HybES: a Hybrid Expert System

I.V. Filis, C.P. Yialouris, M. Sabrakos and A.B. Sideridis

Informatics Laboratory, Agricultural University of Athens
75 Iera Odos, 11855 Athens, Greece
Email: {ioannis,yialouris,marios,as}@aua.gr

Abstract. This paper describes the architecture, representation and operation of a Hybrid Expert System (HybES) in the sense of incorporating methodologies of rule based systems and relational databases within fuzzy logic data sets and criteria. HybES is a Rule Based Expert System, deployed on a relational database with its rules based on the Object-Attribute-Value (O-A-V) triplet representation model. Every rule is valid for a specific time period and, consequently, the system is valid at certain time periods, using different valid-time versions of knowledge. The values of the O-A-V triplets are related to fuzzy sets, handling linguistic variables, therefore these are fuzzy values and each one depends on one or more criteria affecting it. The system infers using either the forward chaining (applying '*rules forward*' or the backward chaining (a bottom-up method, applying the rules in the opposite way). Using a simple technique, the working memory of the system is split into two parts (Conditions' and Rules' Working Memory), which are used according to system's needs.

1 Introduction

An expert system is an intelligent computer programme that accumulates information regarding a particular field and then uses that knowledge and an inference mechanism in order to solve difficult problems requiring special expertise in the given field [7]. A Rule-Based Expert System (RBES) is an ES that represents knowledge by means of rules, aiming to help a non expert user to solve real-world problems. The elements of a RBES are the Knowledge Base (KB) containing the expert's knowledge, the inference engine which decides the appropriate rules to be fired in order to suggest the best solution(s) and the user interface which is the contact between the ES and the end user.

Another part of the ES that is frequently referred as an additional internal component, is the working memory [6, 12], which temporarily keeps those data related to the case being analyzed while a permanent copy of these data are kept in the database. We do not consider the working memory as a separate component, since every single system has its own embedded working memory to manipulate data for the time being, but we use system's memory, separated according to our needs, as Conditions' and Rules' Working Memory, as stated bellow.

The KB of a RBES has as a fundamental component, the production rule, which in the simplest case has the form:

IF <assertionIf> THEN <assertionThen>.

Since a rule is a piece of knowledge, and Valid Time (VT) of a piece of knowledge is the time during which this knowledge is true in the real world [13], VT of a rule is the time interval during which, the rule we are referring to, is true or exists. According to the definitions above, the following rule 'R' is presently valid:

R: "IF a man is middle-aged THEN he has to work in order to live"

But what is the meaning of the word 'middle-aged'? In order to answer this question, we use the membership grade of each crisp value, which derives from a membership function in order to define the linguistic value 'middle-aged'. A membership function is a function that assigns to each possible individual in the universe of discourse, a value representing its grade of membership in the fuzzy set [19]. Larger values denote higher degrees of the membership set. A fuzzy set is a set with the capability to express gradual transition from membership to non-membership and vice versa, and it is defined by the membership function.

Therefore, we can use the membership function 'A_m' to define a middle-aged man as follows:

$$A_{\mathrm{m}}(\mathrm{x}) = \begin{cases} 0 & \textit{when either} \quad x \leq 20 \; \textit{or} \geq 60 \\ (x-20)/15 & \textit{when} \quad 20 < x < 35 \\ (60-x)/15 & \textit{when} \quad 45 < x < 60 \\ 1 & \textit{when} \quad 35 \leq x \leq 45 \end{cases} \qquad \textbf{(1)}$$

By resolving the membership function for x=30 we get that a 30 years old man is a member of the set of middle-aged men, having a membership grade $\mu(30)=0.66$, or in a more meaningful way, 'a 30 years old man belongs by 66% in the set of middle aged men'.

Up to now, according to the rule 'R', which is 'valid' nowadays, and according to the membership function 'A_m', we have concluded that a '30 years old man' is 'middle-aged' so 'he has to work in order to live'.

Nevertheless, is this rule correct for a man born and growing old in Africa? Is this rule correct for a man born two hundred years ago? Alternatively, is this rule correct for a man born two hundred years ago in Africa? All these are criteria (or a combination of criteria) that one has to take into account before reaching to a conclusion.

We must note that these criteria do not affect the assertion (assertion IF) of the rule 'R' directly, but the membership grade that affects the assertion of the rule. We must also note that there is a great difference between the time criterion we have used in the above question (Is this rule correct for the men born two hundred years ago?) and the Valid-Time definition we have introduced at the beginning of this section. The first is a criterion that affects the rule while the latter is the time period during which the rule is valid. If our knowledge about the 'middle-aged man' definition changes later, we shall define a new rule 'R1' with a new membership function and new criteria, keeping the old one as historical knowledge.

In order to represent the above example (rule 'R') with all the mentioned components (validity of time, fuzziness, multi criteria) we introduce an ES based on a Relational Data model, which is a combination of the structural part (in which the DB is a collection of relations), the integrity part (in which are described primary and foreign keys), and the manipulative part (in which relational algebra and relational calculus is used) [9].

Summarizing the above, this is an Expert Database System [17] with multicriteria fuzzy sets aiming to be used in a wide range of decision making problems where there is no a step by step process [14], according to the KB of the system.

In the next Section, we describe the architecture of a hybrid expert system (HybES), expounding its components and the way these are related each other in order to infer. In Section 3, we implement the methodology used, by describing HybES's integration on a relational database and the inference engine that draws the corresponding conclusions. Finally, in Section 4 we discuss the interconnection of the described components, under the specified architecture, to an integrated system.

2 System Architecture

An Expert System (ES) consists of the Knowledge Base (KB) containing the expert's knowledge, the inference engine and the User Interface (UI). In this chapter we will describe the discernible methodologies that constitute the components of a KB, as well as the characteristics and methodologies of inference engines. The UI, the only visible part by the user, which interacts with the KB and inference engine in order to produce the appropriate information / proposals / conclusions to the user.

2.1 Knowledge Base

A Rule is a conditional statement of two parts. The first part, comprised of one or more IF clauses, establishes the conditions of the rule. The second part, comprised of one or more THEN clauses, establishes the actions of the rule that have to be undertaken, and finally, the first part applies the second. The clauses of both parts, are represented in Object – Attribute – Value (O-A-V) triplets, since this representation easily fits into any RBES development tool [18].

The KB of all RBES, share the same most fundamental component, the production rule, which in the simplest case has the following form:

```
IF < assertion_i1 >                    THEN < assertion_t1 >
        AND / OR < assertion_i2 >              AND / OR < assertion_t2 >
        AND / OR ...                           AND / OR ...
        AND / OR < assertion_im >              AND / OR < assertion_tm >
```

In our system the assertions 'assertion_i1, assertion_i2, …, assertion_im' of the 'IF' part of the rule (the antecedents), are kept in the 'Condition' table and these are represented as O-A-V triplets. The Value (V) of each condition may be either a single value, or the conclusion of another existed rule.

The assertions 'assertion_t1, assertion_t2, …, assertion_tm' of the 'THEN' part of the rule (the consequents), are kept in the 'Conclusion' table and these may be either a single value, or a single condition, coded in an O-A-V triplet. We are able to use a single condition as a conclusion to support the forward chaining inference engine. If there is a need for a reaction as an entailment for a specific conclusion, we use the ‘Reaction’ table.

Since a rule consists of condition(s), conclusion(s) and possibly reaction(s), the three above tables are related to the ‘Rule’ table, with intermediate tables by ‘many to many’ relationships. The Validity of Time (VT) is performed at rule level, so using the appropriate fields, a rule is valid in a specific time interval. Another approach of the VT is to affect each one condition instead of the rule. This helps when a condition changes while all the others (both conditions and conclusions) remain unchanged and we don’t want to make a new rule for just one change. This approach is helpful in legal KBs [13], but the procedures of knowledge entry and maintenance are difficult since we must enter valid time interval for each one condition.

There are also fields in the related tables to represent the essential attributes of each rule. More specifically, at condition level, in order to express the uncertainty of the knowledge contained in the knowledge base [16] we have included the missing factor, the weight factor and the user’s certainty factor [15], as well as the order of every condition within a specific rule. At conclusion and reaction levels, we have included the confidence factor and the order within the specific rule, as before.

The described system up to now is a simple RBES on a relational DB. In more complex ES, the values of the condition may be vague. For example in the condition: ‘The tank has water level that is low’, the value ‘low’ can be represented either by a discrete fuzzy set or by a membership function defining vaguely the ‘low’ for the specific condition which is part of a specific rule.

A fuzzy relational data model, as an extension of classical relational model, incorporates the impreciseness in data values and their associations [1]. A Fuzzy Relational DataBase (FRDB) represents imprecise attribute values and close domain elements with possibility distributions and closeness relations respectively [3]. Buckles et al (1982) [2] proposed one of the earliest versions of Fuzzy Relational Database System (FRDBS) by merging the theory of fuzzy set and Relational Database System (RDBS). Chiang et al (1997) [4], following some works, defined an extended fuzzy relational model by the first-order logic.

In our system, we did not use the pre-described fuzzy relational data model, but the values of the O-A-V (condition) triplets are used as separate fuzzy sets related to one or more conditions. Therefore, in order to represent the vagueness of the values, we use a table to store values’ discrete fuzzy sets. From the two alternative ways used to represent a membership function (continuous and discrete), we have used the later one.

The next step we introduce is the way to handle fuzzy values under miscellaneous situations (criteria). At the last example what is the ‘low’ for a refinery’s oil tank, and for a glass of water? In order to represent the shifting of the fuzzy sets under various criteria, we use a table to store the criteria and another one to relate it with the discrete fuzzy sets.

2.2 Inference Engine

Levesque (1984) [10] proposed that the basic interface to any knowledge representation system consist of two kinds of interactions; one to *‘tell’* information to the system and one to *'ask'* whether information follows from what was previously told to the system.

The '*ask'* interaction is unbroken linked to ''inference engine'', which makes inferences by deciding which rules are satisfied by facts, prioritizes the satisfied rules, and executes the rule with the highest priority.

Inference is the process by which new facts are derived form known facts. For example, the rule R: "If the time varies between 22:00 to 05:00 next day, then the sky is black" combined with a rule of inference and a known fact F: "The time is 02:00" results to a new fact F': "The sky is black".

The inference engine must sometimes operate under indeterminate conditions or incomplete information. This requires information to be cross-referenced and needs the application of a number of rules that can complement one another.

Two formats exist for following rules in the process of searching for a solution.

(1) Forward (or data-driven) chaining—applying ''rules forward'': if the ''situation'' condition holds, do ''action''.

(2) Backward (or goal-driven) chaining—applying the rules in the opposite way: if you want ''action'' to happen, do ''situation''.

Application of the rules is not based on data existing in the KB, but on the desired result.

Using the relational data model to represent the ES, the inference is becoming faster than in the text based knowledge bases since the performance of inference engines degrades quickly as the size of the knowledge base increases [11].

A very useful and widely known feature of Rule-Based Expert Systems is their capability to keep a track of the reasoning process and provide the user the 'WHY' and 'HOW' the system came to the proposed conclusions. This facility is provided by making use of chains of rules to move logically from the user's input to the conclusions drawn from that input. Thus, by keeping these rules and their connections in dynamic working memory, the ES can provide the user with information on the logic underlying decisions.

3 System Integration

We present the development of HybES into three steps. Firstly, we introduce a simple Rule Based Expert System based on the relational data model. Next, we extend it using fuzziness in the conditions, and, finally, in order to handle fuzzy values under multiple criteria, we present a Multicriteria Fuzzy Rule Based Expert System.

Update of knowledge is performed through the user interface. Alternatively, we have also build an easy maintained ES (since such a job is critically important but very difficult job) [8].

The Entity Relationship model is presented in Fig. 1, and it is described in the following sections.

3.1 A Rule Based Expert System (RBES)

Section 1 of Fig.1 represents the Entity Relational model of the Rule Based Expert System (RBES).

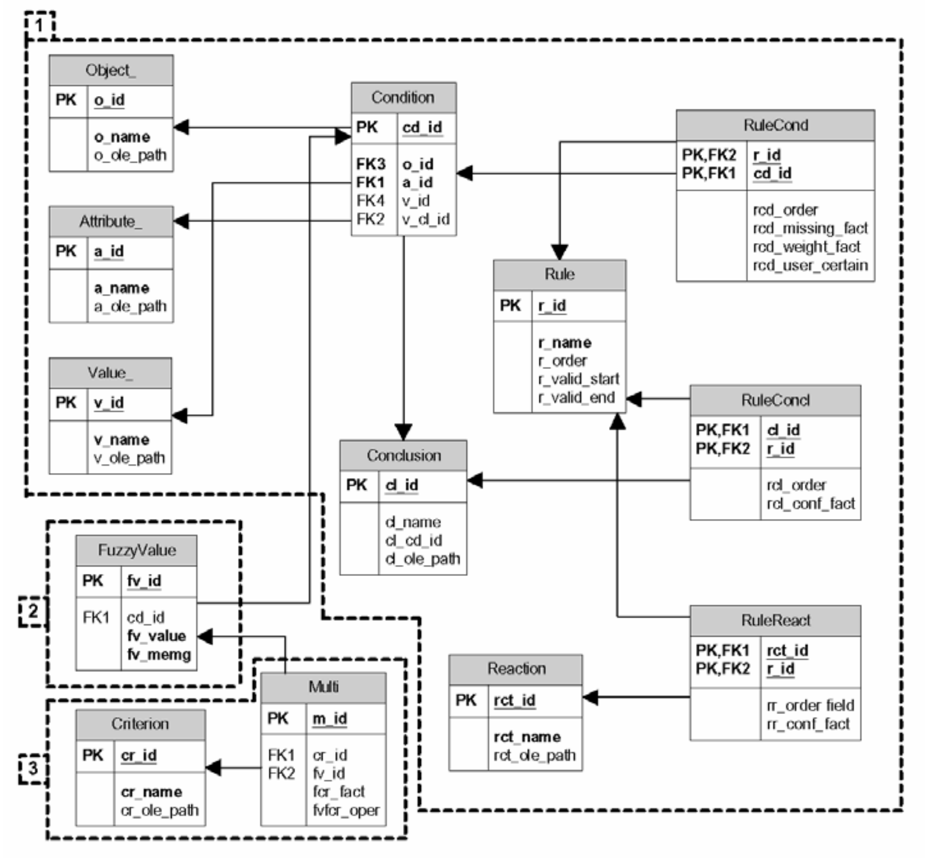

Fig. 1. Entity Relational model of the Hybrid Expert System

Rule is the basic element of the system. It is related to Conditions, in order to build the 'IF' part of the "IF P THEN Q" statement and to Conclusions to build the 'THEN' part of the "IF P THEN Q" statement. Thereinafter, every Rule may (or not) act in response, using a relation to Reactions, building the extension "AND_ACT A" to the classic "IF P THEN Q" statement.

The system is able to use a rule's conclusion as an input condition to another rule, giving the advantage to the Knowledge Engineer to use a conclusion in many rules. For example, in the condition "The hives have honey production that is less than normal honey production", the value derives from another rule, which defines the meaning of 'less than normal honey production'. Additionally, every condition within a specific rule, has its own missing factor, weight factor and user's certainty factor, to help the classification of the rule after inference.

We have also added a feature to the rule that determines system's temporality. It is a time interval during which, the rule is valid helping the user to compare the up-to-date valid knowledge to older conclusions from different time periods. It is very significant since a rule may be dropped during the time and a new rule may be born using similar conditions, reaching in the same or similar conclusion.

The hierarchical representation of the knowledge is represented by ordering every rule, helping the inference to apply the rules 'forward' (forward chaining) or in a bottom-up method (backward chaining).

In Fig. 1, are included only the basic fields in order the system to work. The full ER model includes all the ancillary fields that help the system to keep the changes, fields to specify every user's access, etc.

3.2 A Fuzzy Rule Based Expert System (FRBES)

Suppose we have the rule: "IF the honey production per hive per month is less than 5 kgrs AND the colony is near the pinewood THEN we have less than normal honey production" In this example the second condition, transformed in O-A-V triplet is: "The colony has location that is near the pinewood". It means that the object 'The colony' has the attribute 'location' that is valued 'near the pinewood'. Nevertheless, what is the meaning of the value 'near the pinewood'? One could say 5 km is near while others could consider 1 km is near. Both of them are correct since this distance is 'near' for bees. But the second opinion is 'more near' than the other.

In order to resolve this type of conflicts, we propose an extension to our RBES (Section 2 of Fig. 1) to handle fuzzy values, by linking the value of a condition to fuzzy sets.

3.3 A Multicriteria Fuzzy Rule Based Expert System (MFRBES)

In the fuzzy extension of the RBES we used the fuzzy sets to handle the linguistic variables of a condition. In this way we were able to conclude about the truth of the linguistic variable 'near the pinewood' according to the corresponding membership grade. Nevertheless, how can we handle the conditions when there are criteria that affect them? For example we are able to use a rule to define 'reduced honey production near the pinewood'. There are more criteria that one could take into account in order to make the rule more complete, as 'the times that the beekeeper transports his colony', 'the season of the year' or 'the existence of water near the colony'. These criteria can be included in the rule, so the rule will be more complete having more conditions. But what can we do for the criterion of 'attitude'? We used the example of 'the colony is 2 km distance from the pinewood', with the membership grade '0.7' but we don't know if this distance is horizontal or inclined. The membership grade should not be the same for an inclined bee flight.

In order to resolve this kind of conflicts, we propose an extension in our FRBES (Section 3 of Fig. 1) to handle fuzzy values under multiple criteria, by relating the criteria to the fuzzy sets.

We have also included a fuzzy criterion factor which interacts with the membership grades of the fuzzy sets. This means that the membership grade of a value increases and decreases alternately, according to the fuzzy criterion factor of the criterion that affects the fuzzy value.

3.4 Inference on HybES

Two methods of inference often are used, forward and backward chaining. Forward chaining is a top-down method which takes facts as they become available and attempts to draw conclusions (from satisfied conditions in rules) which lead to actions being executed. Backward chaining is the reverse. It is a bottom-up procedure which starts with goals (or actions) and queries the user about information which may satisfy the conditions contained in the rules. It is a verification process rather than an exploration process.

We have used both of these methods in our system, generating and utilizing the following two kinds of working memory.

a. The conditions' working memory in which there are stored all the initiated conditions, in order to be available to the system at any time, avoiding to be requested again from the user.
b. The rules' working memory in which there are stored all the rules that have been elaborated by the system. In this working memory, the stored rules are grated from 'satisfied' to 'unsatisfied' (according to a repletion factor, with values from 1 to 0), and when they are associated with the existing rules (master or ancillary), they drive the inference engine and help the reasoning process.

The major advantage by using a relational database for knowledge storing, is that we are able to use SQL statements to retrieve the knowledge, fast, simple and accurately. Examples of SQL statements, representing the process of gathering sets of rules from the KB, can be found in [5].

4 Conclusions

In this paper, we integrate various techniques in order to represent a hybrid Rule Based Expert System based on the Relational Data Model.

One critical point is the use of Valid Time knowledge, in order to meet specific needs of various applications in wide areas of soil sciences. We have also used Fuzzy conditions and conclusions in our rules, and we have related our Fuzzy conditions and conclusions to any criteria affecting the fuzzy values.

This methodology is suitable for most of the cases where humans and machines need to communicate using linguistic terms in order to make a decision. It depends on the Knowledge Engineer to acquire knowledge from the expert, and to format it in the proper format so to be stored in our Knowledge Base.

References

1. T.K. Bhattacharjee, A.K. Mazumdar, Axiomatisation of Fuzzy Multivalued Dependencies in a Fuzzy Relational Data Model, *Fuzzy Sets and Systems*, 96:343-352 (1998)
2. B.P. Buckles, Petry F.E., A Fuzzy Representation of Data for Relational Databases, *Fuzzy Sets and Systems,* 7:213-226, (1982)
3. G.Q. Chen, Vandenbulcke J., Kerre E.E., A Step Towards the Theory of Fuzzy Database Design, *Proceedings 4th World Congress of International Fuzzy Systems Association (IFSA),* Brussels, (1991) 44-47
4. D. Chiang, I.R. Chow, N. Hsien, Fuzzy Information in Extended Fuzzy Relational Databases, *Fuzzy Sets and Systems*, 92:1-20 (1997)
5. I. Filis, Rule-Based Valid-Time Expert Systems using Fuzzy Logic and Multicriteria Approaches. TR-161. Informatics Lab. Agricultural Univ. of Athens (2000)
6. Gamble R.F., Stiger P.R., Plant R.T., Rule-based Systems Formalized within a Software Architectural Style, *Knowledge-Based Systems* 12:13-22 (1999)
7. Lazarov A., Shoval P., A Rule-based System for Automatic Assignment of Technicians to Service Faults, *Decision Support Systems*, 32:343–360 (2002)
8. J.K. Lee, J.S. Hong, A Regenerative Expert System Approach for the Maintenance of Expert Systems, *Expert Systems With Applications*, 14:313-321 (1998)
9. M. Levene, G. Loizou, *A Guided Tour of Relational Databases and Beyond*, Springer, (1999)
10. H.J. Levesque. Foundations of a Functional Approach to Knowledge Representation. *Artificial Intelligence*, 23:155-212, (1984)
11. Levy A.Y., Fikes R.E., Sagiv Y., Speeding up Inferences Using Relevance Reasoning- a Formalism and Algorithms, *Artificial Intelligence*, 97:83-136 (1997)
12. Liao T.W., Zhan Z.H., Mount C.R., An Integrated Database and Expert System for Failure Mechanism Identification: Part II - the System and Performance Testing, *Engineering Failure Analysis*, 6:407-421 (1999)
13. N.A. Lorentzos., C.P. Yialouris, A.B. Sideridis, Time-evolving Rule-based Knowledge Bases, *Data & Knowledge Engineering*, 29:313-335 (1999)
14. Y. Siskos, A. Spyridakos, Intelligent Multicriteria Decision Support: Overview and Perspectives, *European Journal of Operational Research*, (1999)
15. G. Vouros, Representing, Adapting and Reasoning with Uncertain, Imprecise and Vague Information, *Expert Systems with Applications*, 19:167-192 (2000)
16. V.C. Yen, Rule Selections in Fuzzy Expert Systems, *Expert Systems with Applications*, (1999)
17. H. Yang, A Simple Coupler to Link Expert Systems with Database Systems, *Expert Systems with Applications,* 12(2):179-188 (1997)
18. C.P. Yialouris, H.C. Passam, A.B. Sideridis, C. Metin, VEGES – a Multilingual Expert System for the Diagnosis of Pests, Diseases and Nutritional Disorders of Six Greenhouse Vegetables, *Computers and Electronics in Agriculture*, 1997.
19. L.A. Zadeh, *Fuzzy sets, Information and Control*, (1965)