Complexity Reduction of Mamdani Fuzzy Systems through Multi-valued Logic Minimization

Marco Cococcioni[1], Luca Foschini[2], Beatrice Lazzerini[1], Francesco Marcelloni[1]

[1]Dipartimento di Ingegneria dell'Informazione: Elettronica, Informatica, Telecomunicazioni
University of Pisa, Via Diotisalvi, 2 – 56122 Pisa, Italy
{m.cococcioni, b.lazzerini, f.marcelloni}@iet.unipi.it

[2]Department of Computer Science
University of California
Santa Barbara, CA 93106, USA
foschini@cs.ucsb.edu

***Abstract*— In this paper, we propose an approach to complexity reduction of Mamdani-type Fuzzy Rule-Based Systems (FRBSs) based on removing logical redundancies. We first generate an FRBS from data by applying a simplified version of the well-known Wang and Mendel method. Then, we represent the FRBS as a multi-valued logic relation. Finally, we apply MVSIS, a tool for circuit minimization and simulation, to minimize the relation and consequently to reduce complexity of the associated FRBS. Unlike similar previous approaches proposed in the literature, the use of MVSIS let us deal with nondeterminism, that is, let us manage rules with the same antecedent but different consequents. To allow nondeterminism guarantees to achieve a higher (or at least not worse) complexity reduction than the one achievable from removing nondeterminism as soon as it appears. We apply our approach to six popular benchmarks. Results show a considerable complexity reduction associated only sporadically with consistent accuracy degradation. Moreover, quite surprisingly, the complexity reduction often comes together with an improvement in the classification accuracy.**

***Keywords*—fuzzy rule-based systems, multi-valued logic optimization, nondeterministic relations minimization, pattern classification.**

I. Introduction

In last decades, Fuzzy Rule-Based Systems (FRBSs) have been widely used in engineering applications (control, regression, classification, etc.) and several approaches to their automatic generation from numerical data have been proposed. The main reason of the success of FRBSs among researchers and practitioners resides in their characteristic of being interpretable, especially when Mamdani-type fuzzy rules are used. On the other hand, high values of accuracy of an FRBS are typically achieved despite high complexity, that is, large numbers of rules and of fuzzy sets in the partitions of input and output variables. Complexity diminishes interpretability of FRBSs and therefore a trade-off between accuracy and interpretability has to be found when developing FRBSs [1]. A common approach to FRBS generation is first to extract a large number of rules from data, for instance, by using the Wang and Mendel approach [2]; then, to refine the rule base by applying some technique of complexity reduction (see [1] for a taxonomy of such techniques).

In this paper we restrict our attention to techniques aimed at simplifying the rule base using logic minimization, i.e., all those approaches which tend to remove logical redundancies as much as possible. Such approaches try to simplify the boolean/multi-valued relation associated with an FRBS, treating it exactly like a boolean/multi-valued function to be minimized, as it happens in the synthesis of VLSI circuits (PLAs, finite-state-machines, etc.). Logic minimization of boolean and multi-valued functions has indeed been tackled by several researchers in the context of circuit design. Only recently, it has been realized that such studies, algorithms and tools can be fruitfully exploited in machine learning [3]. In [4] one of the first attempts to reduce the complexity of a rule base by removing logical redundancies is presented in a neuro-fuzzy system. Here, two or more rules are merged if the following three conditions are true: the rules have the same consequents, some conditions in the antecedents are common to all the rules, and the union of the other conditions in the antecedents coincides with the overall term set of some input linguistic variable. This approach has been applied and extended in [5] and [6].

In [7], complexity reduction of an FRBS is achieved by applying the ESPRESSO-II algorithm [8], which is one of the most widely used algorithms for heuristic binary logic minimization. Here, multi-valued inputs are encoded into binary inputs before applying ESPRESSO-II. Today's availability of multi-valued logic minimization algorithms like ESPRESSO-MV [9] makes binary encoding unnecessary. This allows achieving a number of advantages: to eliminate the time wasted by encoding process and the dependence of the minimization on the encoding, to reduce the search space, making the search process more efficient, and to avoid the unspecified values added by the encoding, thus speeding up the algorithm. On the other hand, Mamdani-type FRBSs can be naturally described through a multi-valued relation. An initial attempt to reduce complexity of an FRBS through multi-valued logic minimization algorithms has been proposed in [10]. Here, the application of the ESPRESSO-MV algorithm has allowed obtaining a complexity reduction ranging from 40% to 68% despite a small loss in accuracy. Nevertheless, this approach does not exploit nondeterminism, which is an interesting feature of multi-valued relations. Indeed, exploiting this feature can lead to a more concise representation of the FRBS. Further, algorithms have been proposed to exactly compute the minimum representation when nondeterminism is tolerated.

In this paper, to exploit nondeterminism we adopt MVSIS as multi-valued logic minimization tool [11]-[14]. MVSIS is a state-of-the-art tool which extends ESPRESSO-MV in essentially two directions which are of interest for complexity reduction of FRBSs. First, it allows the minimization of nondeterministic relations, in contrast to ESPRESSO-MV which can optimize only deterministic relations. Second, it is able to minimize, simulate and verify not only a single multi-valued relation but also a network of such relations. This is a very promising aspect, since the interest in systems composed of multiple and interconnected FRBSs is growing [15][16] and their complexity reduction is a challenging research field. We show the results of the application of our approach to six public available classification benchmarks. We obtained a complexity reduction ranging from 41.8% to 97.9% with a very slight degradation (sometimes even an increase) of classification accuracy. Actually, considerable accuracy degradations are sporadic and always associated with a significant complexity reduction.

II. Fuzzy Rule-Based Systems

Let $\mathbf{X} = \{X_1, ..., X_F\}$ be the set of input variables, X_{F+1} be the output variable and $C = \left\{C_0, ..., C_{T_{F+1}-1}\right\}$ be the set of T_{F+1} output labels. Let U_f, $f = 1, ..., F$, be the universe of the f-th variable X_f. Let $P_f = \left\{A_{f,0}, ..., A_{f,T_f-1}\right\}$ be a fuzzy partition with T_f fuzzy sets on the f-th variable. The m-th rule of a Mamdani FRBS can be expressed as:

R_m: IF X_1 *is* $A_{1,j_{m,1}}$ **and**...**and** X_F *is* $A_{F,j_{m,F}}$ THEN X_{F+1} *is* $C_{j_{m,F+1}}$

where $m = 1, ..., M$, $j_{m,f} \in \left\{0, ..., T_f - 1\right\}$, $f = 1, ..., F$, identifies the index of the fuzzy set which has been selected for X_f in rule R_m among the T_f fuzzy sets in partition P_f, and $C_{j_{m,F+1}}$, $j_{m,F+1} \in \left\{0, ..., T_{F+1} - 1\right\}$, is the class recognized by rule R_m. among the T_{F+1} possible classes.

A Mamdani-type FRBS can be completely described by an $M \times (F+1)$ matrix J of natural numbers, where the generic entries (m, f), $f = 1, ..., F$, and $(m, F+1)$, indicate that fuzzy set $A_{f,j_{m,f}}$ and label $C_{j_{m,F+1}}$ have been selected, respectively, for input variable X_f and output variable X_{F+1} in the m-th rule. Given an input pattern $\mathbf{x} = [x_1, ..., x_F]$, the output of the system is computed by firstly evaluating the degree $w_m(\mathbf{x}) = \prod_{f=1}^{F} A_{f,j_{m,f}}(x_f)$ of activation of each rule (we chose to implement the **and** logical operator as product), and then computing, for each C_i, $i = 0, ..., T_{F+1} - 1$, confidence $s_i = \sum_{l=1}^{L_i} w_l(\mathbf{x})$, where L_i are the rules which recognize C_i as output. Finally, $x_{F+1} = C_{\bar{i}}$, where $\bar{i} = \arg\max_i s_i$.

In this paper, we allow that a condition can be expressed as disjunction of fuzzy sets. For instance, a rule can contain a condition like X_f *is* $(A_{f,1}$ **or** $A_{f,2}$ **or** $A_{f,5})$. The **or** operator is implemented as maximum. Thus, $j_{m,f}$ can be either a single natural value taken from the set $\{0, ..., T_f - 1\}$ or a subset of $\{0, ..., T_f - 1\}$. In these cases, matrix J is not anymore a matrix of natural numbers but rather a two-dimensional cell able to store both single natural numbers and sets of them. This leads to a three-level logic instead of the original two-level one. In the following, we assume that a condition involving all fuzzy sets defined on a given variable can be considered as a *don't care* (*dc*) condition and therefore can be removed from the rule. For instance, in the case partition P_f consists of three fuzzy sets, condition X_f *is* $(A_{f,1}$ **or** $A_{f,2}$ **or** $A_{f,3})$ can be removed from the rule. We remark that in general this assumption is only an approximation. Indeed, disjunction of all fuzzy sets corresponds to the overall domain of the variable only for particular implementations of the **or** operator and specific types of membership functions (on the contrary, it is always true in multi-valued logic). Anyway, the assumption appears quite intuitive and natural.

The complexity of an FRBS can be expressed in terms of the sum of non *don't care* conditions in the antecedents of all rules defined in the FRBS. In the following, the generic condition X_f *is* $(A_{f,a}$ **or** $A_{f,b}$ **or** $A_{f,c})$ is expressed in a notation typically used in multi-valued logic as $X_f^{\{a,b,c\}}$. The symbols $X_f^{\{a,b,c\}}$ are called *literals*. When the variable X_f is binary, that is, partition P_f consists of only two fuzzy sets, its literals $X_f^{\{0\}}$ and $X_f^{\{1\}}$ will be also denoted as $\overline{X_f}$ and X_f, respectively, to resemble classical binary logic.

A. Automatic generation of rules from numerical data

Fuzzy rules can be generated from numerical data in several ways. Here, we follow the intuitive and very popular method proposed by Wang and Mendel (WM) in [2]. The WM method requires to define the partition of each variable at the beginning. Then, given a training set made of input-output pairs, a rule is generated for each pair as follows: for each input variable, the fuzzy set used in the condition of the rule is the one with the highest membership value for the specific input. The label in the consequent of the rule corresponds to the output. This approach to rule generation can produce conflicting rules, that is, rules which share the same antecedent but have different consequents. The original WM method introduces an algorithm to fix conflicts. In this paper, we do not use this algorithm, but we only remove sporadic conflicts as follows. For each pair of conflicting rules, we count the number of duplicates (we recall that we have a rule for each input-output pair). If the number of duplicates of one of the two conflicting rules is lower than a percentage (10% in the experiments) of the number of duplicates of the other rule, the first rule is removed. Otherwise, both the conflicting rules are kept. Indeed, when the amount of duplicates is strongly unbalanced between two conflicting rules, we assume that the rule with fewer duplicates has been actually generated by outliers. Finally, all the duplicates are removed so that the final rule base contains only distinct rules. If an FRBS contains at least a pair of conflicting rules, it is nondeterministic. The choice of preserving conflicting rules and therefore allowing nondeterminism can help find optimal solutions to the problem of complexity reduction. Indeed, in general, postponing the resolution of conflicts provides a higher level of freedom in the optimization process [17]. On the other hand, we do not keep all conflicting rules because the presence of outliers can degrade the performance of the FRBS.

III. MULTI-VALUED RELATION MINIMIZATION

In this Section, we motivate our approach through two examples dealing with two-valued and multi-valued logic, respectively. Let us consider a three-input, two-class classification problem. Further, let each input variable be partitioned by only two fuzzy sets. Let us assume that the application of the WM method generates the following rule base, represented in matrix notation on the left and by using a (partial) truth table on the right:

$$J = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

X_1	X_2	X_3	X_4
0	0	0	0
0	1	0	0
0	1	0	1
1	1	0	1
1	0	0	1
0	0	1	0
1	1	1	1
1	0	1	0

The Karnaugh map [18] associated with the truth table is (the symbol "–" denotes that the relation is *unspecified* for the corresponding combination of the inputs):

X_3 \ X_1X_2	00	01	11	10
0	0	0,1	1	1
1	0	–	1	0

We can observe that in correspondence to the input combination $(X_1, X_2, X_3) = (0,1,0)$, there exist two conflicting rules and consequently the output is not deterministic. In classical boolean function optimization, nondeterminism is not allowed and thus the conflict is resolved by arbitrarily fixing the output either to 0 or to 1. In the former choice, the associated Karnaugh map is given by:

X_3 \ X_1X_2	00	01	11	10
0	0	0	1	1
1	0	–	1	0

We observe that the Karnaugh map is equivalent to remove the rule $(0,1,0,1)$ from J, i.e., the third row. To generate the FRBS, we determine the smallest Sum-Of-Product representation (SOP) both for output 1 and for output 0. Actually, in classical logic minimization the smallest SOP is determined only for one of the two outputs, considering the other as default. In FRBS minimization, both the outputs have to be explicitly synthesized. In the previous map, we have highlighted the cubes (we recall that a cube is defined as a product of literals) corresponding to output 1 and output 0 with dashed and continuous lines, respectively. The resulting minimum SOP representation for output 1 is $X_4 = X_1\overline{X_3} + X_1X_2$ (or using the alternative and more general notation introduced in the previous section $X_4^1 = X_1^1X_3^0 + X_1^1X_2^1$). The minimum SOP representation for output 0 is $X_4 = \overline{X_1} + \overline{X_2}X_3$ (or, alternatively, $X_4^0 = X_1^0 + X_2^0X_3^1$). From the multi-valued Shannon expansion theorem [13], we can write the SOP representation for the overall relation as:

$$\begin{aligned} X_4 &= \sum_{i=0}^{T_{F+1}} C_i X_4^i = C_0X_4^0 + C_1X_4^1 \\ &= C_0(X_1^0 + X_2^0X_3^1) + C_1(X_1^1X_3^0 + X_1^1X_2^1) \end{aligned}$$

which corresponds to the set of rules:

R_1 : IF X_1 *is* $A_{1,0}$ **and** X_2 *is dc* **and** X_3 *is dc* THEN X_4 *is* C_0

R_2 : IF X_1 *is dc* **and** X_2 *is* $A_{2,0}$ **and** X_3 *is* $A_{3,1}$ THEN X_4 *is* C_0

R_3 : IF X_1 *is* $A_{1,1}$ **and** X_2 *is dc* **and** X_3 *is* $A_{3,0}$ THEN X_4 *is* C_1

R_4 : IF X_1 *is* $A_{1,1}$ **and** X_2 *is* $A_{2,1}$ **and** X_3 *is dc* THEN X_4 *is* C_1

or, equivalently, to the minimized matrix J :

$$J = \begin{bmatrix} 0 & dc & dc & 0 \\ dc & 0 & 1 & 0 \\ 1 & dc & 0 & 1 \\ 1 & 1 & dc & 1 \end{bmatrix}.$$

We note that the complexity of the FRBS after optimization is 7 (only 7 entries in the antecedent part are not *don't care*), while the complexity of the initial FRBS was 21. Further, since synthesis of Karnaugh maps tends to generate the minimum number of cubes and each cube corresponds to a rule, we also observe a reduction in the number of rules (from 7 in the original FRBS to 4 in the minimized FRBS). Let us consider now the second choice for removing nondeterminism, that is, to choose 1 as output for the two conflicting rules. This corresponds to eliminate the second row from the original matrix *J*. The new Karnaugh map to be minimized is now:

X_3 \ X_1X_2	00	01	11	10
0	0	1	1	1
1	0	–	1	0

which has the following minimum SOP representation:

$X_4^0 = X_1^0X_2^0 + X_2^0X_3^1$,

$X_4^1 = X_2^1 + X_1^1X_3^0$.

Also in this case, the complexity is 7 and the number of rules is 4. In general, however, different choices in the removal of nondeterminism will lead to different values of complexity.

An important theorem valid even in the multi-valued case states that resolving the nondeterminism before minimization always gives a minimum SOP representation which is never smaller than the minimum SOP representation associated with the initial nondeterministic relation [14]. Let us consider again the three-input, two-class example and allow the relation to be nondeterministic. This implies that we can determine the smallest SOP representation by choosing each time the most favorable output between 0 and 1 to resolve nondeterminism. In this way, we obtain the following optimal cubes:

$X_3 \backslash X_1X_2$	00	01	11	10
0	0	0,1	1	1
1	0	–	1	0

which correspond to the following minimum SOP representations:

$$X_4^0 = X_1^0 + X_2^0 X_3^1,$$

$$X_4^1 = X_2^1 + X_1^1 X_3^0 .$$

The complexity is now 6, although the number of cubes (and consequently of rules) is still 4. By analyzing the minimized map, we can observe how the unspecified value corresponding to the input combination $(X_1, X_2, X_3) = (0,1,1)$ has been considered as 0 and 1 when synthesizing, respectively, the 0 and 1 outputs. Thus, at the end of the minimization process, the minimum SOP representation is nondeterministic also for $(X_1, X_2, X_3) = (0,1,1)$, although initially it was unspecified. In general, minimizing separately the SOP representations for 0 and 1, it may happen that the final relation is nondeterministic, even if it was not originally. To be sure that the final relation is deterministic, we should therefore use a different strategy. A heuristic solution consists of first minimizing the SOP representation for 0. Then, when minimizing the SOP representation for 1, all unspecified input combinations already used for 0 are set to 0. However, doing so, the result will be dependent on the order according to which 0 and 1 are examined. Further, to obtain the smallest deterministic representation we should know the optimal value to assign to each unspecified and nondeterministic input combination. So far, no method to find the optimal solution to this problem has been proposed. On the contrary, this method exists when nondeterminism is allowed [14]. In addition, the smallest nondeterministic representation will never be more complex than the smallest deterministic one [14], since the former has a larger number of freedom degrees than the latter.

It can be now useful to clarify the difference between the symbol "−" (unspecified) and the co-occurrence of different possible output values (in the example 0 and 1). The symbol "−" denotes that the relation has not been determined for the corresponding inputs, due to the absence of training patterns in the corresponding region. Thus we can replace the value "−" with either 0 or 1, depending on which value is better for minimization. On the other hand, the absence of training data points within the region makes us quite confident that a pattern will never fall into that region, thus making the accuracy of the system independent of this choice. Obviously, this confidence increases as much as the training set is a reliable sample of the overall data set. On the other hand, the case of conflicting rules is different, since there is no reason to choose either 0 or 1 as output of the system, since both of them are implied by points in the training set. This is especially true in the worst case, i.e., when the same number of input/output pairs has voted for both 0 and 1. In such cases and in the absence of further information, we can therefore adopt any strategy to choose a deterministic value, or we can keep nondeterminism.

There is another important difference to be considered. In MVSIS, an unspecified input combination can: *i*) be specified as 0 to help the minimization of the SOP representation for 0, *ii*) be specified as 1 to help the minimization of the SOP representation for 1, *iii*) be specified both as 0 and as 1 to help the minimization of both, thus making the system nondeterministic, *iv*) remain unspecified. On the other hand, a nondeterministic input combination can be managed as the unspecified input combination except for item *iv*). Indeed, the nondeterministic input combination always has to be part of at least one cube.

After analyzing the two-valued case, let us now examine the multi-valued case. Let us consider a two inputs, three classes classification problem. Assume that each input variable is partitioned by using three fuzzy sets. It follows that $T_1 = T_2 = T_3 = 3$. Let us suppose that the application of the WM method generates the rule base represented by the following Karnaugh map (we have highlighted the cubes corresponding to outputs 2, 1 and 0 with dotted, dashed and continuous lines, respectively):

$X_2 \backslash X_1$	0	1	2
0	0	0	2
1	0	0,1	1
2	2	1	1

We note that, when nondeterminism is preserved, the minimum SOP representation is described by only 4 cubes, while at least 5 cubes are necessary when nondeterminism is resolved. The optimal multi-valued SOP (MV-SOP) representation is:

$$X_3^0 = X_1^{\{0,1\}} X_2^{\{0,1\}},$$

$$X_3^1 = X_1^{\{1,2\}} X_2^{\{1,2\}},$$

$$X_3^2 = X_1^{\{2\}} X_2^{\{0\}} + X_1^{\{0\}} X_2^{\{2\}}$$

which determines the following rule base:

R_1 : IF X_1 *is* ($A_{1,0}$ **or** $A_{1,1}$) **and** X_2 *is* ($A_{2,0}$ **or** $A_{2,1}$) THEN X_3 *is* C_0

R_2 : IF X_1 *is* ($A_{1,1}$ **or** $A_{1,2}$) **and** X_2 *is* ($A_{2,1}$ **or** $A_{2,2}$) THEN X_3 *is* C_1

R_3 : IF X_1 *is* $A_{1,2}$ **and** X_2 *is* $A_{2,0}$ THEN X_3 *is* C_2

R_4 : IF X_1 *is* $A_{1,0}$ **and** X_2 *is* $A_{2,2}$ THEN X_3 *is* C_2

The complexity is 8 and the number of rules is 4. We can note how this MV-SOP representation is nondeterministic. Indeed, $(X_1, X_2) = (1,1)$ activates both the first and the second rules, thus implying both 0 and 1 as output.

Now, we give some definitions typically used for multi-valued relations. A nondeterministic multi-valued relation H is a mapping: $H : D \to 2^{\{0,...,T_{F+1}-1\}} \cup "-"$, defined on the domain $D = \{0,...,T_1 - 1\} \times ... \times \{0,...,T_F - 1\}$, where $2^{\{0,...,T_{F+1}-1\}}$ is the set of all possible subsets of the set $\{0,...,T_{F+1} - 1\}$ and $"-"$ is the unspecified value. A *minterm* μ is any element belonging to the domain D. A minterm is unspecified if $H(\mu) = "-"$. The *i*-set is the set of minterms that can produce the value i. A multi-valued relation H_1 implies (is contained in) a relation H_2, also denoted as $H_1 \subseteq H_2$, if $H_1(\mu) \subseteq H_2(\mu)$, $\forall \mu \in D$ (in this context, an unspecified is considered equivalent to the set $\{0,...,T_{F+1} - 1\}$).

In multi-valued relation minimization the goal is to find the smallest relation which implies the original relation. Like for the two-valued logic, no algorithm exists to find the smallest deterministic MV-SOP representation. On the contrary, this algorithm exists when nondeterminism is allowed [14]. Thus, we approach complexity reduction of Mamdani-type FRBSs by allowing nondeterminism in the reduction process. To this aim, we adopt MVSIS, which exploits nondeterminism for multi-valued logic circuit synthesis. Actually, MVSIS does not implement the algorithm proposed in [14], but rather a method based on more efficient heuristics. In the experimental section we have used the method corresponding to the MVSIS command *mfs* with the following change. MVSIS avoids the synthesis of the most complex output set, deriving it from all the other *i*-sets. Since this cannot be accepted in our case, we have forced the explicit synthesis of each *i*-set by changing the source code. A detailed introduction to MVSIS is out of the scope of this paper. The interested reader can refer to the corresponding manual [11].

We would like to remark that at the end of the minimization process, the resulting FRBS may contain conflicting rules. In this case, a pattern which falls in the region defined by the antecedent of the conflicting rules is classified with the same confidence into all the output classes recognized by the conflicting rules. The FRBS, however, is able to classify the pattern also in this case. Indeed, since we adopt Gaussian fuzzy sets, all the rules in the FRBS are activated, thus producing different confidences s_i with different classes C_i. In the improbable case in which two or more classes are characterized by the same confidence values, the class is randomly chosen.

IV. EXPERIMENTAL RESULTS

We tested our approach on five classification benchmarks from UCI machine learning repository (ftp://ftp.ics.uci.edu) and one from the free access ELENA database (ftp://ftp.dice.uc1.ac.be). In particular, we used Breast Cancer Wisconsin (BCW), Credit Approval (CRX), Cleveland Heart diseases (HEART), Pima Indians Diabetes (PIMA) and WINE from UCI, and PHONEME from ELENA. Each dataset has been randomly split into two parts for training and test, respectively, with the same number of patterns and keeping the class distribution unchanged. The numbers of available patterns and input variables are, respectively, for each benchmark: BCW(683, 9), CRX(666, 6) (after removal of the 24 patterns having missing values), HEART(297, 13) (after removal of 6 patterns with missing values), PIMA(768, 8), WINE(178, 13) and PHONEME(5404, 5). All datasets are two-class classification problems, except WINE and HEART which are three-class and five-class problems, respectively. For each dataset, uniform fuzzy partitions with the same number of fuzzy sets T ($T_f = T$, $\forall f = 1,...,F$) have been generated for each input variable using Gaussian membership functions. A number of experiments have been performed on each dataset, changing the value of T from 2 to 10 with step = 1. The initial rule base has been generated as explained in subsection II.A.

We have used the number of literals and the number of classification errors on the test set as complexity and accuracy measures, respectively. We have denoted the number of literals (classification errors) of the original and simplified FRBSs as Co (Eo) and Cs (Es), respectively. To prevent the possible bias due to a random choice of patterns in training and test sets, we have repeated the experiments ten times.

To make the effects of the complexity reduction evident, for each data set we compute the Percentage of Complexity Reduction (PCR) and the Percentage of Error Increase (PEI), defined as $PCR = (Co - Cs)/Co$ and $PEI = (Es - Eo)/Eo$. Table I shows the minimum, mean and maximum values of *PCR* and *PEI* obtained in all the ten trials performed for each different value of T.

We observe that the value of *PCR* is considerable for all datasets, since it ranges from a minimum of 41.8% for PHONEME to a maximum of 97.9% for WINE. The mean values of *PCR* range between 62.3% (PIMA) and 90.5% (WINE). This means that, on average, the complexity is reduced at least of 2/3. Surprisingly, this significant complexity reduction does not happen to the detriment of the FRBS accuracy: it may happen that the simplified FRBS performs better than the original one, as highlighted by the numerous negative values in Table I. This phenomenon can be explained by observing that rules with a lower number of literals (i.e., rules with a higher number of *don't care* conditions) are more general and therefore can allow a higher level of generalization on the test set [19]. In other words, a system which uses a lot of local rules is more prone to overfitting than a system with fewer and more general rules. We also observe that, when the accuracy of the simplified version decreases, this decrease is not drastic.

For the sake of space, we present detailed results only for two out the six datasets, namely BCW and WINE. On the other hand, these are the datasets which behave quite differently from the others. Tables II and III show *Co*, *Cs*, *Eo* and *Es* in the form of mean value ± standard deviation, and *PCR* and *PEI* for each value of *T*. We note that when the number *T* of fuzzy sets increases, the original FRBSs tend to overtrain (*Eo* increases), whereas the reduced FRBSs maintain the number of errors quite constant. Further, we observe that the high value of *PEI* computed for WINE is generated for low values of *T*. The explanation of this behavior resides in the high number of conflicting rules generated by the low number of fuzzy sets used in the partitions. In particular, for $T = 2$, PEI is equal to 210.4%, which is undoubtedly very high. On the other hand, we have to consider that this percentage means that the errors pass from 8 to 24 on average, while the complexity passes from 1221 to 101 on average.

Finally, the execution time of the minimization approach proposed in the paper depends on both the number T of fuzzy sets used to partition the input linguistic variables and on the number of classes. We verified that for reasonable values of T (from 3 to 5), the time required by the minimization is on the order of minutes.

TABLE I. MINIMUM, MEAN AND MAXIMUM PCR AND PEI

	Min *PCR*	Mean *PCR*	Max *PCR*	Min *PEI*	Mean *PEI*	Max *PEI*
BCW	85.6%	89.4%	95.6%	-76.0%	-40.6%	0.5%
CRX	50.1%	77.0%	93.7%	-100.0%	3.2%	36.7%
HEART	62.4%	71.8%	94.1%	0.4%	28.4%	46.7%
PIMA	45.2%	62.3%	92.8%	-1.0%	7.6%	18.8%
WINE	86.6%	90.5%	97.9%	-63.7%	35.4%	210.4%
PHONEME	41.8%	69.9%	95.6%	-12.7%	-3.4%	10.7%

TABLE II. PERFORMANCE ON BCW DATASET.

T	*Co* ± *σ*	*Cs* ± *σ*	*Eo* ± *σ*	*Es* ± *σ*	*PCR*	*PEI*
2	939±6	41±4	22±2	22±10	96.0%	-2.0%
3	1418±8	114±5	19±2	19±12	92.0%	+1.0%
4	1583±8	164±4	22±2	18±13	90.0%	-15.0%
5	1730±7	175±5	28±2	22±13	90.0%	-22.0%
6	1822±8	216±7	40±2	24±13	88.0%	-39.0%
7	1837±9	227±6	67±3	22±14	88.0%	-67.0%
8	1888±7	232±7	92±2	26±14	88.0%	-71.0%
9	2406±7	284±8	108±2	29±16	88.0%	-74.0%
10	2401±7	345±9	118±2	28±15	86.0%	-76.0%

TABLE III. PERFORMANCE ON WINE DATASET.

T	*Co* ± *σ*	*Cs* ± *σ*	*Eo* ± *σ*	*Es* ± *σ*	*PCR*	*PEI*
2	998±7	21±3	31±2	32±8	98.00%	6.00%
3	1221±4	101±6	8±2	24±9	92.00%	210.00%
4	1246±0	94±4	7±1	21±9	92.00%	210.00%
5	1246±0	88±4	7±2	12±9	93.00%	67.00%
6	1246±0	122±5	10±1	11±9	90.00%	5.00%
7	1246±0	167±7	15±2	14±9	87.00%	-5.00%
8	1246±0	144±5	24±1	13±9	88.00%	-47.00%
9	1246±0	158±6	34±2	12±9	87.00%	-64.00%
10	1246±0	160±8	39±2	14±9	87.00%	-64.00%

V. CONCLUSIONS

In this paper, the issue of complexity reduction of FRBSs is addressed by an approach based on the use of multi-valued logic relation minimization theory. We represent each FRBS as multi-valued logic relation. Then, we apply MVSIS, a state-of-the-art multi-valued relation simplification tool, to reduce the complexity of the FRBS. The main feature of our approach is that nondeterminism is exploited as a resource to get a more compact representation of the rule base. We have tested our approach on six classification benchmarks. The percentage of complexity reduction ranges from 41.8% to 97.9%. On the other hand, the loss of accuracy is rarely considerable and often complexity reduction corresponds to an increase in accuracy. We consider the preliminary results shown in this paper very promising and think that the complexity reduction approach based on MVSIS can be very useful to minimize multiple, interconnected FRBSs.

ACKNOWLEDGMENT

The authors are grateful to Prof. Alan Mishchenko, University of Berkeley, for the help provided in better understanding the internal working principles of MVSIS.

REFERENCES

[1] A. Gegov, Complexity management in fuzzy systems, Springer, 2007.

[2] L.X. Wang, J.M. Mendel, "Generating fuzzy rules by learning from examples," IEEE Trans. Syst. Man & Cyb., vol. 22, no. 6, pp. 1414–1427, 1992.

[3] C.M. Files, M.A. Perkowski, "Multi-valued functional decomposition as a machine learning method," Proc. ISMVL'98, pp. 173–178, 1998.

[4] C. Lin, C. Lee, "Neural-network-based fuzzy logic control and decision system," IEEE Trans. Comput., vol. 40, no. 12, pp. 1320–1336, 1991.

[5] R. Rovatti, R. Guerrieri, G. Baccarani, "An enhanced two-level Boolean synthesis methodology for fuzzy rules minimization," IEEE Trans. Fuzzy Systems, vol. 3, no. 5, pp. 288–299, 1995.

[6] C.-T. Chao, C.-C. Teng, "Implementation of a fuzzy inference system using a normalized fuzzy neural network," Fuzzy Sets and Systems, vol. 75, pp. 17–31, 1995.

[7] A.F. Goby, W. Pedrycz, "Fuzzy modeling through logic minimization," Proc. of NAFIPS'05, pp. 494–499, 2005.

[8] R.K. Brayton, A. Sangiovanni-Vincentelli, C.T. McMullen, G.D. Hatchel, Logic minimization for VLSI synthesis, Boston, MA, Kluwer Academic Publishers, 1984.

[9] R.L. Rudell, A. Sangiovanni-Vincentelli, "Multiple-valued minimization for PLA optimization," IEEE T. on CAD, vol. CAD-6, no. 5, pp. 727–750, 1987.

[10] R. Rovatti, R. Guerrieri, T. Villa, "Fuzzy rules optimization for analog VLSI implementation," Proc. 4th IEEE Int. Con. Fuzzy Syst., pp. 1194-1204, 1995.

[11] MVSIS (Version 3.0), University of Berkeley, http://www-cad.eecs.berkeley.edu/Respep/Research/mvsis/

[12] M. Gao, J-H. Jiang, Y. Li, A. Mishchenko, S. Sinha, T. Villa, R. Brayton, "Optimization of multi-valued multi-level networks," Proc. of. ISMVL'02, pp. 168–177, 2002.

[13] R.K. Brayton, S.P. Khatri, "Multi-valued logic Synthesis," Proc. 12th International Conference on VLSI Design, pp. 196–205, 1999.

[14] A Mishchenko, R.K. Brayton, "A theory of nondeterministic networks," IEEE Trans. on CAD, vol. 25, no. 6, pp. 977–999, 2006.

[15] H.-P. Chen, T.-M. Parng, "A new approach of multi-stage fuzzy logic inference," Fuzzy Sets and Systems, vol. 78, pp.51–72, 1996.

[16] F. Chung, J. Duan, "On Multistage fuzzy neural network modelling", IEEE Trans. on Fuzzy Systems, vol. 8, no. 2, pp. 125–142, 2000.

[17] F. Marcelloni, M. Aksit, "Leaving inconsistency using fuzzy logic", Information and Software Technology, vol. 43, n. 12, 2001, pp.725-741.

[18] M. Karnaugh, "The map method for synthesis of combinatorial logic circuits," Trans. AIEE, vol. 72, part. I, pp. 593–598, 1953.

[19] H. Ishibuchi, T. Nakashima, T. Murata, "Performance evaluation of fuzzy classifier systems for multidimensional pattern classification problems," IEEE Trans. Syst. Man & Cyb., part-B, vol. 29, no.5, pp. 601–618, 1999.