MINING TRIADIC ASSOCIATION RULES

Sid Ali Selmane[1], Rokia Missaoui[2]
Omar Boussaid[1], Fadila Bentayeb[1]

[1]5 avenue PierreMendès France 69676 Bron Cedex, France
Sid-Ali.selmane@univ-lyon2.fr,
[2]101, rue Saint-Jean-Bosco Gatineau (Québec) Canada, J8X 3X7
http://larim.uqo.ca/index.html

ABSTRACT

*The objective of this research is to extract triadic association rules from a triadic formal context $K := (K_1, K_2, K_3, Y)$ where K_1, K_2 and K_3 respectively represent the sets of objects, properties (or attributes) and conditions while Y is a ternary relation between these sets. Our approach consists to define a procedure to map a set of dyadic association rules into a set of triadic ones. The advantage of the triadic rules compared to the dyadic ones is that they are less numerous and more compact than the second ones and convey a richer semantics of data. Our approach is illustrated through an example of ternary relation representing a set of **C**ustomers who purchase their**P**roducts from **S**uppliers. The algorithms and approach proposed have been validated with experimentations on large real datasets.*

KEYWORDS

Formal Concept Analysis, Galois lattice, triadic association rules.

1. INTRODUCTION

Formal concept analysis and Galois lattice constitute a theoretical basis for solving many problems in different application areas that handle or produce data representing ternary relation (a relationship between three distinct sets) such as networks social. For example, the participation of *researchers to scientific conferences* in various roles (member of the scientific committee, author, organizer, etc.). Another area *Web 2.0* where folksonomies are defined as links between users who annotate tags (keywords) to certain resources (web pages, articles), *OLAPCubes*, eg suppliers who supply products to stores...

Thus the extraction of knowledge from these ternary relationships becomes important in view of the diversity of application domains and their magnitudes on economic and scientific. The dyadic association rules are conditional implications between two sets of proprieties called *items* allows to extracts hidden information on large datasets, however, the large number of rules obtained during the extraction process on several data sets leads us to seek a better match rule type. Hence the idea to propose an approach that produces the triadic association rules are more compact than the dyadic [17].

This research helps to show how we produce triadic association rules (including implications) from a triadic context. We propose an approach based on formal concept analysis and dyadic association rules mining. The work presented here can be useful for data mining models of

Sundarapandian et al. (Eds) : ICAITA, SAI, SEAS, CDKP, CMCA-2013
pp. 305–316, 2013. © CS & IT-CSCP 2013 DOI : 10.5121/csit.2013.3825

ternary relationships between three groups of entities and in particular when one of the three sets describes a collection of individuals while the others sets correspond to their properties (eg, privileges or roles in secure systems) and conditions (space and time constraints) to which they are subject.

The main objective of this work was to improve and simplify the approach proposed by [17] proposing procedures to extract triadic association rules from a formal triadic context which is anensemblist representation of ternary relationship. The second goal was to make this work on the basis of formal concept analysis tools and approaches. Then, we proposed to decompose the extraction process in three stages. First, transform the triadic contexts on equivalent dyadic contexts in order to use formal concept analysis approaches. Then produce concepts, generators and dyadic association rules. And finally, extract from these last triadic association rules.

The paper is organized as follows. Section 2 presents the state of the art of relating works to our problems. Section 3 focuses on the notion and definition of the FCA (Formal Analysis Concepts) which are necessary to study association rules through the definition of dyadic and triadic concepts, of dyadic association rules and triadic association rules. Then, in Section 4, we develop our approach and we roll the proposed algorithms. Finally, in Section 5, we present our experiments and results. We conclude and present the perspectives of our work in the last section.

2. STATE OF THE ART

The formal concept analysis [23], [8] and the Galois lattice theory [2]constitute a theoretical basis for solving numerous problems in the fields of artificial intelligence, software engineering and databases. A Galois lattice is considered as a form of conceptual clustering, in which the elements of the lattice are identified with concepts and the graph to a generalization/specialization relationship between concepts. Each concept corresponds to a pair composed of an extension representing a subset of instances of the application and an intention representing the common properties of these instances. Accordingly, the hierarchical structure of the Galois lattice concepts offers a compact representation of the information conducive to the exploration and knowledge discovery. In this section we present how we use formal concept analysis in association rules discovery.

In the literature, there's various works some of them date back to 1995, as those [24], [16]have focused on triadic contexts analysis, concepts and diagrams and concept lattices. They define, in this way, the theoretical basis for the triadic concept analysis (TCA). The same year, Bidermann proposes a formalism for writing triadic implications. In 2002, Voutsadkis provides ideas on the analysis of polyadic concepts and generalizes the work of [24] for polyadic formal contexts to produce polyadic formal concepts and to obtain n-lattices.

More recent works related to the triadic context analysis exist. In 2004, Ganter and Obiedkov list different types of triadic implications. In 2006, two approaches to the generation of triadic concept are proposed: the first through TRIAS [12], which allows the computation of triadic concepts from dyadic concepts. Then comes RSM and Cube Miner, [13] addresses the same problem of calculating triadic concepts. The first is founded on the frequent dyadic concepts to produce triadic concepts. While the second operates directly the three-dimensional table to calculate the patterns of a more efficient way by exploiting a ternary enumeration that recursively decomposes the data set into smaller groups. Based on these algorithms, [6] propose Data Peeler which performs better than the previous ones and generalized to n-ary relations. [18] propose an approach of discovering rules applied to dynamic relational graphs that can be encoded in n-ary relations ($n \geq 3$) [19], [5] propose the generalization of the concept of association rule in such a multi-dimensional context.

Study directly related to the discovery of association rules in online analytical processing (OLAP) cubes, [14] who introduced the concept of Meta-*rule-guided mining* based on model rules defined by the users, in order to guide the mining process. The authors propose two algorithms to extract association rules from data cubes. The first extracts these rules from a *Multidimensional OLAP* cube, however, the second focuses on the *Relational OLAP* cube. In a study by [25], the author addresses this problematic according to three types of associations: inter-dimensional, intra-dimensional and hybrid rules. Essentially, he generates a cube according to the desired dimensions; transforms it into a tabular form so as to extract the frequent itemsets, and finally explore the association rules. Consequently, this tabular form does not take advantage of the concept of dimension hierarchy; as a result it is performed during the pre-treatment phase. Moreover, [11] propose *Cubegrades* as a generalization of association rules. They focus on significant changes that affect measures when a cube is modified by: specialization (*Drill-down*), generalization (*roll-up*) or even a mutation (*Switch*). Moreover, they believe that classical association rules are limited only to the *COUNT* aggregate and consequently, can only express changes in the body of the rule. [22] present a method of mining association rules in data warehouses. Based on the multi-dimensional structuration of data, it is a method capable of extracting associations from multiple dimensions at multiple levels of abstraction according to the COUNT measure. However, [3] propose, *OLEMAR* (*On-Line An Environment for Mining Association Rules*) an online environment for mining association rules in data cubes. It enables the extraction of inter-dimensional association rules in data cubes according to a more general indicator than aggregate values provided by the traditional *COUNT* measure.

To our knowledge, except for the work [17] no research has been conducted in the problem of mining triadic association rules from ternary relations. Our work helps address this issue by proposing a new, more effective approach.

3. FORMAL CONCEPT ANALYSIS AND ASSOCIATION RULES

Before we describe our process for extracting triadic association rules we will give in this section some definitions necessary for the understanding this process, namely those of a formal triadic context and its equivalent dyadic context. We illustrate these definitions through an example (Figure 1 (a)) triadic context initially proposed by [7] and taken up and adapted by [17]. This example shows a data cube with three dimensions *CUSTOMERS, SUPPLIERS* and *PRODUCTS.* It concerns a group $K_1$of customers (numbered from 1 to 5) those purchases from suppliers in K_2 (Peter, Nelson, Rick, Kevin and Simon) products found in K_3 (accessories, books, computers and digital cameras). For example, the value *ab* at the intersection of the K column and the first row means that the customer 1 buys from the supplier *K*products *a* and *b*. Figure 1 (b) shows the dyadic context (obtained from the triadic context) where the columns (attributes) are $(a_j, a_k) \in K_2 \times K_3$, generally rated $a_j \times a_k$ or simply a_j-a_k. We often use the simplified notations for sets as well as tuples (e.g., 125 stands for {1,2,5}, *ab* for {*a*,*b*}, and P-aP-dN-dR-aK-a for {$P \times a$, $P \times d$, $N \times d$, $R \times a$, $K \times a$}).

K	P	N	R	K	S
1	abd	abd	ac	ab	a
2	ad	bcd	abd	ad	d
3	abd	d	ab	ab	a
4	abd	bd	ab	ab	d
5	ad	ad	abd	abc	a

(a)

$K^{(1)}$	P				N				R				K				S			
	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c	d	a	b	c	d
1	1	1		1	1			1	1			1	1	1			1			
2	1			1		1	1	1	1	1		1	1			1				1
3	1	1		1				1	1	1			1	1			1			
4	1	1		1		1		1	1	1			1	1						1
5	1			1	1			1	1	1		1	1	1		1	1			

(b)

Figure. 1. (a) : A triadic context K := (K_1, K_2, K_3, Y), with $K1 = \{1, 2, 3, 4, 5\}$ (customers), $K_2 = \{P, N, R, K, S\}$ (suppliers) and $K_3 = \{a, b, c, d\}$ (products). **(b)** : The dyadic context $K^{(1)}$ extracted from K. Customers 1 to 5 purchase from suppliers **P**eter, **N**elson, **R**ick, **K**evin and **S**imon the products: **a**ccessories, **b**ooks, **c**omputers and **d**igital cameras.

3.1 Dyadic Formal Contexts and Triadic Formal Contexts

Definition :A formal (dyadic) context is a triple $K:=(G,M,I)$ where G, M and I stand for a set of objects, a set of attributes, and a binary relation between G and M respectively. For $A\subseteq G$ and $B\subseteq M$ two subsets $A'\subseteq M$ and $B'\subseteq G$ are defined as the set of attributes common to objects in A and the set of objects sharing all the attributes in B, respectively.

Formally, the derivation ' is defined by $:A':=\{a\in M|oIa\forall o\in A\}$ $and B':=\{o\in G|oIa\forall a\in B\}$.

This setting defines a pair of mappings (',') between the powerset of G and the powerset of M, which is a Galois connection. The induced closure operators (on G and M) are both denoted by ''. For example, the closure of R-b is :

$$(R-b)''=((R-b)')'=\{2,3,4,5\}'=\{P-a,P-d,N-d,R-a,R-b,K-a\}.$$

Definition:A *formal concept fc* is a pair (A,B) with $A\subseteq G$, $B\subseteq M$, $A=B'$ and $B=A'$. The set A that we denote by Ext(fc) is called the *extent* of fc while B is its *intent* denoted by Int(fc). A formal (dyadic) concept corresponds to a maximal rectangle (full of crosses / ones) in the dyadic context. In the closed *itemset* mining framework [20], G, M, A and B correspond to the transaction database, the set of items (products), the closed *tidset* and the closed *itemset* respectively.

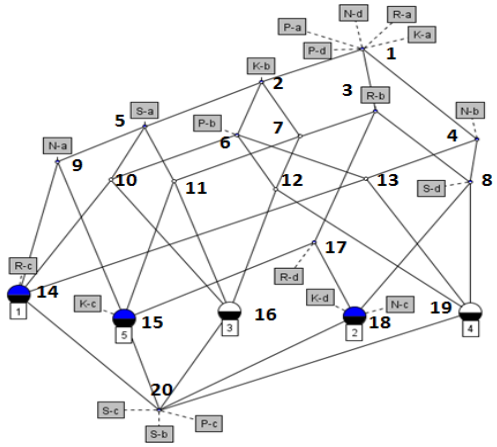

Figure.2 Concept lattice generated from the dyadic context $K^{(1)}$.

The set B(K) of all concepts of the context K, partially ordered by:

$(X_1, Y_1) \leq (X_2, Y_2) \Leftrightarrow X_1\subseteq X_2$ forms a complete lattice, called concept lattice of K and de-noted by B(K). A concept (X_2, Y_2) is called successor of a concept (X_1,Y_1) whenever$(X_1, Y_1) < (X_2, Y_2)$ holds. In this case, (X_1,Y_1) is called predecessor of (X_2,Y_2). The immediate precedence relation $\prec$ is the transitive reduction of $<$, i.e. $fci \prec fc_j$ if $fc_i<fc_j$ and there is no concept between fc_i and fc_j. We then call c_i an immediate predecessor of fc_j and fc_j an immediate successor of c_i. Figure 2 shows the Hasse diagram of the concept lattice corresponding to the dyadic context (Figure 1(b)).

The labeling of the diagram is reduced so that the extent of a concept represented by a node n is given by all labels in G (in white square) from the node n downwards, and the intent by all labels in M (in grey rectangles) from n upwards. For example, node #4 with the label N-b represents the concept ({1, 2, 4}, {P-d, P-a, N-d, R-a, K-a, N-b}).

The top of the lattice (supremum) (P-a, P-d) shows that the supplier P supplies products a and d to all customers and The bottom of the lattice exhibits three attributes: S-b, S-c and P-c which indicate that no customer asks for books or computers from supplierS or for computers from supplier P.

3.2 Dyadic Concept and Triadic Concept

Triadic concept analysis was originally introduced by [16] and [24]as an extension to formal concept analysis, to analyze data described by three setsK_1 (objects), K_2 (attributes) and K_3 (conditions) together with a 3-ary relation $Y \subseteq K_1 \times K_2 \times K_3$.K: = (K_1, K_2, K_3, Y) is called a *triadic context*. A triple (a_1, a_2, a_3) in Y *means* that object a_1 possesses attribute a_2 under condition a_3. For e.g., the Figure 1(a) is a triadic context (K_1, K_2, K_3, Y) representing the purchase of customers in $K_1 = \{1, 2, 3, 4, 5\}$ from suppliers in $K_2 = \{P, N, R, K, S\}$ of products in $K_3 = \{a, b, c, d\}$.

Definition: A *triadic concept* (also called *closed tri-set* or 3-*set* for short) of a triadic context is a triple (A_1, A_2, A_3) with $A_1 \subseteq K_1, A_2 \subseteq K_2, A_3 \subseteq K_3$ and $A_1 \times A_2 \times A_3 \subseteq Y$. It represents a maximal cuboid full with ones (or crosses). The subsets A_1 ,A_2 and A_3 are called the *extent*, the *intent* and the *modus* of the triadic concept (A_1, A_2, A_3) respectively. From Figure 1, we can extract e.g., the closed tri-sets(12345, *P RK, a*) and (14,*P N, bd*). The tri-set (135, *P N, d*) is not closed since its extent can be augmented without violating the ternary relation to get (12345,*P N, d*).

Let K: = (K_1, K_2, K_3, Y) be a triadic context and $\{i, j, k\} = \{1, 2, 3\}$ with $j < k$. For $X_i \subseteq K_i$and $(X_j, X_k) \subseteq K_j \times K_k$[1], an$^{(i)}$ -derivation extending the derivation (see Subsection 3.1) is defined as follows Lehmann (1995):

$$X^{(i)}_i := \{(a_j, a_k) \in K_j \times K_k \mid (a_i, a_j, a_k) \in Y \ \forall a_i \in X_i\}.$$

$$(X_j, X_k)^{(i)} := \{a_i \in K_i \mid (a_i, a_j, a_k) \in Y \ \forall (a_j, a_k) \in X_j \times X_k\}.$$

For example the $^{(1)}$-derivation in a triadic context K: = $(K_1, K_2, K_3, Y)$is the derivation in the dyadic context $K^{(1)}$:= $(K_1, K_2 \times K_3, Y^{(1)})$ with $(a_i, (a_j, a_k)) \in Y^{(1))} \Leftrightarrow (a_i, a_j, a_k) \in Y$.

In practice, the *attribute*×*condition* set can be restricted to the existing combinations (a_j, a_k) instead of all possible ones.

The set of triadic concepts can be ordered and form a complete trilattice [4], [16]. Indeed, for each $i \in \{1, 2, 3\}$, the relation $(A_1, A_2, A_3) \lesssim i (B_1, B_2, B_3) \Leftrightarrow Ai \subseteq Bi$ is a quasi-order whose equivalence relation $\sim i$ is given by: $(A_1, A_2, A_3) \sim i (B_1, B_2, B_3) \Leftrightarrow A_i = B_i$.

These three quasi-orders satisfy the following *antiordinal dependencies*: for$\{i, j, k\} = \{1, 2, 3\}$, $(A_1, A_2, A_3) \lesssim i (B_1, B_2, B_3)$ and $(A_1, A_2, A_3) \lesssim j (B_1, B_2, B_3)$ imply $(B_1, B_2, B_3) \lesssim k (A_1, A_2, A_3)$ for all concepts (A_1, A_2, A_3) and (B_1, B_2, B_3).

[1] *We write* $(X_i, X_k) \subseteq K_i \times K_k$*wich means* $X_i \subseteq K_i$ *and* $X_k \subseteq K_k$

4. TRIADIC ASSOCIATION RULES EXTRACTION

In the following text we present the gait we propose illustrated with examples on the basis of formal definitions. Several approaches to research concepts and triadic analysis are shown in literature. [17] consists of input data represented as a formal triadic context transformed into a dyadic context (figure 3). Next, a Galois lattice is constructed and requires the generation of

dyadic concepts and dyadic generators and to order the succession of the latter. Triadic concepts are then generated from dyadics and triadic generators from dyadics. Once these two sets have been assembled, it is possible to extract triadic association rules. These transactions have a resulting exponential complexity. These operations give good results, however, they are complex because they handle two sets of very large size, the set of dyadic concepts and the set of dyadic generators.

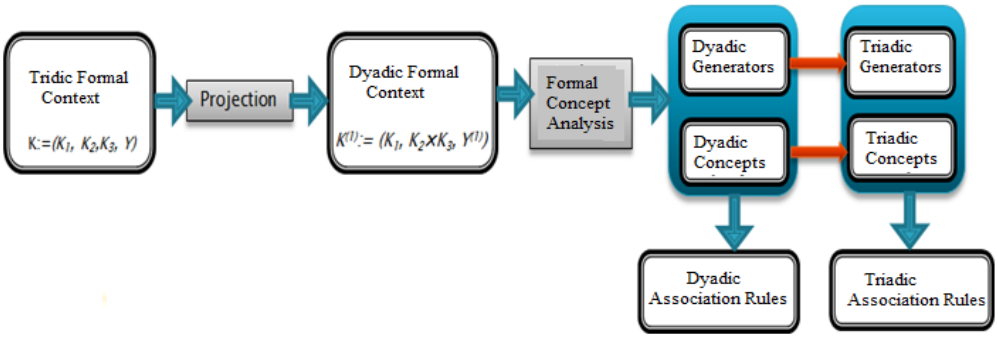

Figure 3 Missaoui et Kwuida approach.

Our approach (figure 4) is based on the same theoretical basis that the proposed approach by [17].Nevertheless, our approach easier extraction. We have in input a formal triadic context $K := (K_1, K_2, K_3, Y)$.We subsequently project the set of properties on the set of conditions to obtain the dyadic equivalent formal context $K^{(1)} := (K_1, K_2 \times K_3, Y^{(1)})$. Then the dyadic association rules are extracted by applying the formal concept analysis. Finally from these last, we apply the algorithms that we have proposed for the research of triadic association rules in their various forms: Biedermann Conditional Attribute Association Rule (BCAAR) Biedermann Attributional Condition Association Rule (BACAR) [7], [4],[17],[8],[12].

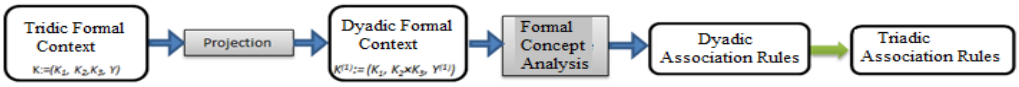

Figure 4 Proposed approach.

Let *(G, M, I)* a formal dyadic context, an association rule has the form $r: B \rightarrow C(s,c)$ *where* $B,C \subseteq M$ *(itemsets) with* $B \cap C = \emptyset$. The parameter *s* that we denote also : $sup(r) = |B' \cap C'| / |G|$ is called the support of the rule *r* while *c* that we denote $conf(r) = |B' \cap C'| / |B'|$ is its confidence[1]. An implication is an association rule whose confidence is equal to 1.

Many studies in Formal Concept Analysis were conducted on the generation of concise representations of rules [10], [15] such as informative rules, Guigues-Duquenne base (stem base),[8],[9], generic base [20], and Luxenburger base. The notions of generator [20],[21], and pseudo-intent [8], [9], play a key role in such studies.

A generic basis [20], associated with a given context is a concise representation of implications $B \rightarrow B'' \backslash B$ such that *B* is a minimal generator for *B"*. An informative basis for approximate association rules takes the following form: $B \rightarrow \text{Int}(cfi) \backslash B$ where B is a minimal generator of Int(*cf*) and *cfi* is an immediate predecessor of *cfi*. The support of r is equal to |Ext (*cfi*)|while its confidence is equal to |Ext(*cfi*)|/|Ext(*cf*)|. For example, there are four (dyadic)generators for the intent of the concept #14 (Figure 5).

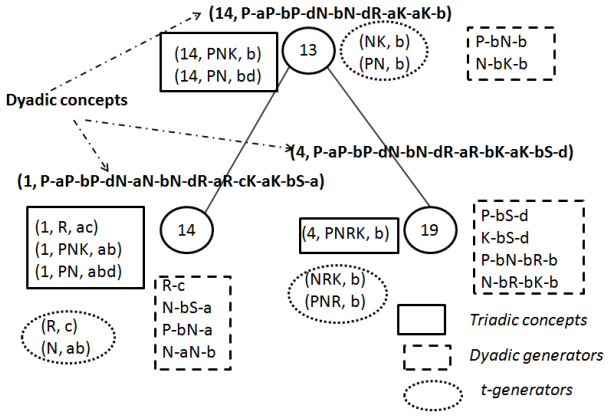

Figure. 5 – Dyadic Concepts, Triadic Concepts and Generators.

The generator *R-c* will produce the implication *R-c →P-aP-bP-dN-aN-bN-dR-aK-aK-bS-a (0.2,1)* while the generator *P-bN-b* associated with node #13 will generate the association rule:

P-bN-b → R-bS-d (0.2, 0.5) when node #19 is considered.

According to the bibliographic study that we conducted, [4] was the first to study the problem of extracting implications in triadic contexts. A triadic implication has the following form:

$$(A \rightarrow D)_C$$

and holds if "whenever *A* occurs under all conditions in *C*, then *D* also occurs under the same conditions". Later on, [7]extended the work of Biedermann and defined three types of implications:

1. **Attributes x Conditions Implications (*AxCIs*):**

An attribute x condition implication (*AxCI*) has the form $A \rightarrow D$, where *A* and *D* are subset of $K_2 \times K_3$. Such implications (rather dyadic) are extracted from the binary context $K^{(1)}$.

E.g. The implication *R-c → P-a P-b P-d N-a N-b N-d R-a K-a K-b S-a* (0.2, 1) is an *AxCI* obtained from the node #14 in Figure 5.

2. **Conditional Attribute Implications (CAIs):**

A conditional attribute implication (*CAI*) takes the form: $A \xrightarrow{C} D$ where *A* and *D* are subsets of K_2 And *C* is a subset of K_3. It means that "*A* implies *D under all conditions in C, then D also occurs under the same conditions and in particular, for any subset in C". Such implication is linked to Biedermann definition of triadic implication As fellow:*

$$A \xrightarrow{C} D \Leftrightarrow (A \rightarrow D)_{C_1} \text{for all } C_1 \subseteq C.$$

E.g. The implication $N \xrightarrow{ad} P$ holds since $(N \rightarrow P)_{C_1}$is true for each $C_1 \subseteq \{a, b\}$. Although $(N \rightarrow P)_{abd}$ holds, $N \xrightarrow{abd} P$ is not true since $(N \rightarrow P)_{C_1}$is not true for each $C_1 \subseteq \{a, b, d\}$ and in particular is not true for $C_1 \in \{b, bd\}$.

3. Attributional Condition Implications (ACIs) :

An Attributional condition (ACI) is an exact association rule of the form $A \xrightarrow{C} D$,where A and D are subsets of K_3 and C is a subset of K_2.

E.g. Using our example, The CAI $N \xrightarrow{ad} P$ states that whenever Nelson supplies accessories and digital cameras (or any one of this two products), then Peter does so.

The ACI : $b \xrightarrow{NP} d$ holds since whenever books are supplied by both Peter and Nelson, then digital cameras are also provided by all those two supplier.

To distinguish triadic association rules (and implications) with Bidermann's meaning from their extensions defined by [7], we will use the prefixe**B** for the first group of patterns. The passage from dyadic association rules to triadic association rules (Figure 4) is carried out using three procedures TRIAR, BCAAR BACAR that we have proposed in the form of three algorithms as we will be rolling out in detail in the following and illustrate with examples.

Algorithme 1 : Main procedure. Computation of triadic association rules (ARs), including implications.

```
1: Procedure TRIAR(D)
2: In : D : a set of dyadic rules where each rule is a tuple : (LHS, RHS, s, c).
3: Out : Σ :triadic ARs where each rule is a tuple (L, R, C, t, s, c) representing association
   rules with left hand-side L, right hand-side R, condition C, type t (= 1 or 2 for BCAAR and
   BACAR resp., including implications), and quality measures : support s and confidence c.
4: Σ ← ∅;
5: for RL = (LHS, RHS, s, c) in D do
6:    {A_L and M_L store the attributes and conditions resp. of the left-hand side LHS of the
      current dyadic rule RL}
7:    A_L ←DISTINCTA(LHS)  {Collect distinct Attribute values in LHS}
8:    M_L ←DISTINCTM(LHS)  {Collect distinct Modus values in LHS}
9:    if Size(A_L) × Size(M_L) = Size(LHS) then
10:      Σ ← Σ ∪ {(BCAARS(A_L, M_L, RHS), 1, s, c)}  {adding BCAARs}
11:      Σ ← Σ ∪ {(BACARS(A_L, M_L, RHS), 2, s, c)}  {adding BACARs}
12: return Σ
```

The main procedure TRIAR (Algorithm 1) is based on two other procedures BCAAR and BACAR (algorithms 2 and 3) to produce all the triadic association rules. We have a set of dyadic association rules *(D)* as input in TRIAR where each rule has the following form *(LHS, RHS, s, c)* representing respectively (the left part of the rule, the right part of the rule the support and the confidence). **Example** : The rule $(\mathrm{P-a} \rightarrow \mathrm{S-a})$ (sup = 0.27 ; conf = 0.27) has the fellowing form (P-a,S-a,0.27,0.27).

At output of the procedure TRIAR we have a set of triadic association rules ($\sum$) where each rule has the following form *(L, R, C, t, s, c)* which represent (the left part of the rule, the right part of the rule the condition of the rule, the rule type (1 for BCAAR, 2 for BACAR), the support and confidence).**Example :**The BCAAR ($N \xrightarrow{ad} P$(sup=0.40; conf=1.0)) has the fellowing form *(N,P,ad,1,0.40,1.0).*

TRIAR corresponds to a screening procedure that can tell if a dyadic association rule is eligible to become a triadic association rule or not. For example, the dyadic rule$(\mathrm{P-a} \rightarrow \mathrm{S-a})$ (sup = 0.27 ; conf = 0.27), from line 5 to 8 of Algorithm 1, we create two set *AL* and *ML* which contain the distinct attributes and distinct conditions of the left part of the rule *LHS*. Therefore, *AL* = P, *ML* = a and their size is equal to 1. This implies that the product *Size (AL) x Size (ML) = 1* (line 9) is equal to *Size (LHS),* so this rule is eligible to become a triadic rule.

Lines 10 and 11 of Algorithm 1 involve both *BCAAR* and *BACAR* procedures to produce both types of triadic rules.

Algorithme 2 : Computation Biedermann Conditional Attribute Association Rule BCAARs type : 1, Form : $(A \rightarrow D)_C$

1: **Procedure** BCAARs(A_L, M_L, RHS)
2: **In :** A_L, M_L, RHS are defined upper.
3: **Out :** a $BCAAR$ is a tuple (L, R, C) representing association rule between attributes with left hand-side L, right hand-side R, condition C belongs to conditions set.
4: $A_R \leftarrow \emptyset$; $Temp \leftarrow \emptyset$
5: **for** e in RHS **do**
6: **if** MODUS$(e) \in M_L$ **then**
7: {MODUS(e) is the condition of the current element in RHS, (e.g. $K - a \in RHS$, MODUS$(K - a) = a$)}
8: $Temp \leftarrow Temp \cup \{e\}$
9: **if** $Temp \neq \emptyset$ **then**
10: Group elements of $Temp$ in buckets $B = b_1, ..., b_n$ with a common attribute part (e.g.$K - a, K - b$)
11: **for** elem in B **do**
12: **if** $Size(elem) = Size(M_L)$ **then**
13: $A_R \leftarrow A_R \cup \{Attr(elem)\}$ {Attr(elem) is the attribute common to elements in the current bucket, (e.g. K)}
14: **if** $A_R \neq \emptyset$ **then**
15: **return** (A_L, A_R, M_L)
16: **out**

BCAAR procedure (Algorithm 2), after initialization parameters (lines 2-4), we take the right-hand side *RHS* of the rule (line 5), which corresponds to *S-a* in our example, and we are looking *Modus* this item corresponding to the condition *Modus (S-a) = a*, which is included in the set *ML*. Since this condition is satisfied. The temp variable is assigned the element *(S - a),* then line 10, we group together noted in a container *B* elements that have the same part attribute in our example *(S - a)* is contained in *(B).*Algorithm 2 checks (lines 11-12) for each element in *(B)* if the size of this element is equal to the size of *ML*. In our example, these two entities are equal. The rule consists of the triplet *(AL, AR, ML) = (P, S, a)* is then formed, which, it is added to the type, support and confidence. The result is: BCAAR$(P \rightarrow S)_a$, type=1, Sup=0.27 and Conf=0.27. Once this rule added to all BCAAR. This is the exit point of the algorithm2.

The *BACAR* procedure is translated by Algorithm 3. Returning to our example of the rule *(P-a;S-a; 0.27;0.27).* After the initialization of variables *(A_L, M_L, RHS)* which take the values *(P, a, S-a)*, the M_R and *RHS* variables are initialized (lines 2 and 4), condition (lines 5-8) if the attribute of the current element *(ATTRIB (S-a) = S)* does not belong to all A_L (equal to *P*), then the rule cannot become BACAR.

Algorithme 3 : Computation Biedermann Attributional Condition Association Rule BA-CARs type : 2, Form : $(A \rightarrow D)_C$

Procedure BACARs(A_L, M_L, RHS)
In : A_L, M_L, RHS are defined upper.
Out : a $BACAR$ is a tuple (L, R, C) representing association rule between conditions with left hand-side L, right hand-side R, condition C belongs to attributes set.
$M_R \leftarrow \emptyset$; $Temp \leftarrow \emptyset$
for e in RHS **do**
  if ATTRIB$(e) \in A_L$ **then**
    {ATTRIB(E) is the attribute of the current element in RHS, (e.g. $K - a \in RHS$,ATTRIB$(K - a) = K$)}
    $Temp \leftarrow Temp \cup \{e\}$
    if $Temp \neq \emptyset$ **then**
      Group elements of $Temp$ in buckets $B = b_1, ..., b_n$ with a common attribute part (e.g.$K - a, P - a$)
      for elem in B **do**
        if $Size(elem) = Size(A_L)$ **then**
          $M_R \leftarrow M_R \cup \{Cond(elem)\}$ {Cond(elem) is the attribute common to elements in the current bucket, (e.g. a)}
if $M_R \neq \emptyset$ **then**
  return (M_L, M_R, A_L)
out

4. EXPERIMENTS

In the last article from [17], the authors argue that there is no implementation or algorithms that handle this type of association rules. Only their works present an algorithm empirically tested on synthetic data.

The obtained results in their work shown that the process the more expensive in terms of time is the generation of triadic concepts and triadic generators from dyadic concepts and dyadic generators. However, in our approach we take input directly dyadic association rules, and we extract our triadic association rules bypassing this step. So the set that we have as input (set of dyadic association rules) is different than the set that [17] have as input (set of dyadic concepts and dyadic generators).

We set up an experimental protocol. We chose the mushroom database[2]which is a reference in the literature and in the field of association rules mining. We pre-treated the database by pruning elements 58 elements from 8124 which included missing values, and then we turned into three triadic contexts. The first contains a quarter of database objects, including 2,014 objects, 16 attributes and 8 conditions noted (*QUART*) and the second half contains the basic objects 4028 is noted (*DEMI*), 16 attributes and 8 conditions, and finally the third dataset contains the entire database with 8056 objects and the same number of attributes and conditions noted (TOT). We show the effectiveness of our algorithms on these three datasets.

We have chosen a real database because the data are highly correlated, which is constitute deal cases more difficult than synthetic databases. Our algorithms have been developed and integrated Lattice Miner[3] is a software tool for visualization and manipulation of concept lattice and giving them a new dimension.

[2] *http://archive.ics.uci.edu/ml/datasets/Mushroom*
[3] *http://sourceforge.net/projects/lattice-miner/*

	QUART	DEMI	TOT
RAD	143192	265649	579199
RC	2100	3331	10181
BCAAR	259	266	2378
BACAR	141	152	81

TAB.1 Number of association rules by rule type

	QUART	DEMI	TOT
RAD	211	532	4330
RC	0,01	0,022	0,073
BCAAR	0,001	0,001	0,002
BACAR	0,001	0,001	0,001

TAB. 2 Execution time by rule type (s).

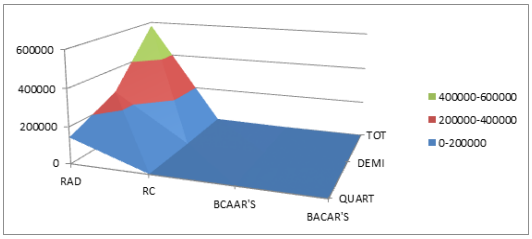

Figure 6. Association rules by rule type.

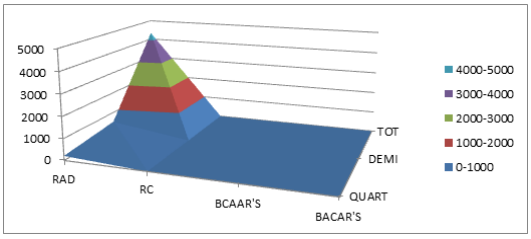

Figure 7. Execution times by rule type (s)

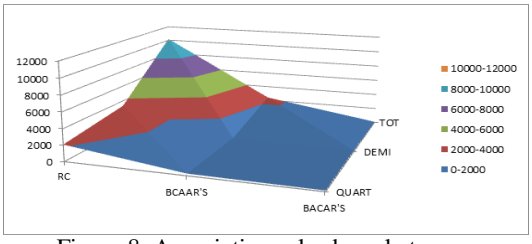

Figure 8. Association rules by rule type.
(without dyadic association rules RAD)

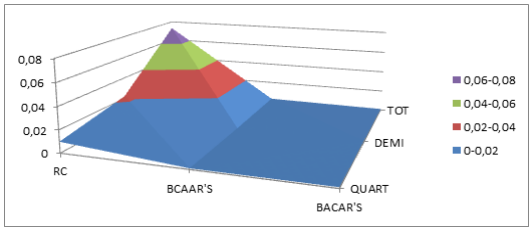

Figure 9. Execution times by rule type (s)
(without dyadic association rules RAD)

The tests were performed on a computer with the following configuration Windows 7 (64bit), Intel(R) Core(TM)7 CPUQ720 @1.60GHZ 1.60GHZwith 4Gb of RAM. For example, the *Mushtot* dataset, we elected in only *73 ms* the number of *10.181* candidates dyadic (RC) association rules to become triadic association rules and that from the initially *579.199*dyadic association rules. Finally, we got *2378BCAAR* and *81BACAR*in only *2 ms*. These results show that the number of triadic rules and considerably smaller than the dyadic association rules average 4 triadic association rules were obtained for 1000 dyadic association rules. In addition, the response times calculated are very encouraging us for future works . Tables (1 and 2) and Figures 6, 7, 8, and 9 summarize all the tests we have carried out and show the performance in terms of execution time and small number of triadic association rules obtained.

5. CONCLUSIONS

In this paper, we proposed an original approach which exploits ideas from the Formal Concept Analysis to generate triadic association rules and bring a new, less expensive solution in terms of time. Through the process and the proposed algorithms we have shown how to get the two types of triadic association rules (BCAARs and BACARs) which are fewer and more compact than dyadic association rules while convey a richer semantics. We showed through experiments conducted on real, dense and highly correlated datasets, the performance of our algorithms.

The prospects of this work are numerous. One of the most important is to generalize the proposed algorithms to polyadic association rules from n-ary relations. This can lead us to the discovery of association rules in OLAP hypercubes for example. Another perspective is to apply these methods of association rules mining in the analysis of communities in social networks, particularly in the field of detection groups.

REFERENCES

[1] Agrawal, R. et R. Srikant (1994). Fast algorithms for mining association rules in large databases.In J. B. Bocca, M. Jarke, et C. Zaniolo (Eds.), VLDB'94, Proceedings of 20thInternational Conference on Very Large Data Bases, September 12-15, 1994, Santiago de Chile, Chile, pp. 487–499. Morgan Kaufmann.

[2] Barbut, M. et B. Monjardet (1970). Ordre et classification, Algèbre et combinatoire. Paris :Hachette.

[3] Ben Messaoud, R., S. LoudcherRabaseda, R. Missaoui, et O. Boussaid (2008). Olemar : An online environment for mining association rules in multidimensional data.

[4] Biedermann, K. (1997). How triadic diagrams represent conceptual structures. In ICCS, pp.304 317.

[5] Cerf, L., J. Besson, T. K. N. Nguyen, et J.-F. Boulicaut (2013). Closed and Noise-TolerantPatterrns in N-ary Relations. Data Mining and Knowledge Discovery 26(3), 574–619.

[6] Cerf, L., J. Besson, C. Robardet, et J.-F. Boulicaut (2008). Data peeler :Contraint-based closedpattern mining in n-ary relations. In SDM, pp. 37–48.

[7] Ganter, B. et S. A. Obiedkov (2004). Implications in triadicformalcontexts. In ICCS, pp.186–195.

[8] Ganter, B. et R.Wille (1999). Formal Concept Analysis : Mathematical Foundations. Springer-Verlag New York, Inc. Translator-C. Franzke.

[9] Guigues, J.-L. et V. Duquenne (1986). Familles minimales d'implications informatives résultantd'un tableau de données binaires. Mathématiques et Sciences Humaines 95(1), 5–18.

[10] Hamrouni, T., P. Valtchev, S. B. Yahia, et E. M. Nguifo (2007). About the lossless reductionof the minimal generator family of a context. In ICFCA, pp. 130–150.

[11] Imielinski, T., L. Khachiyan, et A. Abdulghani (1999). Cubegrades : Generalizing associationrules. Data Mining and Knowledge Discovery 6, 2002.

[12] Jäschke, R., A. Hotho, C. Schmitz, B. Ganter, et G. Stumme (2006). Trias - an algorithm formining iceberg tri-lattices. In ICDM, pp. 907–911.

[13] Ji, L., K.-L. Tan, et A. K. H. Tung (2006). Mining frequent closed cubes in 3d datasets. InVLDB, pp. 811–822.

[14] Kamber, M., J. Han, et J. Y. Chiang (1997). Using data cubes for metarule-guided mining ofmulti-dimensional association rules. Technical report.

[15] Kryszkiewicz, M. et M. Gajek (2002). Concise representation of frequent patterns based ongeneralized disjunction-free generators. In PAKDD '02 : Proceedings of the 6th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining, London, UK, pp.159–171. Springer-Verlag.

[16] Lehmann, F. et R. Wille (1995). A triadic approach to formal concept analysis. In ICCS, pp.32–43.

[17] Missaoui, R. et L. Kwuida (2011). Mining triadic association rules from ternary relations. pp.204–218.

[18] Nguyen, K. N. T., L. Cerf, et J.-F. Boulicaut (2010). Sémantiques et calculs de règles descriptivesdans une relation n-aire. In 26èmes Journées de Bases de Donnéees Avancées BDA'10,pp. 1–20.

[19] Nguyen, T. K. N. (2012). Generalizing Association Rules in N-aryRelations : Application toDynamic Graph Analysis. Thése de doctorat en informatique, INSA de Lyon.

[20] Nicolas, P. (2000). Data Mining : algorithmes d'extraction et de réduction des règles d'associationdans les bases de données. Ph. D. thesis, Université Blaise Pascal Clermont-Ferrand.

[21] Szathmary, L., P. Valtchev, A. Napoli, et R. Godin (2008). Constructing iceberg lattices from frequent closures using generators. In Proceedings of the 11th International Conference onDiscovery Science, DS '08, Berlin, Heidelberg, pp. 136–147. Springer-Verlag.

[22] Tjioe, H. C. et D. Taniar (2005). Mining association rules in data warehouses. IJDWM 1(3),28–62.

[23] Wille, R. (1982). Restructuring lattice theory : An approach based on hierarchies of concepts.In I. Rival (Ed.), Ordered Sets, pp. 445–470. Reidel, Dordrecht-Boston.

[24] Wille, R. (1995). The basic theorem of triadic concept analysis. Order 12(2), 149–158.

[25] Zhu, H. (1998). On-line analytical mining of association rules.