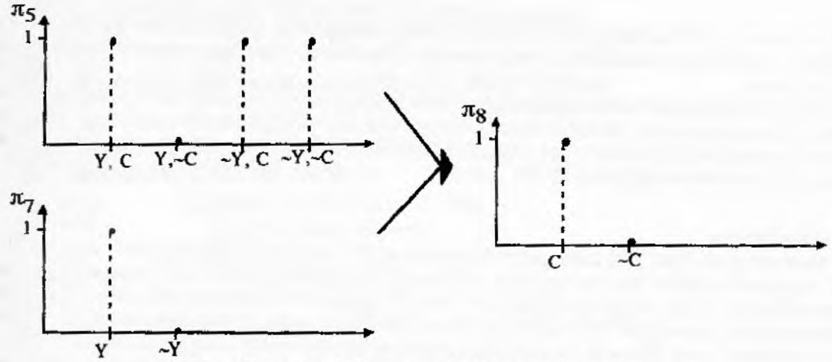


Fig 5. Ilustration of the process of conjunction/projection

4.4.3 Creation and Modification of Possibilistic Semantic Nets

The nodes and links in a semantic net can be created before the execution of the expert system. In this case, the user determines the valuation α to be (initially) attached to each link and the system makes sure that $0 \leq \alpha \leq 1$. The system then proceeds in the following manner.

Let a be the start node and b the end node of a link is-a in the net, and α the valuation on the link. In our new semantic net formalism node a (respec. b) is transformed in a positive node A and a negative node $\neg$A (respec. B and $\neg$B). Then, considering that $\Pi(A \rightarrow B) = 1$ and $N(A \rightarrow B) = \alpha$, a distribution for the relation A -> B is created in the following manner : $\pi(A,B) = \pi(\neg A,B) = \pi(\neg A,\neg B) = 1$, and $\pi(A,\neg B) = 1 - \alpha$. This transformation of necessity and possibility measures into possibility distributions is justified by the principle of minimum specificity [DP 86]. After the distribution is created, inheritance is dealt with as specified in Section 4.4.2. The same reasoning process applies when we have nodes a and b connected by a is-not-a link. In this case we consider that $\Pi(A \rightarrow \neg B) = 1$ and $N(A \rightarrow \neg B) = \alpha$, and construct the possibility distribution accordingly.

The nodes and links can also be created or modified during run time, by performing the actions contained in the right-hand side of a rule R under execution (again we only consider the case in which inference is done using the forward-chaining mechanism [DK 77]).

The creation of a node or link occurs if the right-hand side of rule R contains a clause referencing the (yet inexistant) node or frame. Otherwise the system considers the action as a modification of the semantic net : if a link between two nodes already exists the new information completely overrides the older one. In either case (creation or modification), the valuation to be initially attached to a link is calculated using the valuation calculated for the premise of the rule, combined to the valuation given in the conclusion clause of interest.

The treatment given to the creation and the modification of a net by a rule is uniform. Let R be a rule R = *If* φ_1 *and* φ_2 *and ... and* φ_n *then* ψ_1 *and ... and* ψ_k, where both the φ_j's and the ψ_i's reference nodes and links in the net. Rule R stands in fact for k formulae $\varphi_1 \wedge ... \wedge \varphi_n \rightarrow \psi_i$, $1 \leq i \leq k$. The φ_j's are given by clauses of the type (a l b), and the ψ_i's are given by clauses of the type (a l b α), where a is the start node, b the end node, l the link between a and b (is-a or is-not-a) and α a valuation for the rule considering that particular conclusion.

Let (a is-a b) be a clause in the premise of the rule under consideration. If there is a link between a and b, the system calculates the valuation on the link and sets it as the valuation of the clause. If there is a path between a and b but no direct link, the valuation on the clause is inferred as specified in Section 4.4.2 (using the positive and negative nodes derived from a,b, and the other nodes in the path). The final result obtained on node b is transformed into a valuation, and this is considered as the valuation of the clause. The same reasoning applies for a link is-not-a.

After calculating the valuation for each clause in the premise, we take their minimum as the valuation for the premise, which is then combined with the valuation in the conclusion clause of interest (again with the minimum). The final valuation is then attached to the link between the nodes of that conclusion clause.

5. Conclusion

We have presented the first formal developments of the uncertainty management module of a tool designed to allow the construction of customized expert system shells according to the characteristics of the target problem. The tool is capable of handling different knowledge representation paradigms. Its implementation has been motivated by the need to construct experts systems with different levels of complexity in the domain of image processing.

We presented here the way our tool uses logic PL1 to deal with uncertainty in the logical, frame and semantic net knowledge representation models. To apply logic PL1 to these formalisms it was necessary to take into account the semantics of inheritance adopted in them. Logic PL1 is the single uncertainty model provided by the tool so far; in the future, we intend to enrich the tool with other uncertainty models.

Two initial applications are being developed using expert system shells generated by the tool: a meteorological radar image interpretation system [Sil 92], and a remote sensing satelite image interpretation system.

Acknowledgements

This work was partially supported by FAPESP, contract No. 91/3532-2.

References

[BM 92] Bittencourt G., Marengoni M., *A Customizable Tool for the Generation of Production-Based Systems*, Tech. Rep. INPE-5453-NTC/306, São José dos Campos, Brazil, 1992. Also accepted to the 8th Int. Conf. on Artificial Intelligence in Engineering, to be held in 29/jun-1/jul 1993, in Toulouse, France.

[DK 11] Davis R., King J., "An Overview of Production Systems". Machine Intelligence, vol 8, pp 300-332, 1977.

[DP 86] Dubois D., Prade H., "The principle of minimum specificity as a basis for evidentional reasoning". in *Uncertainty in Knowledge-Based Systems*, B. Bouchon & R.R. Yager (eds), Springer Verlag, 1986.

[DP 88] Dubois D., Prade H. (with the collaboration of H. Farreny, R. Martin-Clouaire, C. Testemale), *Possibility Theory - An Approach to the Computerized Processing of Uncertainty*. Plenum Press, 1988.

[DPL 91] Dubois D., Lang J., Prade H., *Possibilistic Logic*. Report IRIT/91-98/R, IRIT, Toulouse, France, 1991.

[Mar 92] Marengoni M., "Uma Ferramenta para a Geração de Arcabouços em Sistemas Especialistas", MSc Thesis, Tech. Rep. INPE-5493-TDI/510, INPE, São José dos Campos, Brazil, 1992. (in Portuguese)

[Min 92] Minsky M., "A Framework to Represent Knowledge". In *The Psychology of Computer Vision*, (P. Winston, ed), McGraw-Hill, pp. 211-277, 1975.

[Qui 68] Quillian, M.R., "Semantic Memory". In *Semantic Information Processing*, (M.L. Minsky, ed), pp. 216-270, M.I.T. Press, Cambridge, Ma, 1968.

[Sil 92] Silva, F. de A.T.F. da, "A Hybrid Formalism for Representation and Interpretation of Image Knowledge". Proc. Int. Soc. for Photogrammetry and Remote Sensing Congress, Washington, DC, 1992.

[Tou 84] Touretzky D.S., "Implicit Ordering of Defaults in Inheritance Systems". Proc. AAAI '84, pp. 322-325, 1984.

[TRT 87] Touretzky D.S., Horty J.F., Thomason R.H., "A Clash of Intuitions : The Current State of Non-monotonic Multiple Inheritance Systems". Proc. 10th IJCAI, pp. 476-482, 1987.

[Zad 78] Zadeh L.A. (1978) Fuzzy sets as a basis for a theory of possibility. Fuzzy Sets and Systems, 1, 3-28.

[Zad 79] Zadeh L.A. (1979) A theory of approximate reasoning. In : Machine Intelligence, Vol. 9 (J.E. Hayes, D. Mikulich, eds.), Elsevier, Amsterdam, 149-194.