UNIVERSIDADE DE SÃO PAULO

Instituto de Ciências Matemáticas e de Computação

ISSN 0103-2569

AN APPROACH FOR THE EXTRACTION OF CLASSIFICATION RULES FROM FUZZY FORMAL CONTEXTS

MARCOS EVANDRO CINTRA
TREVOR P. MARTIN
MARIA CAROLINA MONARD
HELOISA DE ARRUDA CAMARGO

Nº 368

RELATÓRIOS TÉCNICOS

São Carlos – SP
Ago./2011

An Approach for the Extraction of Classification Rules from Fuzzy Formal Contexts

Marcos Evandro Cintra
Maria Carolina Monard
Trevor P. Martin
Heloisa de Arruda Camargo

[1]Universidade de São Paulo – USP
Instituto de Ciências Matemáticas e de Computação – ICMC
Laboratório de Inteligência Artificial – LABIC
Av. Trabalhador São-carlense, 400 – Centro
Caixa Postal: 668 – CEP: 13560-970 – São Carlos – SP – Brasil

cintra,mcmonard@icmc.usp.br

trevor.martin@bristol.ac.uk, heloisa@dc.ufscar.br

Abstract. *This work describes experiments carried out using fuzzy formal concept analysis for the generation of fuzzy classification rules to be used by a genetic process. These rules are simply the intention of the formal concepts extracted from a fuzzy-based formal context. The motivation we have is the need for a method to generate fuzzy classification rules to be used as the search space of the genetic process with reasonable computational cost. The hypothesis we work on is that the extraction of classification rules from a fuzzy formal context can be a suitable alternative to generate the search space of a genetic process due to its low complexity and straightforwardness/simplicity, generating not only good rules, due to the fact that the support of these rules is a subproduct of the extraction of them, but also better interpretable rules, with a varying number of conjunctions in their antecedent part. In order to reduce the complexity of the extraction of the formal concepts, besides the generation of all existing formal concepts representing classification rules, an alternative approach is also proposed and studied: the use of the lattice in order to sequentially extract classification rules. Both ideas are detailed together with the preliminary experiments and results.*

Contents

1 Introduction **1**

2 Genetic Fuzzy Systems **3**

3 Methods for the Generation of Rules to Form the Search Space of Genetic Fuzzy Systems **5**

3.1 Use of Heuristic Criteria 5

3.2 Association Rules Extraction 6

4 Formal Concept Analysis **7**

4.1 The Extraction of Formal Concepts Using the NextNeighbor Algorithm 11

4.2 Adaptation of the NextClosure Algorithm for Formal Concept Extraction 16

4.3 Proposal for Extracting Fuzzy Rules from a Formal Context . 18

5 Preliminary Experiments **19**

5.1 Experiments and Results 19

5.2 Additional Experiments on Attribute Exploration to Reduce the Number of Extracted Rules 22

6 Conclusions and Future Work **25**

References **28**

1. Introduction

Fuzzy systems, which are basically systems with variables based on fuzzy logic, have been successfully used for the solution of problems in many areas, including pattern classification, optimization, and control of processes (Dumitrescu et al., 2000; Pedrycz, 1996). Recently, fuzzy systems have also been used in conjunction with Formal Concept Analysis (FCA), a theory of data analysis which identifies conceptual structures among data sets.

The focus of this work is the automatic definition of fuzzy systems, especially those known as Rule Based Fuzzy Systems (RBFS), which usually have two main components: a Knowledge Base (KB) and an Inference Mechanism (IM). The KB comprises the Fuzzy Rule Base (FRB), *i.e.*, a set of fuzzy rules that represents a given problem, and the Fuzzy Data Base (FDB), which contains the definitions of the fuzzy sets related to the linguistic variables used in the FRB. The IM is responsible for carrying out the required computation that uses inferences to derive the output (or conclusion) of the system, based on both the KB and the input to the system.

The special term "Genetic Fuzzy System" (GFS) was coined by the community to refer to fuzzy systems that use a genetic algorithm to create or adjust one or more of their components (Cordon et al., 2004). Specifically, the classification of GFSs, according to Herrera (2008), takes into account if the goal is:

- the genetic tuning of an existing knowledge base;
- the genetic learning of components of the knowledge base.

This work is focused on the genetic learning of the rule base, one of the components of the knowledge base.

Regarding the generation of rules to form the search space of a Genetic Algorithm (GA), Ishibuchi and Yamamoto (2004) proposed an approach based on the rules confidence and support to preselect rules. A predefined number of rules with 0, 1, 2, and 3 antecedent conjunctions were generated for the wine dataset (Frank and Asuncion, 2010) and used with a GA. A similar approach, named DOC-BASED, is proposed in (Cintra et al., 2007), in which, after the exhaustive generation of possible rules, a subset of them is selected to form the search space of a GA according to the degree of coverage of the rules. However, since the task of generating

all possible rule combinations has exponential complexity, depending on the number of fuzzy variables and sets, the number of possible rules can be very large, interfering with the codification of the chromosomes, and overloading the whole genetic learning process. Thus, for datasets described by many features, a preselection of the most relevant features is important. Also, preselecting the most important possible rules to form the seach space might be essential for datases described by many features.

The theory of FCA was introduced in 1982 by Wille (1982) and has since then grown rapidly. FCA has successfully been applied to many fields, such as medicine and psychology, musicology, linguistic databases, library and information science, software re-engineering, civil engineering, ecology, and others. A strong feature of FCA is its capability of producing graphical visualizations of the inherent structures among data. Another strong feature of FCA is related to its mathematical lattices since they can be interpreted as classification systems and even used for the extraction of association and classification rules. For the extraction of association rules, these rules are composed of the attributes present in the formal concepts extracted from the formal context, *i.e.*, the intention of the formal concept. Their support is calculated directly from the process of extracting the formal concept, with no extra computation cost. Regarding the classification rules, they can be extracted from the formal concepts containing a class in their intention, discarding their extensions. Several methods for the extraction of formal concepts have been proposed (Ganter, 1984, 2002; Krajca et al., 2010). The one by (Krajca et al., 2010), is particularly interesting because it proposes a new method for the structuring of the formal context that reduces considerably the processing time for the extraction of the formal concepts and allows the parallelization of the process.

The integration of FCA with fuzzy logic has also produced several proposals (Zheng et al., 2009; Sigmund et al., 2010; Wolff, 2002; Belohlávek et al., 2005). One of the reasons for this integration is due to the fact that, since FCA requires discrete attributes, the fuzzy theory can be used to transform continuous attributes into binary ones using fuzzy sets, in order to define the formal context.

Our interest in FCA is related to its simplicity in the extraction of classification rules from data. These rules are particularly interesting be-

cause they present a variable number of conjunctions in their antecedent and are extracted from examples. Thus, our hypothesis is that we can extract fuzzy rules from a fuzzy FCA to populate the search space of a genetic process that, in turn, will be used to induce a fuzzy rule base. We intend to use the genetic rule selection approach (Herrera, 2008) representing each chromosomes as complete rule bases with a predefined number of rules.

The remainder of this work is organized as follows. Section 2 presents the basic concepts of Genetic Fuzzy Systems. Section 3 describes some of the most frequently used methods in the literature for the generation of genetic search spaces. Section 4 introduces the theory of Formal Concept Analysis and fuzzy scaling, as well as our proposal for the use of FCA to extract fuzzy rules. Section 5 presents the preliminary experiments. Section 6 presents the conclusions and future work.

2. Genetic Fuzzy Systems

Accordingly to Herrera (2008), GFSs can be classified as:

1. Genetic tuning: if there exists a knowledge base, a genetic tuning process for improving the rule-based fuzzy system performance is applied without changing the existing fuzzy rule base;
2. Genetic learning: a component of the knowledge base is learnt, either the fuzzy rule base, fuzzy data base, or both. An adaptive inference mechanism can also be included in the process.

The genetic tuning can be further divided according to the taxonomy provided in Figure 1(a), extracted and adapted from Herrera (2008), into: i) genetic tuning of KB parameters; ii) genetic adaptive inference engine. Similarly, the genetic learning can also be further divided into: i) genetic KB learning; and ii) genetic learning of DB components and inference engine parameters.

Our work is related to the genetic learning of the FRB, which is a part of the KB. Figure 1(b) provides the complete classification of the genetic KB learning, according to Herrera (2008), which is also listed with some references next.

2.1 Genetic rule learning with a predefined fuzzy data base (Thrift, 1991);
2.2 Genetic rule selection with a priori rule extraction (Cintra and Camargo, 2007);

2.3 Genetic fuzzy data base learning Cordon et al. (2001);

2.4 Simultaneous genetic learning of knowledge base components (Homaifar and McCormick, 1995).

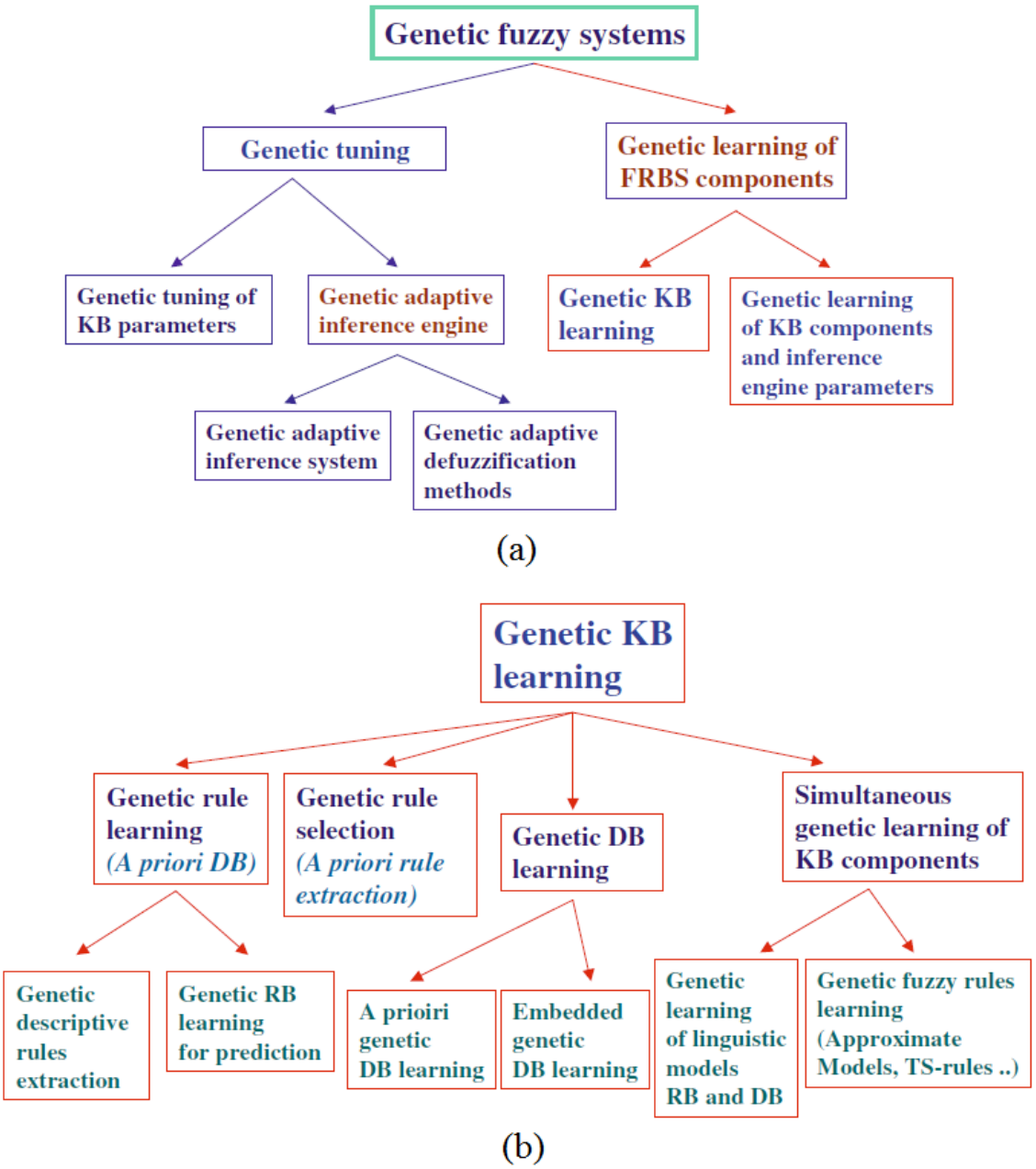


Figure 1. Genetic Fuzzy Systems Classification Herrera (2008).

More specifically, our work focuses on methods to form the search space of a GA in order to build a GFS, which is part of item 2.2 from the previous list. Next section surveys some of the key existing approaches for this task, *i.e.*, the task of generating the search space for a GFS.

3. Methods for the Generation of Rules to Form the Search Space of Genetic Fuzzy Systems

In this section we describe some of the most frequently used methods in the literature for the generation of genetic search spaces, presenting and discussing their pros and cons.

3.1. Use of Heuristic Criteria

Ishibuchi and Murata (1995) propose the generation of classification rules to form the search space of a GA using the following steps:

1. generation of all possible rule antecedent combinations;
2. calculation of the degree of certainty of each antecedent combination with each possible class using a set of training examples;
3. definition of the consequent of each rule antecedent combination as the class with highest degree of certainty.

Although this approach suits low-dimensional domains, it is not scalable to larger domains. Thus, Ishibuchi and Yamamoto (2004) propose the extensive generation of rule antecedent and class calculation using the degree of certainty with preselection of candidate rules as a more feasible alternative to larger domains. The criteria used to preselect rules are the confidence and support measures, as defined in the data mining context for association rules.

Similarly to the approach proposed by Ishibuchi and Yamamoto (2004), Cintra et al. (2007) propose the generation of the search space by extensively generating all possible rules combining all attributes, then calculating their degree of coverage as a criterium to preselect a subset of them. This way, all rules were composed by a combination of valid linguistic values for each attribute. The Degree of Coverage (DoC) was used as an indication of the classification power of these rules.

In both approaches described previously, the main issue is related to the exponential complexity of the exhaustive generation of all possible rules. While these approaches are feasible for domains described by a small number of attributes combined with a reduced set of linguistic values for each attribute, they are not directly scalable for larger domains or when attributes are described by many linguistic values. Another issue is the computational cost required to calculate the degree of certainty and degree of coverage of these rules when combined with the substantial computational effort and processing time required by GAs.

Regarding the advantages of these methods, and others based on heuristic criteria, we can state the following:

1. Since these approaches preselect only relevant rules (according to criteria related to their classification power), the search space can be substantially reduced;
2. Due to the fact that the rules are previously selected according to their classification power, the search process of the GA is reduced in terms of time;
3. With the preselection of rules, each rule in the search space has an index that is used in the chromosomes, thus, the chromosome codification is simplified (each position of the chromosome has an index to a rule or a special value to represent the absence of a rule). This way, the whole genetic process is optimized saving processing time;
4. Although the approach is not scalable to larger domains, or when a large number of fuzzy terms is used, one might argue that it is, nevertheless, totally feasible and produces good results for a large number of real domains. Also, the use of a feature subset selection algorithm reduces the computational cost and time required by this approach, making it scalable to larger domains.

3.2. Association Rules Extraction

Another option for the generation of the rules to form the search space of a GA is to use an association rule extraction algorithm, such as Apriori (Agrawal and Srikant, 1994). Given a set of examples described by a set of attributes, the idea is to find rules based on associated items. The support and confidence values of these rules are usually considered for their evaluation. Although association rules are not concerned with supervised domains, association rule algorithms can be used to extract associative classification rules, or simply classification rules, from a set of examples, by simply fixing the consequent of the rules as the class attribute.

In the literature, it is possible to find proposals to generate fuzzy rule-based classifiers using fuzzy association rules (Hu et al., 2002; Pach et al., 2008). Hu et al. (2002) propose the generation of large fuzzy grids from training examples by fuzzy partitioning each attribute; these grids are then use to generate fuzzy association rules for classification. Pach et al. (2008) use the Fuzzy APriori algorithm to search for frequent fuzzy

item sets to form classification rules. The set of these rules is then pruned using the complexity, importance, and generality measures of the rules, forming a fuzzy classifier.

While approaches for the generation of classifiers using fuzzy association rules are abundant in the literature, to the best of our knowledge, the closest proposal using association rules and GAs to a GFS can be found in (Ishibuchi et al., 2006), in which the authors propose the use of association rule mining to define classifiers using a technique to search for Pareto-optimal rule sets. The authors first mine all possible classification rules using a minimum support and confidence values. These rules are then used by an evolutionary multiobjective optimization algorithm to search for Pareto-optimal rule sets. The three objectives used are: i) number of correctly classified training patterns; ii) number of selected rules (number of rules in the classifier); and iii) total number of antecedent conditions over the selected rules in the classifiers. It is important to notice that the rules generated in (Ishibuchi et al., 2006) are not fuzzy, but classic association rules.

An advantage of this approach based on association rules is related to the fact that the support of these rules can be used in a selection process, discarding rules with very low support, which would not help improve the final rule base, speeding up the genetic search process, and contributing to a better interpretability of the final FRB.

The disadvantages might include the fact that extracting all possible rules is an exponentially complex task. Furthermore, the number of attributes to be included in the extraction process must be defined in advance and, thus, the total number of extracted rules might not be sufficient to form the search space, creating a dilemma: the number of combined attributes to be used *versus* the number of possible extracted rules.

Next, we briefly introduce the topic of Formal Concept Analysis (FCA) and present our proposed approach for the generation of the genetic search space using FCA.

4. Formal Concept Analysis

Formal Concept Analysis (FCA) is a mathematical technique for extracting concepts and structure from data introduced in Wille (1982), which is

becoming increasingly popular, especially for allowing the visualization of structures in data.

FCA transforms a formal context into a concept lattice. A formal context is a representation of the relation between objects and their attributes. The basic data structure in FCA is the context, which is normally represented in a table form where the columns represent the attributes and the rows represent the objects. The context table contains 1 (true) in cell (i, j) if object i has attribute j, and 0 (false) otherwise. Formally, a context is a triple $k = (G, M, I)$, where G is a set of objects, M is a set of attributes, and I is a binary relation $I \subseteq G \times M$. Given a set of objects $A \subseteq G$, the shared image of A in M is defined as:

$$A^{\uparrow} = \{m \in M | (g, m) \in I \ \forall \ g \in A\} \tag{1}$$

Consider the formal context present in Table 1. The set of objects, called G or extension, is formed by $\{O_1, O_2, O_3, O_4\}$. The set of attributes, called M, is formed by $\{At_1, At_2, At_3, At_4\}$. Let us define A as set $\{O_1, O_2, O_4\}$. The shared image of A in M, noted as $A^{\uparrow}$, is defined by the set containing all attributes shared by all elements of A. This way, $A^{\uparrow} = \{At_2\}$.

Table 1. Toy Example.

	At_1	At_2	At_3	At_4
O_1	1	1	0	0
O_2	0	1	1	0
O_3	1	1	0	0
O_4	1	1	0	1

Similarly, for a set of attributes $B \subseteq M$, its shared image in G is:

$$B^{\downarrow} = \{g \in G | (g, m) \in I \ \forall \ m \in B\} \tag{2}$$

Consider the formal context present in Table 1. Let us define B as set $\{At_1, At_2\}$. The shared image of B in G, noted as $B^{\downarrow}$, is defined by the set containing all objects sharing all attributes of B. This way, $B^{\downarrow} = \{O_1, O_2, O_3\}$. Let us now define B as set $\{At_3, At_4\}$. The shared image of B in G is the empty set, since there are no objects that share attributes At_3 and At_4 at the same time. Thus , $B^{\downarrow} = \emptyset$.

The pair $(A, B) \in G \times M$ is a formal concept of (G, M, I) if and only if

$A \subseteq G, B \subseteq M$ and $A = B\downarrow$, $B = A\uparrow$. A is called the ***extent*** of the concept and B is called the ***intent*** of the concept (Wille, 1982). In other words, Equation 1 defines the collection of all attributes shared by all objects from A, and Equation 2 defines the collection of all objects sharing all the attributes from B. $A\uparrow$ and $B\downarrow$ are also commonly noted as A' and B' by the FCA community, and this notation is adopted in the remaining of this work.

In traditional FCA, the attributes is binary, although multi-valued contexts are much more common than binary-valued ones. For attributes that can take a range of values, the idea of "conceptual scaling" that transforms a many-valued attribute (e.g. a number) into a symbolic attribute can be used. For example, an attribute such as "height in centimetres", given an integer or real value between 0 and 200, could be transformed into attributes "height-less-than-50", "height-from-50-to-99", etc. These derived attributes have true/false values and can thus be treated within the FCA framework.

A toy example is illustrated in Table 2, which presents an attribute $\times$ value table with the name, age, sex, and hair colour of six people.

Table 2. Attribute $\times$ value table of a toy example.

Name	Age	Sex	Hair Colour
Andy	48	M	Black
Lina	29	F	Black
Mark	23	M	Brown
Martina	46	F	Blonde
Mike	18	M	Brown
Suzy	17	F	Blonde

In order to create the formal context, once FCA only admits binary attributes, attribute Age is discretized into three attributes, Age "$\leq$ 20", "> 20 & $\leq$ 30", and "> 30". Once attributes Sex and Hair Colour already present nominal values, these values are used to create binary attributes. Table 3 is the resulting table after the scaling process.

Using the formal context, it is possible to generate a conceptual lattice that presents the information in a friendly visual way. Figure 2 shows the generated conceptual lattice of the formal context presented in Table 3. In the lattice structure, formal concepts are represented by nodes. Attributes are noted slightly above nodes while objects are noted

Table 3. Formal context generated by the scaling of Table 2.

	Age			Sex		Hair Colour		
Name	≤ 20	> 20 & ≤ 30	> 30	Male	Female	Black	Blonde	Brown
Andy			⊛	⊛		⊛		
Lina		⊛			⊛	⊛		
Mark		⊛		⊛				⊛
Martina			⊛		⊛		⊛	
Mike	⊛			⊛				⊛
Suzy	⊛				⊛		⊛	

slightly under nodes. This way, each node will contain the labels of the attributes and/or objects it shares. The positioning of the nodes can be arranged in a variety of ways. In the lattice presented in Figure 2, the nodes were arranged in order to minimize intersections, thus, attributes are not displayed in the order they occur in the formal context. Another option would be to arrange the nodes respecting the order of the attributes or the order of the examples in the formal context. In order to retrieve extensions, one must simply trace all paths leading down from the node. To retrieve intentions, on the other hand, one must trace all paths leading up from the node. For example, the intention of the formal concept represented by the node with the label *Mark* is {> 20 & ≤ 30, *Brown*, *Male*}. The extension of the formal concept represented by the node with the label *Blonde* is {*Martina*, *Suzy*}.

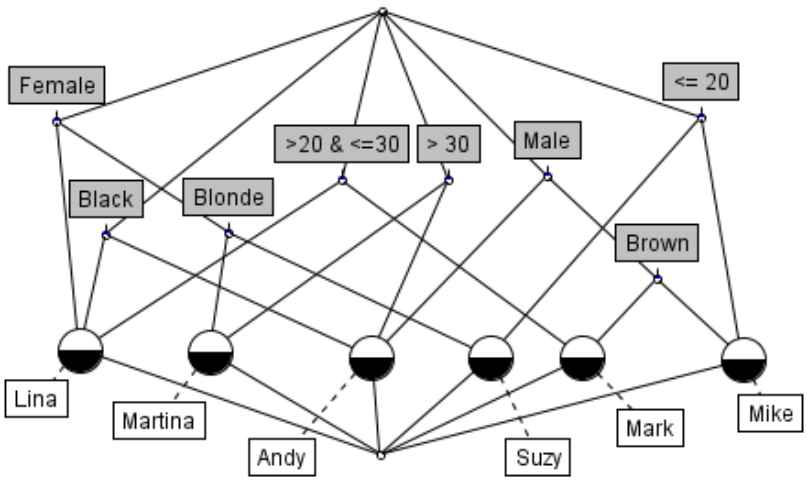


Figure 2. Conceptual lattice.

As stated previously, the transformation of continuous attributes into binary ones is commonly called *scaling* by the FCA community. The fuzzy theory has also contributed to this task. Wolff (2002) presents an introduction to the use of the fuzzy theory for the task of FCA scaling. Some of the advantages of using the fuzzy theory for FCA scaling include:

- the partitioning assume linguistic values, which are easily interpretable by humans, such as *young, old, tall, short*, etc. This way, the formal concepts extracted will convey this highly desirable interpretability characteristic;
- unnatural divisions are avoided, such as the division in Table 3 for the 30 year-old people that are borderlines but must be place in only one category. The fuzzy logic easily avoids this problem with the use of the membership degrees that allow one object to belong to different categories with different degrees of membership;
- it is a natural choice when the interest in FCA is for the extraction of fuzzy rules, since these rules will reflect the fuzzy data base, also used by the inference mechanism of the induced classifier.

4.1. The Extraction of Formal Concepts Using the NextNeighbor Algorithm

For the automatic extraction of formal concepts, several algorithms have been proposed in the literature (Ganter, 2002; Krajca et al., 2010). In the experiments presented in this work, we used the NextNeighbor approach (Ganter, 2002), which optimizes the automatic extraction of formal concepts by finding neighboring concepts. Next, we explain this approach in details based on the author's notes (Ganter, 2006).

As stated previously, for a formal context (G, M, I) and given $A \subseteq G$, we define:

$$A' = \{m \in M | (g, m) \in I \ \forall \ g \in A\} \tag{3}$$

Similarly, given a set of attributes $B \subseteq M$, the shared image of B in G is:

$$B' = \{g \in G | (g, m) \in I \ \forall \ m \in B\} \tag{4}$$

These two operators, A' and B', are the ***derivation operators*** for (G, M, I). The derivation operators can be combined so that, starting with a set $A \subseteq G$, we obtain that A' is a subset of M. By applying the second operator on this set, we get $(A')'$, or A'' for short, both constituting sets of

objects. It is possible to continue with the process, obtaining A''', A'''',and so on. But these sets are not all different, as shown next.

Proposition 4.1. *For subsets A, $A_1, A_2 \subseteq G$, we have:*

1. $A_1 \subseteq A_2 \Rightarrow A_2' \subseteq A_1'$
2. $A \subseteq A''$
3. $A' \subseteq A'''$

Dually, for subsets B, $B_1, B_2 \subseteq M$, we have:

1. $B_1 \subseteq B_2 \Rightarrow B_2' \subseteq B_1'$
2. $B \subseteq B''$
3. $B' \subseteq B'''$

By combining the derivation operators, we obtain two operators of the form $X \Rightarrow X''$, one on G, the other on M. For $A \subseteq G$, we have that $A'' \subseteq G$. The set A'' is called the ***extent closure*** of A. Dually, when $B \subseteq M$, then also $B'' \subseteq M$, and B'' is called the ***intent closure*** of B. Note that A'' and B'' have a clear meaning for the data:

- Whenever all objects from a set $A \subseteq G$ have a common attribute m, then also all objects from A'' have that attribute.
- Whenever an object $g \in G$ has all attributes from $B \subseteq M$, then this object also has all attributes from B''.

Some properties of these operators are listed in the next proposition. They are simple consequences of Proposition 4.1.

Proposition 4.2. *For subsets A, $A_1, A_2 \subseteq G$, we have:*

1. $A_1 \subseteq A_2 \Rightarrow A_1'' \subseteq A_2''$
2. $A \subseteq A''$
3. $(A'')'' \subseteq A''$

Dually, for subsets B, $B_1, B_2 \subseteq M$, we have:

1. $B_1 \subseteq B_2 \Rightarrow B_1'' \subseteq B_2''$
2. $B \subseteq B''$
3. $(B'')'' \subseteq B''$

The technical term for operators satisfying the properties given in Proposition 4.2 is ***closure operators***. The sets which are images of a closure operator are the ***closed sets***. This way, in the case of a closure operator $X \Rightarrow X''$, the closed sets are the sets of the form X''.

Proposition 4.3. *If (G, M, I) is a formal context and $A \subseteq G$, then A'' is an extent. Conversely, if A is an extent of (G, M, I), then $A = A''$. Dually if $B \subseteq M$, then B'' is an intent, and every intent B satisfies $B'' = B$.*

Preposition 4.3 follows from the fact that for each subset $A \subseteq G$, the pair (A'', A') is a formal concept, and that similarly (B'', B') is a concept whenever $B \subseteq M$.

This way, the closed sets of the closure operator $A \Rightarrow A'', A \subseteq G$ are precisely the extents of (G, M, I), and the closed sets of the operator $B \Rightarrow B'', B \subseteq M$, are precisely the intents.

In order to understand the NextNeighbor algorithm, also called NextClosure, let us consider that a base set M has an arbitrary linear order:

$$M = \{m_1 < m_2 < ... < m_n\}$$

Every subset $S \subseteq M$ can conveniently be described by a row of crosses and blanks. For example, if $M = \{a < b < c < d < e < f < g\}$, the subset $S = \{a, c, d, f\}$, can be written as shown in Figure 3:

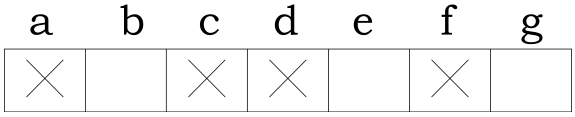


Figure 3. Vector representation of subset S.

Using this notation it is easy to visualize if a given set is a subset of another given set.

It is now possible to define the ***lectic order***, *i.e.*, a linear order of the subsets of M, as follows.

Definition 4.1. *Let $A, B \subseteq M$ be two distinct subsets. We say that A is **lectically smaller** than B, if the smallest element in which A and B differ belongs to B. Formally*

$$A < B \Leftrightarrow \exists_i (i \in B, i \notin A, \forall_{j<i} (j \in A \Leftrightarrow j \in B)).$$

For example, $\{a, c, e, f\} < \{a, c, d, f\}$, because the smallest element in which the two sets differ is d, and this element belongs to the larger set. Figure 4 shows the vector representation of the previous sets.

It is important to notice that the lectic order extends the subset-order, *i.e.*,

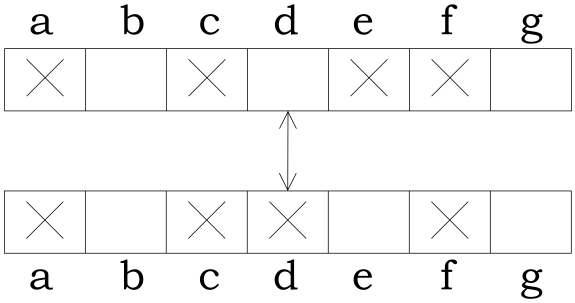


Figure 4. Vector representation of sets $\{a, c, e, f\}$ and $\{a, c, d, f\}$.

$$A \subseteq B \rightarrow A \leq B.$$

To lectically compare two sets, the following notation is helpful:

$$A <_i B \rightarrow (i \in B, i \notin A, \forall_{j<i}(j \in A \Leftrightarrow j \in B)).$$

The previous equation can be read as: $A <_i B$ if and only if i is the smallest element in which A and B differ, and $i \in B$.

Proposition 4.4. *1. $A < B$ if and only if $A <_i B$ for some $i \in M$.*
2. If $A <_i B$ and $A <_j C$ with $i < j$, then $C <_i B$.

For the task of finding closures, and considering the closure operator $A \rightarrow A''$ on the base set M, to each subset $A \subseteq M$ its closure is defined as $A'' \subseteq M$. In order to find the list of all these closures, the naive algorithm is "for each $A \subseteq M$, check if the resulting formal concept is already listed", which requires an exponential number of lookups in a list that may have exponential size. A better idea is to generate the closures in lectic order.

Following, we demonstrate how to compute, given a closed set, the lectically next closed set. Using this approach, no lookups in the list of found solutions is necessary. In fact, it is not even necessary to store the list of solutions. For many applications, it will suffice to generate the list of elements on demand. Therefore, we do not have to store exponentially many closed sets. Instead, only one closed set, the current one, is stored.

To find the next closure we define, for $A \subseteq M$ and $m_i \in M$:

$A \oplus m_i = ((A \cap \{m_1, ..., m_{i-1}\}) \cup m_i)''$.

For example, let $A = \{a, c, d, f\}$ and $m_i = e$. We first remove all elements that are greater or equal to m_i from A, then we insert m_i, and form the closure. These steps are shown in Figure 5.

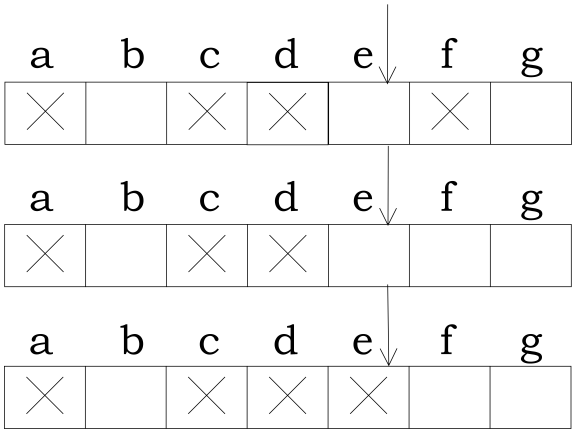


Figure 5. Finding the next closure.

Proposition 4.5. *1. If $i \notin A$ then $A < A \oplus i$.*

2. If B is closed and $A <_i B$ then $A \oplus i \subseteq B$, in particular, $A \oplus i \leq B$.

3. If B is closed and $A <_i B$ then $A <_i A \oplus i$.

Using the statements of Proposition 4.5, it is easy to characterize the "next"closed set:

Theorem 4.1. *The smallest closed set larger than a given set $A \subset M$, with respect to the lectic order, is:*

$$A \oplus i$$

with i being the largest element of M with $A <_i A \oplus i$.

Algorithms 1, 2, and 3 implement the steps of computing closed sets to extract formal concepts.

Input : A closure operator $X \to X''$ on a finite set M.
Output : All closed sets in lectic order.
First_Closure;
repeat
  Output A;
  Next_Closure;
until *not success* ;

Algorithm 1: All Closures Algorithm.

Notice that the so-called lectic order is simply the lexicographic order of the formal concepts (Distel, 2010). Algorithm 1 basically repeatedly evokes the formal concept extraction method, *i.e.*, the Next_Closure method, until no more formal concepts can be found.

The First_Closure method simply returns an empty set representing the union of all attributes.

```
Input  : A closure operator X → X'' on a finite set M.
Output : The closure A of the empty set.
begin
    A := ∅;
end
```

Algorithm 2: First Closure Algorithm.

4.2. Adaptation of the NextClosure Algorithm for Formal Concept Extraction

As stated earlier, the algorithm we adopted was the NextClosure whose details are described in (Ganter, 2002), and presented in Algorithm 1, Algorith 2, and Algorithm 3. In order to illustrate how to modify this algorithm for our purpose, the modified version of its core is presented in Algorithm 4.

Notice that the so-called lectic order is simply the lexicographic order of the formal concepts (Distel, 2010). The First_Closure method simply returns an empty set representing the union of all attributes. This way, Algorithm 1 basically repeatedly evokes the formal concept extraction method, *i.e.*, the Next_Closure method, until no more formal concepts can be found. The Next_Closure method is presented next (Algorithm 3).

```
     Input  : A closure operator X → X'' on a finite set M, and a subset
              A ⊆ B.
     Output : A is replaced by the lectically next closed set.
3.1  i := largest element of M;
3.2  success := false;
3.3  repeat
3.4      if i ∉ A then
3.5          A := A ∩ {1, 2, ..., (i − 1)} ∪ {i};
3.6          B := A'';
3.7          if (B − A) contains no element < i then then
3.8              A := B (A is an extracted concept);
3.9              success := true;
3.10         end
3.11         else
3.12             A := A−{i};
3.13         end
3.14
3.15     end
3.16     i :=pred(i);
3.17 until success or i = smallest element of M ;
```

Algorithm 3: Next Closure Algorithm.

This algorithm can be used to extract formal concepts by analyzing

the attributes or the objects. When it uses the attributes, it is called *Next Intent*, and it is called *Next Extent* when used with the objects. This way, the word *element* is used instead of attributes or objects.

Algorithm 4 presents the modified version of the NextClosure algorithm to extract formal concepts including a class. The process is quite direct: once the algorithm looks for neighboring concepts, we just need to verify whether a found concept has a class or not. If it does, then this formal concept can be stored in a data structure including the extracted concepts, otherwise, the process discards the found formal concept and carries on looking for the next neighboring concept. This modification is represented by line 4.9 of Algorithm 4.

```
      Input  : A closure operator X → X'' on a finite set M, and a
               subset A ⊆ B.
      Output : A is replaced by the lectically next closed set.
4.1   i := largest element of M;
4.2   success := false;
4.3   repeat
4.4       if i ∉ A then
4.5           A := A ∩ {1, 2, ..., (i − 1)} ∪ {i};
4.6           B := A'';
4.7           if (B − A) contains no element < i then
4.8               if B contains a class then
4.9                   A := B (A is an extracted concept);
4.10              end
4.11              success := true;
4.12          end
4.13          else
4.14              A := A− {i};
4.15          end
4.16
4.17      end
4.18      i :=pred(i);
4.19  until success or (i > (numElements - numClasses)) ;
```

Algorithm 4: Adaptation of the Next Closure Algorithm.

Another interesting possibility for the restriction of formal concepts is the evaluation of their support, which, for a formal context, is just the number of objects a given formal concept shares divided by the number of all objects. This way, considering the balancing of the classes, formal concepts with low support could be discarded, reducing the number of extracted formal concepts.

4.3. Proposal for Extracting Fuzzy Rules from a Formal Context

As discussed in the previous sections, a formal context is the base for the extraction of formal concepts. These formal concepts can be seen as associations between attributes based on the existence of objects sharing these attributes. It is also important to notice that in the process of extracting formal concepts, their support is calculated automatically.

Our proposal is based on the fuzzy definition of a problem in an attribute $\times$ value table to create a formal context and then obtain the classification rules. Considering a general dataset for classification purposes with N examples and M attributes, the fuzzy scaling procedure of our proposal is performed by the following steps:

1. Define the fuzzy database, *i.e.*, the partitions that will define the fuzzification of the continuous attributes;
2. Create a binary attribute defined by each fuzzy set of each continuous attribute and each value of each discrete attribute;
3. Calculate the membership degree of the input values for each example in each binary attribute (notice that this step is only required for continuous attributes, once discrete ones will be automatically defined as true of false);
4. Define a minimum value A_{min} to guide the scaling of the real values so that if the membership degree of a certain value for a particular fuzzy set is equal or higher than A_{min}, the corresponding attribute is set to true in the formal context;
5. Use an algorithm to extract formal concepts to extract all existing classification rules from the formal context;
6. Define a minimum support value to select a subset of the extracted rules;

The final rule set can then be used as the search space of a GA to generate a fuzzy system, according to the genetic rule selection with a priori rule extraction approach, described in Section 2.

In order to reduce the number of possible formal concepts, the fuzzy sets defining each attribute can be evenly distributed in the partition, so the maximum possible membership degree in the intersections is 0.5. This way, if A_{min} is set to 0.5, for each original attribute only one of its binary attributes will be activated. Notice that ties must also be handled, thus, in our implementation we used a random variable to activate one of 2 tied fuzzy sets.

An important issue regarding the extraction of the formal concepts when using a binary fuzzification of the formal context is related to the increase in the number of attributes. This total number of attributes will be equal to the sum of all fuzzy sets describing each attribute, each value of each discrete attribute, and the number of classes. This increase in the number of attributes leads to an increase in the total number of formal concepts that can be extracted. Nevertheless, it is important to notice that for our purpose, the total number of formal concepts is much larger than the number of formal concepts we are interested in extracting: since we want to extract classification rules, we are only interested in extracting formal concepts that have a class in their intention.

Regarding the total number of formal concepts existing in a formal context, this number can be estimated using the Metropolis-Hastings algorithm for sampling formal concepts described by Boley et al. (2010).

The process of extracting formal concepts and, consequently, classification rules, can be done by adapting any algorithm for the extraction of formal concepts. In our experiments we used the NextClosure algorithm described by Ganter (2002), as presented in the previous section. Since the NextClosure can be used to extract formal concepts by analyzing either the attributes or the objects, when it uses the attributes, it is called *Next Intent*, and it is called *Next Extent* when it uses the objects.

Next, we present preliminary experiments.

5. Preliminary Experiments

Our main goals when carrying on the experiments were threefold:

1. to verify whether the resulting rule set would contain a suitable number of rules to form the search space of a GA;
2. to evaluate the time taken to extract the formal concepts;
3. to evaluate the idea of using the support of the rules for a preselecting process.

Next, we present the experiments carried out, discuss the results, and present additional experiments on attribute exploration in order to reduce the number of extracted rules.

5.1. Experiments and Results

To evaluate the proposed modification of the NextClosure algorithm we used 10 datasets from the UCI - Machine Learning Repository (Frank

and Asuncion, 2010) in order to analyze the number of concepts. Table 4 summarizes the dataset characteristics giving the total number of examples (Examples); total number of features (Features), including the number of continuous and discrete features in brackets; number of classes (Classes); the majority error (ME); and the number of fuzzy sets for each of the attributes (FS) predefined using the Fuzzy–DBD algorithm (Cintra et al., 2009). Examples with missing values were removed from the datasets.

Table 4. General characteristics of the datasets.

Dataset	Examples	Features	Classes	ME	FS
Credit	653	15 (6 9)	2	45.33	2
Cylinder	277	32 (19 13)	2	35.74	2
Dermatology	358	34 (33 1)	6	68.99	2
Diabetes	769	8 (8 0)	2	34.90	2
Glass	220	9 (9 0)	7	65.46	7
Heart	270	13 (13 0)	2	44.44	2
Ionosphere	351	34 (34 0)	2	35.90	3
Iris	150	4 (4 0)	3	66.60	3
Vehicle	846	18 1(8 0)	4	74.23	2
Wine	178	13 (13 0)	3	59.74	3

Table 5 presents the total number of formal concepts (TNFC), the total number of formal concepts with a class (FCwC), *i.e.*, the total number of classification rules, and the percentage it represents of the total number of formal concepts. In order to allow further comparisons, Table 5 also presents the total number of formal concepts for 50%, 20%, 10%, and 5% support values and the percentage they represent of the total number of formal concepts with a class (FCwC). The bigger the support, the smaller the number of rules. Support values greater than 50% were not considered as the number of rules was not enough to form the search space of a GA.

Table 5. Extracted Formal Concepts Information

Dataset	TNFC	FCwC	50%	20%	10%	5%
Credit	20,083	9,843 49.01	4 0.04	825 8.38	2,410 24.48	4,235 43.03
Cylinder	7,041,110	1,944,271 27.61	884 0.05	113,474 5.84	546,265 28.10	1,236,470 63.60
Dermatology	21,896,570	312,177 1.43	0 0.00	553 0.18	38,141 12.22	140,502 45.01
Diabetes	2,172	1,279 58.89	32 2.50	308 24.08	669 52.31	758 59.27
Glass	4,054	1,863 45.95	0 0.00	38 2.04	301 16.16	670 35.96
Heart	81,935	36,942 45.09	9 0.02	1,648 4.46	8,342 22.58	19,854 53.74
Ionosphere	102,641,179	2,076,229 2.02	12 0.00	1,187,827 57.21	1,197,307 57.67	1,649,406 79.44
Iris	93	65 69.89	0 0.00	11 16.92	27 41.54	41 63.08
Segmentation	10,785	1,437 13.32	0 0.00	0 0.00	162 11.27	54 3.76
Vehicle	86,918	28,979 33.34	0 0.00	91 0.31	1,780 6.14	6,625 22.86
Wine	21,000	9,802 46.68	0 0.00	423 4.32	3,338 34.05	6,676 68.11

Regarding our first goal (the suitability of the rule set extracted in terms of the number of rules), we verified that an appropriate number of rules was extracted (FCwC). Our verification takes into consideration an estimated number of rules to populate 50 chromosomes, which was the total population used in previous experiments with the DoC-Based algorithm (Cintra et al., 2007). Considering that the smallest rule set was obtained with the Iris dataset (65 rules), for this specific dataset, the total number of rules in each chromosome must be carefully defined in order to avoid lack of diversity in the population. The number of rules obtained for the remaining datasets, except Iris, can be considered sufficient to populate a genetic search space.

Table 6 shows the time taken, in minutes, for the extraction of the formal concepts (column TNFC of Table 5) and calculation of their support. The process was executed in an *Intel*® Core™2 Duo T7250 (2.00GHz, 2MB L2 Cache, 800MHz FSB) machine. The time taken to extract the rules can also be considered appropriate for this approach to be used together with a genetic algorithm search process. The whole process took a matter of seconds to finish for all datasets but Cylinder, Dermatology, and Ionosphere, due to the total number of attributes and examples for these databases. It is also important to notice that our experiments were carried out with the NextClosure algorithm (Ganter, 2002), although a faster algorithm was recently proposed which has a parallel search process (Krajca et al., 2010).

Table 6. Time (in minutes) taken to extract all formal concepts (column TNFC).

Dataset	Time	Dataset	Time
Credit	0.50	Heart	0.55
Cylinder	126.00	Ionosphere	144.63
Dermatology	168.73	Iris	0.02
Diabetes	0.05	Vehicle	1.90
Glass	0.12	Wine	0.18

If the support is taken into consideration for the selection of formal concepts including a class, it is possible to reduce even more the number of extracted formal concepts, giving the user more flexibility to decide on the number of extracted rules. The only issue one has to bear in mind when using the support in order to reduce the number of extracted rules is related to classes with few examples, which, in turn, will result in relative low support. In fact, as stated previously, in (Ishibuchi and Yamamoto, 2004), the authors use the support, confidence, and the

product of these two measures to select rules. These measures will be investigated in future work.

Next, we present a discussion on the number of rules to be extracted and an approach we worked on to reduce the number of extracted rules.

5.2. Additional Experiments on Attribute Exploration to Reduce the Number of Extracted Rules

Although the proposal for fuzzy classification rule extraction from formal contexts extracts a limited number of rules and there is the possibility of using the support value of the formal concepts to further reduce the total number of extracted rules, we decided to investigate an alternative to reduce even more the number of possible extracted rules related to the various possibilities lattices provide.

The idea came from a visual analysis of resulting lattices. By visually analyzing a lattice and the set of extracted rules, we could observe that some attributes (notice that an attribute in the formal context is defined by a single fuzzy set) were related to a single class. Thus, we started analyzing the idea that if an attribute was only related to a single class, it would be possible to generate a rule from this relation and prevent this attribute from being used in other rules. Similarly, if two or more attributes were related to a single class, they could be used to generate a single rule and would not be used in any other rule generated from that moment on.

This removal of attributes from the search space is easily understood by a simple analysis of the lattice. In order to illustrate this idea, consider the Iris dataset which contains 4 continuous attributes (petal length, petal width, sepal length, and sepal width) and 3 classes (iris virginica, iris versicolor, iris setosa). Figure 6 presents the resulting lattice for this domain. Each attribute was defined in fuzzy terms by 3 fuzzy sets, named *1_Low*, *1_Med* (medium), *1_High*,*2_Low*, *2_Med*, *2_High*,*3_Low*, *3_Med*, *3_High*, *4_Low*, *4_Med*, and *4_High*, where 1, 2, 3, and 4 represent the sequence number of the original attribute. We used the ConExp software[1] to generate the lattices and proceed experimenting with the removal of attributes.

[1]More information on this free licensed software can be found on http://conexp.sourceforge.net/

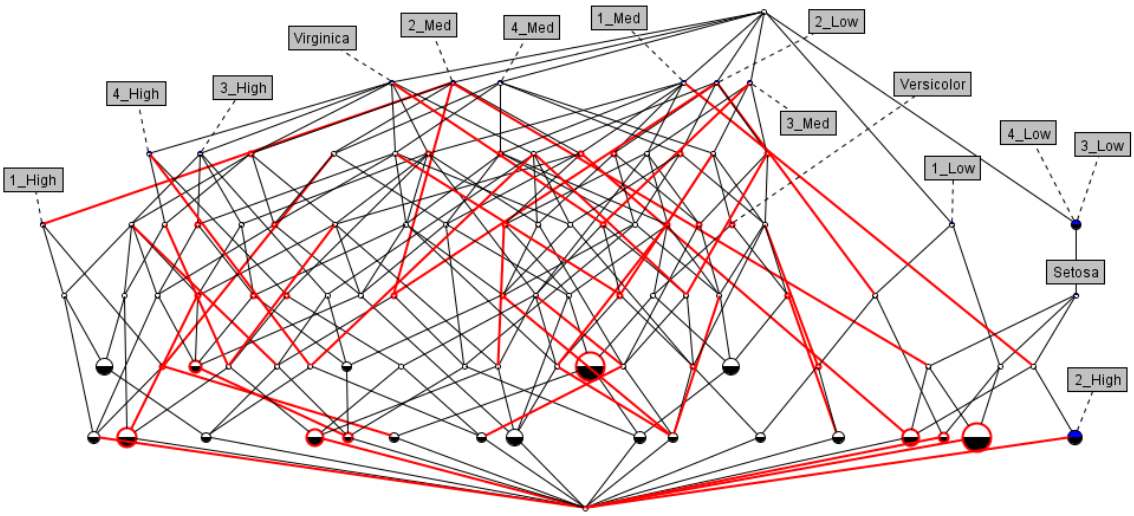


Figure 6. Complete lattice for Iris dataset.

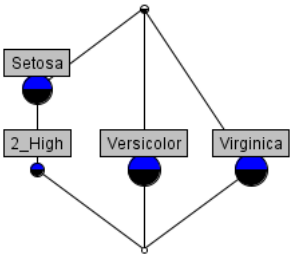
Although the total lattice is messy and confusing, the Conexp software allows the exploration of associations among classes and attributes by presenting an updated lattice by simply removing attributes (using a check box). By leaving single attributes and the classes, interesting associations can be found, such as the ones presented in Figure 7.

Observe that for each figure in Figure 7, one class is connected to one attribute, the remaining two classes have no connections. This happens because they are not related to the only attribute selected to build the lattice.

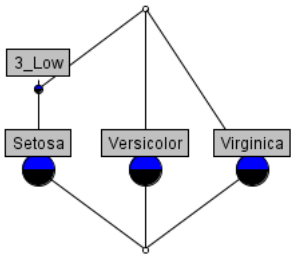
Figure 7 presents 4 attributes, each related to only one class. However, for some particular cases, one class might be related to two or more attributes. This case is shown in Figure 8, where attributes *2_High* and *4_Low* are only related to class Iris Setosa.

In order to find associations of attributes and classes, considering a dataset with M attributes, the following algorithm was proposed:

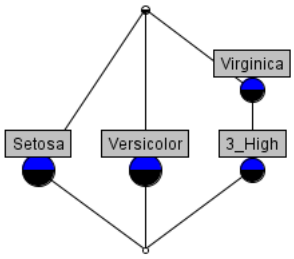
1. Generate the lattice using each single attribute alone and check if an association between a single attribute and class was found; if so, create a rule using this attribute and class, and remove the attribute from the set of available attributes so it cannot be used in future lattices;
2. Repeatedly generate lattices using combinations of 2, 3,..., $M-1$ attributes and check if associations among the set of attributes and a single class is found; if so, create a rule using the attribute set



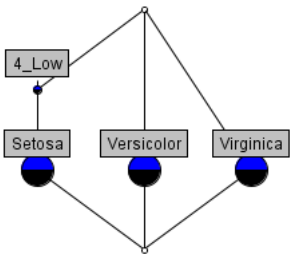
(a) Lattice for attribute *2_High*.



(b) Lattice for attribute *3_Low*.



(c) Lattice for attribute *3_High*.



(d) Lattice for attribute *4_Low*.

Figure 7. Latices of attributes *2_High* (a), *3_Low* (b), *3_High* (c), and *4_Low* (d).

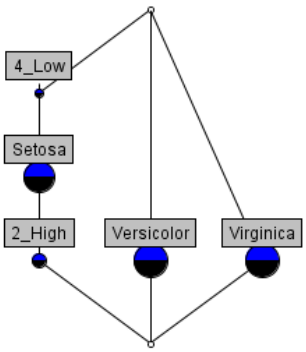


Figure 8. Attributes 2_High and 4_Low, related to class Iris Setosa.

and the class, and remove the attribute set from the set of available attributes.

Although the cost of combining all possible attributes is high, we expect that, in practice, the total number of combinations can be rapidly

reduced, due to the fact that attributes which participate in rules are discarded from future combinations.

In order to evaluate this idea, preliminary experiments were carried out using the Iris and Diabetes datasets to further analyzing this idea. However, results with the Iris and Diabetes datasets showed that the final sets of rules extracted were quite small and would not suit a genetic process. It is important to notice that these two datasets, Iris and Diabetes, are described by 4 and 8 attributes, which can be considered a small number. This way, we intend to use other datasets to evaluate this idea as a means of extracting classification rules and building classifiers, in future work.

Next, we present the conclusions and future work.

6. Conclusions and Future Work

The field of genetic fuzzy systems has produced some promising results in the area of fuzzy systems and many new approaches have been proposed in the literature. Specifically for the genetic definition of the rule base, a possible approach is the genetic rule selection, with the previous definition of the fuzzy database and the generation of fuzzy rules to form the search space of the genetic selection process. In this work we presented some approaches found in the literature for this task of forming the search space of a genetic algorithm.

We also presented a new proposal and preliminary experiments and results on the use of formal concept analysis for this task. The use of formal concept analysis can be considered a new area of research, with applications in various domains and with an increasing interest due to its visual benefits and powerful mathematical basis.

Preliminary results show that our proposal is suitable for the task of forming the search space of a genetic algorithm in terms of the number of rules extracted, processing time, and also the use of the support measure to preselect a reduced number of rules if necessary.

As future work, we intend to adopt the DoC-Based method, proposed in (Cintra et al., 2007; Cintra and Camargo, 2007), to generate fuzzy rule bases using the proposal presented here to form the search space. This method includes the number of rules and the accuracy of the rule base in the fitness calculation in order to induce a rule base with

high accuracy and interpretability rates. We intend to compare the formal concept extraction time taken by the NextClosure algorithm (Ganter, 2002) and the parallel approach proposed by Krajca et al. (2010). We also intend to empirically evaluate the use of the support and confidence measures, and the product of them, to select rules.

References

Agrawal, R. and Srikant, R. (1994). Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, pages 487–499, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc. Cited in page 6.

Belohlávek, R., Sklenar, V., and Zacpal, J. (2005). Crisply generated fuzzy concepts. In *International Conference on Formal Concept Analysis - ICFCA*, pages 269–284. Cited in page 2.

Boley, M., Grosskreutz, H., and Gärtner, T. (2010). Formal concept sampling for counting and threshold–free local pattern mining. In *Proceedings of the SIAM Int. Conf. on Data Mining (SDM 2010)*, pages 177–188. Cited in page 19.

Cintra, M. E. and Camargo, H. A. (2007). Fuzzy rules generation using genetic algorithms with self-adaptive selection. *Proceedings of the IEEE International Conference on Information Reuse and Integration - IRI*, pages 261–266. Cited in pages 3 and 25.

Cintra, M. E., Camargo, H. A., and Martin, T. P. (2009). Optimising the fuzzy granulation of attribute domains. *Proceedings of the International Fuzzy Systems Association World Conference*, pages 742–747. Cited in page 20.

Cintra, M. E., Camargo, H. A., R. Hruschka Jr, E., and Nicoletti, M. C. (2007). Fuzzy rule base generation through genetic algorithms and Bayesian classifiers - a comparative approach. *Proceedings of the International Conference on Intelligent Systems Design and Application - ISDA*, 1:315–320. Cited in pages 1, 5, 21, and 25.

Cordon, O., Gomide, F., Herrera, F., Hoffmann, F., and Magdalena, L. (2004). Special issue on genetic fuzzy systems. Cited in page 1.

Cordon, O., Herrera, F., and Villar, P. (2001). Generating the knowledge base of a fuzzy-based system by the genetic learning of the data base. *IEEE Transactions on Fuzzy Systems*, 9(4):667–674. Cited in page 4.

Distel, F. (2010). Hardness of enumerating pseudo-intents in the lectic

order. In *International Conference on Forma Concept Analysis - ICFCA*, pages 124–137. Cited in pages 15 and 16.

Dumitrescu, D., Lazzerini, B., and Jair, L. (2000). *Fuzzy Sets and Their Application to Clustering and Training*. Int. Series on Computational Intelligence. CBC Press. Cited in page 1.

Frank, A. and Asuncion, A. (2010). UCI machine learning repository. Cited in pages 1 and 19.

Ganter, B. (1984). Two basic algorithms in concept analysis. Technical report, Technische Hochschule Darmstadt. 28 p. Cited in page 2.

Ganter, B. (2002). Formal concept analysis: Algorithmic aspects. Technical report, Technische Universitat Dresden. Cited in pages 2, 11, 16, 19, 21, and 26.

Ganter, B. (2006). Finger exercises in formal concept analysis - Dresden ICCL Summer School. Cited in page 11.

Herrera, F. (2008). Genetic fuzzy systems: taxonomy, current research trends and prospects. *Evolutionary Intelligence*, 1(1):27–46. Cited in pages 1, 3, and 4.

Homaifar, A. and McCormick (1995). Simultaneous design of membership functions and rule sets for fuzzy controller using genetic algorithms. *IEEE Transactions on. Fuzzy Systems*, 3(2):129–139. Cited in page 4.

Hu, Y.-C., Chen, R.-S., and Tzeng, G.-H. (2002). Mining fuzzy association rules for classification problems. *Computing and Industrial Engineering*, 43:735–750. Cited in page 6.

Ishibuchi, H., Kuwajima, I., and Nojima, Y. (2006). Multiobjective association rule mining. In *Proceedings of the Workshop on Multiobjective Problem Solving from Nature - PPSN*, pages 12–24. Cited in page 7.

Ishibuchi, H. and Murata, T. (1995). Selecting linguistic classification rules by two-objective genetic algorithms. In *Proceedings of the 1995 IEEE Int. Conf. on Systems, Man and Cybernetics*, pages 1410–1415. Cited in page 5.

Ishibuchi, H. and Yamamoto, T. (2004). Fuzzy rule selection by multi-objective genetic local search algorithms and rule evaluation measures in data mining. *Fuzzy Sets and Systems*, 141:59–88. Cited in pages 1, 5, and 21.

Krajca, P., Outrata, J., and Vychodil, V. (2010). Parallel algorithm for computing fixpoints of galois connections. *Annals of Mathematics and Artificial Intelligence*, 59:257–272. Cited in pages 2, 11, 21, and 26.

Pach, F. P., Gyenesei, A., and Abonyi, J. (2008). Compact fuzzy associa-

tion rule-based classifier. *Expert Syst. Appl.*, pages 2406–2416. Cited in page 6.

Pedrycz, W. (1996). *Fuzzy Modelling: Paradigms and Practice*. Kluwer Academic Press. Cited in page 1.

Sigmund, E., Zacpal, J., and Sigmundová, D. (2010). The application of formal concept analysis and the importance of scale selection in the evaluation of physical activity data in relation to the body mass index. *Acta Universitatis Palackianae Olomucensis. Gymnica*, 39(4):41–51. Cited in page 2.

Thrift, P. (1991). Fuzzy logic synthesis with genetic algorithms. In *Proceedings of the 4th International Conference on Genetic Algorithms*, pages 509–513. Morgan Kaufmann. Cited in page 3.

Wille, R. (1982). Restructuring lattice theory: An approach based on hierarchies of concepts. In *Ordered Setsand in I. Rivals (Ed.)*, volume 23, pages 445–470. Cited in pages 2, 7, and 9.

Wolff, K. E. (2002). Concepts in fuzzy scaling theory: order and granularity. *Fuzzy Sets and Systems*, 132(1):63–75. Cited in pages 2 and 11.

Zheng, S., Zhou, Y., and Martin, T. (2009). A new method for fuzzy formal concept analysis. In *Proceedings of the International Joint Conferences on Web Intelligence and Intelligent Agent Technologies*, pages 405–408. Cited in page 2.