An Inference Engine based on Specialisation with Uncertainty[*]

Josep Puyol-Gruart

Institut d'Investigació en Intel.ligència Artificial (IIIA)
Centre d'Estudis Avançats de Blanes
Camí de Santa Bàrbara
17300 Blanes (Girona)
SPAIN
E-mail: puyol@ceab.es

1. Introduction

Looking at an Expert System (ES) as a *blackbox*, the standard behaviour we can observe is the following one: The user asks the system for the value of a fact. If the system can deduce it, it gives the value to the user, otherwise the answer is *unknown*.

In the following we explain why this behaviour is rather poor and we propose a more informative input/output ES behaviour. Even if the system can not give a value to a fact, it has usually much more information related to the fact that could be useful to the user. In fact the standard behaviour of ES modelises only one aspect of the way human experts communicate. Two more cases are not usually taken into account:

a) In the case the system is able to answer the user's question, we could also be interested in knowing other explored but not successful deductive paths. It could also be useful to the user to know conclusions that are deducible from the reached goal.

b) A more evident case of poor communicative behaviour of an ES is when the system answers *unknown* to a question. Usually the system deduces *unknown* because the user has not given enough information to the system, i.e., he has not answered to all the questions the system has asked. Maybe because he did not know the relevance of some questions. It will be then much more informative if the system is able to deduce, not *unknown*, but with the set of facts the user should know to come up with a value for the question.

These are the reasons why we introduce an enriched communication scenario and the possibility of reconsidering information that has been previously answered with *unknown*.

On the other hand, the expert who develops an ES can not look at the ES under current development as a *blackbox*. The expert needs to validate it considering an opened perspective of the running ES. The standard behaviour offered to the experts consists of a trace of the execution, that is, which is the rule the system tries to fire, which is the value of a deduced

[*] This research has been supported by the CICYT TIC91-0430 project TESEU.

fact, and so on. This trace gives to the expert only an idea of the execution flow and the expert only can compare the system answer (a fact value) with his own one.

Given the problems presented above, in section 2 we propose an enriched behaviour of an ES. In section 3 we explain the inference mechanism that deals with this ideas, and in section 4 a brief example of communication.

2. Proposal

In this section we propose two extensions of an ES: the enriched communication scenario and the expert development tool.

2.1. Enriched Communication Scenario

Looking carefully at how experts communicate their knowledge and at their problem solving procedures we can find much more complex communication mechanism. Sometimes experts can not reduce their interaction only to the communication of certainty values of predicates. For instance, when communicating, experts in medical diagnosis also need:

1) To condition their answers.

Suppose that it is not known if a patient is allergic to penicillin. An ES deducing the possibility of giving penicillin can answer: *Penicillin is a good treatment from a clinical point of view if there is no allergy to it*.

2) To give conclusions that have to be considered with the answer.

The user would be interested in knowing other conclusions (*antibiogram*) that are deducible form the reached goal (*pneumococcus*). For instance the system can answer: *Culture of sputum pneumococcus has been isolated, then it is strongly suggested to make an antibiogram to the patient.*

3) To give conditioned conclusions to be considered with the answer.

Another example of complex communication is the combination of the above two examples: *Ciprofloxacine is a good treatment, but if the patient is a woman on breast-feeding period she must stop breast-feeding.*

4) To give a more general answer.

Imagine that Gram positive coccus are detected. An answer to the predicate pneumococcus is at that moment too precise and cannot be given, but at least the morphological classification can be answered, for example: *Coccus is definite*.

To model such communication protocols, we need to extend the ES answering procedure. What we need is to answer a given question with a set of formulas (rules and facts).

2.2. Expert Development Tool

Normally experts validate their KB's by testing cases. Given a case the expert can only analyze the relation between the case, the answer and the trace given by the ES. Another form of validating the KB is by specialising it. Suppose that the expert has a general KB for pneumonia treatment, he can validate it in a context like a woman 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 new KB because it is a specialisation of its original KB, otherwise he must revise it. This mechanism provides a more powerful method of validating a KB that the standard trace of the execution.

3. Milord II: A Language for Knowledge Engineering

We propose an inference engine based on the notion of specialisation of rule bases as a deductive mechanism. All the previous ideas about communication and validation are contained in this inference engine. It is not necessary to program special explanations for the user and the expert can specialise its KB for development purposes.

Specialisation is based on the notion of partial evaluation of Kleene's Theorem [Kleene,52]:

Given any computable function f of n variables $f=f(x_1,x_2,...,x_n)$, and k (k n) values $a_1,...,a_k$ for $x_1,...,x_k$ we can compute a new function f' such that,

$$f'(x_{k+1},...,x_n)=f(a_1,...,a_k,x_{k+1},...,x_n)$$

We say that f' is a *specialisation* of f. A partial evaluation algorithm is an implementation of this theorem.

Partial evaluation algorithms have been used in different areas as functional programming and logic programming [Venken,84] [Gallagher,86] mainly for efficiency purposes. We are interested in using this technique only for explanation capabilities (user communication and development tools), but efficiency is an advantageous lateral effect.

Milord II is a modular language for knowledge engineering that includes an inference engine based on a specialisation mechanism [Puyol et al.,91].

Milord II uses rules with uncertainty and the *specialisation inference rule* used is the following one:

Given a fact 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 ρ'= MP[1]*(α,ρ) is the new value of the rule.*

The *specialisation inference rule* is based on the well known logical equivalence property $(A \rightarrow B) \rightarrow C \equiv A \rightarrow (B \rightarrow C)$ which also applies in the Milord multiple-valued logical framework, based on residuated implications [Agustí,91]. Notice that in the particular case of B=Ø, we recover the usual modus ponens rule.

The specialisation of a KB consists on specialising rules and deducing facts. Given a set of rules whose conditions contain a fact with a known value, we can obtain a new set of

[1] MP (modus ponens) is a function that takes the truthvalue of the condition A, and the truthvalue of the rule ρ, and returns the truth value of the conclusion B.

specialised rules by applying the above inference rule. Rules that only has a condition will be eliminated and a new fact value will be deduced. This new fact will be used again to specialise the KB. The process will finish when the KB has no rule containing a known fact.

To answer a question, the rules considered are those in deductive paths to and from the question. The facts in the answer are those that have been obtained in the application of such rules. The rules in the answer are those which could not be applied because they used unknown knowledge. We only consider the rules in the deduction tree of the question because we assume that when a user makes a question it expects eventually all the relevant information associated with.

An important characteristic of the inference engine is that the search and the deductive processes are independent. In classical inference engines the deductive mechanism has a strategy that determines the ordering of obtaining the information from the user. In Milord II there are two processes, one decide the next question (search) to ask and the second one is the specialisation mechanism (deduction). Then we can use multiple strategies by changing the first process, and we are not limited to the classical backward and forward chaining.

4. Example

Consider the following KB for women pnemonia treatment:

R1- If Gram_negative_rods then Treatment_ciprofloxacine is possible
R2- If Treatment_ciprofloxacine is possible and breast-feeding then stop_breast-feeding is definite.

with the fact: Gram_negative_rods is definite

The specialised KB that results is the following one:

R1- Treatment_ciprofloxacine is possible
R2- If breast-feeding then stop_breast-feeding is definite.

Now if the question made is Treatment_ciprofloxacine, the answer will be: *Treatment with ciprofloxacine is possible and if you use this treatment if the woman is on breast-feeding period, she must stop breast-feeding*. With a classical ES the answer would be: *Treatment with ciprofloxacine is possible*.

5. Conclusions

In this extended abstract we have presented a short description of the specialization-based inference engine used in Milord II. This inference engine has independent search and deductive processes and allows to have an enriched communication protocol and a validation development tool.

6. References

Agustí, J., Esteva, F., Garcia, P., Godo, Ll., Sierra, C.: Combining Multiple-valued Logics in Modular Expert Systems. Proceeding 7th Conference on Uncertainty in AI. July 1991.

Gallagher, J: *Transforming logic programming by specialising interpreters.* Proceedings ECAI'86, pp. 109-122. 1986

Kleene, S.C.: *Introduction to Metamathematics.* Van Nostrand. 1952.

Puyol, J., Sierra, C. Agusti, J.: *Partial Evaluation in Milord II: A Language for Knowledge Engineering*. Proceedings Europ-IA'91. 1991.

Venken R.: *A Prolog meta-interpreter for partial evaluation and its application to source transformation and query-optimisation*. Proceedings ECAI'84, pp. 91-100. 1984