Attribute classification and reduct computation in multi-adjoint concept lattices

Lubomir Antoni, M. Eugenia Cornejo, Jesús Medina, Eloísa Ramírez-Poussa

***Abstract*—The problem of reducing information in databases is an important topic in Formal Concept Analysis, which has been studied in several papers. In this work, we consider the fuzzy environment of the multi-adjoint concept lattices, since it is a general fuzzy framework that allows us to easily establish degrees of preference on the elements of the considered database. We introduce algorithms to discover the information contained in the relational system. By means of these algorithms, we classify the attributes of a multi-adjoint context, and build a minimal subset of attributes preserving the information of the original knowledge system.**

***Index Terms*—Fuzzy sets, formal concept analysis, concept lattice reduction**

I. Introduction

Different areas such as software engineering [1], information retrieval [2], [3], data mining [4], [5], knowledge discovery [6], [7], [8], machine learning [9], [10] and fuzzy rough sets theory [11], [12], [13], [14] handle databases with helpful information for a multitude of real applications. The treatment of such databases could be a complex task, because they often include data that provide redundant information what makes difficult the collection of relevant information in which the previous areas are interested in. Therefore, it has become indispensable to use tools that allow to process and extract information from such databases. Several techniques have been proposed to address the challenging tasks involving many irrelevant and redundant data. Attribute selection has become the focus for applications which include the acquisition of spacial fuzzy decision rules to analyze social and environmental causes of neural tube birth defects [15] or the reductions of systems of fuzzy relation equations to simplify the computation of their solutions [16].

Formal Concept Analysis (FCA) [17] is one of the branch of mathematics that pursues to obtain knowledge from relational databases. Specifically, FCA organizes the information contained in databases by means of a mathematical structure called concept lattice. At the computational level, databases containing data that does not provide extra information only hinders the construction of the concept lattice. Consequently, such data should be deleted or obviated, as long as the main information in the considered databases is preserved. Attribute reduction is a fundamental part in FCA in charge of selecting the main information and removing the unnecessary and redundant one.

There exists a large amount of papers on this topic. Some works deal with this problem by means of the use of techniques from rough set theory. Liu et al study the reduction of the concept lattices based on rough set theory and propose two kinds of reduction methods for the concept lattices in [18]. First, they present the sufficient and necessary conditions for justifying whether an attribute and an object are dispensable or indispensable in the concept lattices. Based on the justifying conditions, they propose a kind of multi-step attribute reduction method and object reduction method for the concept lattices, respectively. In [19], the authors propose the use of notion of reduct considered in rough sets theory to reduce formal context. Li [20] focuses on attribute reduction of formal concepts via covering rough set theory. Shao and Zhang [21] present algorithms for attribute and object reduction in concept lattices. Attribute reduction methods based on discernibility matrices are introduced by Zhang et al. [22] and improved by Qi [23]. Dias and Vieira [24] present a survey on concept lattices reduction.

The current papers in the literature are proposed in restrictive frameworks, such as in the classical - boolean - or one-sided formal contexts, using thresholds, etc., or they are focused on the reduction of the whole concept lattice (once it is computed) instead of the set of attributes or objects.

This paper considers the multi-adjoint concept lattice environment [25], [26] which provides a more flexible framework that to allows us apply this tool to a wider range of situations, as well as establishing preferences on the elements of the considered database. To know in more detail how the multi-adjoint framework allows to assign preferences on the elements of the database to be considered, readers are referred to [26]. In particular, we are going to consider the characterization of meet-irreducible elements of a concept lattice and attribute classification theorems given in [27], [28], in order to obtain algorithms to compute reducts. Reducts are minimal subsets of attributes containing the main information, that is, subsets from which we can build a concept lattice isomorphic to the original one. It is important to emphasize that we cannot remove any attribute from a reduct because in such case the isomorphism is not preserved. In addition, more than one reduct can be obtained from a multi-adjoint context, and these different reducts can have different cardinalities, as it was proven in [28].

L. Antoni is with the Institute of Computer Science, Faculty of Science, P. J. Safarik University in Kosice (email: lubomir.antoni@upjs.sk), and he is partially supported by the Slovak Research and Development Agency under the contract No. APVV-15-0091; M. E. Cornejo, J. Medina and E. Ramírez-Poussa are with Dept. Matemáticas, Univ. de Cádiz, Facultad de Ciencias, Campus Universitario de Puerto Real 11510 Puerto Real, Cádiz, Spain (email: {mariaeugenia.cornejo,jesus.medina,eloisa.ramirez}@uca.es), and they are partially supported by the Spanish Economy and Competitiveness Ministry (MINECO) project TIN2016-76653-P, and by the European Cooperation in Science & Technology (COST) Action CA17124.

Corresponding author: E. Ramírez-Poussa, eloisa.ramirez@uca.es

In this paper, based on the strength and robustness of theoretical results, we introduce several algorithms to classify the set of attributes of a multi-adjoint context. From this classification, another algorithm to obtain a reduct of the multi-adjoint context is presented, which follows the philosophy of the linear QUICKREDUCT II algorithm given in [29], removing attributes that are not necessary from the original database. A formal proof of this algorithm is also introduced. The same study and algorithms work when the set of objects needs to be reduced, to do that we only need to interchange both sets, the set of attributes and the set of objects.

Furthermore, we present a comparison between our algorithms and other algorithms given in the literature which also chase the same goal. We will see that the attribute reduction algorithms presented in this paper consider the most flexible environment. As a consequence, this work brings novelty and original advances on this research topic.

The paper is organized in the following manner: Section II includes a brief summary with preliminary notions about multi-adjoint concept lattices as well as different notions and results corresponding to attribute reduction in this environment. Section III includes the algorithms to obtain the classification of the set of attributes of a context. An algorithm to compute reducts and some useful properties are presented in Section IV. A useful example has been included to illustrate all the algorithms introduced in this work. In Section V a comparative theoretical study among our algorithms and other related works is shown. We finish our study showing some conclusions and prospects for future works in Section VI.

II. Multi-adjoint concept lattices and attribute classification

Multi-adjoint concept lattices framework uses as calculation operators an interesting generalization of triangular norms and their residuated implications [30], which are called adjoint triples [31], [32].

Definition 1: Let $(P_1, \leq_1)$, $(P_2, \leq_2)$, $(P_3, \leq_3)$ be posets and $\&\colon P_1 \times P_2 \to P_3$, $\swarrow\colon P_3 \times P_2 \to P_1$, $\nwarrow\colon P_3 \times P_1 \to P_2$ be mappings, then $(\&, \swarrow, \nwarrow)$ is an *adjoint triple* with respect to P_1, P_2, P_3 if:

$$x \leq_1 z \swarrow y \quad \text{iff} \quad x \,\&\, y \leq_3 z \quad \text{iff} \quad y \leq_2 z \nwarrow x \qquad (1)$$

where $x \in P_1$, $y \in P_2$ and $z \in P_3$. The condition (1) is called *adjoint property*.

Once we have recalled the previous operators, the definitions of multi-adjoint frame is given below.

Definition 2: A *multi-adjoint frame* is a tuple $(L_1, L_2, P, \&_1, \dots, \&_n)$, where $(L_1, \preceq_1)$ and $(L_2, \preceq_2)$ are complete lattices, $(P, \leq)$ is a poset and $(\&_i, \swarrow^i, \nwarrow_i)$ is an adjoint triple with respect to L_1, L_2, P, for all $i \in \{1, \dots, n\}$.

Fixed a multi-adjoint frame, we can introduce the notion of multi-adjoint context, which considers a relational database with an extra mapping.

Definition 3: Let $(L_1, L_2, P, \&_1, \dots, \&_n)$ be a multi-adjoint frame, a *context* is a tuple (A, B, R, σ) such that A and B are non-empty sets (usually interpreted as attributes and objects, respectively), R is a P-fuzzy relation $R\colon A \times B \to P$ and $\sigma\colon A \times B \to \{1, \dots, n\}$ is a mapping which associates any element in $A \times B$ with some particular adjoint triple.

In the multi-adjoint concept lattice environment, the concept-forming operators ${}^{\uparrow}\colon L_2^B \to L_1^A$ and ${}^{\downarrow}\colon L_1^A \to L_2^B$ are defined as follows

$$g^{\uparrow}(a) = \inf\{R(a,b) \swarrow^{\sigma(a,b)} g(b) \mid b \in B\} \qquad (2)$$

$$f^{\downarrow}(b) = \inf\{R(a,b) \nwarrow_{\sigma(a,b)} f(a) \mid a \in A\} \qquad (3)$$

for all $g \in L_2^B$, $f \in L_1^A$ and $a \in A$, $b \in B$, where L_2^B and L_1^A denote the set of mappings $g\colon B \to L_2$ and $f\colon A \to L_1$, respectively. We call *multi-adjoint concepts* to pairs $\langle g, f\rangle$, where $g \in L_2^B$ is a fuzzy subset of objects and $f \in L_1^A$ is a fuzzy subset of attributes, verifying the equalities $g^{\uparrow} = f$ and $f^{\downarrow} = g$. Specifically, we call *extent* to the first component of a multi-adjoint concept g and *intent* to the second component f. The extent and intent of a concept C are denoted by $\mathfrak{E}(C)$ and $\mathfrak{I}(C)$, respectively. Note that the concept-forming operators form an antitone Galois connection [26].

Definition 4: The *multi-adjoint concept lattice* associated with a multi-adjoint frame $(L_1, L_2, P, \&_1, \dots, \&_n)$ and a context (A, B, R, σ) given, is the set

$$\mathcal{M} = \{\langle g, f\rangle \mid g \in L_2^B, f \in L_1^A \text{ and } g^{\uparrow} = f, f^{\downarrow} = g\}$$

where the ordering is defined by $\langle g_1, f_1\rangle \preceq \langle g_2, f_2\rangle$ if and only if $g_1 \preceq_2 g_2$ (equivalently $f_2 \preceq_1 f_1$).

Characterizing the *meet-irreducible elements* of a multi-adjoint concept lattice was a fundamental task in order to categorize the attributes of the associated multi-adjoint context, as it was showed in [33], [27]. Before including the attribute classification theorems, we need to recall the notion of meet-irreducible element and the characterization theorem of this special type of concepts.

Definition 5: Given a lattice $(L, \preceq)$, such that $\wedge, \vee$ are the meet and the join operators, and an element $x \in L$ verifying

1) If L has a top element $\top$, then $x \neq \top$.
2) If $x = y \wedge z$, then $x = y$ or $x = z$, for all $y, z \in L$.

we call x *meet-irreducible ($\wedge$-irreducible) element* of L. Condition (2) is equivalent to

2′. If $x < y$ and $x < z$, then $x < y \wedge z$, for all $y, z \in L$.

A *join-irreducible ($\vee$-irreducible) element* of L is defined dually.

According to the previous definition, we say that a concept is a meet-irreducible element if cannot be expressed as infimum of strictly greater concepts of the lattice.

The following specific family of fuzzy subsets of attributes play an important role in the characterization theorem of meet-irreducible elements.

Definition 6: For each $a \in A$, the fuzzy subsets of attributes $\phi_{a,x} \in L_1^A$ defined, for all $x \in L_1$, as

$$\phi_{a,x}(a') = \begin{cases} x & \text{if } a' = a \\ \bot_1 & \text{if } a' \neq a \end{cases}$$

will be called *fuzzy-attributes*, where $\bot_1$ is the minimum element in L_1. The set of all fuzzy-attributes will be denoted as $\Phi = \{\phi_{a,x} \mid a \in A, x \in L_1\}$.

Theorem 1 ([27]): The set of $\wedge$-irreducible elements of $\mathcal{M}$, $M_F(A)$, is formed by the pairs $\langle \phi_{a,x}^{\downarrow}, \phi_{a,x}^{\downarrow\uparrow}\rangle$ in $\mathcal{M}$, with $a \in A$ and $x \in L_1$, such that

$$\phi_{a,x}^{\downarrow} \neq \bigwedge\{\phi_{a_i,x_i}^{\downarrow} \mid \phi_{a_i,x_i} \in \Phi, \phi_{a,x}^{\downarrow} \prec_2 \phi_{a_i,x_i}^{\downarrow}\}$$

and $\phi_{a,x}^{\downarrow} \neq g_{\top_2}$, where $\top_2$ is the maximum element in L_2 and $g_{\top_2}\colon B \to L_2$ is the fuzzy subset defined as $g_{\top_2}(b) = \top_2$, for all $b \in B$.

After introducing the characterization of meet-irreducible elements of a multi-adjoint concept lattice, we need to recall other important notions and results associated with the attribute classification of a multi-adjoint context. The first definitions are the notions of consistent set and reduct [27].

Definition 7: A set of attributes $Y \subseteq A$ is a *consistent* set of (A, B, R, σ) if the following isomorphism holds:

$$\mathcal{M}(Y, B, R_Y, \sigma_{Y\times B}) \cong_E \mathcal{M}(A, B, R, \sigma)$$

This is equivalent to say that, for all $\langle g, f\rangle \in \mathcal{M}(A, B, R, \sigma)$, there exists a concept $\langle g', f'\rangle \in \mathcal{M}(Y, B, R_Y, \sigma_{Y\times B})$ such that $g = g'$.

In addition, if the isomorphism does not hold when we remove every attribute $a \in Y$, that is, if:

$$\mathcal{M}(Y \setminus \{a\}, B, R_{Y\setminus\{a\}}, \sigma_{Y\setminus\{a\}\times B}) \ncong_E \mathcal{M}(A, B, R, \sigma)$$

for all $a \in Y$, then Y is called *reduct* of (A, B, R, σ). Notice that $Y \setminus \{a\}$ is the relative complement of the element $\{a\}$ in Y, also called the set difference of Y and $\{a\}$.

The *core* of (A, B, R, σ) is the intersection of all the reducts of (A, B, R, σ).

The following technical results are very useful in order to demonstrate several results presented in Section IV. The first one characterizes the consistent sets in the multi-adjoint environment.

Lemma 1 ([27]): $Y \subseteq A$ is a consistent set of (A, B, R, σ) if and only if, for every $\langle \phi_{a_i,x_i}^{\downarrow}, \phi_{a_i,x_i}^{\downarrow\uparrow}\rangle \in M_F(A)$, there exists $\langle \phi_{a_j,x_j}^{\downarrow}, \phi_{a_j,x_j}^{\downarrow\uparrow}\rangle \in M_F(Y)$, such that $\phi_{a_i,x_i}^{\downarrow} = \phi_{a_j,x_j}^{\downarrow}$.

Following the same philosophy, the next result characterizes the reducts associated with a multi-adjoint framework and a multi-adjoint context.

Lemma 2 ([27]): Given a reduct $Y \subseteq A$ of (A, B, R, σ). For each attribute $a_i \in Y$, there exists $x_i \in L_1$, such that $\langle \phi_{a_i,x_i}^{\downarrow}, \phi_{a_i,x_i}^{\downarrow\uparrow}\rangle \in M_F(A)$ and $\phi_{a_i,x_i}^{\downarrow} \neq \phi_{a_j,x_j}^{\downarrow}$, for all $a_j \in Y \setminus \{a_i\}$ and $x_j \in L_1$.

Considering the reducts of the multi-adjoint context related to a multi-adjoint concept lattice, the following definition presents a reducts-based categorization of the set of attributes A.

Definition 8: Given a formal context (A, B, R, σ) and the set $\mathcal{Y} = \{Y \subseteq A \mid Y \text{ is a reduct}\}$ of all reducts of (A, B, R, σ). The set of attributes A can be divided into the following parts:

1) Absolutely necessary attributes (core attributes) $C_{\mathrm{f}} = \bigcap_{Y\in\mathcal{Y}} Y$.
2) Relatively necessary attributes $K_{\mathrm{f}} = (\bigcup_{Y\in\mathcal{Y}} Y) \setminus (\bigcap_{Y\in\mathcal{Y}} Y)$.
3) Absolutely unnecessary attributes $I_{\mathrm{f}} = A \setminus (\bigcup_{Y\in\mathcal{Y}} Y)$.

The following definition will be useful to recall the three attribute classification theorems proved in [28], as well as to present the algorithms introduced along this paper.

Definition 9: Given a multi-adjoint frame $(L_1, L_2, P, \&_1, \dots, \&_n)$, a context (A, B, R, σ) associated with the concept lattice $(\mathcal{M}, \preceq)$ and a concept C of $(\mathcal{M}, \preceq)$, *the set of attributes generating* C is defined as the set:

$$\begin{aligned}\mathrm{Atg}(C) \;\; = \;\; & \{a \in A \mid \text{there exists } \phi_{a,x} \in \Phi \\ & \qquad \text{such that } \langle \phi_{a,x}^{\downarrow}, \phi_{a,x}^{\downarrow\uparrow}\rangle = C\}\end{aligned}$$

The three classification theorems are recalled in the following. The first one characterizes the absolutely necessary attributes of the formal context.

Theorem 2 ([28]): Given an attribute $a \in A$, we have that $a \in C_{\mathrm{f}}$ if and only if there exists a meet-irreducible concept C of $(\mathcal{M}, \preceq)$ satisfying that $a \in \mathrm{Atg}(C)$ and $\mathrm{card}(\mathrm{Atg}(C)) = 1$.

The following one shows the characterization of the relatively necessary atributes.

Theorem 3 ([28]): Given an attribute $a \in A$, we have that $a \in K_{\mathrm{f}}$ if and only if $a \notin C_{\mathrm{f}}$ and there exists $C \in M_{\mathrm{F}}(A)$ with $a \in \mathrm{Atg}(C)$ and $\mathrm{card}(\mathrm{Atg}(C)) > 1$, satisfying that $\big(A \setminus \mathrm{Atg}(C)\big) \cup \{a\}$ is a consistent set.

Finally, the last theorem shows the characterization of the completely unnecessary attributes.

Theorem 4 ([28]): Given an attribute $a \in A$, we have that $a \in I_{\mathrm{f}}$ if and only if, for every $C \in M_{\mathrm{F}}(A)$, $a \notin \mathrm{Atg}(C)$, or if $a \in \mathrm{Atg}(C)$ then $\big(A \setminus \mathrm{Atg}(C)\big) \cup \{a\}$ is not a consistent set.

In the next section, we introduce the necessary algorithms to obtain the set of meet-irreducible concepts associated with a multi-adjoint context, and the classification of the attributes.

III. Algorithms for the attribute classification of a multi-adjoint context

To begin with, we present an auxiliar algorithm which will be useful to obtain the set of irreducible elements of a given multi-adjoint context. Specifically, Algorithm 1 returns a list composed of pairs, such that each pair contains the extent of a concept, C, generated from at least a fuzzy-attribute (the first component is $\mathfrak{E}(C)$), together with the list of attributes associated with such fuzzy-attributes (the second component is $\mathrm{Atg}(C)$). This composite list is denoted as Φ^2.

This algorithm will be helpful in order to obtain the list of $\wedge$-irreducible concepts since, according to Theorem 1, every $\wedge$-irreducible element is generated from at least one fuzzy-attribute. Specifically, from line 3 to line 7, the algorithm takes into account every possible combination of attributes $a \in A$ and values $x \in L_1$ and computes $\phi_{a,x}^{\downarrow}$. Line 8 includes the generated extension together with the associated attribute in the set Φ^2. Due to different fuzzy-attributes can generate the same concept of the concept lattice, lines 9 to 14 are intended to avoid duplications, we search the elements in Φ^2 that match in the first component (that is, with the same extension) and join the attributes that appears in the second component. Although this module can be optimized in different aspects, such as including the avoid duplications in the first computation of Φ^2, we have preferred to leave an easier representation.

Let us show how this algorithm works in a particular example.

Algorithm 1: Computing Φ^2

input : (A, B, R, σ), $(L_1, L_2, P, \&_1, \ldots, \&_n)$
output: Φ^2

$\Phi^2 := []$;
$\phi^{\downarrow}_{a,x} := []$;
for each $a \in A$ **do**
    for each $x \in L_1 \setminus \{\perp_1\}$ **do**
        for each $b \in B$ **do**
            $z =$ **Compute**$(R(a,b) \nwarrow_{\sigma(a,b)} x)$
            add z **to** $\phi^{\downarrow}_{a,x}$
        add $\{(\phi^{\downarrow}_{a,x}, a)\}$ **to** Φ^2
for each $(g, Y) \in \Phi^2$ **do**
    for each $(g', Y') \in \Phi^2 \setminus \{(g, Y)\}$ **do**
        if $g = g'$ **then**
            remove $\{(g', Y')\}$ **from** Φ^2
            remove $\{(g, Y)\}$ **from** Φ^2
            add $\{(g, Y \cup Y')\}$ **to** Φ^2
return Φ^2

R	H	Ba	F	C	St	Sn	T
James	0.6	0.2	0.2	1	0.6	0.2	0
Adam	0.8	0.4	0.4	1	0.8	0.6	0.6
Lily	0.6	0.6	0.2	0	0	0	0

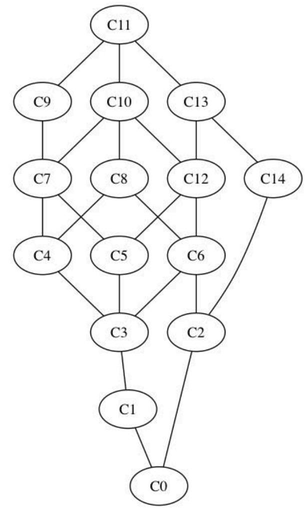

Fig. 1. Relation R and Hasse diagram of $(\mathcal{M}, \preceq)$ of Example 1.

Example 1: Let $(L_1, L_2, L_3, \preceq, \&^*_G, \&^*_L)$ composed of regular partitions of the unit interval in 10, 4 and 5 pieces, that is, $L_1 = [0,1]_{10}$, $L_2 = [0,1]_4$, $L_3 = [0,1]_5$, respectively, and the discretizations of the Gödel and Łukasiewicz conjunctors, $\&^*_G$ and $\&^*_L$, respectively (for more details, see [34]). The considered context (A, B, R, σ) is composed of the set of attributes A={Headache (H), Body aches (Ba), Fever (F), Cough (C), Sore throat (St), Stuffy nose (Sn), Tiredness (T)}, the set of objects is composed of 3 patients B={James, Adam, Lily}, the relation $R\colon A \times B \to L_3$ displayed in Figure 1 and the mapping σ is defined, for all $a \in A$ and $b \in B$, as:

$$\sigma(a,b) = \begin{cases} a \,\&^*_L\, b & \text{if } a \in \{\text{H, C, St}\} \text{ and } b \in \{\text{James, Adam}\} \\ a \,\&^*_G\, b & \text{otherwise} \end{cases}$$

Note that, we have assigned different operators to different objects and attributes of the considered context, the Łukasiewicz adjoint triple is associated with certain symptoms of certain patients and the Gödel adjoint triple is associated with another patient. This fact let us give more importance to certain symptoms in certain patients depending on, for example, the medical history of each patient, which could be essential to deliver a proper diagnosis of these patients.

Applying Algorithm 1, we obtain that Φ^2 is composed of the following list of pairs:

$$\begin{aligned}
(\mathfrak{E}(C_0), \text{Atg}(C_0)) &= \big(\{\text{Adam}/0.25\}, \{\text{F}\}\big)\\
(\mathfrak{E}(C_1), \text{Atg}(C_1)) &= \big(\{\text{Adam}/0.5\}, \{\text{Sn}, \text{T}\}\big)\\
(\mathfrak{E}(C_2), \text{Atg}(C_2)) &= \big(\{\text{Adam}/0.25, \text{Lily}/0.5\}, \{\text{Ba}\}\big)\\
(\mathfrak{E}(C_4), \text{Atg}(C_4)) &= \big(\{\text{Adam}/0.75, \text{James}/0.5\}, \{\text{St}\}\big)\\
(\mathfrak{E}(C_5), \text{Atg}(C_5)) &= \big(\{\text{Adam}\}, \{\text{F}, \text{Sn}, \text{T}\}\big)\\
(\mathfrak{E}(C_7), \text{Atg}(C_7)) &= \big(\{\text{Adam}, \text{James}/0.75\}, \{\text{St}\}\big)\\
(\mathfrak{E}(C_8), \text{Atg}(C_8)) &= \big(\{\text{Adam}/0.75, \text{James}/0.5, \text{Lily}/0.5\}, \{\text{H}\}\big)\\
(\mathfrak{E}(C_9), \text{Atg}(C_9)) &= \big(\{\text{Adam}, \text{James}\}, \{\text{C}, \text{St}, \text{Sn}\}\big)\\
(\mathfrak{E}(C_{10}), \text{Atg}(C_{10})) &= \big(\{\text{Adam}, \text{James}/0.75, \text{Lily}/0.5\}, \{\text{H}\}\big)\\
(\mathfrak{E}(C_{11}), \text{Atg}(C_{11})) &= \big(\{\text{Adam}, \text{James}, \text{Lily}\}, \{\text{Ba}, \text{F}, \text{H}\}\big)\\
(\mathfrak{E}(C_{13}), \text{Atg}(C_{13})) &= \big(\{\text{Adam}, \text{Lily}\}, \{\text{Ba}\}\big)\\
(\mathfrak{E}(C_{14}), \text{Atg}(C_{14})) &= \big(\{\text{Adam}/0.25, \text{Lily}\}, \{\text{Ba}\}\big)
\end{aligned}$$

As we previously commented, in Φ^2 we can observe the extents of all the concepts generated from fuzzy-attributes (on the left), together with the subset of attributes from which these extents are obtained (on the right). Note that, C_3, C_6 and C_{12} are not generated from fuzzy-attributes. ■

However, not all extents included in the set Φ^2 correspond to extents of irreducible concepts. Thus, we need to eliminate from Φ^2 those extensions that do not correspond to irreducible concepts, this procedure is carried out by Algorithm 2.

The output of Algorithm 2 is the set $\{(C_i, \text{Atg}(C_i)) \mid C_i \in M_F(A)\}$, which is denoted as $\Phi^2(M_F)$ because it is a subset of Φ^2. It is easy to check that this algorithm is also based on Theorem 1 since it selects from Φ^2 each extension that cannot be expressed as infimum of extensions greater than it. In particular, from line 3 to line 6, Algorithm 2 takes an element $(g, Y) \in \Phi^2$ and builds the upper bounds of the first component of this element, denoting this set as UB. In line

7 the algorithm checks if g is the infimum of UB, if it is not, this element is stored in $\Phi^2(M_F)$. Therefore, according to Theorem 1, the output $\Phi^2(M_F)$ only contain those elements in Φ^2 whose extents (first components of each pair) belong to $\wedge$-irreducible concepts.

Algorithm 2: Computing $\Phi^2(M_F)$

input : Φ^2
output: $\Phi^2(M_F)$

1 UB $:= \varnothing$;
2 $\Phi^2(M_F) := []$;
3 **for each** $(g, Y) \in \Phi^2$ **do**
4   **for each** $(g', Y') \in \Phi^2 \setminus \{(g, Y)\}$ **do**
5   **if** $g < g'$ **then**
6   **add** $\{g'\}$ **to** UB
7   **if** $g \neq \bigwedge UB$ **then**
8   **add** $\{(g, Y)\}$ **to** $\Phi^2(M_F)$
9 **return** $\Phi^2(M_F)$

In the following example, Algorithm 2 is applied to the list Φ^2, obtained in Example 1.

Example 2: Considering the input Φ^2 obtained from the multi-adjoint frame and the context of Example 1, the output $\Phi^2(M_F)$ that arises applying Algorithm 2, is displayed below:

$$(\mathfrak{E}(C_1), \text{Atg}(C_1)) = (\{\text{Adam}/0.5\}, \{\text{Sn}, \text{T}\})$$
$$(\mathfrak{E}(C_8), \text{Atg}(C_8)) = (\{\text{Adam}/0.75, \text{James}/0.5, \text{Lily}/0.5\}, \{\text{H}\})$$
$$(\mathfrak{E}(C_9), \text{Atg}(C_9)) = (\{\text{Adam}, \text{James}\}, \{\text{C}, \text{St}, \text{Sn}\})$$
$$(\mathfrak{E}(C_{10}), \text{Atg}(C_{10})) = (\{\text{Adam}, \text{James}/0.75, \text{Lily}/0.5\}, \{\text{H}\})$$
$$(\mathfrak{E}(C_{13}), \text{Atg}(C_{13})) = (\{\text{Adam}, \text{Lily}\}, \{\text{Ba}\})$$
$$(\mathfrak{E}(C_{14}), \text{Atg}(C_{14})) = (\{\text{Adam}/0.25, \text{Lily}\}, \{\text{Ba}\})$$

From the previous list and Figure 1, it is easy to check that extents contained in $\Phi^2(M_F)$ are the extents of the $\wedge$-irreducible elements of the concept lattice. For example, we have that the extent of C_9, C_{10} and C_{11} are greater than the extent of C_7, hence they belong to UB (line 6) and the infimum is C_7. Therefore, $(\mathfrak{E}(C_7), \text{Atg}(C_7)) = (\{\text{Adam}, \text{James}/0.75\}, \{\text{St}\})$ is not added to $\Phi^2(M_F)$. ■

From the set $\Phi^2(M_F)$ we can compute the set C_{f} by using Algorithm 3. According to Theorem 2, we only need to know the list of elements in $\Phi^2(M_F)$ satisfying that the cardinality of the second component is equal to one, as Algorithm 3 does. Then, the attributes appearing in these elements form the core of the context.

In the next example, we will compute the core set of the context of Example 1.

Example 3: In order to obtain the set C_{f}, we consider as input of Algorithm 3 the list $\Phi^2(M_F)$ obtained in Example 2. It is clear to see that the output of this algorithm indicates that the set of absolutely necessary attributes is $C_{\text{f}} = \{\text{H}, \text{Ba}\}$. ■

According to Theorems 3 and 4, in order to obtain the sets K_{f} and I_{f} we need to make use of two auxiliary algorithms because we need to take into account two important aspects:

1) We should be able to identify when a subset of attributes $D \subseteq A$ is consistent or not with respect to the set of meet-irreducible elements appearing in $E \subseteq \Phi^2(M_F)$, which is the aim of Algorithm 4. This algorithm tests if every element in $M_F(A)$ can be generated from the set D. To be allowed to do it, Algorithm 4 checks if the set D has a non-empty intersection with the second component of each element in $\Phi^2(M_F)$. In the case that an intersection is empty, that means that this irreducible concept cannot be generated from the attribute contained in D and, consequently, the set D is not a consistent set.
2) We do not have to consider those $\wedge$-irreducible concepts that can be obtained from the attributes in the core. Thus, we need to create a filtered list that contains the set of irreducible concepts that cannot be obtained from attributes in C_{f}. This is the purpose of Algorithm 5, which remove from the set $\Phi^2(M_F)$ those pairs whose second components contain any attribute in the core. The resulting subset of $\Phi^2(M_F)$ is denoted as $\Phi^2_C(M_F)$.

Algorithm 3: Computing C_{f}

input : $\Phi^2(M_F)$
output: C_{f}

$C_{\text{f}} := \varnothing$;
for each $(g, Y) \in \Phi^2(M_F)$ **do**
    if $\text{card}(Y) = 1$ **then**
        add Y **to** C_{f}
return C_{f}

Algorithm 4: CONSISTENCY(D,E)

input : $D \subseteq A$, $E \subseteq \Phi^2(M_F)$
output: True or False

Consistency := True
break=False
for each $(g, Y) \in \Phi^2(M_F)$ **and break=False do**
    if $D \bigcap Y = \varnothing$ **then**
        Consistency = False
        break=True
return *Consistency*

Algorithm 5: Computing $\Phi^2_C(M_F)$

input : C_{f}, $\Phi^2(M_F)$
output: $\Phi^2_C(M_F)$

$\Phi^2_C(M_F) := \Phi^2(M_F)$;
for each $(g, Y) \in \Phi^2(M_F)$ **do**
    if $C_{\text{f}} \bigcap Y \neq \varnothing$ **then**
        remove $\{(g, Y)\}$ **from** $\Phi^2_C(M_F)$
return $\Phi^2_C(M_F)$

In the following example, we remove from the list $\Phi^2(M_F)$ obtained in Example 2 those elements whose second component contains elements in the core.

Example 4: From the list $\Phi^2(M_F)$, obtained in Example 2, and the subset of attributes in the core, obtained in Example 3, we can apply Algorithm 5. This algorithm provides the set $\Phi^2_C(M_F)$, which is composed of the elements $\big(\{\text{Adam}/0.5\},\{\text{Sn},\text{T}\}\big)$ and $\big(\{\text{Adam},\text{James}\},\{\text{C},\text{St},\text{Sn}\}\big)$. ■

Once we have the auxiliary algorithms, we can present the algorithm to compute the subsets of unnecessary attributes and relatively necessary attributes. Algorithm 6 is based on the attribute classification results given in Theorems 3 and 4.

Algorithm 6: Computing K_f and I_f

---

input : A, C_f, $\Phi^2_C(M_F)$
output: K_f, I_f

$I_\mathrm{f} := \varnothing$;
$K_\mathrm{f} := \varnothing$;
$A' := \bigcup\{Y \mid (g,Y) \in \Phi^2_C(M_F)\}$;
for each $a \in A'$ **do**
    add a **to** K_f
    break=False
    for each $(g,Y) \in \Phi^2_C(M_F)$ **and break=False do**
        if $a \in Y$ **then**
            if *CONSISTENCY*$\big((A' \backslash Y)\bigcup\{a\}, \Phi^2_C(M_F)\big)$=*False* **then**
                remove a **from** K_f
                break=True
$I_\mathrm{f} = A \setminus \{C_\mathrm{f} \bigcup K_\mathrm{f}\}$
return K_f, I_f

Considering again the example we are using to illustrate the results obtained with the algorithms introduced in this section, we obtain the subsets K_f and I_f from Algorithms 4 and 6.

Example 5: In Table I, we can observe the steps that are carried out when we apply Algorithm 6 considering the inputs we have obtained in the previous examples. In this case, the input of Algorithm 6 is composed of the sets:

$$A = \{\text{H},\text{Ba},\text{F},\text{C},\text{St},\text{Sn},\text{T}\}$$
$$C_\mathrm{f} = \{\text{H},\text{Ba}\}$$
$$\Phi^2_C(M_F) = \Big\{\big(\{\text{Adam}/0.5\},\{\text{Sn},\text{T}\}\big),\big(\{\text{Adam},\text{James}\},\{\text{C},\text{St},\text{Sn}\}\big)\Big\}$$

Taking into consideration the previous sets, we have that $A' = \{\text{Sn},\text{T},\text{C},\text{St}\}$.

In this case, the final attribute classification is as follows:

$$\begin{aligned} C_\mathrm{f} &= \{\text{H},\text{Ba}\} \\ K_\mathrm{f} &= \{\text{C},\text{St},\text{Sn},\text{T}\} \\ I_\mathrm{f} &= \{\text{F}\} \end{aligned}$$

■

In the following we present Algorithm 7, which supposes an alternative option to Algorithm 4 and 6. This new algorithm has the advantage that do not need to consider the auxiliar Algorithm 4 to compute the subsets K_f and I_f and it can obtain the final classification only from the filtered list of meet-irreducible concepts (provided by Algorithm 5). Hence, we obtain the final classification in a more direct way. In order to understand the development of Algorithm 7, it is necessary to introduce the following result.

Proposition 1: Let $(L_1, L_2, P, \&_1, \dots, \&_n)$ be a multi-adjoint frame, a context (A, B, R, σ) and an attribute $a \in A$. The attribute $a \in I_\mathrm{f}$ if and only if one of the following statements holds:

- $a \notin \mathrm{Atg}(C)$ for any concept $C \in M_F(A)$.
- for all $C \in M_F(A)$, such that $a \in \mathrm{Atg}(C)$, there exists $C' \in M_F(A)$ satisfying that $\mathrm{Atg}(C') \subseteq \mathrm{Atg}(C) \setminus \{a\}$.

Proof: First of all, we will prove the first implication. Applying Theorem 4 we straightforwardly obtain that a does not generate any $\wedge$-irreducible concept or, if $a \in \mathrm{Atg}(C)$, with $C \in M_F(A)$ then the set $\big(A \backslash \mathrm{Atg}(C)\big) \cup \{a\}$ is not a consistent set. If $\big(A \backslash \mathrm{Atg}(C)\big) \cup \{a\}$ is not consistent then there exists at least one $\wedge$-irreducible concept $C' \in M_F(A)$ which cannot be generated from the attributes that belong to $\big(A \backslash \mathrm{Atg}(C)\big) \cup \{a\}$, which is equivalent to say that $\mathrm{Atg}(C') \subseteq \mathrm{Atg}(C) \setminus \{a\}$.

Now, we will prove the opposite implication. If $a \notin \mathrm{Atg}(C)$ for any concept $C \in M_F(A)$, by Theorem 4, we have that $a \in I_\mathrm{f}$. Now, we suppose that for all $C \in M_F(A)$ such that $a \in \mathrm{Atg}(C)$, there exists $C' \in M_F(A)$ satisfying that $\mathrm{Atg}(C') \subseteq \mathrm{Atg}(C) \setminus \{a\}$. In this case, we have that $\big(A \backslash \mathrm{Atg}(C)\big) \cup \{a\}$ is not a consistent set because the concept C' cannot be generated from the attributes in $\big(A \setminus \mathrm{Atg}(C)\big) \cup \{a\}$. Consequently, by Theorem 4, we conclude that $a \in I_\mathrm{f}$. ■

Specifically, to obtain the sets K_f and I_f, Algorithm 7 also considers the filtered list $\Phi^2_C(M_F)$. Therefore, Algorithm 7 is based exclusively on the second item of the previous result. In particular, from the perspective of the relatively necessary attributes, the construction process of K_f considered in Algorithm 7 is shown in the following lemma.

Lemma 3: Let $(L_1, L_2, P, \&_1, \dots, \&_n)$ be a multi-adjoint frame, a context (A, B, R, σ) and one attribute $a \in A$. The atribute $a \in K_\mathrm{f}$ if for each $C \in M_F(A)$, such that $a \in \mathrm{Atg}(C)$ the following statements hold:

- $\mathrm{Atg}(C) \bigcap C_\mathrm{f} = \varnothing$.
- There is no $C_i \in M_F(A)$, satisfying that $\mathrm{Atg}(C_i) \subseteq \mathrm{Atg}(C) \setminus \{a\}$.

Proof: If for each $C \in M_F(A)$, such that $a \in \mathrm{Atg}(C)$, there is no $C_i \in M_F(A)$, satisfying that $\mathrm{Atg}(C_i) \subseteq \mathrm{Atg}(C) \setminus \{a\}$, by Proposition 1 we have that $a \notin I_\mathrm{f}$. In addition, the condition $\mathrm{Atg}(C) \bigcap C_\mathrm{f} = \varnothing$ means that $a \notin C_\mathrm{f}$. Consequently, we can conclude that $a \in K_\mathrm{f}$. ■

In line 11 of Algorithm 7 we can see whether $Y'' \subseteq Y'$. In such a case, applying Lemma 3, we have that $a \in A'$ is an unnecessary attribute and, consequently, this attribute is removed from K_f in line 12.

In the following example, we specify how Algorithm 7 works when the same inputs, that were considered in Example 5, are provided.

Example 6: As we have already mentioned, the inputs coincide with those given in the previous example and, as a consequence, the set A' is also the same as the one shown in Example 5. Table II shows step by step the procedure to obtain the classification, according to Algorithm 7.

We can see in Table II that the number of loops that this alternative algorithm requires to obtain the final classification

TABLE I
ALGORITHM 6 APPLIED TO EXAMPLE 1

loop	a	(g, Y)	membership to Y	CONSISTENCY$(((A' \setminus Y) \bigcup\{a\}), \Phi^2_C(M_F))$	K_{f}
1	Sn	$(g_1, Y_1) = (\{\text{Adam}/0.5\}, \{\text{Sn}, \text{T}\})$	Sn $\in Y_1$	CONSISTENCY$(\{\text{C}, \text{St}, \text{Sn}\}, \Phi^2_C(M_F))$=True	
		$(g_2, Y_2) = (\{\text{Adam}, \text{James}\}, \{\text{C}, \text{St}, \text{Sn}\})$	Sn $\in Y_2$	CONSISTENCY$(\{\text{Sn}, \text{T}\}, \Phi^2_C(M_F))$=True	{Sn}
2	T	$(g_1, Y_1) = (\{\text{Adam}/0.5\}, \{\text{Sn}, \text{T}\})$	T $\in Y_1$	CONSISTENCY$(\{\text{C}, \text{St}, \text{T}\}, \Phi^2_C(M_F))$=True	
		$(g_2, Y_2) = (\{\text{Adam}, \text{James}\}, \{\text{C}, \text{St}, \text{Sn}\})$	T $\notin Y_2$		{Sn,T}
3	C	$(g_1, Y_1) = (\{\text{Adam}/0.5\}, \{\text{Sn}, \text{T}\})$	C $\notin Y_1$		
		$(g_2, Y_2) = (\{\text{Adam}, \text{James}\}, \{\text{C}, \text{St}, \text{Sn}\})$	C $\in Y_2$	CONSISTENCY$(\{\text{C}, \text{T}\}, \Phi^2_C(M_F))$=True	{Sn,T,C}
4	St	$(g_1, Y_1) = (\{\text{Adam}/0.5\}, \{\text{Sn}, \text{T}\})$	St $\notin Y_1$		
		$(g_2, Y_2) = (\{\text{Adam}, \text{James}\}, \{\text{C}, \text{St}, \text{Sn}\})$	St $\in Y_2$	CONSISTENCY$(\{\text{St}, \text{T}\}, \Phi^2_C(M_F))$=True	{Sn,T,C,St}
$I_{\mathrm{f}} = A \setminus \{C_{\mathrm{f}} \bigcup K_{\mathrm{f}}\} = \{\text{F}\}$					
Output: $K_{\mathrm{f}} = \{\text{Sn}, \text{T}, \text{C}, \text{St}\}$ $I_{\mathrm{f}} = \{\text{F}\}$					

Algorithm 7: Computing I_{f} and K_{f}

```
input : A, C_f, Φ²_C(M_F)
output: I_f, K_f
I_f := ∅;
K_f := ∅;
A' := ⋃{Y | (g, Y) ∈ Φ²_C(M_F)};
for each a ∈ A' do
    add a to K_f
    break=False
    for each (g, Y) ∈ Φ²_C(M_F) and break=False do
        if a ∈ Y then
            Y' = Y \ a
            for each (g'', Y'') ∈ Φ²_C(M_F) and
              break=False do
                if Y'' ⊆ Y' then
                    remove a from K_f
                    break=True
I_f = A \ (C_f ⋃ K_f)
return I_f, K_f
```

coincides with the number of loops required by Algorithm 6. Clearly, applying Algorithm 7, we obtain the same attribute classification that in the previous example. However, this time, we avoid having to resort to an auxiliar algorithm (since Algorithm 6 resorts to Algorithm 4). ■

A categorization of the set of attributes of a multi-adjoint context to absolutely necessary, relatively necessary and absolutely unnecessary attributes help us to obtain reducts. These reducts may decrease the computational complexity of the concept lattice in a relevant way. In the next section, we will present the algorithm to compute reducts of formal contexts in the multi-adjoint environment.

IV. ALGORITHM TO COMPUTE REDUCTS OF A MULTI-ADJOINT CONTEXT

As we have previously mentioned, the elimination of superfluous attributes is an essential procedure in order to reduce the computational complexity to obtain the knowledge stored in a relational dataset. In this section, we introduce an algorithm to compute a reduct from a given multi-adjoint context.

The sets presented in the following will be useful in this section. These sets were used in [28] to analyze reducts of a multi-adjoint context. In this paper, we have removed the subscripts f of the notation of these sets for the sake of simplicity.

$$\begin{aligned} \mathcal{G}^C_K &= \{\text{Atg}(C) \mid C \in M_F(A) \text{ such that } \text{Atg}(C) \cap K_{\mathrm{f}} \neq \varnothing \\ &\qquad \text{and} \quad \text{Atg}(C) \cap C_{\mathrm{f}} = \varnothing\} \\ \mathcal{G}^{C,I}_K &= \{\text{Atg}(C) \setminus I_{\mathrm{f}} \mid \text{Atg}(C) \in \mathcal{G}^C_K\} \end{aligned}$$

Notice that the set $\mathcal{G}^C_K$ coincides with the union of the second components of the elements in $\Phi^2_C(M_F)$, that is, it can be also expressed as $\mathcal{G}^C_K = \bigcup\{Y \mid (g, Y) \in \Phi^2_C(M_F)\}$.

Taking into account the outputs provided by the previous algorithms, the set $\mathcal{G}^{C,I}_K$ can be obtained by means of Algorithm 8.

Algorithm 8: Computing $\mathcal{G}^{C,I}_K$

```
input : I_f, Φ²_C(M_F)
output: G^{C,I}_K
G^{C,I}_K := [];
for each (g, Y) ∈ Φ²_C(M_F) do
    Y' = Y \ I_f
    add {Y'} to G^{C,I}_K
return G^{C,I}_K
```

Example 7: Considering as input of Algorithm 8 the subsets obtained from the previous examples:

$$I_{\mathrm{f}} = \{\text{F}\}$$
$$\Phi^2_C(M_F) = \{(\{\text{Adam}/0.5\}, \{\text{Sn,T}\}), (\{\text{Adam}, \text{James}\}, \{\text{C,St, Sn}\})\}$$

We obtain that $\mathcal{G}^{C,I}_K = \{\{\text{Sn,T}\}, \{\text{C,St, Sn}\}\}$. This is because, in this particular example, the second component of each pair in $\Phi^2_C(M_F)$ has an empty intersection with the subset of attributes I_{f}. Therefore, according to the obtained attribute classification, we have that $\mathcal{G}_K = \mathcal{G}^C_K = \mathcal{G}^{C,I}_K$. ■

From now on, the elements belonging to $\mathcal{G}^{C,I}_K$ will be denoted as $\text{Atg}(C)^K$. In addition, we need to introduce the following notations. We denote the particular elements of

TABLE II
ALGORITHM 7 APPLIED TO EXAMPLE 1

loop	a	(g, Y)	membership to Y	$Y' = Y \setminus a$	inclusion in Y'	K_{f}
1	Sn	$(g_1, Y_1) = (\{\text{Adam}/0.5\}, \{\text{Sn}, \text{T}\})$	Sn $\in Y_1$	{T}	$Y_1 \not\subseteq Y'$	
		$(g_2, Y_2) = (\{\text{Adam}, \text{James}\}, \{\text{C}, \text{St}, \text{Sn}\})$	Sn $\in Y_2$	{C,St}	$Y_2 \not\subseteq Y'$	{Sn}
2	T	$(g_1, Y_1) = (\{\text{Adam}/0.5\}, \{\text{Sn}, \text{T}\})$	T $\in Y_1$	{Sn}	$Y_1 \not\subseteq Y'$	
		$(g_2, Y_2) = (\{\text{Adam}, \text{James}\}, \{\text{C}, \text{St}, \text{Sn}\})$	T $\notin Y_2$			{Sn,T}
3	C	$(g_1, Y_1) = (\{\text{Adam}/0.5\}, \{\text{Sn}, \text{T}\})$	C $\notin Y_1$			
		$(g_2, Y_2) = (\{\text{Adam}, \text{James}\}, \{\text{C}, \text{St}, \text{Sn}\})$	C $\in Y_2$	{St, Sn}	$Y_2 \not\subseteq Y'$	{Sn,T,C}
4	St	$(g_1, Y_1) = (\{\text{Adam}/0.5\}, \{\text{Sn}, \text{T}\})$	St $\notin Y_1$			
		$(g_2, Y_2) = (\{\text{Adam}, \text{James}\}, \{\text{C}, \text{St}, \text{Sn}\})$	St $\in Y_2$	{C, Sn}	$Y_2 \not\subseteq Y'$	{Sn,T,C,St}
$I_{\mathrm{f}} = A \setminus \{C_{\mathrm{f}} \bigcup K_{\mathrm{f}}\} = \{\text{F}\}$						
Output: $K_{\mathrm{f}} = \{\text{Sn}, \text{T}, \text{C}, \text{St}\}$ $I_{\mathrm{f}} = \{\text{F}\}$						

the set of relatively necessary attributes by $k_1, k_2, \ldots, k_m$, i.e. $K_{\mathrm{f}} = \{k_1, k_2, \ldots, k_m\}$ and the set of attributes $A = \{a_1, a_2, \ldots, a_n\}$, where $m \leq n$.

For each relatively necessary attribute $a \in K_{\mathrm{f}}$, we need to count how many times the attribute a appears in elements $\mathrm{Atg}(C)^K \in \mathcal{G}_K^{C,I}$. Given $H \subseteq \mathcal{G}_K^{C,I}$, we need to define the following function $q_H \colon K_{\mathrm{f}} \to \mathcal{P}(A)$ as follows:

$$q_H(k_j) = \{\mathrm{Atg}(C)^K \mid \mathrm{Atg}(C)^K \in H \text{ and } k_j \in \mathrm{Atg}(C)^K\}$$

for all $j \in \{1, 2, \ldots, m\}$. Particularly $q_H(k_j)$ represents the subsets $\mathrm{Atg}(C)^K \in H$ containing the attribute k_j.

Taking into account the previous notational conventions, in order to compute a reduct $Y \subseteq A$, the instructions shown in Algorithm 9 should be applied.

Algorithm 9: Computing a reduct Y

```
   input : C_f, K_f = {k_1, k_2, ..., k_m}, G_K^{C,I}, q_H
   output: Y
1  H := G_K^{C,I};
2  T := K_f;
3  Y := K_f ;
4  repeat
5     v_min := min{card(q_H(k_i)) | k_i ∈ T};
6     i_min := j, such that card(q_H(k_j)) = v_min;
7     if min{card(Atg(C)^K) | Atg(C)^K ∈ q_H(k_{i_min})} = 2
      then
8        for X ∈ q_H(k_{i_min}) do
9           if card(X) = 2 then
10             remove X from T
11    remove k_{i_min} from T;
12    remove k_{i_min} from Y;
13    H = {Atg(C)^K \ k_{i_min} | Atg(C)^K ∈ H};
14 until T = ∅;
15 return Y = Y ⋃ C_f
```

Algorithm 9 considers the philosophy of the linear QUICKREDUCT II algorithm [29] in the multi-adjoint concept lattice framework. Specifically, it removes from the set of relatively necessary attributes K_{f}, the dispensable attributes. Now, we explain in detail how Algorithm 9 works.

In lines 1, 2, 3, the auxiliary sets H, T and Y are initially defined as $\mathcal{G}_K^{C,I}$, K_{f} and K_{f}, respectively.

Line 5 computes the value $v_{\min}$, which expresses the minimal number of sets $\mathrm{Atg}(C)^K$ containing some particular relatively necessary attribute k_j.

Line 6 defines the value $i_{\min}$ representing the subscript of a relatively necessary attribute which has obtained the previous minimal value $v_{\min}$. If two or more indices satisfy a minimal value $v_{\min}$, it is taken an arbitrary one.

In line 7, the algorithm checks if there exists an element in $q_H(k_{i_{\min}})$ whose cardinality is equal to 2.

When the condition given in line 7 is satisfied, the for-loop in line 8 goes over each component of $q_H(k_{i_{\min}})$. These components are denoted as X.

In lines 9 and 10, if a component X has cardinality equal to 2, that is $X = \{k_{i_{\min}}, k'\}$, with $k' \in K_{\mathrm{f}}$, then the algorithm removes the elements of that component of the set T (the idea of the step given in line 10, is to avoid removing the attribute k' from Y in the subsequent iterations, since the attribute $k_{i_{\min}}$ is always removed from the set Y).

The element $k_{i_{\min}}$ is removed from T in line 11. With this step we guarantee that in the following loop the value $k_{i_{\min}}$ is different from the value considered in the previous loop.

In line 12, we reduce the set Y because the attribute $k_{i_{\min}}$ is not necessary to build the concept lattice.

Since $k_{i_{\min}}$ is not needed, line 13 removes this attribute from each element $\mathrm{Atg}(C)^K \in H$. This fact decreases the cardinality of these sets and they can satisfy the condition in line 7 in future loops.

The algorithm finishes in line 14 when the set T is empty. As output (line 15), the algorithm includes in the set Y the elements in the core.

Concerning the termination of the algorithm, we can ensure that it terminates after a finite number of steps, because the set T is reduced by at least one element (see lines 10 and 11 of the loop repeat-until) in each step of the algorithm. Notice that if $K_{\mathrm{f}} = \varnothing$, then $\mathcal{G}_K^{C,I} = \varnothing$ and $Y = C_{\mathrm{f}}$.

In order to prove that Algorithm 9 provides a reduct, we need to introduce the following results.

The first one states that, we can always find at least two different relatively necessary attributes generating a meet-irreducible concept C when the condition $\mathrm{Atg}(C) \cap C_{\mathrm{f}} = \varnothing$ holds.

Proposition 2: Let C be a meet-irreducible concept. If $\mathrm{Atg}(C) \cap C_{\mathrm{f}} = \varnothing$ then $\mathrm{card}(\mathrm{Atg}(C) \cap K_{\mathrm{f}}) \geq 2$.

Proof: If $\mathrm{Atg}(C) \cap C_{\mathrm{f}} = \varnothing$ then, for all $a \in \mathrm{Atg}(C)$ we have that $a \notin C_{\mathrm{f}}$. In addition, since $C \in M_F(A)$ then, by Theorem 3, there exist at least two attributes $a_i, a_j \in K_{\mathrm{f}}$ such that $a_i, a_j \in \mathrm{Atg}(C)$. Consequently, $\mathrm{card}(\mathrm{Atg}(C) \cap K_{\mathrm{f}}) \geq 2$. ∎

The following results relate Definition 9 to the notions of consistent set and reduct.

Proposition 3: A subset $Y \subseteq A$ is a consistent set if and only if for all $C \in M_F(A)$ the condition $\mathrm{card}(\mathrm{Atg}(C) \bigcap Y) \geq 1$ holds.

Proof: If the subset of attributes Y is a consistent set, by Lemma 1, we have that for every $C = \langle \phi^{\downarrow}_{a_i,x_i}, \phi^{\downarrow\uparrow}_{a_i,x_i} \rangle \in M_F(A)$, there exists $\langle \phi^{\downarrow}_{a_j,x_j}, \phi^{\downarrow\uparrow}_{a_j,x_j} \rangle \in M_F(Y)$, such that $\phi^{\downarrow}_{a_i,x_i} = \phi^{\downarrow}_{a_j,x_j}$. Therefore, according to Definition 9 we have that $a_j \in \mathrm{Atg}(C)$ and, consequently, $\mathrm{card}(\mathrm{Atg}(C) \bigcap Y) \geq 1$.

Now, we will prove the second implication. We will assume that for all $C \in M_F(A)$ the inequality $\mathrm{card}(\mathrm{Atg}(C) \bigcap Y) \geq 1$ holds. Hence, given any meet-irreducible concept C, we have that there exists an attribute $a \in Y$ such that $a \in \mathrm{Atg}(C)$. Then, by Definition 9, we can find a fuzzy-attribute $\phi_{a,x} \in \Phi$ such that $\langle \phi^{\downarrow}_{a,x}, \phi^{\downarrow\uparrow}_{a,x} \rangle = C$. Therefore, by Lemma 1, we can conclude that the subset Y is a consistent set. ∎

Proposition 4: A subset $Y \subseteq A$ is a reduct if and only if Y is a consistent set and for all $a \in Y$ there exists $C \in M_F(A)$ such that $\mathrm{Atg}(C) \bigcap (Y \setminus \{a\}) = \varnothing$.

Proof: First of all, we will suppose that the subset Y is a reduct. Then, Y is a consistent set. In addition, applying Lemma 2, for each attribute $a_i \in Y$, there exists $x_i \in L_1$, such that $\langle \phi^{\downarrow}_{a_i,x_i}, \phi^{\downarrow\uparrow}_{a_i,x_i} \rangle \in M_F(A)$ satisfying that $\phi^{\downarrow}_{a_i,x_i} \neq \phi^{\downarrow}_{a_j,x_j}$, for all $a_j \in Y \setminus \{a_i\}$ and $x_j \in L_1$. Considering $C = \langle \phi^{\downarrow}_{a_i,x_i}, \phi^{\downarrow\uparrow}_{a_i,x_i} \rangle$ and according to Definition 9, this is equivalent to say that there exists $C \in M_F(A)$ satisfying that $a_i \in \mathrm{Atg}(C)$ and $\mathrm{Atg}(C) \bigcap (Y \setminus \{a_i\}) = \varnothing$.

Now, we prove the converse. Given $a \in Y$, by hypothesis, if we remove the atribute a from the subset Y, there exists a concept $C \in M_F()A$ which can not be generated from any attribute $Y \setminus \{a\}$. Therefore, the set $Y \setminus \{a\}$ is not a consistent set, which leads us to a contradiction. ∎

As direct consequences of the above results, we obtain the following corollaries.

Corollary 1: Given a subset $Y \subseteq A$, if for all $a \in Y$ there exists $C \in M_F(A)$ such that the equality $\mathrm{card}(\mathrm{Atg}(C) \bigcap Y) = 1$ holds, then the subset Y is a reduct.

Corollary 2: Given a subset $Y \subseteq A$, if for all $a \in Y \setminus C_{\mathrm{f}}$ there exists $C \in M_F(A)$ such that the equality $\mathrm{card}(\mathrm{Atg}(C)^K \bigcap Y) = 1$ holds, then the subset Y is a reduct.

Once we have presented the previous results, we can prove that the subset of attributes Y, provided by Algorithm 9, is a reduct.

Proposition 5: Algorithm 9 provides a reduct.

Proof: First of all, we will prove that the subset Y provided by Algorithm 9 is a consistent set, which is equivalent by Proposition 3 to prove that, for all $C \in M_F(A)$ there is an attribute $a \in Y$ satisfying that $a \in \mathrm{Atg}(C)$. For that purpose, we will assume that there exists a concept $C \in M_F(A)$ such that $\mathrm{Atg}(C) \bigcap Y = \varnothing$ and we will obtain a contradiction. Then, since $C_{\mathrm{f}} \subseteq Y$ we have that $\mathrm{Atg}(C) \bigcap C_{\mathrm{f}} = \varnothing$ and, applying Proposition 2, the inequality $\mathrm{card}(\mathrm{Atg}(C) \cap K_{\mathrm{f}}) \geq 2$ holds, that is, $\mathrm{card}(\mathrm{Atg}(C)^K) \geq 2$.

If $\mathrm{card}(\mathrm{Atg}(C)^K) = 2$, then there exist $a_i, a_j \in A$ such that $\mathrm{Atg}(C)^K = \{a_i, a_j\}$ and, without lack of generality, we can assume that $\mathrm{card}(q_H(a_i)) \leq \mathrm{card}(q_H(a_j))$. Now, we have to distinguish two different cases:

(a) If $a_i = k_{i_{\min}}$, then the attribute a_i satisfies the conditions displayed in lines 7 and 9 of our algorithm. Consequently, by line 10 the attribute a_j is removed from the set T. Therefore, a_j can never be considered equal to $k_{i_{\min}}$ and, by line 13, we can conclude that $a_j \in Y$ which lead us to a contradiction since $Y \bigcap \mathrm{Atg}(C) = \varnothing$ and $a_j \in \mathrm{Atg}(C)^K \subseteq \mathrm{Atg}(C)$.

(b) If the attribute a_i is never equal to $k_{i_{\min}}$, then, according to line 13, this element is never removed from Y, that is, $a_i \in Y$, which is a contradiction since $Y \bigcap \mathrm{Atg}(C) = \varnothing$ and $a_i \in \mathrm{Atg}(C)^K \subseteq \mathrm{Atg}(C)$.

Now, we will suppose that $\mathrm{card}(\mathrm{Atg}(C)^K) > 2$, that is, $\mathrm{Atg}(C)^K = \{a_{i_1}, \ldots, a_{i_l}\}$, where $a_{i_1}, \ldots, a_{i_l} \in A$. We have to consider two different situations again:

(c) We will suppose that there exists at least one attribute a_{i_j} satisfying that it is always different from $k_{i_{\min}}$. Then, by line 13, this attribute a_{i_j} is not removed from Y, that is, $a_{i_j} \in Y$ and we obtain a contradiction.

(d) Otherwise, we suppose that, for all $j \in \{1, \ldots, l\}$ the elements a_{i_j} are equal to $k_{i_{\min}}$. In such a case, we can consider again that $\mathrm{card}(q_H(a_{i_1})) \leq \ldots \leq \mathrm{card}(q_H(a_{i_l}))$, without lack of generality. Hence $a_{i_1} = k_{i_{\min}}$, then the attribute a_{i_1} satisfies the conditions displayed in line 7, but as $\mathrm{card}(\mathrm{Atg}(C)^K) > 2$, this element does not satify the condition given in line 9 of our algorithm. As a consequence, according to lines 11 and 13, the attribute a_{i_1} is removed from T and Y, respectively. Now, we have two different possibilities again:

  (d1) If $\mathrm{card}(\mathrm{Atg}(C)^K \setminus \{a_{i_1}\}) = 2$, we come back to the situation of item (a).

  (d2) If $\mathrm{card}(\mathrm{Atg}(C)^K \setminus \{a_{i_1}\}) > 2$, we are again in the situation shown in item (d).
  If we repeat this process, due to we reduce one unit the cardinality of $\mathrm{Atg}(C)^K$ in each loop, we have the set $\mathrm{Atg}(C) = \{a_{i_{l-1}}, a_{i_l}\}$, after a finite number of steps. Then, we are in the case of item (a) and we have a contradiction.

Finally, we can conclude that the set Y provided by Algorithm 9 is a consistent set.

Now, we will prove that the set Y is also a reduct of the multi-adjoint context using Corollary 2. Hence, we have to prove that for all $a \in Y \setminus C_{\mathrm{f}}$ there exists $C \in M_F(A)$ such that the equality $\mathrm{card}(\mathrm{Atg}(C)^K \bigcap (Y \setminus C_{\mathrm{f}})) = 1$ holds.

If $a_j \in Y \setminus C_{\mathrm{f}}$, we know that the attribute a_j was removed from T in line 10 in a step N of Algorithm 9. Therefore, there exists a meet-irreducible concept C such that $\mathrm{Atg}(C)^K = \{a_{i_1}, \ldots, a_{i_l}, a_{i_{l+1}}, a_j\}$. We also know that before step N a finite number of loops in Algorithm 9 should be given in order to obtain the set $\mathrm{Atg}(C)^K \setminus \{a_{i_1}, \ldots, a_{i_l}\} = \{a_{i_{l+1}}, a_j\}$, satisfying that $\{a_{i_1}, \ldots, a_{i_l}\} \notin Y$ and the element $k_{i_{\min}}$ is currently $a_{i_{l+1}}$. As a consequence, at the end of step N, by line 12, we obtain $\{a_{i_1}, \dots, a_{i_l}, k_{i_{\min}}\} \notin Y$. Thus, by Corollary 2, we can conclude that the subset Y is a reduct. ■

Henceforth, we will continue our example and we will show how the last algorithm is applied on it.

Example 8: Coming back to the environment of Example 1 and, in order to preserve the considered notations in Algorithm 9, we provide the attributes with an ordering making an identification as follows:

$$A = \{\mathrm{H}, \mathrm{Ba}, \mathrm{F}, \mathrm{C}, \mathrm{St}, \mathrm{Sn}, \mathrm{T}\} = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7\}$$

Therefore, the inputs of our algorithm are the following:

$$\begin{array}{lcl} C_{\mathrm{f}} & = & \{\mathrm{H}, \mathrm{Ba}\} = \{a_1, a_2\} \\ K_{\mathrm{f}} & = & \{\mathrm{C}, \mathrm{St}, \mathrm{Sn}, \mathrm{T}\} = \{a_4, a_5, a_6, a_7\} \\ \mathcal{G}_K^{C,I} & = & \{\{\mathrm{Sn}, \mathrm{T}\}, \{\mathrm{C}, \mathrm{St}, \mathrm{Sn}\}\} = \{\{a_6, a_7\}, \{a_4, a_5, a_6\}\} \end{array}$$

Table III represents the values that Algorithm 9 computes in each loop for v_{min}, i_{min}, and the sets T and Y.

Furthermore, in order to clarify the performance of the algorithm, it is interesting to show the computation of the cardinality of the function q_H in each element of T, see Table IV.

As a result, the algorithm has provided the minimal reduct, $Y = \{a_1, a_2, a_6\} = \{\mathrm{H}, \mathrm{Ba}, \mathrm{Sn}\}$, of the multi-adjoint formal context.

TABLE III
ALGORITHM 9 APPLIED TO EXAMPLE 1

loop	$v_{\min}$	$i_{\min}$	T	Y
0			$\{a_4, a_5, a_6, a_7\}$	$\{a_4, a_5, a_6, a_7\}$
1	1	4	$\{a_5, a_6, a_7\}$	$\{a_5, a_6, a_7\}$
2	1	5	$\{a_7\}$	$\{a_6, a_7\}$
3	1	7	$\varnothing$	$\{a_6\}$
Output: Y=$\{a_1, a_2, a_6\}$				

TABLE IV
COMPUTING THE CARDINALITY OF THE SETS $q_H(k_i)$ WITH ALGORITHM 9

loop	q_H
1	$\mathrm{card}(q_H(a_4)) = \mathrm{card}(q_H(a_5)) = \mathrm{card}(q_H(a_7)) = 1$
	$\mathrm{card}(q_H(a_6)) = 2$
2	$\mathrm{card}(q_H(a_5)) = \mathrm{card}(q_H(a_7)) = 1$
	$\mathrm{card}(q_H(a_6)) = 2$
3	$\mathrm{card}(q_H(a_7)) = 1$

■

The computational complexity of the whole algorithm depends on the number of attributes, objects and elements in the lattice associated with the attributes. In usual datasets the number of attributes is very small with respect to the number of objects, which also implies that the elements in the lattice is also small with respect to the number of objects. As a consequence, we can consider that the cardinality of A and L_1 is negligible with respect to the cardinality of B, that is, $|A| \lll |B|$ and $|L_1| \lll |B|$. Therefore, we will give the complexity of the algorithm with respect to the cardinality of B, which will be denoted as $|B| = n$.

The different modules introduced in Algorithms 1-9 are computed sequentially and so, the final complexity will be the greatest complexity of the different modules. We will detail the complexity of the first module and the rest are computed analogously. The first part is introduced in Algorithm 1, which is split in two for loops. Clearly, the complexity of the first part (lines 3-8) is $|A| \times |L_1| \times |B|$. Since $|A|$ and $|L_1|$ are considered constants, then the complexity order is $O(n)$. Due to the elements in Φ^2 depend on the elements in L_1 and A, the maximum number of elements in this set is $|A| \times |L_1|$, which is a constant (product of two constants). Therefore, the complexity of the second part (Lines 9-14) is $|A| \times |L_1| \times |A| \times |L_1| \times |B|$, that is, the complexity order of this part is also $O(n)$. Thus, the complexity order of Algorithm 1 is $O(n)$.

Algorithm 2 makes two comparison between two fuzzy subsets of objects in Lines 5 and 7. Since they are in different loops, the complexity order of this algorithm is also $O(n)$. The rest of algorithms neither compare fuzzy subsets of objects nor consider loops with respect to objects, the complexity order is constant. As a consequence, we can ensure that the final complexity order of the whole algorithm is $O(n)$.

In the following section, we gather several related works and bring to light the main differences that we find, as well as the main advantages that our work provides.

V. Comparison with related works

There exist several papers which deal with the problem of obtaining reducts from formal contexts. In these works, the authors present different mechanisms to reduce the contexts, even some algorithms that pursue this goal. In this section, we collect some of these reduction mechanisms and we highlight the main advantages of our mechanism compared to them.

For instance, in [35], [36] the reducts are computed considering a classic formal context. The reducts are the prime implicants of the discernibility function obtained from a discernibility matrix. This discernibility matrix is built considering the attributes that distinguish the intensions of each pair of concepts of the concept lattice. Obviously, for the computation of this matrix, all the concepts of the concept lattice must be considered, which considerably increases the computational cost. In addition, the computational cost of turning a conjunctive normal form into a reduced disjunction normal form, is also very high. The algorithms presented in our work only need to consider the $\wedge$-irreducible elements of the concept lattice to obtain reducts, whose computation does not require the whole set of concepts of the concept lattice.

Another paper that introduces an algorithm to compute the reducts of a formal context considering a crisp setting is [37]. In this work, a study about attribute reduction and the computation of reducts is presented, which is based on dominance relations obtained from the formal context. This reduction mechanism considers the notion of dominance consistent set, that is, a subset of attributes from which we can define the same dominance relation that the one obtained from the original set of attributes. When a dominance consistent set is minimal with respect to the number of attributes, it is called dominance reduct. They compute the dominance reducts by means of a discernibility matrix with respect to the relation of dominance. These reducts are the prime implicants of the discernibility function derived from the discernibility matrix. Therefore, this reduction mechanism suffers the same drawbacks from the paper mentioned above. Furthermore, the authors prove an equivalence between the set of granular reducts and the set of dominance reducts. Due to this equivalence and the fact that the framework considered in our paper is more general, we can ensure that all the results and algorithms of our work can be successfully applied to the environment of [37].

On the other hand, as we mentioned in the introduction, the algorithms introduced in our study can also be analogously applied for the reduction of the set of objects. For example, in [38] the authors present an algorithm to reduce objects of a formal decision context, from the definitions of consistent set and reduct based on decision rules. This reduction guarantees that the set of valid decision rules in the reduced context is also valid in the original. However, they also work in a crisp environment and the original structure of the concept lattice is not preserved after the reduction, unlike our approach.

Other works analyze the reduction of the set of attributes from the perspective provided by other frameworks. For example, in [39], [40] the covering generalized rough sets framework is considered to study the reduction of formal contexts. In these works, considering a crisp environment, the authors define formal contexts from a universe together with a covering of the universe. From that universe and that covering, they define a relationship R, that relates the elements in the universe to the covering. Hence, the elements of the universe are the objects, the elements of the covering are considered as the set of attributes and R is the relation of the formal context. In this way, the concept-forming operators coincide with the approximation operators of the covering-based rough set. Consequently, the reducts of the covering coincide with the reducts of the derived formal context. Hence, the given reduction follows a different philosophy of FCA and so, they provide an incomparable reduction to our mechanism. On the other way, the algorithms introduced in our work can also be applied to other frameworks, such as to obtain reducts in the covering generalized rough sets framework.

Other papers consider fuzzy environments, as for example [41], [4]. In [41], the authors consider crisp subsets of objects and fuzzy subsets of attributes. They also introduce an algorithm whose reduction is carried out through the discernibility matrix and the derived discernibility function. Although they guarantee that the structure of the original concept lattice is preserved after the reduction, the computational cost of obtaining the prime implicants is high, as it was previously mentioned. In [4], the authors also work in a fuzzy environment in which the concepts have a crisp set of objects and a fuzzy set of attributes. They also compute the reducts from the discernibility matrix and the discernibility function. In order to obtain the discernibility matrix, all the extensions of the concepts of the corresponding concept lattice need to be compared. Once again, we find the same drawbacks that we have already stated, which are overcome with the algorithms introduced in our work.

As it was proved in [42] those mechanisms to reduce formal contexts based on discernibility matrices are less efficient than those based on clarification and reduction, that is, based on Ganter and Wille's original method [17]. All algorithms presented in this paper are based on the results given in [27], [28], which were proven to be generalizations of the mechanism given by Ganter and Wille to the multi-adjoint framework in [27].

VI. Conclusions and future work

In this work, we have considered the fuzzy framework of multi-adjoint concept lattices. Basing on the characterization of the meet-irreducible elements of a concept lattice and the attribute classification theorems presented in [27], [28], we have introduced new algorithms to know the set of irreducible elements, the classification of the set of attributes as well as to obtain reducts from any multi-adjoint framework and context. The proposed algorithms are interesting because they consider a general fuzzy environment, that requires few restrictions and, therefore, can be applied in a wider variety of real situations. In addition, this fuzzy framework is capable of encompassing different frameworks considered in the theory of formal concept analysis. Consequently, all the algorithms introduced in this work can be successfully applied to other existing frameworks of FCA.

In addition, we have presented several results in order to relate the notions of reduct and consistent set to the notion of attribute generating a meet-irreducible element. We have also presented an illustrative example to clarify how all the introduced algorithms work. In addition, we have included a comparison with other related works given in the literature.

As future work, we are interested in an algorithm to get reducts, with the goal to obtain a minimal reduct, that is, the reduct with a smaller number of attributes among all the possible reducts of the considered context. In addition, we will apply these algorithms and results to solve real-life problems. In particular, we would like to explore the potential of the application in the digital forensics field in which the authors coordinate an European network.

References

[1] T. Tilley, R. Cole, P. Becker, and P. Eklund, *A Survey of Formal Concept Analysis Support for Software Engineering Activities*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 250–271.

[2] S. M. Dias and N. J. Vieira, "Applying the JBOS reduction method for relevant knowledge extraction," *Expert Systems with Applications*, vol. 40, no. 5, pp. 1880 – 1887, 2013.

[3] C. A. Kumar and S. Srinivas, "Concept lattice reduction using fuzzy k-means clustering," *Expert Systems with Applications*, vol. 37, no. 3, pp. 2696–2704, 2010.

[4] M.-W. Shao and K.-W. Li, "Attribute reduction in generalized one-sided formal contexts," *Information Sciences*, vol. 378, pp. 317 – 327, 2017.

[5] M.-W. Shao, H.-Z. Yang, and W.-Z. Wu, "Knowledge reduction in formal fuzzy contexts," *Knowledge-Based Systems*, vol. 73, pp. 265 – 275, 2015.

[6] W. S. Du and B. Q. Hu, "Attribute reduction in ordered decision tables via evidence theory," *Information Sciences*, vol. 364-365, pp. 91 – 110, 2016.

[7] J. Li, C. Mei, and Y. Lv, "Incomplete decision contexts: Approximate concept construction, rule acquisition and knowledge reduction," *International Journal of Approximate Reasoning*, vol. 54, no. 1, pp. 149–165, 2013.

[8] F. Pacheco, M. Cerrada, R.-V. Sánchez, D. Cabrera, C. Li, and J. V. de Oliveira, "Attribute clustering using rough set theory for feature selection in fault severity classification of rotating machinery," *Expert Systems with Applications*, vol. 71, pp. 69 – 86, 2017.

[9] M. Liu, M. Shao, W. Zhang, and C. Wu, "Reduction method for concept lattices based on rough set theory and its application," *Computers & Mathematics with Applications*, vol. 53, no. 9, pp. 1390–1410, 2007.

[10] R. Zhang, S. Xiong, and Z. Chen, "Construction method of concept lattice based on improved variable precision rough set," *Neurocomputing*, vol. 188, pp. 326 – 338, 2016.

[11] C. Cornelis, R. Jensen, G. Hurtado, and D. Ślęzak, "Attribute selection with fuzzy decision reducts," *Information Sciences*, vol. 180, pp. 209–224, 2010.

[12] C. Cornelis, J. Medina, and N. Verbiest, "Multi-adjoint fuzzy rough sets: Definition, properties and attribute selection," *International Journal of Approximate Reasoning*, vol. 55, pp. 412–426, 2014.

[13] R. Jensen and C. Cornelis, "Fuzzy-rough nearest neighbour classification and prediction," *Theoretical Computer Science*, vol. 412, no. 42, pp. 5871–5884, 2011.

[14] J. Medina, "Multi-adjoint property-oriented and object-oriented concept lattices," *Information Sciences*, vol. 190, pp. 95–106, 2012.

[15] H. Bai, Y. Ge, J. Wang, D. Li, Y. Liao, and X. Zheng, "A method for extracting rules from spatial data based on rough fuzzy sets," *Knowledge-Based Systems*, vol. 57, pp. 28–40, 2014.

[16] J. C. Díaz-Moreno and J. Medina, "Multi-adjoint relation equations: Definition, properties and solutions using concept lattices," *Information Sciences*, vol. 253, pp. 100–109, 2013.

[17] B. Ganter and R. Wille, *Formal Concept Analysis: Mathematical Foundation*. Springer Verlag, 1999.

[18] M. Liu, M. Shao, W. Zhang, and C. Wu, "Reduction method for concept lattices based on rough set theory and its application," *Computers and Mathematics with Applications*, vol. 53, no. 9, pp. 1390 – 1410, 2007.

[19] M. J. Benítez-Caballero, J. Medina, E. Ramírez-Poussa, and D. Ślęzak, "A computational procedure for variable selection preserving different initial conditions," *International Journal of Computer Mathematics*, vol. 0, no. 0, pp. 1–18, 2019.

[20] T.-J. Li, "Knowledge reduction in formal contexts based on covering rough sets," in *Rough Sets and Knowledge Technology*, P. Wen, Y. Li, L. Polkowski, Y. Yao, S. Tsumoto, and G. Wang, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 128–135.

[21] M.-W. Shao and W.-X. Zhang, "Approximation in formal concept analysis," in *Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing*, D. Ślęzak, G. Wang, M. Szczuka, I. Düntsch, and Y. Yao, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 43–53.

[22] W.-X. Zhang, L. Wei, and J.-J. Qi, "Attribute reduction in concept lattice based on discernibility matrix," in *Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing*, D. Ślęzak, J. Yao, J. F. Peters, W. Ziarko, and X. Hu, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 157–165.

[23] J.-J. Qi, "Attribute reduction in formal contexts based on a new discernibility matrix," *Journal of Applied Mathematics and Computing*, vol. 30, no. 1, pp. 305–314, 2009.

[24] S. M. Dias and N. J. Vieira, "Concept lattices reduction: Definition, analysis and classification," *Expert Systems with Applications*, vol. 42, no. 20, pp. 7084 – 7097, 2015.

[25] J. Medina and M. Ojeda-Aciego, "Multi-adjoint t-concept lattices," *Information Sciences*, vol. 180, no. 5, pp. 712–725, 2010.

[26] J. Medina, M. Ojeda-Aciego, and J. Ruiz-Calviño, "Formal concept analysis via multi-adjoint concept lattices," *Fuzzy Sets and Systems*, vol. 160, no. 2, pp. 130–144, 2009.

[27] M. E. Cornejo, J. Medina, and E. Ramírez-Poussa, "Attribute reduction in multi-adjoint concept lattices," *Information Sciences*, vol. 294, pp. 41 – 56, 2015.

[28] M. E. Cornejo, J. Medina, and E. Ramírez-Poussa, "Characterizing reducts in multi-adjoint concept lattices," *Information Sciences*, vol. 422, pp. 364 – 376, 2018.

[29] Q. Shen and A. Chouchoulas, "A rough-fuzzy approach for generating classification rules," *Pattern Recognition*, vol. 35, no. 11, pp. 2425 – 2438, 2002.

[30] P. Hájek, *Metamathematics of Fuzzy Logic*, ser. Trends in Logic. Kluwer Academic, 1998.

[31] M. E. Cornejo, J. Medina, and E. Ramírez-Poussa, "Multi-adjoint algebras versus extended-order algebras," *Applied Mathematics & Information Sciences*, vol. 9, no. 2L, pp. 365–372, 2015.

[32] ——, "Multi-adjoint algebras versus non-commutative residuated structures," *International Journal of Approximate Reasoning*, vol. 66, pp. 119–138, 2015.

[33] ——, "On the classification of fuzzy-attributes in multi-adjoint concept lattices," *Lecture Notes in Computer Science*, vol. 7903, pp. 266–277, 2013.

[34] ——, "A comparative study of adjoint triples," *Fuzzy Sets and Systems*, vol. 211, pp. 1–14, 2013.

[35] J.-S. Mi, Y. Leung, and W.-Z. Wu, "Approaches to attribute reduction in concept lattices induced by axialities," *Knowledge-Based Systems*, vol. 23, no. 6, pp. 504 – 511, 2010.

[36] W. Zhang, L. Wei, and J. Qi, "Attribute reduction theory and approach to concept lattice," *Science in China Series F: Information Sciences*, vol. 48, no. 6, pp. 713–726, Dec 2005.

[37] M.-W. Shao and Y. Leung, "Relations between granular reduct and dominance reduct in formal contexts," *Knowledge-Based Systems*, vol. 65, pp. 1 – 11, 2014.

[38] J. Li, C. Mei, J. Wang, and X. Zhang, "Rule-preserved object compression in formal decision contexts using concept lattices," *Knowledge-Based Systems*, vol. 71, pp. 435 – 445, 2014.

[39] J. Chen, J. Li, Y. Lin, G. Lin, and Z. Ma, "Relations of reduction between covering generalized rough sets and concept lattices," *Information Sciences*, vol. 304, pp. 16 – 27, 2015.

[40] A. Tan, J. Li, and G. Lin, "Connections between covering-based rough sets and concept lattices," *International Journal of Approximate Reasoning*, vol. 56, pp. 43 – 58, 2015.

[41] M.-W. Shao, Y. Leung, X.-Z. Wang, and W.-Z. Wu, "Granular reducts of formal fuzzy contexts," *Knowledge-Based Systems*, vol. 114, pp. 156 – 166, 2016.

[42] J. Konecny and P. Krajca, "On attribute reduction in concept lattices: Experimental evaluation shows discernibility matrix based methods inefficient," *Inf. Sci.*, vol. 467, pp. 431–445, 2018.