A Specialisation Calculus to improve Expert Systems Communication *

Josep Puyol-Gruart, Lluís Godo, Carles Sierra

Institut d'Investigació en Intel·ligència Artificial(IIIA)
Centre d'Estudis Avançats de Blanes (CSIC)
Camí de Santa Bàrbara, 17300 Blanes (Girona), Spain

May 1992

*This research has been supported by the CICYT TIC91–0430 project TESEU and the Esprit Basic Research Action number 3085 (DRUMS).

Abstract

The motivation of this work is the improvement of the classical input/output expert systems behaviour. In an uncertain reasoning context this behaviour consists of just getting certainty values for propositions. Instead, the answer of an expert system will be a set of formulas: a set of propositions and a set of specialised rules containing unknown propositions in their left part. This type of behaviour is much more informative than the classical one because gives to users not only the answer to a query but all the relevant information to improve the solution. A family of propositional rule-based languages founded on multiple-valued logics is presented and formalised. The deductive system defined on top of it is based on a *Specialisation Inference Rule* (SIR): $(A_1 \wedge A_2 \cdots \wedge A_n \rightarrow P, V), (A_1, V') \vdash (A_2 \wedge \ldots \wedge A_n \rightarrow P, V'')$, where V, V' and V'' are uncertainty intervals. This inference rule provides a way of obtaining rules containing unknown conditions in their premise as the result of the deductive process. The soundness and literal completeness of the deductive system are proved. The implementation of this deductive calculus is based on techniques of partial evaluation. Moreover, the specialisation mechanism provides an interesting way of validating knowledge bases. Keywords: Partial Evaluation, Expert Systems, Multiple-valued Logic.

1 Introduction and Motivation

Looking at an Expert System (ES) as a *blackbox*, the standard behaviour we can observe is as follows. The user queries to the system whether a given proposition can be deduced. If the system is able to deduce the proposition, its certainty value is given back. Otherwise the answer is *unknown* (open world assumption).

This behaviour is rather poor because the system usually has much more information that could be useful to the user, for instance:

1. When the system is able to answer the user's query, the user might also be interested in knowing other deductive paths that would be useful to improve the solution, or to know other conclusions that are deducible from the proposition answered.

2. When the system is not able to answer a query, it gives back the value *unknown* maybe because the user did not provided enough information to the system. Thus, the communication will be much more informative if the system is able to answer, not *unknown*, but with the information the user should know to come up with a value for the query.

All this hidden information can be used to better modelise communication among human experts. Looking carefully at how experts communicate their knowledge and at their problem solving procedures, we can find complex communication patterns. Sometimes experts cannot reduce their interaction only to the communication of certainty values for propositions. For instance, in medical diagnosis, when experts communicate, they also need:

1. **To condition their decisions**. Suppose that it is not known whether a patient is allergic to penicillin. An expert considering the possibility of giving penicillin as treatment would say: *Penicillin is a good treatment from a clinical point of view provided that the patient has no allergy to penicillin.*

2. **To give suggestions that must be considered with solutions**. Experts usually give other suggestions (*antibiogram*) that are related to the solution (*pneumococcus*). For instance the expert might say: *Pneumococcus has been isolated in the culture of sputum. In this case it is strongly suggested to make an antibiogram to the patient.*

3. **To give conditioned suggestions to be considered together with decisions**. Another example of complex communication is the combination of the above two communication patterns: *Ciprofloxacine is a good treatment, but if the patient is a woman on breast-feeding period she must stop breast-feeding.*

To model such communication protocols, we need to extend the ES answering procedure, by allowing to answer queries with sets of formulas (rules and propositions). We propose to do it by means of an Specialization Calculus of KBs.

Specialisation is based on the notion of partial evaluation expressed in the well known Kleene's Theorem. Partial evaluation algorithms have been intensively in logic programming [9] [2] [3] [8] [4] mainly for efficiency purposes. In this paper we propose the use of this technique to improve the communication behavior of ESs. With this purpose in section 2 we propose a partial evaluation mechanism for rule bases. In section 3 we formalise an Specialisation Calculus. Finally a little example and conclusions are presented in sections 4 and 5 respectively.

2 Proposal: Partial Evaluation in Rule Bases with Uncertainty

In rule bases, deduction is mainly based on the modus ponens inference rule:

$$A, A \rightarrow B \vdash B$$

This inference rule is only applicable when every condition of the premise is satisfied, otherwise nothing can be inferred. We will use partial evaluation to extract the maximum information from incomplete knowledge.

We base the partial evaluation in a rule base context on the well known logical equivalence $(A \wedge B) \rightarrow C \equiv A \rightarrow (B \rightarrow C)$ which leads to the following boolean specialisation inference rule:

$$A, A \wedge B \rightarrow C \vdash B \rightarrow C$$

The rule $B \rightarrow C$ is called the *specialisation* of $A \wedge B \rightarrow C$ with respect to the proposition A. Notice that in the particular case of $B = \emptyset$, we recover the usual modus ponens rule.

In a more formal way we give the following definitions.

Definition 1 (Rule Specialisation) *Let R be a set of rules and P a set of literals. We note rules as pairs, $r = (m_r, c_r)$ where m_r is the premise (a set of literals) and c_r is the conclusion (a literal). The rule specialisation is defined as a function:*

$$\mathcal{S}_{\mathcal{R}} : R \times P \rightarrow R \times P$$

$$\mathcal{S}_{\mathcal{R}}(r,p) = \begin{cases} (r, \emptyset) & \textit{if } p \notin m_r \\ (\emptyset, c_r) & \textit{if } m_r = \{p\} \\ ((m_r - \{p\}, c_r), \emptyset) & \textit{otherwise} \end{cases}$$

Definition 2 (KB Specialisation) *Let KB be a set of knowledge bases. We note KBs as pairs $kb = (R_{kb}, P_{kb})$ where R_{kb} is a set of rules and P_{kb} is a set of propositions. KB specialisation is defined as a function:*

$$\mathcal{S}_{\mathcal{KB}} : KB \rightarrow KB$$

$$\mathcal{S}_{\mathcal{KB}}(kb) = \begin{cases} \mathcal{S}_{\mathcal{KB}}((R_{kb} - \{r\} + \{r'\}, P_{kb} + \{p'\})), (*) \\ kb \text{ , otherwise} \end{cases}$$

(*) if $P_{kb} \neq \emptyset$ and $\exists p \in P_{kb}$ and $\exists r \in R_{kb}$ such that $\mathcal{S}_{\mathcal{R}}(r, p) = (r', p')$ and $r' \neq r$

In other words, the specialisation of a *kb* consists on the exhaustive specialisation of its rules. Rules whose conditions contain propositions with known values are replaced by their specialisations. Rules that only have one condition will be eliminated and a new proposition will be added. This new proposition will be used again to specialise the *kb*. The process will finish when the *kb* has no rule containing on its conditions a known proposition. This approach is different for instance from the logic programming one used in [4]. There, partial evaluation is goal driven, whereas here partial evaluation is data driven.

In an uncertain reasoning context we propose to extend the above boolean specialisation inference rule as follows:

Definition 3 (SIR) *Given a proposition A with certainty value α, and a rule with certainty value ρ, then*

$$(A, \alpha), (A \wedge B \rightarrow C, \rho) \vdash (B \rightarrow C, \rho')$$

where $\rho' = MP^1(\alpha, \rho)$ is the new value of the rule.

Therefore we need to extend the previous definition of the function $\mathcal{S}_{\mathcal{R}}$ to allow the handling of certainty values.

Definition 4 (Specialisation of Uncertain Rules) *Let R^* be a set of weighted rules and P^* a set of weighted literals. We note weighted rules as pairs, $r^* = (r, u_r)$ where r is a classical rule and u_r is the certainty value of r. And we note weighted literals as pairs, $p^* = (p, u_p)$ where p is a classical literal and u_p is the certainty value of p.*

$$\mathcal{S}_{\mathcal{R}} : R^* \times P^* \rightarrow R^* \times P^*$$

$$\mathcal{S}_{\mathcal{R}}(r^*, p^*) = \begin{cases} (r^*, \emptyset) & \text{if } p \notin m_r \\ (\emptyset, p^{*'}) & \text{if } m_r = \{p\} \\ (r^{*'}, \emptyset) & \text{otherwise} \end{cases}$$

where $r^{*'} = (m_r - \{p\}, c_r, MP(u_p, u_r))$ and , $p^{*'} = (c_r, MP(u_p, u_r))$.

[1] SIR is parametric on the uncertainty propagation function MP (modus ponens), particular for each uncertainty calculus.

It is easy to extend (not included here) the classical KB specialisation to an uncertain KB specialisation.

Now, the answer to a query can be considered as a specialised kb: The specialised kb obtained from $kb = (R^*, P^*)$ where R^* is the set of rules in deductive paths to and from the query. And P^* is the set of propositions defining a case.

3 Formalisation of a Specialisation Calculus for Rule Bases

In this section we present the definition of a family of multiple-valued logics with a deductive system based on a specialisation inference rule. Some aspects of these logics have been already described in [1]. Each logic is determined by a particular algebra of truth-values from a parametric family that is described next.

An **algebra of truth-values** is a finite algebra $\mathcal{A}_T^n =< A_n, N_n, T, I_T >$ such that:

- The set of truth-values A_n is a chain:

$$0 = a_1 < a_2 < \cdots < a_n = 1$$

  where 0 and 1 are the booleans False and True respectively.

- The negation operator N_n is an unary operation defined as

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

  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: $N_n^2 = Id$.

- The conjunction operation T is a binary operation satisfying $\forall a, b, c \in A_n$:

  T1: $T(a, b) = T(b, a)$

  T2: $T(a, T(b, c)) = T(T(a, b), c)$

  T3: $T(0, a) = 0$

  T4: $T(1, a) = a$

  T5: if $a \leq b$ then $T(a, c) \leq T(b, c)$ for all c

- The implication operator I_T is defined by residuation with respect to T, i.e.

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

  and satisfies the following properties:

I1: $I_T(a, b) = 1$ if, and only if, $a \leq b$.

I2: $I_T(1, a) = a$

I3: $I_T(a, I_T(b, c)) = I_T(b, I_T(a, c))$

I4: If $a \leq b$, then $I_T(a, c) \geq I_T(b, c)$ and $I_T(c, a) \leq I_T(c, b)$

I5: $I_T(T(a, b), c) = I_T(a, I_T(b, c))$

As it is easy to notice from the above definition, any of such truth-values algebras is completely determined as soon as the set of truth-values A_n and the conjunction operator T are determined. So, varying this two parameters we obtain a family of multiple-valued logics, including, among others, Kleene's and Lukasiewicz's logics.

In the following description of the language, the semantics and the deduction system (specialisation calculus) of a particular logic, we suposse fixed an algebra A_T^n. This calculus is proved to be sound and also complete if constrained to the case of literals.

3.1 Syntax

A propositional language $\mathcal{L}_n = (A_n, \Sigma, \mathcal{C}, \mathcal{S}_n)$ is defined by:

- A signature Σ consisting on a set of atomic symbols plus *true* and *false*.
- A set of Connectives: $\mathcal{C} = \{\neg, \wedge, \rightarrow\}$
- A set of Sentences: $\mathcal{S}_n =$ Mv-Literals $\cup$ Mv-Rules

  Sentences are pairs of classical-like propositional sentences and intervals of truth-values. The classical-like propositional sentences are restricted to be literals or rules. Thus, the sentences of the language are of the following types:

  Mv-Atoms: $\{(p, V) \mid p \in \Sigma$ and V is an interval of truth-values of $A_n\}$

  Mv-Literals: $\{(p, V) \mid (p, V) \in$ Mv-Atoms or $p = \neg q$ and $(q, V) \in$ Mv-Atoms$\}$

  Mv-Rules: $\{(p_1 \wedge p_2 \wedge \cdots \wedge p_n \rightarrow q, V) \mid p_i$ and q are literals, $V = [a, 1]$ is an upper interval of truth-values of A_n where $a > 0$, and $\forall i, j (p_i \neq p_j, p_i \neq \neg p_j, q \neq p_j, q \neq \neg p_j)\}$

3.2 Semantics

- Models M_ρ are defined by valuations ρ, i.e. mappings from the first components of sentences to A_n such that:

  $\rho(\neg p) = N_n(\rho(p))$

  $\rho(p_1 \wedge p_2) = T(\rho(p_1), \rho(p_2))$

  $\rho(p \to q) = I_T(\rho(p), \rho(q))$

  $\rho(true) = 1$

  $\rho(false) = 0$

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

$$M_\rho \models (p, V) \text{ iff } \rho(p) \in V$$

- Semantical entailment between sets of sentences and sentences is defined as usual:

  $\Gamma \models A$ iff for any model $M_\rho \models \Gamma$ implies $M_\rho \models A$,

  for any set of sentences Γ and sentence A.

Definition 5 *We define on the set of intervals of A_n the functions N_n^* and T^* as those functions that give the minimal interval containing the point-wise extensions of N_n and T respectively. That is:*

- $N_n^*([a, b]) = [N_n(b), N_n(a)]$
- $T^*([a, b], [c, d]) = [T(a, c), T(b, d)]$

In order to get a functional expression of the multiple-valued version of the Modus Ponens rule, we also define on the set of intervals of A_n the function MP_T^* as follows:

Definition 6 *For any truth-intervals V and W, we define $MP_T^*(V, W)$ as the minimal interval containing all solutions for z in the familly of functional equations*

$$I_T(a, z) = b$$

varying $a \in V$ and $b \in W$.

This definition can be made more explicit when taking into account that truth-intervals W accompanying Mv-Rules are always upper intervals, i.e. W is of the form $W = [c, 1]$.

Proposition 1 $MP_T^*([a, b], [c, 1]) = [T(a, c), 1]$

Proof. It reduces to find all solutions of the functional inequations

$$I_T(x, z) \geq c$$

being $a \leq x \leq b$. However, given a residuated pair (T, I_T) it is well known [5] that the following relation holds:

$$T(x, y) \leq z \text{ iff } I_T(x, z) \geq y$$

Then, the solution for the first equation is $z \geq T(x, c)$, and taking into account that $x \geq a$, and that $z = 1$ is always a solution, the minimal interval that will contain all the solutions for z is $[T(a, c), 1]$.□

A set of interesting properties of the semantic entailment that will play a major role in later proofs is presented next.

Proposition 2 *If $p, q, p_1, \ldots, p_n$ denote literal symbols then the following properties are fulfilled:*

SR1: $(p, v) \models (p, W) \Leftrightarrow V \subseteq W$

SR2: $(p, V) \models (\neg p, W) \Leftrightarrow N_n^*(V) \subseteq W$

SR3: $(p, V), (p, W) \models (p, U) \Leftrightarrow V \cap W \subseteq U$

SR4: $(p_i, V_i), (p_1 \wedge \cdots \wedge p_n \rightarrow q, V) \models (p_1 \wedge \cdots \wedge p_{i-1} \wedge p_{i+1} \wedge \cdots \wedge p_n \rightarrow q, W) \Leftrightarrow MP_T^*(V_i, V) \subseteq W$

SR5: $MP_T^*(T^*(V_1, \ldots, V_n), W) = MP_T^*(V_1, MP_T^*(V_2, \ldots, MP_T^*(V_n, W) \ldots))$, *if* $W = [w, 1]$

Proof.

SR1: Straightforward from the satisfaction relation definition.

SR2: Follows from SR1 and the fact that a valuation ρ satisfies $\rho(p) \in V$ if, and only if, $\rho(\neg p) \in N^*(V)$.

SR3: Straightforward from the satisfaction relation definition.

SR4: First we prove the property for the simplest Modus Ponens case, i.e.,

$$\{(p, U), (p \rightarrow q, V)\} \models (q, W) \text{ iff } MP_T^*(U, V) \subseteq W$$

By definition of the function MP_T^*, $MP_T^*(U, V)$ is the minimal interval containing all the solutions for $\rho(q)$ in the following family of functional equation system:

$$\begin{cases} \rho(p) = a \\ \rho(p \to q) = I_T(\rho(p), \rho(q)) = b \end{cases}$$

for any $a \in U$, and $b \in V$. Thus, for any model ρ satisfying $\rho(p) \in U$, and $\rho(p \to q) \in V$ and $\rho(q) \in W$, it must be only the case that $\rho(p) \in MP^*_T(U, V)$, and thus $MP^*_T(U, V) \subseteq W$. Moreover, if $U = [x, y]$ and $V = [z, 1]$, then $MP^*_T(U, V) = [T(x, z), 1]$.

Now property SR4 follows straightforward from the associativity of the t-norm T, used to interpret conjunctions, and from the fact that a residuated pair (T, I_T) satisfies the following equality:

$$I_T(T(x, y), z) = I_T(x, I_T(y, z))$$

SR5: From proposition 1, it follows that, if $U = [x, y]$ and $V = [z, 1]$, then $MP^*_T(U, V) = [T(x, z), 1]$. Then, it is easy to see that, due to the associativity of the t-norm T, if $V_i = [a_i, b_i]$, for $i = 1, \ldots, n$, then

$$T^*(V_1, \ldots, V_n) = [T(a_1, \ldots, a_n)^2, T(b_1, \ldots, b_n)]$$

and thus, on the one hand

$$\begin{aligned} MP^*_T(V_1, MP^*_T(V_2, \ldots, MP^*_T(V_n, W)\cdots)) &= [T(a_1, \ldots, T(a_n, w)\cdots), 1] \\ &= [T(a_1, \ldots, a_n, w), 1] \end{aligned}$$

and on the other hand

$$\begin{aligned} MP^*_T(T^*(V_1, \ldots, V_n), W) &= MP([T(a_1, \ldots, a_n), T(b_1, \ldots, b_n)], [w, 1]) \\ &= [T(a_1, \ldots, a_n, w), 1] \end{aligned}$$

3.3 Specialisation Calculus

The specialisation calculus is based on:

1. The following axioms:

   AS1: $(\neg\neg p \to p, [1, 1])$

   AS2: $(p, [0, 1])$

   A1: $(true, [1, 1])$

   A2: $(false, [0, 0])$

[2]The expression $T(r_1, r_2, r_3, \ldots)$ is the recurrent application of T as $T(r_1, T(r_2, T(r_3 \ldots) \ldots)$

2. The following inference rules:

   Weakening: $(p, V_1) \vdash (p, V_2)$ where $V_1 \subseteq V_2$

   Not-introduction: $(p, V) \vdash (\neg p, N_n^*(V))$

   Composition: $(p, V_1), (p, V_2) \vdash (p, V_1 \cap V_2)$

   SIR: $(p_i, V_i), (p_1 \wedge \cdots \wedge p_i \wedge \cdots \wedge pn \rightarrow q, V_r) \vdash (p_1 \wedge \cdots \wedge p_{i-1} \wedge p_{i+1} \wedge \cdots \wedge p_n \rightarrow q, MP_T^*(V_i, V_r))$

3.4 Soundness Theorem

From properties SR1, SR2, SR3 and SR4 of the semantical entailment, it is easy to check that the above specialisation calculus is sound.

Theorem 1 (Soundness) *Let A be a sentence and Γ a set of sentences. Then $\Gamma \vdash A$ implies $\Gamma \models A$*

Proof. The properties S1-S4 show that the inference rules are locally sound and complete. So, we need only to show that the axioms are sound to have the proof of the theorem.

1. If A is the axiom AS1, i.e. $A = (\neg\neg p \rightarrow p, [1, 1])$ then for every model M_ρ, $\rho(p) = N(N(\rho(p))) = N(\rho(\neg p)) = \rho(\neg(\neg(p))) \Rightarrow I(\neg\neg p \rightarrow p) = I(\rho(\neg\neg p), \rho(p)) = 1$. Then, for all M_ρ, $M_\rho \models (\neg\neg p \rightarrow p, [1, 1])$.

2. If A is the axiom AS2, i.e. $A = (p, [0, 1])$, it is the case that every model M_ρ satisfies $\rho(p) \in [0, 1]$. Then for all M_ρ, we have trivially that $M_\rho \models (p, [0, 1])$.

3. If A is the axiom A1, i.e., $A = (true, [1, 1])$ then, by definition, for every model M_ρ, $\rho(true) = 1 \Rightarrow M_\rho \models (true, [1, 1])$.

4. If A is the axiom A2, the proof is analogous to the previous case.

3.5 Literal Completeness

It is straightforward to see that our deductive system is not complete. For instance, we have $\{(p \rightarrow q, 1), (q \rightarrow r, 1)\} \models (p \rightarrow r, 1)$ but $\{(p \rightarrow q, 1), (q \rightarrow r, 1)\} \not\vdash (p \rightarrow r, 1)$. It is also the case that the language is not complete for literal deduction in general. For instance, we have $\{(p \rightarrow q, 1), (\neg p \rightarrow q, 1)\} \models (q, 1)$ but $\{(p \rightarrow q, 1), (\neg p \rightarrow q, 1)\} \not\vdash (q, 1)$. However, it can be proved that the system is complete for literal deduction in the context of a restricted language setting, as it will be shown in this section.

3.5.1 Previous definitions

Definition 7 (Mv-Horn-Rules) *We define the set Mv-Horn-Rules as the set* $\{(p_1 \wedge p_2 \wedge \cdots \wedge p_n \rightarrow q, V) \mid p_i$ *and* q *are atomic symbols,* $V = [a, 1]$ *is an interval of truth-values of* A_n *with* $a > 0$*, and* $\forall i, j(p_i \neq p_j, q \neq p_j)$ $\}$

Definition 8 (Restricted Language) *Given the propositional language*

$$\mathcal{L}_n = (A_n, \Sigma, \mathcal{C}, \mathcal{S}_n)$$

we define a restricted propositional language as:

$$\mathcal{RL}_n = (A_n, \Sigma, \mathcal{C}, \mathcal{RS}_n)$$

where $\mathcal{RS}_n$ = *Mv-Atoms* $\cup$ *Mv-Horn-Rules*

For any $\Gamma \subset \mathcal{RS}_n$ the next notation will be used:

- $\Gamma = \Gamma_L \cup \Gamma_R$
- $\Gamma_L = \{\gamma \in \Gamma | \gamma \text{ are mv-Atoms}\}$
- $\Gamma_R = \{\gamma \in \Gamma | \gamma \text{ are mv-Horn-Rules}\}$
- *Prem*: Is a function that given a rule returns its conditions.
- *Cond*: Is a function that given a rule returns its conclusion.
- $\Gamma_L^A = \{p \in \Sigma | \exists V \text{ interval of } A_n : (p, V) \in \Gamma_L\}$
- $\Gamma_R^A = \{p \in \Sigma | \exists r \in \Gamma_R : p \in Prem(r) \text{ or } p = Concl(r)\}$
- $\Gamma^A = \Gamma_L^A \cup \Gamma_R^A$

3.5.2 Previous Lemmas

Proposition 3 *The inference rules (Weakening, Composition, Negation and Specialisation) are locally complete, i.e., they verify the following equivalences:*

1. $(p, V) \vdash (p, W)$ *iff* $(p, V) \models (p, W)$
2. $(p, V) \vdash (\neg p, W)$ *iff* $(p, V) \models (\neg p, W)$
3. $\{(p, V), (p, W)\} \vdash (p, U)$ *iff* $\{(p, V), (p, W)\} \models (p, U)$
4. $\{(p_1, V_1), \ldots, (p_n, V_n), (p_1 \wedge \cdots \wedge p_n \rightarrow q, W)\} \vdash (q, U)$ *iff* $\{(p_1, V_1), \ldots, (p_n, V_n), (p_1 \wedge \cdots \wedge p_n \rightarrow q, W)\} \models (q, U)$

Proof. Straightforward from properties SR1-SR5 and from the definition of the four inference rules in Section 3.3.

Lemma 1 *Let Γ_L be a set of Mv-Atoms, and let R_1, R_2 be two sets of mv-Horn-Rules. R_1 and R_2 rules have as premisses conjunctions of atoms belonging to Γ_L^A, and share the same conclusion, an atom p not belonging to Γ_L^A.*

If $V_1 = \bigcap\{V''|\{\Gamma_L, R_1\} \models (p, V'')\}$, $V_2 = \bigcap\{V''|\{\Gamma_L, R_2\} \models (p, V'')\}$ and $W = \bigcap\{V''|\{\Gamma_L, R_1, R_2\} \models (p, V'')\}$, then $W \supseteq V_1 \cap V_2$.

Proof. By *reductio ad absurdum*. Suppose that $W \not\supseteq V_1 \cap V_2$. Then $\exists \alpha \in V_1 \cap V_2$ and $\alpha \notin W$. Because of V_1 and V_2 are minimals, we have that:

- $\alpha \in V_1 \Rightarrow \exists M_\rho$ such that $\rho(p) = \alpha$, $M_\rho \models \Gamma_L$ and $M_\rho \models R_1$
- $\alpha \in V_2 \Rightarrow \exists M_{\rho'}$ such that $\rho'(p) = \alpha$, $M_{\rho'} \models \Gamma_L$ and $M_{\rho'} \models R_2$

We will prove that there always exists a model $M_{\rho''}$ such that $\rho''(p) = \alpha$, $M_{\rho''} \models \Gamma_L$, $M_{\rho''} \models R_1$ and $M_{\rho''} \models R_2$. Define $\rho''(p) = \alpha$, and $\rho''(a) = min(\rho(a), \rho'(a))$, $\forall a \in \Gamma_L^A$. $M_{\rho''}$ easily extends to the Mv-Horn-Rules by the implication function I_T. Then, for this model $M_{\rho''}$ we have:

1. $\rho''(p) = \alpha$.
2. $M_{\rho''} \models \Gamma_L$: due to the fact that $\rho'' = min(\rho, \rho')$ over Γ_L^A.
3. $M_{\rho''} \models R_1$: $\forall r \in R_1$, where $r = (q_1 \wedge \cdots \wedge q_m \rightarrow p, [v_r, 1])$, we have that $M_\rho \models R_1$ implies $\rho(r) \geq v_r$. Given that we work with Mv-Horn rules, i.e. q_i are not negated literals, and the monotonicity property of function T, it always holds that:

   $$\rho''(q_1 \wedge \cdots \wedge q_n) = T(\rho''(q_1), \ldots, \rho''(q_n)) \leq T(\rho(q_1), \ldots, \rho(q_n)) = \rho(q_1 \wedge \cdots \wedge q_n)$$

   and, given that I_T is not increasing in the first argument, it always holds that:

   $$\rho''(r) = I_T(\rho''(q_1 \wedge \cdots \wedge q_n), \alpha) \geq I_T(\rho(q_1 \wedge \cdots \wedge q_n), \alpha) = \rho(r) \geq v_r$$

   that is, $M''_\rho \models R_1$.
4. $M_{\rho''} \models R_2$: Analogously to the previous case.

Summarizing, we have found $M_{\rho''}$ such that $\rho''(p) = \alpha$, $M_{\rho''} \models \Gamma_L$, $M_{\rho''} \models R_1$ and $M_{\rho''} \models R_2$, i.e. $\{\Gamma_L, R_1, R_2\} \not\models (p, W)$ which is in contradiction with the enunciate of the lemma.□

Next Lemma shows that previous deductions over a Mv-Atom p do not restrict the models of Mv-Atoms belonging to premisses of other rules concluding the same Mv-Atom p. In practical terms, having previous deductions over an atom r means that we know r with an interval of truth values of type $[v, 1]$. Otherwise, if we knew r with a general interval of type $[b, c]$ it could be the case that premises of rules concluding r would be semantically deduced with intervals different of $[0, 1]$. On the contrary, it would not be possible to syntactically deduce them. So, next Lemma allows us, when considering atom deducibility, to only consider those rules that deduce it and not the rules that use it as a premise.

Lemma 2

$$\bigcap\{V''|\{(p \wedge q_1 \wedge \cdots \wedge q_n \rightarrow r, [a, 1]), (r, [b, 1])\} \models (p, V'')\} = [0, 1]$$

Proof. It is sufficient to prove that $\forall\alpha \in [0, 1]$, we can find a model M_ρ such that $\rho(p) = \alpha$ and that $M_\rho \models (p \wedge q_1 \wedge \cdots \wedge q_n \rightarrow r, [a, 1])$ and $M_\rho \models (r, [b, 1])$. Actually, every model M_ρ such that $\rho(p) = \alpha$ and $\rho(r) = 1$, satisfies that $\rho(p \wedge q_1 \wedge \cdots \wedge q_n \rightarrow r) = 1$, and thus $M_\rho \models (p \wedge q_1 \wedge \cdots \wedge q_n \rightarrow r, [a, 1])$ and $M_\rho \models (r, [b, 1])$. □

3.5.3 Restricted Literal Completeness Theorem

Theorem 2 (Restricted Literal Completeness) *If $\Gamma \models (p, V)$, then $\Gamma \vdash (p, V)$, provided that $p \in \Gamma^A$, where Γ is such that the following conditions hold:*

1. $\Gamma \subset \mathcal{RS}_n$
2. $\forall r \in \Gamma_R : concl(r) \notin \Gamma_L^A$
3. *The deductive and/or graph associated to Γ is acyclic.*

Proof. Given that the and/or deductive graph associated to Γ is acyclic, we can decompose the set Γ^A of atomic symbols appearing in Γ in a set of disjoint layers. The definition of the layers is the following:

- $S_0 = \{q \in \Gamma^A |\ \nexists r \in \Gamma_R : Concl(r) = q\}$
- $S_1 = \{q \in \Gamma^A | \forall r \in \Gamma_R : Concl(r) = q \Rightarrow \forall x \in Prem(r), x \in S_0\}$
- ...
- $S_i = \{q \in \Gamma^A | \forall r \in \Gamma_R : Concl(r) = q \Rightarrow \forall x \in Prem(r), x \in S_j,$ being $j < i$ and $\exists r : \exists x \in Prem(r), x \in S_{i-1}\}$
- ...

The proof of the theorem is by induction over the layer number n to which p belongs. Suppose that $V \neq [0,1]$, otherwise the proof of the theorem is trivial.

The set Γ^A is decomposed in layers $\Gamma^A = \bigcup_{i=1,n} S_i$. Because of Lemma 2, in order to deduce p we only need to consider that part of Γ containing rules using atoms belonging to layers lower than the layer of p. That is, we consider only those rules of Γ belonging to the deductive subgraph of p.

Case $n = 0$: In this case Γ contains a set of mv-atoms as

$$\{(p, V_i) | i \in I\} \subseteq \Gamma_L$$

Then, it is easy to see that every model M_ρ that satisfies Γ must hold $\rho(p) \in (\bigcap_{i \in I} V_i)$, and then $(\bigcap_{i \in I} V_i) \subseteq V$. Therefore, we can assure that if we apply repeatedly the *composition* and *weakening* rules, we also can deduce syntactically (p, V). Then the theorem is true for $n = 0$.

Induction hypothesis: The theorem is true for $n - 1$.

Case n: Suppose that $p \in S_n$. Given R_p, the set of rules of Γ with conclusion p, then $\forall r \in R_p$, the premisses of r belong to lower layers. Let be $V_q = \bigcap\{V | \Gamma \models (q, V)\}$, $\forall q \in Prem(r)$, $\forall r \in R_p$. By the induction hypothesis we have that $\Gamma \models (q, V_q) \Rightarrow \Gamma \vdash (q, V_q)$, $\forall q \in Prem(r)$, $\forall r \in R_p$.

By induction over nrp, the number of rules of R_p, and together with the conditions of the theorem, we will prove that $\Gamma \models (p, V)$ implies $\Gamma \vdash (p, V)$.

1. $nrp = 1$. In this case we have $R_p = \{(q_1 \wedge \cdots \wedge q_m \rightarrow p, W)\}$, $\Gamma \models (q_i, V_{q_i})$ for $i = 1, \ldots, m$, where V_{q_i} are minimals. From Lemma 2 we have that $\Gamma \models (p, V)$ if and only if

$$\{(q_1 \wedge \cdots \wedge q_m \rightarrow p, W)(q_1, V_{q_1}), \ldots, (q_m, V_{q_m})\} \models (p, V)$$

but from properties SR-4 and SR-5 of proposition 2, this holds if and only if

$$V \supseteq MP_T^*(T^*(V_{q_1}, V_{q_2}, \ldots, V_{q_m}), W)$$

Given that

$$MP_T^*(T^*(V_{q_1}, V_{q_2}, \ldots, V_{q_m}, W) = MP_T^*(V_{q_1}, MP_T^*(V_{q_2}, \ldots, MP_T^*(V_{q_m}, W) \cdots))$$

it is easy to see that by succesive applications of SIR inference rule we can obtain $\Gamma \vdash (p, V)$, and thus for $nrp = 1$ the theorem is true.

2. Suppose that the theorem is true for $nrp = k - 1$.

3. $nrp = k$. In this case we have that $R_p = \{r_1, r_2, \ldots, r_k\}$ and $\Gamma \models (q_{ij}, W_{q_{ij}})$ for all $q_{ij} \in Prem(r_i)$, $i = 1, \ldots, k$, being $W_{q_{ij}}$ minimal. Let be $r_i = (\wedge q_{ij} \rightarrow p, V_i)$ and $A_p = \bigcup_{i,j}(q_{ij}, W_{q_{ij}})$. Again Lemma 2 allows us to state that $\Gamma \models (p, V)$ if and only if $V \supseteq U$, where $\{r_1, \ldots, r_k\} \cup A_p \models (p, U)$ and U minimal,. Consider $R_p = R^*_p \cup r_k$, where $R^*_p = \{r_1, \ldots, r_{k-1}\}$ and let be $V^* = \bigcap\{V''|\{R^*_p, A_p\} \models (p, V'')\}$. By induction hypothesis we have also $R^*_p \vdash (p, V^*)$. Furthermore we have $\{r_k, A_p\} \vdash (p, MP^*_T(T^*(W_{q_{k1}}, \ldots, W_{q_{kj_k}}, V_k))$, and because of Lemma 1 we know that $MP^*_T(T^*(W_{q_{k1}}, \ldots, W_{q_{kj_k}}, V_k) = \bigcap\{V''|\{r_k, A_p\} \models (p, V'')\}$. Then from Lemma 1, we have:

$$\{R_p, A_p\} \models (p, V) \text{ iff } V \supseteq V^* \cap MP^*_T(T^*(W_{q_{k1}}, \ldots, W_{q_{kj_k}}, V_k)$$

   Finally it is easy to notice that $V^* \cap MP^*_T(T^*(W_{q_{k1}}, \ldots, W_{q_{kj_k}}, V_k)$ can be obtained by succesive applications of SIR and *composition* inference rules, that is, we can finally conclude that $\Gamma \vdash (p, V)$.□

4 Example

Milord II is a modular language for knowledge engineering that manages uncertainty and reflection. It includes an inference engine that implements the specialisation calculus described in this paper [7] [6]. In this section an example will be presented. This example is part of a real application for pneumonia treatment written in Milord II, named Terap-IA. When writting the example we will use some extensions of the language described in section 3.

The set of truth-values used is A_n =*(impossible, slightly-possible, possible, very-possible, definite)* where $impossible = 0$ and $definite = 1$.

Consider the following rules for pneumonia treatment [3]:

R0 (H-Influenzae → Quinolones, *possible*)
R1 (female ∧ young ∧ pregnant ∧ Legionella-sp → Co-trimoxazole, *slightly-possible*)
R2 (female ∧ young ∧ breast-feeding ∧ ≥(Quinolones, *possible*)[4] → stop-breast-feeding, *definite*)
R3 (breast-feeding ∧ Co-trimoxazole → stop-breast-feeding, *definite*)

[3]In this rules *H-Influenzae* and *Legionella-sp* are possible diagnosis, and *Quinolones* and *Co-trimoxazole* are antibiotics. Also, we have simplified the intervals syntax, as it is done in Milord II. Intervals of the type $[a, 1]$ appearing as values of rules and propositions are written just as a.

[4]"≥" is a boolean metapredicate of Milord II such that "$\geq (p, a)$" is true if and only if $(p, [b, c])$ has been deduced with $b \geq a$.

Consider the case of a young female patient with a diagnosis of *H-Influenzae*. The propositions representing this case are:

(H-Influenzae, *very-possible*)
(female, *definite*)
(young, *definite*)

If we specialise the *kb* composed by the rules and the propositions above presented, it is easy to see that the final set of rules obtained is:

R1' (pregnant $\wedge$ Legionella-sp $\rightarrow$ Co-trimoxazole, *slightly-possible*)
R2' (breast-feeding $\rightarrow$ stop-breast-feeding, *definite*)
R3 (breast-feeding $\wedge$ Co-trimoxazole $\rightarrow$ stop-breast-feeding, *definite*)

and the the final set of propositions is.

(H-Influenzae, *definite*)
(female, *definite*)
(young, *definite*)
(Quinolones, *possible*)

Then, we can interpret this result as a new *kb* specialised for a particular patient. On the other hand, for the same example of specialisation we can see an example of communication. Suppose that the user queries the system for a certainty value for *Quinolones*.

Then, the system shows the propositions and rules related to the query *Quinolones*. That is,

(Quinolones, *possible*)
R1" (breast-feeding $\rightarrow$ stop-breast-feeding, *definite*)

In natural language the answer would be: *For the case of a H. Influenzae diagnosis for a young female, quinolones is possible, and if she is on breast-feeding period, she has to stop breast-feeding.*

5 Discussion

In this paper a new communication protocol for ES's is presented. It is based on an inference calculus containing an Specialisation Inference Rule in the paradigm of multiple-valued logics. This specialisation calculus is implemented using techniques of partial evaluation, and it is shown to be sound and complete for literals.

The communication so obtained is much more cooperative with users than the classical one: The answer to a query is a set of specialised rules and propositions.

This specialisation calculus can also be used to make validation of *kb*s. Consider that the expert has a general *kb* for pneumonia treatment, and that he wants to check the *kb* in a restricted context such as: women with gramnegative rods. The specialisation mechanism allows to obtain a new *kb* that is a *kb* for pneumonia treatment in the case of a *woman* with *gramnegative rods*. The expert should agree with the behaviour of the new *kb* so obtained because it is a specialisation of its original *kb*, otherwise he must revise it. To check the behaviour of this reduced *kb* he can apply any classical method, but to a much more reduced *kb*. This method can also be understood as a way of modularisation, by contexts, of flat and non-structured *kb*s. This methodology gives then a more comprehensive and systematic way of validating *kb*s than the standard methods.

6 Acknowledgements

The authors would like to thanks Gonzalo Escalada for many discussions on the literal completeness theorem proof. Thanks to Pere Garcia for helpful discussions on specialisation rule.

References

[1] J. Agusti, J. Esteva, P. Garcia, L. Godo, and C. Sierra. Combining multiple-valued logics in modular expert systems. In *Proceedings 7th Conference on Uncertainty in AI*, 1991.

[2] J. Gallagher. Transforming logic programming by specialising interpreters. In *Proceedings ECAI'86*, pages 109–122, 1986.

[3] H. J. Komorowski. *A specification of an abstract Prolog machine and its application to partial evaluation*. PhD thesis, Linkoping University, 1981.

[4] J. W. Lloyd and J. C. Shepherson. Partial evaluation in logic programming. *Logic Programming*, 11(3/4):217–242, October/November 1991.

[5] J. Pavelka. On fuzzy logic i. *Zeitschr. f. math. Logik und Grundingen d. Math.*, pages 45–52, 1979.

[6] J. Puyol, C. Sierra, and J. Agusti. Partial evaluation in MILORD II: A language for knowledge engineering. In *Proceedings Europ-IA'91*, pages 193–207, 1991.

[7] C. Sierra and J. Agusti. Colapses: Towards a methodology and a language for knowledge engineering. In *Proceedings AVIGNON'91*, pages 407–423, 1991.

[8] A. Takeuchi and K. Furukawa. Partial evaluation of prolog programs and its application to meta programming. In *Information Processing 86*, 1986.

[9] R. Venken. A prolog meta-interpreter for partial evaluation and its application to source transformation and query-optimisation. In *Proceedings ECAI'84*, pages 91–100, 1984.