Soft Comput (2009) 13:451–465
DOI 10.1007/s00500-008-0361-5

FOCUS

Learning consistent, complete and compact sets of fuzzy rules in conjunctive normal form for regression problems

Jorge Casillas · Pedro Martínez · Alicia D. Benítez

Published online: 2 September 2008

Abstract When a flexible fuzzy rule structure such as those with antecedent in conjunctive normal form is used, the interpretability of the obtained fuzzy model is significantly improved. However, some important problems appear related to the interaction among this set of rules. Indeed, it is relatively easy to get inconsistencies, lack of completeness, redundancies, etc. Generally, these properties are ignored or mildly faced. This paper, however, focuses on the design of a multiobjective genetic algorithm that properly considers all these properties thus ensuring an effective search space exploration and generation of highly legible and accurate fuzzy models.

Keywords Genetic fuzzy systems · Regression problems · Multiobjective optimization · Flexible fuzzy rules · Interpretability constrains

1 Introduction

In knowledge discovery in databases we can distinguish between two different approaches (Lavrač et al. 2004): *predictive* induction and *descriptive* induction. The difference lies in the main objective pursued and, therefore, the learning method used to attain that. On the one hand, predictive induction looks for generating legible models that describe with the highest reliability the data set that represents the analyzed system. In that case, the goal is to use the obtained model to simulate the system, thus getting an explanation of its complex behavior. On the other hand, descriptive induction looks for particular (interesting) patterns of the data set. In that case, we do not get a global view of the relationships among variables but we discover a set of rules (different enough among them) statistically significant.

This paper focuses on the former approach, the predictive induction, to deal with regression problems where both input and output are real-valued and where the knowledge obtained is important to understand better the analyzed system. To represent the knowledge, and with the aim of generating legible enough models (which, no doubt, is one of the fundamental premises in any knowledge extraction process), we propose the use of fuzzy rule-based systems. These systems use IF–THEN fuzzy rules and linguistic variables to express the knowledge about the problem.

The automatic extraction of fuzzy rule-based systems can be done with different learning methods, either greedy algorithms (Nozaki et al. 1997; Wang and Mendel 1992) or other methods such as neural networks (Fullér 2000; Nauck et al. 1997) and genetic algorithms (GAs) (Cordón et al. 2001). Due to the aim of this paper on generating knowledge with good interpretability, we propose the use of GAs because it holds a sort of useful features for our purpose. Firstly, they have a powerful search capacity that allows us to work with multiobjective optimization. Secondly, they can manage flexible representation structures mixing coding schemes or including restrictions. From the beginning of the 90s many researchers have drawn their attention to the use of GAs to automatically design different components of a fuzzy rule-based system (Karr 1991; Thrift 1991; Valenzuela-Rendón 1991). These learning systems are usually known as genetic fuzzy systems (Cordón et al. 2001).

Regardless the learning tool used, a crucial problem emerges: to obtain both an accurate and an understandable model. Indeed, fuzzy modeling (i.e., the process of deriving

J. Casillas (✉) · P. Martínez · A. D. Benítez
Department of Computer Science and Artificial Intelligence, University of Granada, 18071 Granada, Spain
e-mail: casillas@decsai.ugr.es

fuzzy systems) usually comes with two contradictory requirements to the obtained model: the *interpretability*, capability to express the behavior of the real system in a comprehensible way, and the *accuracy*, capability to faithfully represent the real system. Of course, the ideal thing would be to satisfy both criteria to a high degree but, since they are contradictory issues, it is generally not possible. The quest of a good trade-off between interpretability and accuracy is target of numerous research works nowadays (Casillas et al. 2003a, b).

To reach this balance, we propose in this paper the use of fuzzy rules with antecedent in conjunctive normal form (i.e, where the antecedent is the conjunction of a set of propositions, each of them associating an input variable with a set of linguistic terms connected by disjunction), usually known as DNF-type fuzzy rules. This representation provides a high degree of compactness and knowledge synthesis. Since we are interested in predictive induction, the Pittsburgh-style GA (where each individual encodes a complete set of rules) seems to be the best approach to properly assess the interaction among the different fuzzy rules to perform interpolative reasoning.

However, the combination of DNF-type fuzzy rules and Pittsburgh-style GA are far from being easy since several difficulties arise:

- *Consistency*: each combination of antecedents (one label per input variable) should have only one possible consequent label.
- *Completeness*: every training data example should fire at least one fuzzy rule.
- *Compactness*: the lowest number of rules to accurately represent input-output relationships should be obtained. Among other issues, it involves avoiding redundant rules.
- *Non-overgeneral rules*: a DNF-type fuzzy rule should be general enough as to represent in a compact way the input-output relationship but specific enough as to avoid covering input areas without data.

Although it is relatively easy to comply with these conditions when using simple (Mamdani-style) fuzzy rules [see for example (Jin et al. 1999), where measures of incompleteness and inconsistency are used as penalty in the rule's fitness], it becomes more complex in the case of DNF-type fuzzy rules. Most of the methods that deal with some kind of generalization of the antecedent of the fuzzy rule (e.g., DNF-type rules or rules with "do not care") do not address properly the problem (Casillas and Martínez-López 2008; Castro et al. 1999; González and Pérez 1998, 1999; Ishibuchi et al. 2006; Ishibuchi and Nojima 2007; Magdalena 1997; Otero and Sánchez 2006; Sánchez et al. 2001; Xiong and Litz 2000). Indeed, some of these proposals use a penalty fitness to correct these deficiencies, others infer a default output when no rules are fired, others tend to generate a high number of rules, some other simply do not prevent the system from generating inconsistencies or redundancies...

There are few proposals that explicitly try to hold one or more of the consistency, completeness and compactness properties with a fuzzy rule structure with generalization capability of the antecedent. For example, Wang et al. (2005) use the same functions defined in Jin et al. (1999) to detect conflicts with an agent-based evolutionary approach in which the agents were multiobjective Pittsburgh-style genetic fuzzy systems. However, they use simple crossover and mutation operators and a posteriori reparation to solve inconsistencies and redundancies. Other solution is proposed in Wang et al. (1998), where redundancy by subsumption is removed by a specific a posteriori operator. However, consistency of the rule set is not ensured.

We take a completely different approach from the above methods and propose an algorithm that intrinsically explores feasible solutions (according to the mentioned consistency, completeness, non-redundancy, and non-overgenerality restrictions), thus avoiding the use of penalties, reparations, or additional function objectives. It considers a multiobjective optimization process which generates a large range of solutions with different interpretability-accuracy balances under the mentioned restrictions.

The paper is organized as follows: Section 2 briefly presents the difficulties that appear when using DNF-type fuzzy rules. Section 3 describes the proposed algorithm, called Pitts-DNF. Section 4 shows the results obtained in a set of real-world problems compared with other fuzzy rule learning methods. Finally, Sect. 5 concludes and suggests some further works.

2 Some properties to be considered when learning DNF-type fuzzy rules

In order to obtain a high degree of knowledge synthesis, thus improving the interpretability, we opted by a compact description based on DNF-type fuzzy rules where the antecedent is described in conjunctive normal form. This kind of fuzzy rule structure is defined as follows:

$$\text{IF } X_1 \text{ is } \widetilde{A_1} \text{ and } \ldots \text{ and } X_n \text{ is } \widetilde{A_n} \text{ THEN } Y \text{ is } B,$$

where each input variable X_i takes as a value a set of linguistic terms $\widetilde{A_i} = \{A_{i1}\ or \ldots or\ A_{il_i}\}$, whose members are joined by a disjunctive (T-conorm) operator, whilst the output variable remains an usual linguistic variable with a single label associated. The structure is a natural support to allow the absence of some input variables in each rule

(simply making $\widetilde{A}_i$ be the whole set of linguistic terms available).

When a whole set of such a kind of rules is simultaneously learnt (as recommended in predictive induction tasks), collisions easily appear. Basically, these collisions are of two types: *inconsistency* and *redundancy*. Furthermore, to have a fuzzy rule structure that allows a variable generality degree of the antecedent may lead the learning algorithm to two undesirable situations: *over-generality* and *incompleteness*. These four cases are discussed in the following sections.

2.1 Consistency

The first kind of collision is the inconsistency. Two rules are inconsistent between them when their antecedents overlap themselves, i.e., their antecedents are the same, they coincide in some labels for each input variable, or one is subsumed by the other (i.e., an antecedent is completely contained in a larger and more comprehensive antecedent) but the consequent is different. For instance, the two following rules are inconsistent:

Example 1

R_1 : IF X_1 is A_1 and X_2 is B_1 THEN Y is C_1

R_2 : IF X_1 is A_1 and X_2 is B_1 THEN Y is C_2

and the same in this second example where the antecedents are partially overlapped:

Example 2

R_1 : IF X_1 is $\{A_1$ or $A_2\}$ and X_2 is B_1 THEN Y is C_1

R_2 : IF X_1 is A_1 and X_2 is $\{B_1$ or $B_2\}$ THEN Y is C_2

or in this third case where the antecedent of the former rule subsumes the latter:

Example 3

R_1 : IF X_1 is A_1 and X_2 is $\{B_1$ or $B_2\}$ THEN Y is C_1

R_2 : IF X_1 is A_1 and X_2 is B_1 THEN Y is C_2

All these cases of inconsistency cause a linguistic contradiction that should be avoided for the sake of a better interpretability.

To solve these inconsistencies sometimes involves making more specific a general rule (as R_1 or R_2 in the example 2, or R_1 in the example 3) or removing the more specific rule (as R_1 or R_2 in the example 1, or R_2 in the example 3). Therefore, in these cases, to solve the inconsistency also helps to reduce the complexity of the rule set.

In other situations, however, to solve the inconsistency may imply the necessity of a higher number of rules as shown in Fig. 1. Therefore, the interpretability improvement obtained when solving the inconsistency may involve a more complex rule set. This fact could be solved by considering a firing-level-based hierarchy of the rule set (being the more specific rules in an upper position) (Yager 1993), discounting the strength of the more general rules when they are inconsistent with more specific ones (Ishibuchi et al. 2006), or even by extending the knowledge representation to consider rules with exceptions (Carmona et al. 2004). These issues are out of the scope of this paper.

2.2 Redundancy

A second, less serious collision is when the antecedent is overlapped as in any of the above examples but the

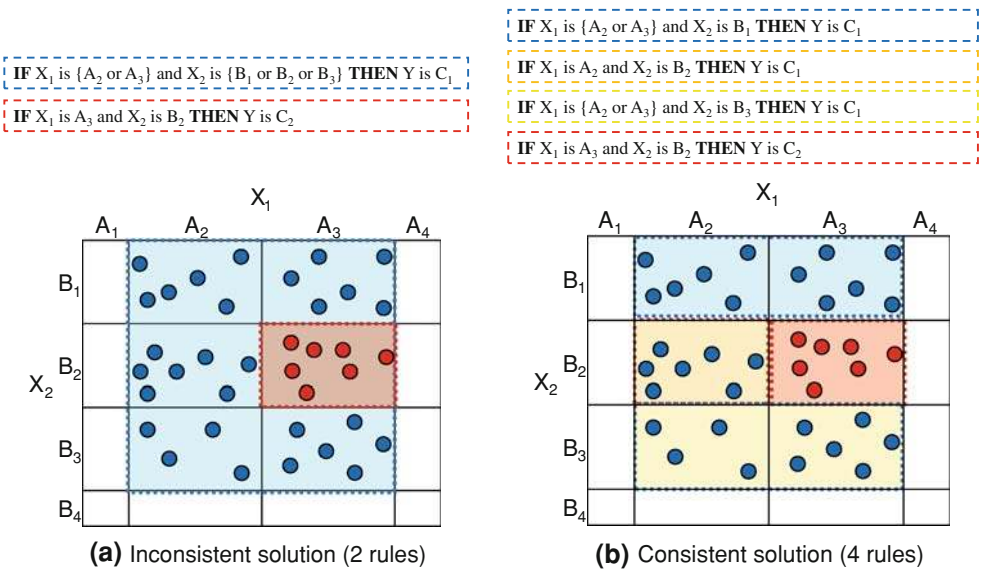

Fig. 1 Example of **a** an inconsistent solution and **b** a consistent solution. Notice that in this case more rules are needed to hold consistency

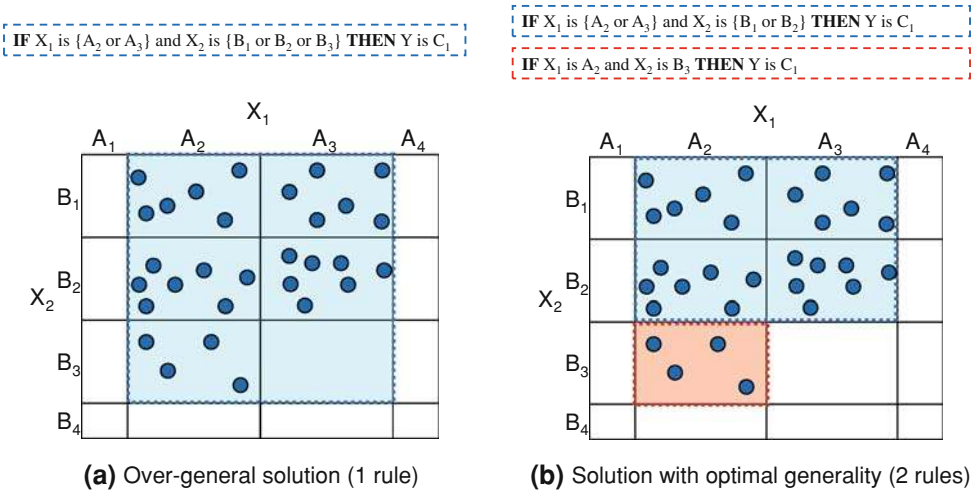

Fig. 2 Example of a solution with **a** over-generality and **b** optimal generality according to training data. Notice that in this case more rules are needed to hold optimal generality

consequent is the same. In that case, we have a redundancy. For instance:

Example 4

R_1 : IF X_1 is A_1 and X_2 is $\{A_2$ or $B_2\}$ THEN Y is C_1

R_2 : IF X_1 is A_1 and X_2 is A_2 THEN Y is C_1

Redundancy increases the fuzzy rule set size unnecessarily and can even provoke some undesirable interaction effects with some inference engines. When both rules have the same antecedent or one subsumes the other, the fuzzy rule set can be easily fixed by removing the repeated or the most specific rule (R_2 in the Example 4), respectively.

2.3 Over-generality

The use of a flexible structure to represent the antecedent of the fuzzy rule can also lead the algorithm to generate over-general rules. This fact appears when a DNF-type fuzzy rule includes a higher number of linguistic terms per variable than the ones strictly necessary according to the training data as illustrated in Fig. 2. It makes the fuzzy rule cover input regions where there is no available information, which is not desirable at all since the quality of the rule in those regions can not be tested.

If the learning algorithm does not care about that, the antecedent structure of some generated fuzzy rules is actually designed in a random way, since any consequent used in the empty regions would return exactly the same accuracy degree. This fact is even more serious if the learning algorithm is oriented by an objective function designed to increase the generality degree of the fuzzy rules.

To ensure optimal generality, however, may provoke some undesirable effects. The first one is that sometimes a higher number of rules is needed (as shown in Fig. 2). Another drawback is that generating a fuzzy rule set that only covers the training input regions may worsen the prediction generality (i.e., the capability to accurately predict the output in unforeseen input data), typically measured by the test error, if there is data on these areas.

Therefore, the question whether keeping optimal generality is recommendable or not is a controversial issue. We believe, nonetheless, that from the knowledge discovery field point of view it is preferable to provide the expert with a set of rules that properly represent the analyzed data instead of doing a pseudo-random generalization of the rule antecedents.

In the last resort, and always considering the expert interests, a good solution would be to provide a fuzzy rule set strictly generated from the data and another one to cover the input gaps. This latter fuzzy rule set with total completeness could be generated by doing linguistic extrapolation (e.g., Wang 2003). We leave this last approach for a further work.

A simpler solution, but extensively done in the literature, is to return a conservative output (e.g., in regression problems it is usual to give the mean value of the output domain) when no rules are fired due to the fact that the data lies in an uncovered input region. If it is done during the learning process, an incomplete rule set may be generated (as discussed in the next section) and, therefore, it is not recommended at all. However, it can be considered in the inference engine once the learning has finished and the final fuzzy system is used to predict the output for new input data. We will follow this approach for reporting test results in this paper.

2.4 Completeness

The last interesting property is completeness. This means that no input regions where there is data should be uncovered, i.e., with no rules being triggered. In

classification tasks this fact is not usual since an uncovered training example is considered as misclassified, so these kinds of solutions are penalized. However, in regression problems where an output must be always returned, if the inference is designed to give a default answer (e.g., the mean value as said in the previous section), incomplete fuzzy rule sets may be generated.

Moreover, if the learning system seeks for an optimal generality (see the previous section) and/or with a reduced number of rules, the risk of obtaining incomplete rule sets is higher. We will endow our proposed algorithm with an effective way of ensuring completeness.

3 Pitts-DNF algorithm

The learning algorithm we propose in this paper, called Pitts-DNF, has been designed to avoid generating DNF-type fuzzy rule sets with inconsistencies, redundancies, over-generality, or training incompleteness. Its scheme is shown in Algorithm 1.

Algorithm 1: Pitts-DNF algorithm

Parameters: Population size, crossover probability, antecedent mutation probability, and consequent mutation probability

Input: Data set: $D = \{(x, y) \mid x \in \mathbb{R}^n,\ y \in \mathbb{R}^m\}$. Membership function definitions

Output: Set of non-dominated solutions, each one with a different number of rules/accuracy trade-off. Each solution is a consistent, non redundant, non over-general, and complete DNF-type fuzzy rule set

```
begin
  Initialization(P)
  CH ⟵ Covering_Hypermatrix(D)
  Evaluation(P, D)
  While not stop condition do
    P1 ⟵ Multiobjective_Selection(P)
    P2 ⟵ Crossover(P1)
    P3 ⟵ Antecedent_Mutation(P2, CH)
    P4 ⟵ Consequent_Mutation(P3)
    P5 ⟵ Completeness_Operator(P4, D)
    Evaluation(P5, D)
    P ⟵ Multiobjective_Replacement(P5, P)
  end
end
```

3.1 Coding scheme

Each individual of the population represents a set of fuzzy rules (i.e., Pittsburgh style). Therefore, each chromosome consists of the concatenation of a number of rules. The number of rules is not fixed a priori so, the chromosome size is variable-length. Each rule (part of the chromosome) is encoded by a binary string for the antecedent part and an integer coding scheme for the consequent part. A binary coding could also be used for the consequent part without influencing on the algorithm behavior, but since we use a fuzzy rule structure where only one label is associated to each output variable, integer coding seems to be more appropriate.

The antecedent part has a size equal to the sum of the number of linguistic terms used in each input variable. The allele '1' means that the corresponding linguistic term is used in the corresponding variable. The consequent part has a size equal to the number of output variables. In that part, each gene contains the index of the linguistic term used for the corresponding output variable.

For example, assuming we have three linguistic terms (S [small], M [medium], and L [large]) for each input/output variable, the fuzzy rule

[IF X_1 is S and X_2 is {M or L} THEN Y is M]

is encoded as

[100|011||2].

A chromosome will be the concatenation of a variable number of these fuzzy rules, e.g.,

[100|011||2 010|111||1 001|101||3]

for a set of three rules. Notice that we do not fix a maximum number of rules per chromosome. Since our algorithm removes any unnecessary redundancy and unfired fuzzy rules, the number of rules is restrained all the time in a natural way.

It is allowed a variable with all the labels set to '1' (which means the variable is not considered in the corresponding rule), but it is forbidden a variable with all the labels set to '0'. It is so because, although one could think of assigning this latter combination to the fact of not using the variable (as in the case of all the labels set to '1'), then we would have solutions genotypically closer but phenotypically far, which is not advisable.

3.2 Initialization

Since we are looking for optimal completeness, we need to start with rules which cover all the examples. Because of that, we use the well-know Wang–Mendel algorithm (Wang and Mendel 1992), briefly described in Appendix, to generate the antecedent structure.

Specifically, every chromosome is generated with the minimum number of rules that cover the examples according to this algorithm and with a random consequent for every rule (except one chromosome that uses the consequents provided by Wang–Mendel). In this way, all chromosomes start with the same number of rules, being as specific as possible (i.e., with Mamdani-type structure instead the DNF one).

3.3 Covering hypermatrix computation

The objective of this step is to generate a data structure which will be used when generating new rules to efficiently avoid over-generality or generating rules in regions without training data. This structure, that we have called *covering hypermatrix*, stores the label combinations of the antecedent that cover all the examples in the training data set. Notice that the hypermatrix represents the highest allowed input covering, but it does not show whether a lower number of rules would completely cover the training data set or not, so it can not be used to ensure completeness.

The structure of this hypermatrix is an array, which dimension is equal to the number of input variables, containing '1' in a cell if the corresponding input combination covers at least a training example and containing '0' in other case. With this structure it is possible to design an efficient mutation operator to avoid over-general rules.

The implementation of this structure must be specially efficient, because of its high requirements of access time to the information. In this work we decided implement the hypermatrix using a hash table, which keys are built with the label concatenation of the contained rules. In order to optimize the table in space and information retrieve time, the combinations '0' are not stored. We consider that if a particular key does not exist then its value is '0'.

3.4 Crossover operator

The crossover operator is applied with a given probability rate to each randomly mated pair of parents. It only interchanges rules between the two parents, but it does not modify them. Furthermore, it guarantees the children does not present neither inconsistencies nor redundancies. The pseudo-code is included in Fig. 3 and an example of its application is shown in Fig. 4.

Function: Crossover Operator
Input: Two individuals (parents)
Output: Two new individuals (children)
Preconditions: The received individuals have not internal inconsistencies
Postconditions: The generated individuals do not have either inconsistencies or redundancies by subsumption, but redundancies by partial overlapping are possible. Lack of completeness can also appear

1. Put all the rules of the two parents into a set, S.
2. Analyze those inconsistent rules among them (which always will come from two parents, due to the preconditions). These rules do not have to be inconsistent in pairs. For instance, a rule of the first parent can be inconsistent with two rules of the second one.
3. Divide every group of inconsistent rules into two subsets depending on the parent of which they come from and take these rules out of S. Assign each subset to a different child.
4. Take an uniform random number $r \sim U[1, |S|]$, which will give the number of rules that will be assigned to the first child, being the rest $(|S| - r)$ the number of rules assigned to the second child.
5. Choose at random r rules from S and assign them to the first child.
6. Assign the rest to the second child.
7. This process can generate redundancies in the children. To avoid it, for every child, check if the antecedent of every rule is subsumed by another one and, if so, delete the more specific rules.

Fig. 3 Crossover operator

3.5 Antecedent mutation operator

This operator together with the consequent mutation are the ones that create new rules. It is applied with a given probability rate to each individual. As its name suggests, it acts on input variables. When a gene in the antecedent part of a fuzzy rule is chosen to be mutated, the operator analyzes among the available movements (it will be explained below) those that ensure to keep consistency and non-overgenerality (this later case is quickly checked with the covering hypermatrix). The consistency is checked by analyzing the collision of the candidate mutate rule with the rest of them. An option among the permitted ones is randomly chosen. Therefore, the antecedent mutation operator only explores feasible solutions, thus constraining the search space and ensuring a better exploration.

Figure 5 shows the pseudo-code of the operator. The two different actions are explained in the following.

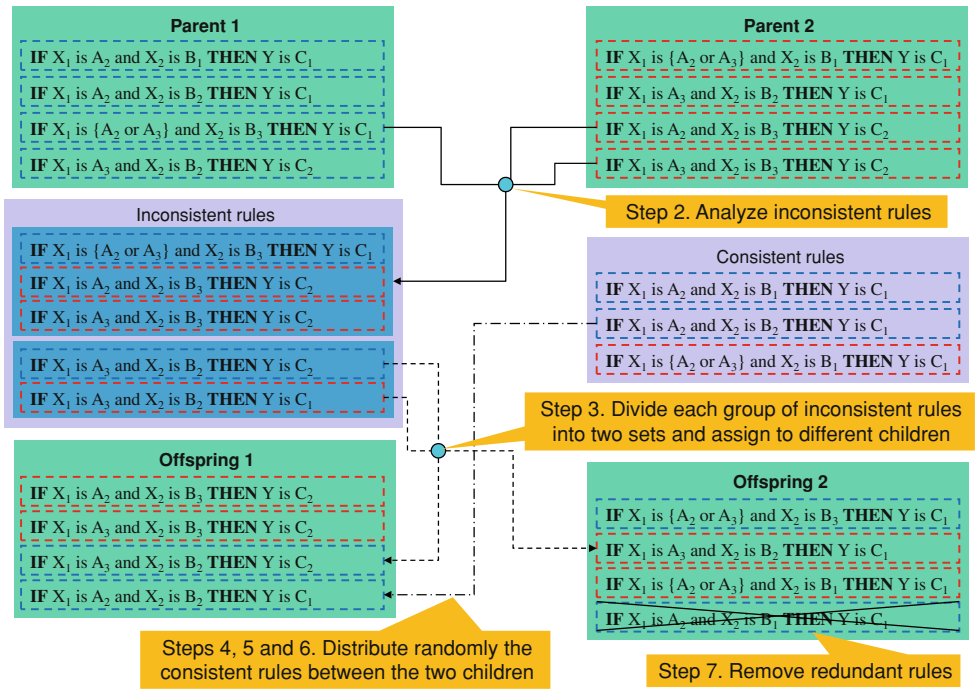

Fig. 4 Example of crossover operator application

Function: Antecedent Mutation Operator
Input: One individual and the covering hypermatrix
Output: The input individual mutated in its antecedent
Preconditions: The received individual has no internal inconsistencies
Postconditions: The generated individual has not neither inconsistencies nor subsumed rules, but redundancies by partial overlapping are possible. Lack of completeness can also appear

1. Randomly choose a specific input variable of the rule set to be mutated.
2. Choose at random one of the two following operations: contraction or expansion. Sometimes, it will not be possible to apply some of these operations, i.e., contraction is not possible if the variable only takes a label and expansion is not possible if the variable already takes all the labels. In such a case, directly choose the available operation.
3. If contraction is chosen, apply a movement at random.
4. If expansion is chosen, compute all the set of candidate movements which do not provoke inconsistency or over-generality. If this set is not empty, choose a candidate movement at random and apply it. If the set is empty and contraction can be done, go to step 3; otherwise, if there are still no analyzed input variables, go to step 1, else skip this mutation.
5. Redundancies by subsumption may appear in the rule set after applying expansion, remove the rules subsumed by the mutated rule.

Fig. 5 Antecedent mutation operator

3.5.1 Contraction operation

It converts the mutated rule into a more specific one by choosing a gene of the selected variable with a '1' and flipping to '0'. Clearly, the contraction operator can only be applied when there are, at least, two '1', because on the contrary all the genes of this variable will be '0' and, as mentioned in Sect. 3.1, it is not allowed.

This operator will never cause inconsistency, redundancy or over-generality since it generates a more specific rule, thus avoiding to go into conflict with other rules. The only undesired property it could cause is lack of completeness, but it will be solved by the completeness operator later.

3.5.2 Expansion operation

This operation carries out the opposite process to contraction operator, making the rule be more general. It chooses a gene with allele '0' and flips it to '1'. In this case, the restriction is that the expansion operation can only be applied when there is, at least, a '0' in the genes of the variable.

Unfortunately, this operator can cause collision problems with other rules or generate over-general rules. Therefore, firstly the set of expansion movements that can be applied to the selected variable without causing inconsistencies or creating over-generality (this latter case is checked using the covering hypermatrix) are generated, and then one of them is randomly chosen. In case there are

no allowed movements, and if the variable contains at least two linguistic terms, the contraction operation is applied.

If after performing expansion the mutated rule subsumes other rules, the more specific ones are removed. With this operation it is not possible to get lack of completeness.

3.6 Consequent mutation operator

This operator, which is applied with a given probability rate to each individual, creates new rules by changing the consequent. It simply consists on randomly selecting an output variable of a rule that is not partially overlapped with other rules (it would be the only problematic case since the consequent mutation operator receives consistent and non-subsumed rules). Then, the consequent is randomly changed to the immediately higher or lower linguistic term. The operation does not cause over-generality or lack of completeness since the fuzzy rule antecedent structures are kept invariable.

3.7 Completeness operator

The crossover operator and the antecedent mutation by contraction can produce fuzzy rule sets that do not cover some specific data set examples. It is fixed with this operator by adding rules to patch the uncovered input subspaces. It can be considered a reparation operator with a low incidence since it does not change the generated rules, it only adds new ones. Figure 6 shows its operation mode.

Function: Completeness Operator
Input: One individual and the training data set
Output: The input individual with some fuzzy rules added, if needed
Preconditions: None
Postconditions: The generated individual covers the whole data set, i.e., at least a fuzzy rule matches each training example

1. From the data set, extract such examples that are not covered by any fuzzy rule encoded in the analyzed individual.
2. Apply Wang-Mendel algorithm (see Appendix A) over this example subset to generate the minimum number of Mamdani-type fuzzy rules that represent them.
3. Add these fuzzy rules to the rule base encoded in the individual. Since these rules come from data that were not covered by any previous rule, neither consistency nor redundancy problems arise.

Fig. 6 Completeness operator

3.8 Inference mechanism

We consider the Max–Min inference scheme (i.e., T-conorm of maximum as aggregation and T-norm of minimum as relational operator), and the T-norm of minimum as conjunction, T-conorm of maximum as disjunction, and center-of-gravity as defuzzification. Moreover, in test mode the mean of the output domain is returned when no rules are fired for the given test example as explained in Sect. 2.3.

3.9 Multiobjective approach

A generational approach with the multiobjective elitist replacement strategy of NSGA-II (Deb et al. 2002) is used. Crowding distance in the objective function space is considered. Binary tournament selection based on the nondomination rank (or the crowding distance when both solutions belong to the same front) is applied. The crowding distance is normalized for each objective according to the extreme values of the solutions contained in the analyzed front.

3.10 Objective functions

We consider two objective functions to assess the quality of the generated fuzzy systems, the former (approximation error) to improve the accuracy and the latter (complexity) to improve the interpretability.

- *Approximation error*: The mean squared error (MSE) is used. It is computed as follows:

$$F_1(S) = \frac{1}{N}\sum_{i=1}^{N}(S(x^i) - y^i)^2, \quad (1)$$

with S being the fuzzy system to be evaluated, N the data set size and (x^i,y^i) the ith input-output pair of the data set. The objective is to minimize this function.

- *Complexity*: As complexity measure, we use the number of DNF-type fuzzy rules:

$$F_2(S) = |S|. \quad (2)$$

The objective is to minimize this function.

Since the algorithm is designed to ensure optimal covering, i.e., without lack of completeness or with over-generalization, we do not care on the linguistic complexity (i.e., generalization) of each fuzzy rule. In a natural way, the more general (i.e., with more labels considered in each rule) the fuzzy rules, the fewer the number of rules.

It is an advantage of our approach that simplifies the design of an interpretability-based objective function. For example, Ishibuchi and Nojima (2007) need to consider a third objective to evaluate the generality degree of "do not

care" fuzzy rules in classification since their algorithm does not hold this correspondence between number of rules and generality.

4 Experimental results

This section includes the obtained results of the proposed algorithm in five real-world regression problems (i.e, with real-valued input and output), and compares them with other fuzzy rule learning methods.

4.1 Problems and learning methods

We have considered the following regression problems:

- The Diabetes problem concerns the study of the factors affecting patterns of insulin-dependent diabetes mellitus in children (Hastie and Tibshirani 1990). The objective is to investigate the dependence of the level of serum C-peptide on the various other factors in order to understand the patterns of residual insulin secretion. The response measurement is the logarithm of C-peptide concentration (pmol/ml) at the diagnosis, and the predictor measurements age and base deficit, a measure of acidity. The data set has been obtained from L. Torgo's website.[1]
- The Ele1 problem relates to the estimation of the low voltage electrical line length in rural towns (Cordón et al. 1999). We were provided with a sample of real measurements from 495 towns in Spain. The estimation is based on the *number of inhabitants of the town* and the *distance from the center of the town to the three furthest clients*. The data set (and the partitions used in this paper) is available at the authors' website.[2]
- The Laser problem uses a set of laser data from the Santa Fe Institute (SFI) time series prediction and analysis competition (Weigend and Gershenfeld 1993). The original laser data set from the SFI competition consisted of 1000 observations of the fluctuations in a far-infrared laser. This time series data has been adapted for regression by considering the four last values as input and the next value as output. The data set is available at KEEL website.[3]
- The Ele2 problem concerns the estimation of electrical network maintenance costs of medium voltage line based on *sum of the lengths of all streets* in the town, *total area* of the town, *area occupied* by buildings, and *energy supply* to the town (Cordón et al. 1999). The information is obtained by sampling a model of the optimal electrical network for a town. The data set (and the partitions used in this paper) is available at the authors' website [2].
- The DEE problem involves predicting the daily average price of TkWhe electricity energy in Spain. The data set contains real values from 2003 about the daily consumption in Spain of energy from hydroelectric, nuclear electric, carbon, fuel, and natural gas. The data set has been obtained from KEEL website [3].

Table 1 collects the main features of each problem, where *#InputVar* stands for number of input variables, *#Exam* for total number of examples, and *#LingTerms* for the number of triangular-shaped uniformly distributed linguistic terms considered for each variable in this experimental analysis.

The experiments shown in this paper have been performed with a fivefold cross validation. Thus, the data set is divided into five subsets of (approximately) equal size. The algorithm is then applied five times to each problem, each time leaving out one of the subsets from training, but using only the omitted subset to compute the test error.

We have considered several learning methods for comparison (all of them use the same inference engine described in Sect. 3.8 for our proposal):

- *Wang and Mendel* (Wang and Mendel 1992): It is a simple algorithm that, in spite of not obtaining accurate results, is a traditional reference in the research community. The algorithm has been implemented by us.
- *COR-BWAS* (Casillas et al. 2005b): It is an ant colony optimization-based learning algorithm with a great performance between interpretability and accuracy. We have disabled fuzzy rule selection since the algorithm does not guarantee total completeness, so the results could not be directly compared with our proposal.
- *Thrift* (Thrift 1991): It is a classic Pittsburgh-style GA-based Mamdani-type fuzzy rule learning method. The mean output value is provided to compute MSE when no fuzzy rules are fired for a training example. The algorithm has been implemented by us.

Table 1 Data sets considered in the experimental analysis

Problem	#InputVar	#Exam	#LingTerms
Diabetes	2	43	7
Ele1	2	495	7
Laser	4	993	5
Ele2	4	1,066	5
DEE	6	365	5

[1] L. Torgo. Collection of regression datasets. http://www.liacc.up.pt/~ltorgo/Regression/DataSets.html

[2] J. Casillas. FMLib: fuzzy modeling library. http://decsai.ugr.es/~casillas/FMLib/

[3] KEEL: Knowledge extraction based on evolutionary learning. http://www.keel.es

- *Pittsburgh* (Casillas and Martínez-López 2008): It is a Pittsburgh-style GA that also learns DNF-type fuzzy rules. A generational approach and direct replacement are performed, with elitism of the best solution. The fitness is the MSE (Eq. 1). The pool is randomly initialized and binary tournament selection is done. The same length-variable coding scheme used in this paper is considered. Specific genetic operators for this representation are used. As in Thrift, the mean value is provided when no fuzzy rules are fired.
- *Fuzzy-GAP* (Sánchez and Couso 2000): This method employs a genetic programming algorithm hybrided with a GA (i.e., GA-P) to learn a fuzzy regression model. The algorithm generates complex fuzzy rules with any combination of conjunction and/or disjunctions in the antecedent part. The number of fuzzy rules must be fixed a priori. We have used the implementation of this algorithm available at KEEL software[3].

Our algorithm has been run with the following parameter values: 300 iterations, 60 as population size, 0.7 as crossover probability, and 0.2 as antecedent and consequent mutation probability per chromosome. We have not performed any previous analysis to fix these values, so better results may probably be obtained by tuning them though we have informally noticed our algorithm is not specially sensitive to any parameter. The same parameter values are also used in Thrift and Pittsburgh algorithms. For COR-BWAS, we have fixed standard values [mostly the ones recommended in Casillas et al. (2005b)], i.e., 100 iterations, number of ants equal to the number of input subspaces (defined by the Wang–Mendel algorithm), heuristic 1 (max value), $\rho = 0.20$, $\alpha = 2$, $\beta = 1$, local search depth 10, local search neighbor size equal to the number of ants, mutation probability 0.3, $\sigma = 4$, and restart after 10 iterations without improvement. Fuzzy-GAP is run with the default parameter values suggested in KEEL software[3], except the number of linguistic terms per variable that is the same that in the rest of algorithms and the number of fuzzy rules, that is set to about half of the number of rules used by the Wang–Mendel method (note Fuzzy-GAP uses flexible structures based on conjunctions and disjunctions to express the antecedent of the fuzzy rules).

4.2 Obtained results

Table 2 collects the obtained results for each problem, where #R stands for the number of fuzzy rules and MSE_{tra} and MSE_{tst} the approximation error (Eq. 1) values over the

Table 2 Results obtained in the different problems

Method	*#R*		MSE_{tra}		MSE_{tst}	
	$\bar{x}$	σ	$\bar{x}$	σ	$\bar{x}$	σ
Diabetes problem						
Wang–Mendel	18.6	1.4	0.22836	0.0425	1.40241	0.6890
COR-BWAS	18.6	1.4	0.17496	0.0250	1.45869	0.7091
Thrift	46.2	0.7	**0.07448**	0.0098	0.87825	0.3575
Pittsburgh	15.0	2.9	0.10398	0.0182	0.95088	0.7881
Fuzzy-GAP	10.0	0.0	0.14292	0.0376	0.50141	0.3014
Pitts-DNF min	**1.6**	0.5	0.41624	0.1231	0.45392	0.1288
Pitts-DNF med	5.4	0.5	0.12958	0.0136	**0.32134**	0.1922
Pitts-DNF max	9.6	1.2	0.10656	0.0150	0.63396	0.5276
Ele1 problem						
Wang–Mendel	22.0	1.4	423466	8069	455262	19943
COR-BWAS	22.0	1.4	354304	7065	417142	9823
Thrift	46.4	1.0	335086	5285	435373	57252
Pittsburgh	17.2	4.3	342464	19209	738691	543165
Fuzzy-GAP	11	0	481603	58989	548122	70968
Pitts-DNF min	**2.0**	0.0	767922	55787	760271	56310
Pitts-DNF med	8.2	0.7	344636	8999	**415266**	71200
Pitts-DNF max	14.0	1.1	**330496**	4815	440692	40370
Laser problem						
Wang–Mendel	58.4	1.0	265.21	20.68	278.58	45.55
COR-BWAS	58.4	1.0	220.83	8.06	232.77	54.16
Thrift	517.8	10.1	461.24	95.05	490.10	114.73
Pittsburgh	196.8	2.9	231.30	31.56	311.88	132.51
Fuzzy-GAP	29.0	0.0	540.20	200.95	567.61	279.50
Pitts-DNF min	**11.4**	1.6	641.70	258.86	633.88	258.98
Pitts-DNF med	20.6	1.0	163.01	11.13	234.69	72.53
Pitts-DNF max	33.6	3.2	**109.16**	11.39	**199.19**	90.74
Ele2 problem						
Wang–Mendel	65.0	0.0	112270	1498	112718	4685
COR-BWAS	65.0	0.0	102664	1080	102740	4321
Thrift	524.6	6.4	146305	12991	168472	20135
Pittsburgh	240.0	21.1	210717	32027	265130	30161
Fuzzy-GAP	33.0	0.0	279166	90017	290062	89155
Pitts-DNF min	**12.2**	0.7	202943	43684	212018	44616
Pitts-DNF med	18.6	1.4	86930	3955	99310	12996
Pitts-DNF max	32.4	6.6	**70207**	1658	**88017**	8968
DEE problem						
Wang–Mendel	178	2	0.14117	0.0074	0.22064	0.0264
COR-BWAS	178	2	0.12463	0.0052	**0.20513**	0.0231
Thrift	13020	33	0.38778	0.0357	0.45830	0.0804
Pittsburgh	982	56	0.42111	0.0784	0.72109	0.3263
Fuzzy-GAP	89	0	0.17751	0.0130	0.20633	0.0172
Pitts-DNF min	**34**	1	0.22073	0.0219	0.30635	0.0884
Pitts-DNF med	57	3	0.13821	0.0060	0.27465	0.1366
Pitts-DNF max	98	5	**0.11267**	0.0035	0.21692	0.0359

training and test data set, respectively. Since our algorithm performs multiobjective optimization, several solutions are returned in each run. Therefore, we show three representative solutions from the final Pareto-optimal set, those with the minimum number of rules (i.e., the worst MSE_{tra}), the median solution, and the maximum number of rules (i.e., the best MSE_{tra}). $\bar{x}$ represents the arithmetic mean of each value over the five partitions and σ the corresponding standard deviation. The best mean results for each problem among the analyzed methods are shown in boldface.

4.3 Analysis

From the obtained results we can observe that the proposed method generates fuzzy models with a good degree of accuracy and interpretability. The most accurate solutions provided by our method (Pitts-DNF max) obtain the best training errors in four problems. Good test errors are also achieved.

Compared to the rest of the methods, we can observe the following:

- As regards Wang–Mendel and COR-BWAS, our method not only outperforms them in accuracy but also uses about a 50% of the number of rules needed by them, thus improving the interpretability of the obtained fuzzy models. To illustrate the capability of Pitts-DNF to compactly represent the rules, Table 3 shows an example of the fuzzy rule set obtained by COR-BWAS and the median solution of Pitts-DNF in a data partition of the Ele1 problem. Both solutions offer similar degrees of accuracy, however the rule set obtained by Pitts-DNF is much more compact, only consisting of seven DNF-type fuzzy rules (they are identified with a subindex in the consequent linguistic term for each cell of Table 3b). Note also that this set of rules are the optimal ones to represent the obtained table decision.
- Thrift and the Pittsburgh method show serious difficulties to generate compact fuzzy rule sets, being this fact more significant as the complexity of the problem increases. They both tend to generate a huge number of fuzzy rules, even taking into account that the Pittsburgh method uses DNF-type fuzzy rules. This fact shows how the constraints of the search space imposed by our Pitts-DNF algorithm dramatically improve the search process, being significantly more accurate and interpretable than these other two methods. This leads us to think our algorithm deals better with the curse of dimensionality.
- Finally, even considering our median results from the Pareto sets (Pitts-DNF med), our method outperforms Fuzzy-GAP generating fuzzy models more accurate and with a lower number of rules.

Table 3 Fuzzy rule set obtained in the first data partition of Ele1 problem by (a) COR-BWAS and (b) Pitts-DNF (median solution)

		X_1						
		XS	*VS*	*S*	*M*	*L*	*VL*	*XL*
(a) COR-BWAS $MSE_{tra\ /\ tst}$ = 356137/370626 #R = 22]								
X_2	*XS*	XS	VS					
	VS	XS	VS	VS	VS			
	S	VS	S	S	S	M		
	M	VS	M	S	VL			
	L		M	S	XL			
	VL	VS		L	M			
	XL	M						
(b) Pitts-DNF [$MSE_{tra\ /\ tst}$ = 348914/390556 #R = 7 ($\#R_{Mamdani}$ = 16)]								
X_2	*XS*	XS_1			VL_7			
	VS	XS_1	VS_3	VS_3	VS_3			
	S	VS_2	S_5	S_5	S_5	S_5		
	M	VS_2	M_6					
	L				VL_7			
	VL		M_6		VS_4			
	XL							

Analyzing our median results (Pitts-DNF med) we can observe that the algorithm is able to derive fuzzy models with a very low number of fuzzy rules preserving a good accuracy degree. It is important to remark that, due to the design of the proposed algorithm, these small fuzzy rule sets still completely cover the training data sets, which is not ensured by the other two Pittsburgh-style algorithms.

Furthermore, Figs. 7, 8, 9, 10, and 11 show the average Pareto fronts and generality degrees obtained by the

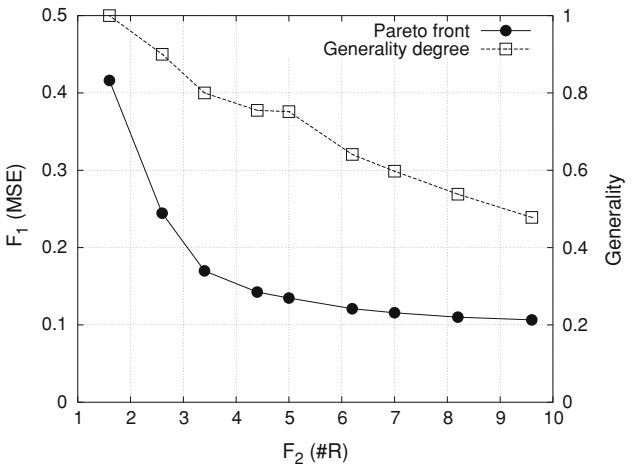

Fig. 7 Average Pareto front (*solid circles*) and generality degrees (*empty squares*) obtained in the Diabetes problem

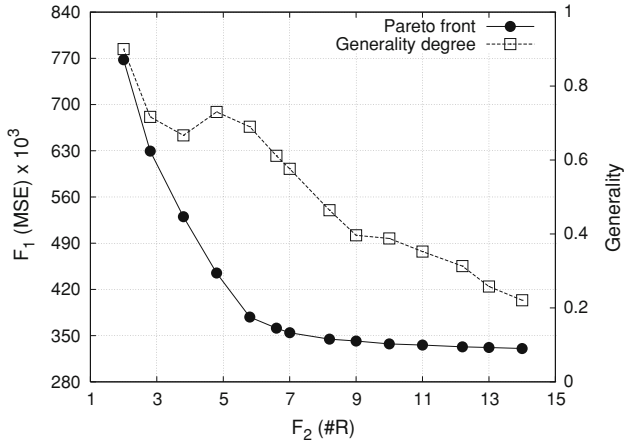

Fig. 8 Average Pareto front (*solid circles*) and generality degrees (*empty squares*) obtained in the Ele1 problem

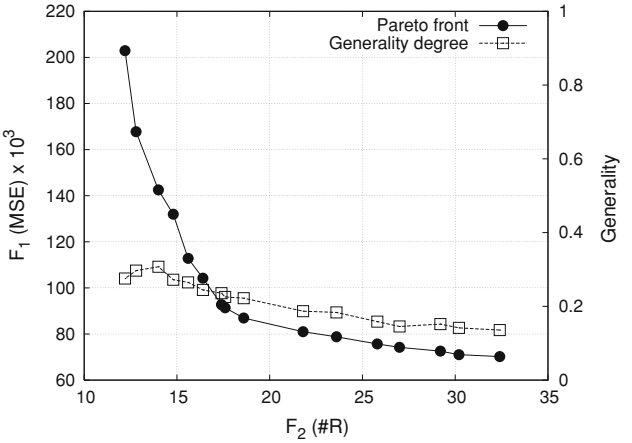

Fig. 10 Average Pareto front (*solid circles*) and generality degrees (*empty squares*) obtained in the Ele2 problem

proposed Pitts-DNF algorithm. The generality degree is computed by counting the mean number of linguistic terms used per variable in each rule. It is normalized to be between 0 (maximum specificity, i.e., where all the fuzzy rules are Mamdani-style) and 1 (maximum generality, i.e., where only one rule that covers the whole input space is used).

The generality degrees are plotted to show the correspondence between number of rules and generality kept by the algorithm without needing to consider this third objective. Naturally, as the number of rules increases, they become more specific so the mean generality degree decreases. As it can be observed, the algorithm generates a large range of solutions with different interpretability-accuracy balances.

5 Self-analysis: strengths, weaknesses, opportunities, and threats

A honest self-analysis of the proposed algorithm is described in Table 4, where *strengths* represent the main advantages of Pitts-DNF, *weaknesses* show its drawbacks, *opportunities* outline some suggested further lines of investigation, and *threats* include some optional approaches that could compete with our proposal.

Pitts-DNF has some important strengths. Firstly, the experiments show that the algorithm performance, both in interpretability and accuracy, is competitive compared with other approaches. Moreover, it uses a flexible fuzzy rule structure for a better knowledge synthesis which increases the interpretability. Besides, it generates consistent fuzzy rule sets, which improves the interpretability since the rules do not interfere among them. It also generates both

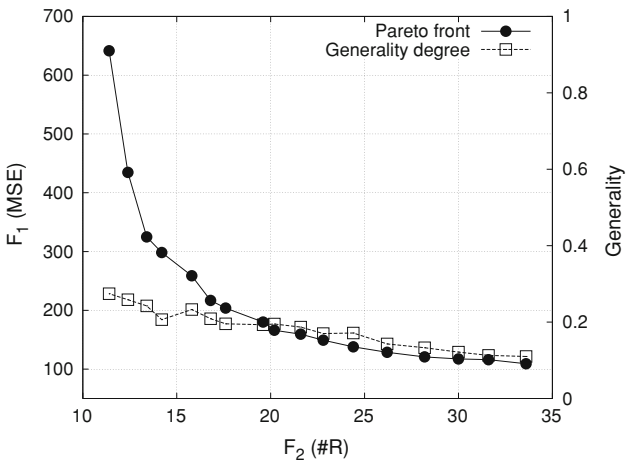

Fig. 9 Average Pareto front (*solid circles*) and generality degrees (*empty squares*) obtained in the Laser problem

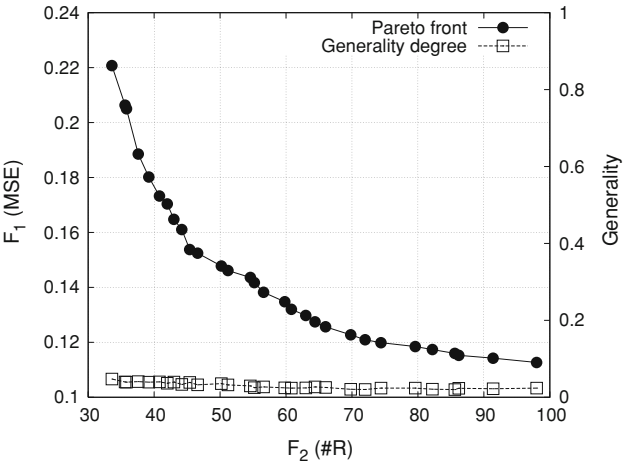

Fig. 11 Average Pareto front (*solid circles*) and generality degrees (*empty squares*) obtained in the DEE problem

Table 4 SWOT analysis of Pitts-DNF

Strengths	Weaknesses
Its performance, both in interpretability and accuracy, is competitive compared with other approaches	It only works in data-driven regression problems
It uses a flexible fuzzy rule structure for a better knowledge synthesis	The considered properties (consistency, completeness and optimal generality) may involve to generate a higher number of rules is some problems
It generates consistent, complete, compact, and non over-general fuzzy rule sets	Although it is better prepared for dealing with high dimensional problems than other approaches, it still needs to be improved to properly generalize the fuzzy rules
It performs multiobjective optimization to return solutions with different interpretability-accuracy trade-offs	
Opportunities	**Threats**
To adapt the algorithm to classification problems	Solutions based on guiding the search process by objective functions that penalize the lack of some of the analyzed properties may obtain good solutions in the practice due to the search flexibility
To adapt the algorithm to learn Takagi-Sugeno fuzzy rules	Consistency may be obtained by applying a two-stage sequential approach: generation of Mamdani-type fuzzy rules plus an *a posteriori* rule grouping process
To combine Pitts-DNF with a membership function parameter learning/tuning process	
To study more complex solutions for avoiding over-generality without leaving uncovered regions	
To analyze other fuzzy rule structures even more flexible than DNF-type for a more compact knowledge representation	

complete and non over-general fuzzy rule sets; therefore, the expert is sure that the provided fuzzy rule sets always cover the whole training data set and they do not cover areas where there are not training data, which helps to gain additional insight into the data set. It generates compact fuzzy rule sets without redundancies. Finally, it performs multiobjective optimization, so a large range of different interpretability-accuracy trade-offs are returned.

The main weaknesses of the method are the following. Firstly, it only works in regression problems. Although it can be easily adapted to classification, to get a competitive classification algorithm may involve more effort than a simple adaptation. Secondly, as discussed in Sect. 2, the fact of constraining the fuzzy rule sets to be consistent and non over-general may imply generating a higher number of rules is some problems. Finally, even when the algorithm is better prepared for dealing with high dimensional problems than other approaches, it still needs to be improved to properly generalize the fuzzy rules.

We also want to mention some possible threats to Pitts-DNF. On the one hand, other approaches based on guiding the search process by objective functions that penalize inconsistencies of the fuzzy rule set (Wang et al. 2005), though they seem to be worst approaches since unfeasible solutions are explored, can obtain good solutions in practice due to the search flexibility. On the other hand, consistency may be faced by applying a two-stage sequential approach: generation of Mamdani-type fuzzy rules (e.g., by Wang and Mendel 1992) plus an *a posteriori* rule grouping process (e.g., by Carmona and Castro 2005). However, we think this two-stage approach, although it is useful to look for accurate solutions, it is not able to balance the accuracy with the linguistic generalization capability since it keeps the original decision table unaltered.

As further work, we intend to adapt the algorithm to classification problems (where the output is a class instead of a real value) and to learn Takagi-Sugeno fuzzy rules, to combine Pitts-DNF with a membership function parameter learning/tuning process (e.g., Casillas et al. 2005a), to study other solutions for avoiding over-generality without leaving uncovered regions (e.g., by doing linguistic extrapolation), and to analyze other fuzzy rule structures even more flexible than DNF-type for a more compact knowledge representation (such as using more relational operators in the antecedent or local exceptions to general rules).

Acknowledgments This work was supported in part by the Spanish Ministry of Education and Science under grant no.

TIN2005-08386-C05-01 and the Andalusian Government under grants no. P05-TIC-00531 and P07-TIC-3185.

Appendix: Wang–Mendel algorithm

The ad hoc data-driven Mamdani-type fuzzy rule set generation process proposed by Wang and Mendel(1992) is widely known and used because of its simplicity. In our algorithm, Pitts-DNF, it is used in the initialization process and completeness operator. Therefore, for the sake of readability we briefly introduce the algorithm in this appendix.

It is based on working with an input-output data pair set representing the behavior of the problem being solved:

$$E = \{e_1, \ldots, e_N\},\ e_l = (x_1^l, \ldots, x_n^l, y_1^l, \ldots, y_m^l),$$

with N being the data set size, n the number of input variables, and m the number of output variables. The algorithm consists of the following steps:

1. Consider a fuzzy partition (definitions of the membership functions parameters) for each input/output variable.
2. Generate a candidate fuzzy rule set: This set is formed by the rule best covering each example contained in E. Thus, N candidate fuzzy rules, CR^l, are obtained. The structure of each rule is generated by taking a specific example, i.e., an $(n + m)$-dimensional real vector, and setting each one of the variables to the linguistic term (associated fuzzy set) best covering every vector component:

$$CR^l: \text{IF } X_1 \text{ is } A_1^l \text{ and } \ldots \text{ and } X_n \text{ is } A_n^l \text{ THEN } Y_1 \text{ is } B_1^l \text{ and } \ldots \text{ and } Y_m \text{ is } B_m^l$$

$$A_i^l = \arg \max_{A' \in \mathbf{A}_i} \mu_{A'}(x_i^l), \quad B_j^l = \arg \max_{B' \in \mathbf{B_j}} \mu_{B'}(y_j^l)$$

3. Give an importance degree to each candidate rule:

$$D(CR^l) = \prod_{i=1}^{n} \mu_{A_i^l}(x_i^l) \cdot \prod_{j=1}^{m} \mu_{B_j^l}(y_j^l)$$

4. Obtain a final fuzzy rule set from the candidate fuzzy rule set: To do so, the N candidate rules are first grouped in g different groups, each one of them composed of all the candidate rules containing the same antecedent combination. To build the final fuzzy rule set, the rule with the highest importance degree is chosen in each group. Hence, g will be both the number of different antecedent combinations in the candidate rule set and the number of rules in the Mamdani-type fuzzy rule set finally generated.

References

Carmona P, Castro J (2005) Interpretability enhancement of fuzzy modeling using ant colony optimization. In: Proceedings of the 1st international workshop on genetic fuzzy systems (GFS 2005), Granada, Spain, pp 148–153

Carmona P, Castro J, Zurita J (2004) Learning maximal structure fuzzy rules with exceptions. Fuzzy Sets Syst 146:63–77

Casillas J, Martínez-López F (2008) Mining uncertain data with multiobjective genetic fuzzy systems to be applied in consumer behaviour modelling. Expert Syst Appl (in press). doi:10.1016/j.eswa.2007.11.035

Casillas J, Cordón O, Herrera F, Magdalena L (eds) (2003a) Accuracy improvements in linguistic fuzzy modeling. Springer, Heidelberg

Casillas J, Cordón O, Herrera F, Magdalena L (eds) (2003b) Interpretability issues in fuzzy modeling. Springer, Heidelberg

Casillas J, Cordón O, del Jesus M, Herrera F (2005a) Genetic tuning of fuzzy rule deep structures preserving interpretability and its interaction with fuzzy rule set reduction. IEEE Trans Fuzzy Syst 13(1):13–29

Casillas J, Cordón O, de Viana IF, Herrera F (2005b) Learning cooperative linguistic fuzzy rules using the best-worst ant system algorithm. Int J Intell Syst 20:433–452

Castro J, Castro-Schez J, Zurita J (1999) Learning maximal structure rules in fuzzy logic for knowledge acquisition in expert systems. Fuzzy Sets Syst 101(3):331–342

Cordón O, Herrera F, Sánchez L (1999) Solving electrical distribution problems using hybrid evolutionary data analysis techniques. Appl Intell 10(1):5–24

Cordón O, Herrera F, Hoffmann F, Magdalena L (2001) Genetic fuzzy systems: evolutionary tuning and learning of fuzzy knowledge bases. World Scientific, Singapore

Deb K, Pratap A, Agarwal S, Meyarevian T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evolut Comput 6(2):182–197

Fullér R (2000) Introduction to neuro-fuzzy systems. Physica-Verlag, Heidelberg

González A, Pérez R (1998) Completeness and consistency conditions for learning fuzzy rules. Fuzzy Sets Syst 96(1):37–51

González A, Pérez R (1999) SLAVE: a genetic learning system based on an iterative approach. IEEE Trans Fuzzy Syst 7(2):176–191

Hastie T, Tibshirani R (1990) Generalized additive models. Chapman & Hall, London

Ishibuchi H, Nojima Y (2007) Analysis of interpretability-accuracy tradeoff of fuzzy systems by multiobjective fuzzy genetics-based machine learning. Int J Approx Reason 44(1):4–31

Ishibuchi H, Yamamoto T, Nakashima T (2006) An approach to fuzzy default reasoning for function approximation. Soft Comput 10:850–864

Jin Y, von Seelen W, Sendhoff B (1999) On generating FC^3 fuzzy rule systems from data using evolution strategies. IEEE Trans Syst Man Cybern B Cybern 29(4):829–845

Karr C (1991) Genetic algorithms for fuzzy controllers. AI Expert 6(2):26–33

Lavrač N, Cestnik B, Gamberger D, Flach P (2004) Decision support through subgroup discovery: three case studies and the lessons learned. Mach Learn 57(1–2):115–143

Magdalena L (1997) Adapting the gain of an FLC with genetic algorithms. Int J Approx Reason 17(4):327–349

Nauck D, Klawonn F, Kruse R (1997) Foundations of neuro-fuzzy systems. Wiley, New York

Nozaki K, Ishibuchi H, Tanaka H (1997) A simple but powerful heuristic method for generating fuzzy rules from numerical data. Fuzzy Sets Syst 86(3):251–270

Otero J, Sánchez L (2006) Induction of descriptive fuzzy classifiers with the logitboost algorithm. Soft Comput 10(9):825–835

Sánchez L, Couso I (2000) Fuzzy random variables-based modeling with GA-P algorithms. In: In Information, uncertainty and fusion. Kluwer, Dordrecht, pp 245–256

Sánchez L, Couso I, Corrales JA (2001) Combining GP operators with SA search to evolve fuzzy rule based classifiers. Inf Sci 136(1–4):175–191

Thrift P (1991) Fuzzy logic synthesis with genetic algorithms. In: Belew R, Booker L (eds) Proceedings of the 4th international conference on genetic algorithms. Morgan Kaufmann Publishers, San Mateo, pp 509–513

Valenzuela-Rendón M (1991) The fuzzy classifier system: a classifier system for continuously varying variables. In: Proceedings of the 4th international conference on genetic algorithms. Morgan Kaufmann Publishers, San Mateo, pp 346–353

Wang C, Hong T, Tseng S, Liao C (1998) Automatically integrating multiple rules sets in a distributed-knowledge environment. IEEE Trans Syst Man Cybern C Appl Rev 28(3):471–476

Wang H, Kwong S, Jin Y, Wei W, Man KF (2005) Agent-based evolutionary approach for interpretable rule-based knowledge extraction. IEEE Trans Syst Man Cybern 35(2):143–155

Wang LX (2003) The WM method completed: a flexible fuzzy system approach to data mining. IEEE Trans Fuzzy Syst 11(6):768–782

Wang LX, Mendel J (1992) Generating fuzzy rules by learning from examples. IEEE Trans Syst Man Cybern 22(6):1414–1427

Weigend A, Gershenfeld N (eds) (1993) Time series prediction: forecasting the future and understanding the past. In: 1992 NATO Advanced Research Workshop on Comparative Time Series Analysis. Addison-Wesley, Santa Fe

Xiong N, Litz L (2000) Fuzzy modeling based on premise optimization. In: Proceedings of the 9th IEEE international conference on fuzzy systems, San Antonio, TX, USA, pp 859–864

Yager R (1993) On a hierarchical structure for fuzzy modeling and control. IEEE Trans Syst Man Cybern 23(4):1189–1197