PONTIFICAL CATHOLIC UNIVERSITY OF MINAS GERAIS

Graduate Program in Informatics

Julio Cesar Vale Neves

APPLYING BINARY DECISION DIAGRAM TO EXTRACT CONCEPTS AND ASSOCIATION RULES FROM A TRIADIC CONTEXTS

Belo Horizonte

2021

Julio Cesar Vale Neves

APPLYING BINARY DECISION DIAGRAM TO EXTRACT CONCEPTS AND ASSOCIATION RULES FROM A TRIADIC CONTEXTS

Thesis presented to the Graduate Program in Informatics of the Pontifical Catholic University of Minas Gerais in partial fulfillment of the requirements for the degree of Doctor in Informatics.

Advisor: Prof. Dr. Mark Alan Junho Song

Belo Horizonte

2021

FICHA CATALOGRÁFICA
Elaborada pela Biblioteca da Pontifícia Universidade Católica de Minas Gerais

N518a Neves, Julio Cesar Vale
Applying binary decision diagram to extract concepts and association rules from a triadic contexts / Julio Cesar Vale Neves. Belo Horizonte, 2021.
116 f. : il.

Orientador: Mark Alan Junho Song
Tese (Doutorado) – Pontifícia Universidade Católica de Minas Gerais. Programa de Pós-Graduação em Informática

1. Algorítmos computacionais. 2. Matemática - Técnica. 3. Circuitos lógicos. 4. Conceitos - Análise. 5. Redes sociais on-line. 6. Percepção de padrões. 7. Banco de dados da Web. 8. Usuários da Internet. I. Song, Mark Alan Junho. II. Pontifícia Universidade Católica de Minas Gerais. Programa de Pós-Graduação em Informática. III. Título.

CDU: 681.3.011

Ficha catalográfica elaborada por Fernanda Paim Brito - CRB 6/2999

Julio Cesar Vale Neves

APPLYING BINARY DECISION DIAGRAM TO EXTRACT CONCEPTS AND ASSOCIATION RULES FROM A TRIADIC CONTEXTS

Thesis presented to the Graduate Program in Informatics as a partial requirement for the degree of Doctor in Informatics from the Pontifical Catholic University of Minas Gerais.

Belo Horizonte, April 28th, 2021

To my parents, José Maria and Sonia,
and siblings, Luis and Marco,
with my love and respect.
Also to my wife Cinthia,
my son Rodrigo,
and daughter Juliana,
the reasons of my life.

ACKNOWLEDGMENTS

First of all, I would like to thank God for giving me the wisdom, strength, support, and expertise in exploring new knowledge, for the guidance in helping me surpass all the trials I encountered during this journey and for giving me the determination to pursue this research and make it possible.

I want to thank my advisor Professor Mark Alan Junho Song, Professor of the Department of Computer Science, Pontifical Catholic University of Minas Gerais (PUC Minas), Belo Horizonte, Brazil. It has been an honor to be his first Ph.D. student. I appreciate all his contributions of time, ideas to make my experience productive and stimulating. The joy and enthusiasm he has were contagious and motivational for me, even during tough times in the Ph.D. pursuit. I am also thankful for the excellent example he has provided as a successful professor. Thank you!

Thanks to colleagues and professors of PUC Minas for sharing knowledge and academic experience. I also thank everyone from the secretariat and members of the program collegiate, especially Giovana and Mauricio. Also, Professor Luis Zarate for some important advice. Thank you!

Also, I would like to thank the Coordination of High Education Personnel Improvement - Brazil (CAPES), National Council for Scientific and Technological Development (CNPq) and Minas Gerais State Research Foundation (FAPEMIG) for financing this research.

Lastly, I would like to thank my family for all their love and encouragement. For my parents who raised me with a love of science and supported me in all my pursuits. And most of all for my loving, supportive, encouraging, and patient wife Cinthia whose faithful support during the final stages of this Ph.D. Thank you!

“Ask not what your country can do for you –
ask what you can do for your country.”

John Fitzgerald Kennedy

RESUMO

A complexidade dos problemas reais tem chamado a atenção de muitos tipos de pesquisa, especialmente aquelas que lidam com grandes conjuntos de dados. A dificuldade de extrair conhecimento de um banco de dados e devido ao aumento de dados de redes sociais (por exemplo, Facebook e Twitter) e profissionais (por exemplo, LinkedIn), cada vez mais aplicações de análise de dados em ambientes com alto dimensional têm sido discutidas na literatura. A Triadic Concept Analysis (TCA) é uma técnica matemática aplicada à análise de dados em que as relações entre objetos, atributos e condições permitem extrair o conhecimento da base de dados em uma representação hierárquica e sistematizada. É considerada uma importante teoria para formalizar a representação do conhecimento. No entanto, o volume de informações a ser processado pode tornar o TCA impraticável, pois exige recursos computacionais poderosos. Existem alguns algoritmos para extrair conceitos, mas eles não são eficientes em grandes conjuntos de dados porque os custos computacionais se tornam exponenciais.

Este trabalho tem como objetivo adicionar uma nova estrutura de dados, denominada Binary Decision Diagram (BDD), no algoritmo TRIAS para extrair conceitos triádicos em contextos de alta dimensionalidade. O BDD é usado para representar contextos formais, objetos, atributos e condições. Além disto, este representa informações de forma canônica e simplificada para reduzir os recursos necessários para manipular grandes conjuntos de dados, mesma estrutura para problemas de grandes conjuntos de dados. Os experimentos mostram que a abordagem proposta tem um desempenho melhor - até 56 % mais rápido - do que o algoritmo original. Além disso, foram encontrados conceitos que não eram capazes de atingir anteriormente, considerando contextos dimensionais elevados.

Nesta pesquisa apresenta-se o comportamento dos algoritmos TRIAS e TRIAS BDD em diferentes contextos. Dentre os 45 contextos sintéticos triádicos avaliados, entre 500 e 10.000 objetos, em apenas dois o TRIAS BDD apresentou tempo maior de processamento que o algoritmo original (TRIAS). Realizou-se também um estudo do uso de memória RAM para ambos os algoritmos, além da ordenação das variáveis do contexto triádico por atributos e condições para analisar o impacto desta ordenação na geração dos BDD's. De acordo com os resultados obtidos, notou-se que em todos os experimentos o TRIAS BDD utilizou menos quantidade de memória RAM quando comparada ao algoritmo TRIAS. Além disso, quando utilizado as ordenações crescentes e decrescentes, a utilização de memória RAM foi ainda menor quando comparada a abordagem sem ordenação. Os resultados apresentam consumo de até 46% menos memória RAM no algoritmo proposto quando comparado ao algoritmo original.

Palavras-Chave: Análise Formal de Conceitos. Análise Formal Triádica. Diagrama de Decisão Binária. Algoritmo TRIAS.

ABSTRACT

The complexity of real problems has directed the attention of many types of research, especially those handling large datasets. The difficulty to extract knowledge from a database and due to the increase of social network (e.g. Facebook and Twitter) and professional (e.g. LinkedIn) data, more and more applications of data analysis on environments with high-dimensional have been discussed in the literature. Triadic Concept Analysis (TCA) is an applied mathematical technique for data analysis in which the relations between objects, attributes and conditions allow extracting database knowledge in a hierarchical and systematized representation. It is considered an important theory to formalize the representation of knowledge. However, the volume of information to be processed could make TCA impracticable because it demands powerful computational resources. There are some algorithms to extract concepts, but they are not efficient in large datasets because the computational cost is usually exponential.

The main objective of this thesis is to use Binary Decision Diagrams (BDD's) as a structure to represent contexts and extract concepts and implication rules in high-dimensional datasets. BDD's are used to represent formal contexts, objects, attributes and conditions in a canonical and simplified way in order to reduce the resources needed to manipulate large datasets. The experiments show that our approach has better performance – up to 56% faster – than the original algorithm used in this work (TRIAS). Also, our approach discovered concepts that were not able to achieve previously, considering high-dimensional contexts.

This research presents the behavior of the TRIAS and TRIAS BDD algorithms in different contexts. Among 45 triadic synthetic contexts evaluated, between 500 and 10,000 objects, only two TRIAS BDD has a longer processing time than the original algorithm (TRIAS). Also, a detailed study of the RAM memory usage for both algorithms was accomplished, in addition to the ordering of the variables of the triadic context by attributes and conditions to analyze the impact of this ordering on the BDD's creation. According to the results obtained, it was noted that in all experiments, TRIAS BDD used less RAM memory when compared to the TRIAS Algorithm. Furthermore, when using ascending and descending orderings, the use of RAM was even less when compared to the unordered approach. The results show consumption up to 46% less RAM in the proposed algorithm when compared to the original algorithm.

Keywords: Formal Concept Analysis. Triadic Concept Analisys. Binary Decision

Diagram. TRIAS Algorithm.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

ACIs – Attributional Condition Implications

AxCIs – Attributes x Condition Implications

ADDs – Algebraic Decision Diagrams

BACAR – Biedermann Attributional Condition Association Rule

BCAAR – Biedermann Conditional Attribute Association Rule

BDD – Binary Decision Diagram

CAIs – Conditional Attribute Implications

CUDD – Colorado University Decision Diagram

FCA – Formal Concept Analysis

RAM – Random Access Memory

RBAC – Role-Based Access Control

SCGaz – Synthetic Context Generator a-z

ZDDs – Zero-suppressed Binary Decision Diagrams

CONTENTS

1 INTRODUCTION

Currently, advanced technologies have facilitated to store dense volumes of information in many diverse environments. Information previously stored only in textual format, can be found nowadays in different formats such as images, audio clips, video streaming and others. It is perceived an important problem which is to deal with the extraction of information and knowledge from a database that, due to the volume, makes the task complex.

Several areas of knowledge, for example, Data Mining, Ontology, Bioinformatics, Information Retrieval and Formal Concept Analysis (FCA) have been concentrating efforts to address this problem (TANG; HUIA; FONG, 2015)(MAIO et al., 2014)(FAN et al., 2013)(SENATORE; PASI, 2013)(LI; TSAI, 2013)(ZERARGA; DJOUADI, 2013). Also, due to the fast development of network technology (HE et al., 2015) (HE; WANG; HUANG, 2016), more and more approaches to dealing with large datasets (ASHFAQ et al., 2017)(ASHFAQ; HE; GHEN, 2017)(WANG, 2015)(WANG; ASHFAQ; FU, 2015) have been emerging. Although many approaches have been developed and presented, it still not achieving the main purpose which is to extract knowledge from large data sets.

Information has become an important asset in all organizations. Thus, the ability to properly manage information has become the biggest difference for the companies. Referring to Information Science, information is data structured that can provide some knowledge. This same science defines the following cycle: Data $\rightarrow$ Information $\rightarrow$ Knowledge. From this perspective, data generate information that generates knowledge which completes the cycle by generating more data (DAVENPORT; PRUSAK, 1998).

The advance of technology has facilitated the process of collecting and storing data which has resulted in an increase in storage requirements. This amount of data from various sources becomes impractical the analysis of it in order to generate intelligent information. Therefore, techniques are needed to automate or assist the process of obtaining information from data.

Several techniques have been proposed in the area of data mining. These techniques are mostly based on statistical concepts. However, there are also those based on natural or bio-inspired computing. In general, although distinct, they have some points in common, such as the fact that they use the probability distribution to analyze the data. Alternatively, Formal Concept Analysis (FCA) uses structural similarities of data, such as the set inclusion ratio, for the analysis. Thus, it is known that FCA-based methods can provide better results in data analysis by enhancing structural relationships. This argument justifies the analysis and comparison of FCA based methods. This technique basically tries to find the relationship between data. Once a relationship is discovered, it can be expressed through rules.

FCA and TCA (formal analysis of triadic concepts) which are the focus of this thesis, are areas of applied mathematics that use the framework of lattice theory to design and analyze the concept hierarchies (DAVEY; PRIESTLEY, 2001). The starting point of the FCA dates from the early 1980s. More precisely, the FCA began with Rudolf Wille (WILLE, 1982) which proposed a formal framework for the use of lattice theory.

Lately, FCA/TCA has migrated from mathematics to computer science (STUMME, 2002). Initially, works related to FCA/TCA were presented at conferences in the area of Mathematics, while currently, the works have been mostly published in conferences in the Computer Science area. This feature shows a transformation in the view of FCA/TCA: in the past, FCA/TCA was studied only from a theoretical point of view and in recent years it has been explored from a practical point of view, for example beyond the development of theories related to FCA/TCA.

Currently, it investigates its use in various areas of human knowledge, especially in Computer Science. FCA and TCA's applications in computer science include those for information retrieval (CARPINETO; ROMANO, 2004)(KOESTER, 2006) and for mining and knowledge discovery in databases (STUMME; WILLE; WILLE, 1998)(WILLE, 2001). The Conceptual Data Analysis (CARPINETO; ROMANO, 2004) confirms FCA's approach to the Computer Science area by presenting the technique in a more algorithmic rather than theoretical and mathematical as seen in Ganter and Wille (GANTER; WILLE, 1999a).

Despite the benefits that FCA brought which is to transform data into intelligible information (knowledge), it has some limitations. For example, the limitation to represent only the relationship between attribute and objects. Although the dyadic approaches have been successful in many applications, there have been situations suggesting an extension of formal concepts by a third component. The triadic approach (TCA) to FCA is based on a formalization of the triadic relation connecting formal objects, attributes and conditions (LEHMANN; WILLE, 1995a)(WILLE, 1995)(WILLE, 1996). TCA, for example, can provide an approach to solve existing problems among three dimensions data, such as Folksonomy, in which users, tabs, and sources have a relation between them (three dimensions). Both techniques (FCA and TCA) can be used to discover, sort and display concepts (WILLE, 1982)(WU; LEUNG; MI, 2009)(WU, 2008).

However, there are several obstacles before making the FCA / TCA application viable when used in large databases. For example, in FCA the algebraic structure called concept is defined according to a set of objects, a set of attributes and a binary relationship that links attributes and objects. In this case, there is a problem with generating all formal concepts and classifying them hierarchically due to the exponential behavior presented in the worst case (NETO; ZÁRATE; SONG, 2018). When dealing with TCA, the problem is even more evident, since a new dimension is inserted: there is now a set of objects, attributes, conditions and a ternary relationship that interconnects them. Consequently, the computational cost increases even more, making it difficult to apply TCA on high-dimensional bases. This fact justifies several studies and alternative approaches, whether dyadic or triadic, for the generation of formal concepts, the construction of the conceptual lattice and the extraction of rules (SELMANE et al., 2013).

1.1 Motivation and problem

The motivation for this research is defined by the need to expand existing technique to extract knowledge in order to make explicit the interactions between elements and to define patterns that represent the behavior of the network and/or dataset. The goal of several research areas is the formal discovery of valid, understandable and useful knowledge from data. Due to fact that databases are becoming larger, to accomplish it is getting even harder. One of the challenges that impulses this research is finding relationships and rules that describe the behavior of the objects contained in a large dataset, as an example, the access to the social networks, such as Linkedin and Facebook, and the volume of data generated by their users, the collaborative network (Wikipedia), professional network (ResearchGate), among others.

One possible solution to resolve what was described previously, is use Formal Concept Analysis (FCA), which is a technique based on the formalization of the notion of concepts and the structuring of concepts into a conceptual hierarchy. Applying FCA is possible to analyze the data through associations and dependencies of objects and attributes formally described from a dataset (WILLE, 1982) (BERNHARD; RUDOLF, 1999).

However, in a high-dimensional context, FCA has not presented an expected computational behavior. In this way, several papers have been presented in order to overcome this limitation, either by changing existing solutions via code parallelization (ANDREWS, 2011)(KODAGODA; ANDREWS; PULASINGHE, 2017), including additional structures to speed up the extraction of concepts (NETO; ZÁRATE; SONG, 2018), rules (SANTOS et al., 2018), or even performing both inclusion of new data structure and parallelization of the solution (NEVES et al., 2020b).

Thus, in some situations, it is required to describe the condition that connects the relationship between the different objects and their attributes. An extension of the classical (dyadic) FCA, called Triadic Concept Analysis (TCA) was proposed to deal with this situation (LEHMANN; WILLE, 1995b). Although it is from the FCA, the triadic approach is more complex because it deals with three-dimensional data.

In the same situation as we have with the FCA, the TCA approach has to deal with computational processing restrictions in which databases are considered high-dimensional. Researchers are still proposing new approaches for extracting information from data contexts (MISSAOUI; KWUIDA, 2011), but although several algorithms have been proposed in the literature in order to extract information from triadic concepts, neither one directly attacks the high-dimensional contexts (JASCHKE et al., 2006) (CERF; BESSON J.AND ROBARDET; BOULICAUT, 2009) (TRABELSI; JELASSI; YAHIA, 2012b). Therefore, this is an important aspect which motivates the development of this work.

It is important to highlight that a dyadic approach (FCA) allows an analysis that can be used only between objects and attributes. A triadic analysis (TCA) allows the extraction of concepts that are not possible with FCA. TCA, in addition to objects and attributes, allows the inclusion of conditions, which model the relationship between attributes and objects, improving the analysis significantly. Adding conditions expands the possibilities of usage in many applications, supporting the extraction of more relevant information from a given context, allowing a more complete study of the information from a given dataset.

For example, in a dyadic context, where objects are represented by users and attributes by the social networks accessed (Linkedin or Facebook), it is possible to know only which social networks are accessed by each user. However, this information is limited and does not provide information of extreme relevance for decision making.

In a triadic context, in addition to the user object and the social network attribute, conditions can be included. Considering only the inclusion of a condition for example, timeslot, it can offer relevant information about who accesses, which social network in which timeslot. The inclusion of only one condition would generate relevant information, allowing actions to be taken, in that timeslot, for certain users on a social network. A practical application of this model would be, for example, to publicize advertising campaigns to users of Linkedin and Facebook, based on what time users spent most of their time.

Another important point to be observed in this study, is to understand that this is not a new subject. Before the development of computational architectures, mathematical studies were used and were able to work with a certain amount of information, promoting the generation of relevant information. Despite the arrival of computers, it is possible to process information faster, more efficiently and by increasing the number of data to be processed, the use of computers has also allowed a very relevant increase in the amount of data generated. Thus, the technology that has facilitated processing and made it possible to analyze so much data, has created another problem: an increasing amount of data to be worked on, which has gone beyond its processing capacity.

As described previously, the computational limitation widely known and discussed in computer science due to the increasing complexity and increase of databases, has become a new challenge to be studied - which also motivated this research. There are several approaches to solve this problem, however, when applied to large datasets, they often do not generate the expected results. In this work, for example, there are cases where, after 14 days of execution, using the original algorithm, no results were generated. Despite formally solving it, when data are applied to an algorithm, the results were not obtained as expected.

From studies carried out in this research and the available literature, it was observed that the application of projections of triadic contexts in dyadic offers a significant improvement in the processing of the algorithms. The projection basically implies the transformation of the triadic model into a dyadic, generating the same possible combinations, with object, attribute and condition, in a model of objects and attributes, where the attributes are stratified in each of the conditions.

Therefore, the concepts extraction was proposed and rules from high-dimensional triadic contexts using the TRIAS (JASCHKE et al., 2006) as the basis, which, in its essence, projects triadic context into dyadic. Modifications were applied to the original algorithm to support the BDD structure (BRYANT, 1986a), representing data as a BDD. From this representation, several logical operations with BDD were included to extract the concepts.

1.2 Objectives

The following objectives were defined to achieve the contributions:

- Represent triadic contexts using Binary Decision Diagram (BDD) in order to store and manipulate high-dimensional contexts efficiently (AKERS, 1978). In this case, a set of boolean operations was implemented to be able to retrieve objects, attributes and conditions (ANANIAS et al., 2018) (ANANIAS et al., 2019);
- Modify the TRIAS algorithm using BDD structure in order to analyze its efficiency when comparing with the original algorithm;
- Extract association rules from dyadic and triadic formal concepts;

- Random Access Memory (RAM) analysis during execution to understand its behavior;
- Manipulate the order of variables to be inserted in the BDD based on the attributes and conditions' density;
- Time execution comparison between TRIAS and the proposed algorithm TRIAS BDD and between TRIAS BDD and the TRIAS BDD Descending and Ascending order.

Based on the behavior of the algorithm, the objective was to apply a data structure called BDD in order to represent triadic contexts and concepts since the behavior of the algorithm for high-dimensional contexts was not able to return concepts after a defined period of time. Finally, TRIAS was extended in order to extract triadic rules - the original algorithm was able to extract only concepts from a dataset.

1.3 Contributions

The main contribution of this thesis is an approach to extract concepts and rules from high-dimensional triadic datasets using Binary Decision Diagrams based on TRIAS algorithm. As pointed out, one of its potentials lies in the use of TCA and its mathematical foundation that enables the generation of knowledge in the form of formal concepts and rules (NEVES et al., 2020a).

In order to analyze the behavior of algorithms in high-dimensional triadic contexts, triadic contexts were created using a synthetic tool called Synthetic Context Generator a-z (SCGaz). These contexts are available for future use for any researcher who needs to deal with high-dimensional datasets. Modifications to the dyadic synthetic context generator *SCGaz* were applied to generate triadic concepts by adding a third dimension, not developed by the tool. The reducibility rules defined in (RIMSA; SONG; ZÁRATE, 2013) were maintained for triadic contexts. Only the notion of a third dimension to be chosen was added, then the number of dyadic context objects was replicated to the number of defined conditions.

1.4 Thesis outline

The remainder of this thesis is organized as follows. Chapter 2 introduces the main definitions about FCA, TCA, BDD, TRIAS and SCGaz. Chapter 3 describes some important related work. The details of the proposed approach are described in Chapter 4. The results and analysis are presented in Chapter 5. Finally, conclusions and future work are drawn in Chapter 6.

2 LITERATURE REVIEW

2.1 Formal Concept Analysis

FCA is a technique based on formalizing the notion of concept and structuring concepts in a conceptual hierarchy (BERNHARD; RUDOLF, 1999). The FCA relies on lattice theory (BIRKHOFF, 1940) (DAVEY; PRIESTLEY, 2001) to structure formal concepts and enable data analysis. The capability to hierarchize concepts extracted from data turns the FCA an interesting tool for dependency analysis. With the increase of social networks and due to the large amount of data generated by users, the study and improvement of techniques to extract knowledge are becoming increasingly justified. Also, it permits the data analysis through associations and dependencies attributes and objects, formally described, from a dataset.

2.1.1 Formal Context

Formally, a dyadic context is formed by a triple (G,M,I), where G is a set of objects (rows), M is a set of attributes (columns) and I is defined as the binary relationship (incidence relation) between objects and their attributes where $I \subseteq G \times M$ (GANTER; STUMME; WILLE, 2005). Table 1 exemplifies a formal context. In this example, objects correspond to planets, attributes are the characteristics, and the relationship of incidence represents whether or not the planet has that characteristic. A planet has that characteristic if and only if there is an $'X'$ at the intersection between the row and the respective column.

2.1.2 Formal Concepts

Let (G,M,I) be a formal context, $A \subseteq G$ a subset of objects and $B \subseteq M$ a subset of attributes. Formal concepts are defined by a pair (A,B) where $A \subseteq G$ is called extension and $B \subseteq M$ is called intention. This pair must follow the conditions where $A = B'$ and $B = A'$ (GANTER; STUMME; WILLE, 2005). The relation is defined by the derivation operator ($'$):

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

Table 1: Formal Context - Planets

Planets	Small	Medium	Big	Near	Far	Moon_Yes	Moon_No
Mercury	×			×			×
Venus	×			×			×
Earth	×			×		×	
Mars	×			×		×	
Jupiter			×		×	×	
Saturn			×		×	×	
Uranus		×			×	×	
Neptune		×			×	×	
Pluto	×				×	×	

If A ⊆ G, then A' is a set of attributes common to the objects of A. The derivation operator ($'$) can be reapplied in A' resulting in a set of objects again (A''). Intuitively, A'' returns the set of all objects that have in common the attributes of A'; note that $A \subseteq A''$. The operator is similarly defined for the attribute set. If $B \subseteq M$, then B' returns the set of objects that have the attributes of B in common. Thus, B'' returns the set of attributes common to all objects that have the attributes of B in common; consequently, $B \subseteq B''$. An example, given the set of objects A ={Mercury, Venus, Earth, Mars}, applying the derivation operator, we will have A' = {Small, Near}. If we apply the derivation operator again, we will have A'' = {Mercury, Venus, Earth, Mars}. Similarly, if we consider the set of attributes B ={Small, Near} and apply the derivation operator, we will have B' = {Mercury, Venus, Earth, Mars}. Then, if we apply the derivation operator again, we will have B'' = {Small, Near}. Finally, it can be said that {{Mercury, Venus, Earth, Mars}, {Near, Small}} is a concept. All the concepts are described in Table 2 were extracted from Table 1.

Table 2: Concepts extracted from the Planets Formal Context - Table 1

Objects	Attributes
{Mercury,Venus,Earth,Mars,Jupiter,Saturn,Uranus,Neptune,Pluto}	{}
{Mercury,Venus,Earth,Mars,Pluto}	{Small}
{Earth,Mars,Jupiter,Saturn,Uranus,Neptune,Pluto}	{Moon_Yes}
{Mercury,Venus,Earth,Mars}	{Near,Small}
{Earth,Mars,Pluto}	{Small,Moon_Yes}
{Jupiter,Saturn,Uranus,Neptune,Pluto}	{Far,Moon_Yes}
{Mercury,Venus}	{Big,Near,Small}
{Earth,Mars}	{Small,Near,Moon_Yes}
{Pluto}	{Small,Far,Moon_Yes}
{Jupiter,Saturn}	{Big,Far,Moon_Yes}
{Uranus,Neptune}	{Medium,Far,Moon_Yes}
{}	{Small,Medium,Big,Near,Far,Moon_Yes,Moon_No}

2.1.3 Dyadic Rules

Dyadic Rules are dependencies between elements of a set of attributes obtained from formal contexts (R.; L., 2011), formal concepts (K.; W, 2005) and concept lattice (BERTET; MONJARDET, 1999). In our study, the implication rules extraction will be based on formal contexts. Using a formal context (G, M, I) an implication rule would be $A \rightarrow B$ where $A, B \subseteq M$ (A and B are defined, respectively, as the rule's premise and conclusion).

Dyadic Rules rule $A \rightarrow B$ is considered valid for the context (G, M, I) if, and only if, every object that has the attributes of A also has the attributes of B. Formally $\forall g \in G[\forall a \in A\ gIa \rightarrow \forall b \in B\ gIb]$.

Let (G,M,I) be a formal dyadic context. A rule is of the form r : A $\rightarrow$ B (s, c) where $A, B \subseteq M$ (itemsets) with A $\cap$ B = $\emptyset$. The parameter s = supp(r)= $\frac{|A' \cap B'|}{|G|}$ is called the support of the rule r while c = conf(r)= $\frac{|A' \cap B'|}{|A'|}$ is its confidence (AGRAWAL; SRIKANT, 1994).

Table 3 shows the examples of all dyadic rules of Table 1 where rules have, as parameters, support greater than 10.00% and confidence greater than 50.00%. Given the dyadic rule {Moon_Yes} → {Far} with 55.55% support and 71.42% confidence. In this example, it can be said that in 55.55% (support) of cases (5 objects {Jupiter, Saturn, Uranus, Neptune, Pluto} of 9 total) where it has the attribute {Moon_Yes}, it also has the attribute {Far}, and this occurs 71.42% (confidence) in the dataset where it has at least the attribute {Moon_Yes} (5 of 7 objects subset {Earth, Mars, Jupiter, Saturn, Uranus, Neptune, Pluto}).

Another example in Table 3 is the dyadic rule {Small, Moon_Yes} → {Near} with 22.22% support and 66.66% confidence. It can be said that in 22.22% of cases (2 objects of 9 total) where it has the attribute {Small, Moon_Yes}, it also has the attribute {Near}, and this occurs 66.66% (confidence) in the dataset where it has at least the attribute {Small, Moon_Yes} (2 objects of 3).

Implication rules are association rules that have 100% confidence. As an example of implication rules (Table 4), we can consider the context described in Table 1. In this context, every planet which is near the sun is also small. This type of relationship can be described as an implication: $\{Near\} \rightarrow \{Small\}$.

2.2 Triadic Concept Analysis

The TCA was introduced by Lehmann and Wille (LEHMANN; WILLE, 1995b), extends the classic FCA, but a new dimension was added (WILLE, 1995).

Table 3: Example of Dyadic Rules of the Table 1

Dyadic Rule	Support	Confidence
{Small} → {Near}	44.44%	79.99%
{Small} → {Moon_Yes}	33.33%	59.99%
{Moon_Yes} → {Far}	55.55%	71.42%
{Near} → {Moon_No}	22.22%	50.00%
{Near} → {Moon_Yes}	22.22%	50.00%
{Near} → {Small}	44.44%	100.00%
{Small, Moon_Yes} → {Near}	22.22%	66.66%
{Far} → {Moon_Yes}	55.55%	100.00%
{Moon_No} → {Small, Near}	22.22%	100.00%
{Near, Moon_Yes} → {Small}	22.22%	100.00%
{Small, Far} → {Moon_Yes}	11.11%	100.00%
{Big} → {Far, Moon_Yes}	22.22%	100.00%
{Medium} → {Far, Moon_Yes}	22.22%	100.00%

Table 4: Implication Rules extracted from the Planets Formal Context

$A \rightarrow B$
{Near} → {Small}
{Far} → {Moon_Yes}
{Moon_No} → {Small, Near}
{Near, Moon_Yes} → {Small}
{Small, Far} → {Moon_Yes}
{Big} → {Far, Moon_Yes}
{Medium} → {Far, Moon_Yes}

2.2.1 Triadic Formal Context

Formally, a triadic context is defined as a quadruple (K_1,K_2,K_3,Y), where K_1, K_2 and K_3 are sets and Y is a ternary relation between K_1, K_2 and K_3, i.e., $Y \subseteq K_1$ x K_2 x K_3, the elements of K_1, K_2, and K_3 are called (formal) objects, attributes, and conditions, respectively, and $(g,m,b) \in Y$ is read: the object g has the attribute m under the condition b. An example of a triadic context is represented in Table 5 - it was originally presented in (MISSAOUI; KWUIDA, 2011) and adapted on this work. This example shows the dataset with 3 dimensions: Customers, Suppliers and Products. Then we have the Customers {*1,2,3,4,5*} as objects where {*1*} is Rod, {*2*} is Anna, {*3*} is Sarah, {*4*} is Josh and {*5*} is Jim, the Suppliers {*P,N,R*} as attributes where {*P*} is Pearson Limited, {*N*} is Norway Incorporation and {*R*} is Randy Incorporation and the products {*a,b,c,d*} as conditions where {*a*} is antenna, {*b*} is bumper, {*c*} is car cover, {*d*} is door.

Table 5: **Left**: An example of triadic formal context - Customers, Suppliers and Products. **Right**: Dyadic context extracted from $\mathscr{K}$.

$\mathscr{K}$	P	N	R
1	abd	abd	ac
2	ad	bcd	abd
3	abd	d	ab
4	abd	bd	ab
5	ad	ad	abd

$\equiv$

$\mathscr{K}$	P				N				R			
	a	b	c	d	a	b	c	d	a	b	c	d
1	X	X		X	X	X		X	X		X	
2	X			X		X	X	X	X	X		X
3	X	X		X				X	X	X		
4	X	X		X		X		X	X	X		
5	X			X	X			X	X	X		X

2.2.2 Triadic Concepts

Let $\mathscr{K} = (K_1, K_2, K_3, Y)$ be a triadic context, $\{i, j, k\} = \{1,2,3\}$. Based on the triadic context described previously, it gives rise to the following dyadic contexts: $\mathscr{K}^{(1)} = (K_1, K_2$ x $K_3, Y^{(1)})$, see Table 6, $\mathscr{K}^{(2)} = (K_2, K_1$ x $K_3, Y^{(2)})$ - see Table 7, $\mathscr{K}^{(3)} = (K_3, K_1$ x $K_2, Y^{(3)})$ - see Table 8, where $gY^{(1)}\ (m,b) \Longleftrightarrow mY^{(2)}(g,b) \Longleftrightarrow bY^{(3)}(g,m) \Longleftrightarrow (g,m,b) \in Y$.

Table 6: Customers, Suppliers, Products context projected by Suppliers x Products

$\mathscr{K}^{(1)}$	Pa	Pb	Pc	Pd	Na	Nb	Nc	Nd	Ra	Rb	Rc	Rd
1	X	X		X	X	X		X	X		X	
2	X			X		X	X	X	X	X		X
3	X	X		X				X	X	X		
4	X	X		X		X		X	X	X		
5	X			X	X			X	X	X		X

Table 7: Customers, Suppliers, Products context projected by Customers x Products

$\mathscr{K}^{(2)}$	(1,a)	(1,b)	(1,c)	(1,d)	(2,a)	(2,b)	(2,c)	(2,d)	(3,a)	(3,b)	(3,c)	(3,d)	(4,a)	(4,b)	(4,c)	(4,d)	(5,a)	(5,b)	(5,c)	(5,d)
P	X	X		X	X			X	X	X		X	X	X		X	X			X
N	X	X		X		X	X	X				X		X		X	X			X
R	X		X		X	X		X	X	X			X	X			X	X		X

Table 8: Customers, Suppliers, Products context projected by Suppliers x Customers

$\mathscr{K}^{(3)}$	(1,P)	(2,P)	(3,P)	(4,P)	(5,P)	(1,N)	(2,N)	(3,N)	(4,N)	(5,N)	(1,R)	(2,R)	(3,R)	(4,R)	(5,R)
a	X	X	X	X	X	X				X	X	X	X	X	X
b	X		X	X		X	X		X			X	X	X	X
c							X				X				
d	X	X	X	X	X	X	X	X	X	X		X			X

Moreover, based on the structure of triadic contexts, another derivation operators is given as follows. Let $\mathscr{K}$ = (K_1,K_2,K_3,Y) be a triadic context, $\{i,j,k\} = \{1,2,3\}$. $X_i \subseteq K_i, X_j \subseteq K_j$ and $X_k \subseteq K_k$, the (i,j,X_k)-derivation operators are defined by: $\mathscr{K}^{ij}_{X_k} = (K_i,K_j,Y^{ij}_{X_k})$, where $(a_i,a_j) \in Y^{ij}_{X_k}$ if and only if a_i,a_j,a_k are related by Y for any $a_k \in X_k$. Then, the basic and the important definition, triadic concept, is given as follows. For $X_i \subseteq K_i$ and $X_k \subseteq K_k$ with {i,j,k} = {1,2,3}, let $A_j = X_i^{(i,j,X_k)}$, $A_i = A_j^{(i,j,X_k)}$ and $A_k = (A_i\text{x}A_j)^{(k)}$. As an example, we have $X_3 = \{a,d\} \subseteq K_3$, we obtain the dyadic context $\mathscr{K}^{12}_{X_3} = (K_1,K_2,Y^{12}_{X_3})$ - Table 9.

Table 9: Table $\mathscr{K}^{12}_{X_3}$ projected where X_3 = {a,d}

$\mathscr{K}^{12}_{X_3}$	P	N	R
1	X	X	
2	X		X
3	X		
4	X		
5	X	X	X

Let $\mathscr{K}$ = (K_1,K_2,K_3,Y) be a triadic context, $A_i \subseteq K_i, = 1,2,3$. If $A_i = (A_j,A_k)^{(i)}$, for $\{i,j,k\} = \{1,2,3\}$ and $j < k$, then (A_1,A_2,A_3) is called a triadic concept, where A_1,A_2 and A_3 are called the extent, the intent, and the modus of (A_1,A_2,A_3), respectively.

Consider the triadic context in Table 5. Let X_1 = $\{5\} \subseteq K_1$, X_3 = $\{\text{a,d}\} \subseteq K_3$, we have $A_2 = X_1^{(1,2,X_3)}$, $A_1 = A_2^{(1,2,X_3)}$ and $A_3 = (A_1\text{x}A_2)^{(3)}$. By the Table 9, we obtain A_2 = $\{5\}^{(1,2,\{ad\})}$ = {P,N,R}. A_1 = $\{\text{P,N,R}\}^{(1,2,\{ad\})}$ = {5}. Then by the Table 8, we have A_3 = $(\{5\}\text{x}\{\text{P,N,R}\})^{(3)})$ = {a,d}. So, {5,PNR,ad} is a triadic concept of the Table 5. In other words, firstly determines the set of all attributes which all objects of A_2 have under all conditions of X_3; secondly, A_1 is extended to the set of all objects having all those attributes under all conditions of X_3; thirdly, A_3 is extended to the set of all conditions under which each of the derived objects has each of the derived attributes. The Table 10 presents all the concepts found.

2.2.3 Triadic Implication Rules

Many studies in FCA were conducted on the generation of precise representation of rules (HAMROUNI; VALTCHEV; YAHIA, 2007), (KRYSZKIEWICZ; GAJEK, 2002) such as informative rules, Guigeus-Duquenene base (stem base),(GANTER; WILLE, 1999b),(DUQUENNE; GUIGUES, 1986), generic base (NICOLAS, 2000), and Luxenburger base (LUXENBURGER, 1991). However, those rules apply only to the dyadic approach. According to the literature, (BIEDERMANN, 1997) was the first

Table 10: Triadic Concepts extracted from the Table 5

Object(s)	Attribute(s)	Condition(s)
{}	{P,N,R}	{a,b,c,d}
{5}	{P,N,R}	{a,d}
{4}	{P,N,R}	{b}
{3,4}	{P,R}	{a,b}
{2}	{N}	{b,c,d}
{2}	{N,R}	{b,d}
{2,5}	{R}	{a,b,d}
{2,5}	{P,R}	{a,d}
{2,5}	{P,N,R}	{d}
{2,4}	{N,R}	{b}
{2,3,4,5}	{R}	{a,c}
{1}	{R}	{a,c}
{1}	{P,N}	{a,b,d}
{1,5}	{P,N}	{a,d}
{1,5}	{P,N,R}	{a}
{1,4}	{P,N}	{b,d}
{1,3,4}	{P}	{a,b,d}
{1,2,4}	{N}	{b,d}
{1,2,3,4,5}	{P}	{ad}
{1,2,3,4,5}	{P,R}	{a}
{1,2,3,4,5}	{P,R}	{a}
{1,2,3,4,5}	{}	{a,b,c,d}
{1,2,3,4,5}	{P,N,R}	{}

work to deal with the problem to extract implication rules from a triadic context. After that, different types were developed by (GANTER; OBIEDKOV, 2004), such as: Attributes x Condition Implications (AxCIs), Conditional Attribute Implications (CAIs) and Attributional Condition Implications (ACIs).

In the following sections, it will be described two types of triadic association rules that can be extracted from a triadic context $\mathscr{K} = (K_1,K_2,K_3,Y)$ by combining ideas from (BIEDERMANN, 1997) (GANTER; OBIEDKOV, 2004). The two types are: Biedermann Conditional Attribute Association Rule (BCAAR) and Biedermann Attributional Condition Association Rule (BACAR). In our approach, it was aggregated those two rules into the TRIAS BDD algorithm.

2.2.4 Conditional Attribute Association Rule

A Biedermann Conditional Attribute Association Rule (BCAAR) has the form: $(A_{2,1} \rightarrow A_{2,2})_C(sup,\ conf)$, $A_{2,1}, A_{2,2} \subseteq K_2$ and $C \subseteq K_3$ (MISSAOUI; KWUIDA, 2011). Its meaning is as follows: whenever $A_{2,1}$ occurs under all conditions in C, then $A_{2,2}$ also occurs under the same conditions with support $supp$ and confidence $conf$. Table 11 presents an example of the association rule (BCAAR) of the triadic context showed in Table 5.

Table 11: Biedermann Conditional Attribute Association Rule (BCAAR) of the Table 5

Implication Rule	Support	Confidence
(N → P) d	100.0%	100.0%
(P → R) a	100.0%	100.0%
(P → N) d	100.0%	100.0%
(R → P) a	100.0%	100.0%
(N → P) b	40.0%	66.7%
(N → R) b	40.0%	66.7%
(P → N) b	40.0%	66.7%
(P → R) b	40.0%	66.7%
(N → PR) a	40.0%	100.0%
(R → N) b	40.0%	50.0%
(R → P) b	40.0%	50.0%
(NP → R) b	20.0%	50.0%
(NR → P) b	20.0%	50.0%
(PR → N) b	20.0%	50.0%
(N → P) ab	20.0%	100.0%
(R → NP) d	40.0%	100.0%

An example presented in the Table 11, the Biedermann Conditional Attribute Association Rule (N → PR) a with 40.0% support and 100.0% confidence, it can be said that in 40% (2 of 5 total objects which is the support) of the dataset, always (confidence of 100%) that customers purchase the *antenna* (a) product from the Norway Incorporation (N) Supplier, they also purchase the same product (*antenna*) from the Pearson Limited (P) and Randy Incorporation (R) Suppliers.

Another example, (R → N)b with 40% support and 50% confidence. In this case, it can be said that in 50% of cases where customers buy *bumper* (b) from Randy Incorporation (R) Supplier, they also buy the same product from Norway Incorporation (N), and this occurs 40% of the time in the dataset (2 objects from 5 that is the support).

2.2.5 Attributional Condition Association Rule

A Biedermann Attributional Condition Association Rule (BACAR) has the form: $(C_{3,1} \rightarrow C_{3,2})_A(sup,\ conf)$, $C_{3,1}, C_{3,2} \subseteq K_3$ and $A \subseteq K_2$ (MISSAOUI; KWUIDA, 2011). Its meaning is as follows: whenever the condition in $C_{3,1}$ occurs for all attributes in A, then the condition in $C_{3,2}$ also occurs for the same attributes with support *sup* and a confidence *conf*. Table 12 presents an example of the association rule (BACAR) of the triadic context showed in Table 5.

Table 12: Biedermann Attributional Condition Association Rule (BACAR) of the Table 5

Implication Rule	Support	Confidence
(d → b) N	60.0%	60.0%
(a → b) P	60.0%	60.0%
(a → d) P	100.0%	100.0%
(d → b) P	60.0%	60.0%
(d → a) P	100.0%	100.0%
(a → b) R	80.0%	80.0%
(b → d) N	60.0%	100.0%
(b → ad) P	60.0%	100.0%
(a → b) N	20.0%	50.0%
(a → d) N	40.0%	100.0%
(b → d) R	40.0%	50.0%
(b → a) R	80.0%	100.0%
(b → d) NP	40.0%	100.0%
(b → a) PR	40.0%	100.0%
(c → a) R	20.0%	100.0%
(ab → d) N	20.0%	100.0%
(d → ab) R	40.0%	100.0%
(c → bd) N	20.0%	100.0%

Given the association rule (b → d)N with 60.0% support and 100.0% confidence, it can be said that whenever (100% confidence) that any customer buys *bumper* (*b*) from Norway Incorporation (*N*) supplier the customer will buy also the *door* (*d*), and this occurs on 40% of the dataset (2 of 5 total objects which is the support).

Another example, (a → b)N with 20.0% support and 50.0% confidence. In this case, it can be said that when a customer buys an *antenna* (*a*) from Norway Incorporation (*N*) Supplier, the customer can buy a *bumper* (*b*) in 50% of cases (confidence). Additionally, it can be said that it occurs in 20% (1 of 5 objects that is the support) of the dataset.

2.3 Binary Decision Diagram

Introduced by (AKERS, 1978) and further developed by (BRYANT, 1986a), binary decision diagrams (BDD) provide a canonical representation for a more compact boolean formula than normal conjunctive and disjunctive forms - It is substantially more compact than these traditional structure forms and it can be efficiently manipulated (BRYANT, 1986b) (MO et al., 2014). So, in general, it is more efficient to handling data than, for example, bit array.

Table 13: An example of Binary Decision Tree Table

X	**Y**	**Z**	**F (X, Y, Z)**
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

A BDD is a directed acyclic graph with two types of nodes: terminal and nonterminal. The nonterminal nodes represent the variables of the boolean formula and the only two terminal nodes represent the values 0 or 1, when the function assumes a true or false value. As in the decision tree representation, dotted and continuous transitions represent false and positive transitions, respectively.

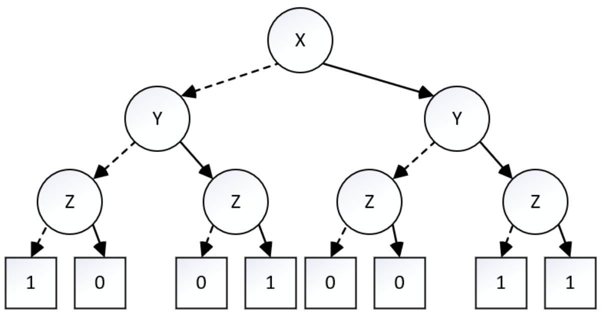


Figure 1: An example of Binary Decision Tree

Figure 1 represents the formal context presented in Table 13 as a binary decision tree and Figure 2 provides an example of which the BDD is used to represent a binary decision tree described in Figure 1. In this diagram, lines represent that the object has the attribute and the dotted line represents the lack of that attribute.

Note that, it is possible to represent the same information using a structure more compact than the original. In our approach, the formal context was represented as a BDD. Given Equation 2.1 representing a Boolean formula correspondent to Table 14. For a better representation, attributes names have been replaced by letters as follows: Google (a), YouTube (b), Facebook (c), Wikipedia (d).

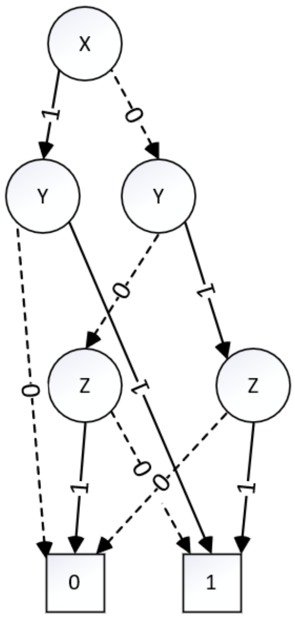


Figure 2: Example of a BDD from Figure 1

$$f(a,b,c,d) = a\overline{b}c\overline{d} + \overline{a}bcd + \overline{a}b\overline{c}\overline{d} + a\overline{b}c\overline{d} + \overline{a}bcd \tag{2.1}$$

Note that Equation 2.1 represents all the formal context in Table 14. The attribute a means that in this part of the function the attribute is true (this object has this attribute), whereas $\overline{a}$ means the opposite. The part $a\overline{b}c\overline{d}$ of the equation was created to validate the ***James*** and ***David*** objects, $\overline{a}bcd$ was created to validate the ***Shannon*** and ***Greg*** objects and the last part $\overline{a}b\overline{c}\overline{d}$ validates the ***Catheryn*** object.

Table 14: An example of the formal context

	Google (a)	**Youtube** (b)	**Facebook** (c)	**Wikipedia** (d)
James	X	.	X	.
Greg	.	X	X	X
Catheryn	.	X	.	.
David	X	.	X	.
Shanon	.	X	X	X

The generated BDD corresponding to the formal context presented in Table 14 (and described by Equation 2.1) can be seen in Figure 3. For a better understanding of each object in the BDD, object names have been replaced by numbers and attributes replaced by letters: *James* (1), *Greg* (2), *Catheryn* (3), *David* (4), *Shannon* (5) and *Google* (a), *YouTube* (b), *Facebook*(c), *Wikipedia* (d). Take as an example, object 1 (*James*) which is represented by the path *Google*, $\overline{\textit{YouTube}}$, *Facebook*, $\overline{\textit{Wikipedia}}$.

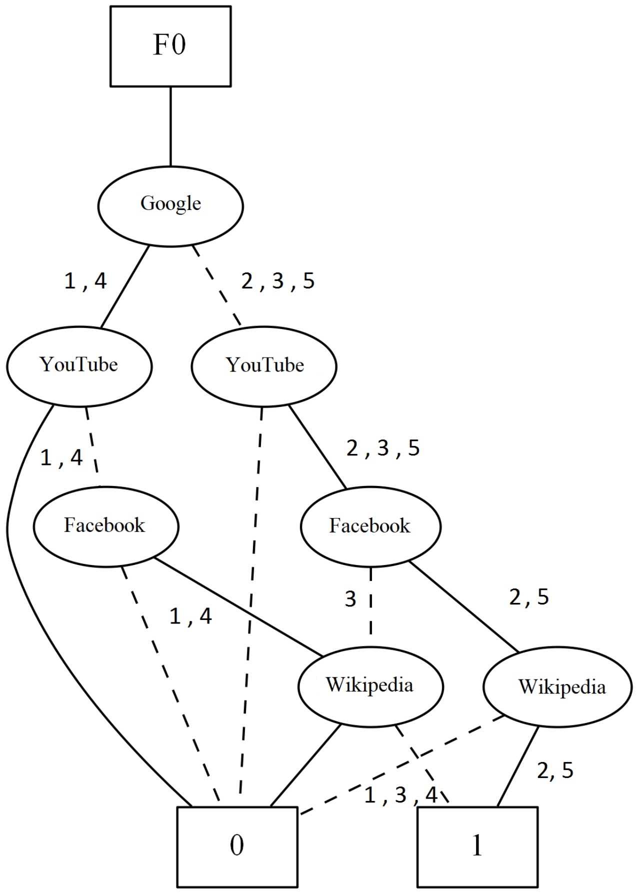


Figure 3: The BDD representing the formal context of the Table 14

Although BDD is a substantially more compact structure than traditional forms, some limitations should be considered during its use since spatial complexity depends on the order in which its variables are added. (BRYANT, 1986b) uses, as an example, a two-bit comparator with variables a_1, a_2, b_1 and b_2. The values are identical if a1 is equal to b_1 and a_2 is equal to b_2, otherwise, they are different. If the BDD is built using the order of $a_1 < a_2 < b_1 < b_2$ the result has 11 nodes. However, if we kept the related variables close together, in the order $a_1 < b_1 < a_2 < b_2$ the resulting BDD would have 8 nodes. Depending on the order in which variables are added to the BDD complexity can become exponential.

In order to build the BDD structure for a formal context, the BDD library Colorado University Decision Diagram (CUDD) was chosen for providing function packages to work with Binary Decision Diagrams (BDDs) besides having more recent updates (JAVABDD, 2019). The CUDD also has functions for Algebraic Decision Diagrams (ADDs) and Zero-suppressed Binary Decision Diagrams (ZDDs) that can be used to represent formal contexts.

2.4 TRIAS Algorithm

In (JASCHKE et al., 2006), the authors defined the problem of mining all triadic concepts of a formal context and proposed a solution called TRIAS based on dyadic projections to resolve the problem.

The authors adapted the dyadic notion of mining all item sets of a formal dyadic context, defined in (PEI et al., 2000) for a triadic approach. They also introduced the TRIAS algorithm to compute all the frequent triadic concepts of a *folksonomy* formal context. Given $\mathscr{K} = (K_1, K_2, K_3, Y)$ a triadic context, the TRIAS algorithm builds a dyadic context $\mathscr{T} = (K_1, K_2 \times K_3, Y)$ where its columns correspond to pairs of elements that belong to K_2 and K_3 and via projection, it extracts all the formal concepts.

The TRIAS algorithm was developed using *NextClosure* to generate concepts (JASCHKE et al., 2006). Thus, it was decided to use the same approach. However, the original implementation, defined in (GANTER, 2010) uses a bit structure to store objects and their attributes and conditions. The proposed approach was implemented using BDD that will be explained in the following section.

Basically, **Algorithm 1** (*Extent*) receives a formal context (*G, M, B, Y*) and an attribute and condition set X $\subseteq$ (*M,B*) as parameters. From these parameters, it returns a set of objects $X' = \{g \in G | \forall m, b \in X : gIm, b\}$ containing these attributes and conditions. It manipulates a dynamic list of objects G for each searched attribute and condition - Line 1. The algorithm runs through the list G - Line 4, adding to X' the objects that have the attribute and condition - Line 8. Before going to the next attribute and condition, X' is assigned to G (line 9), and, in Line 3, it is restarted with an empty set. The algorithm loops through all objects in the formal context and checks whether or not they have the attributes and conditions in the provided attribute set. The computational complexity of the algorithm is $\theta(|G|$ x $|X|)$.

Algorithm 2 describes the *Intent* (FormalContext, X′) function. The function parameters are the formal context (*G, M, B, Y*) and the set of objects $X' \subseteq G$ returned by function *Extent* (FormalContext, X). The function returns a set of attributes and conditions shared by all objects of the provided object set $X'' = \{m \in M, b \in B | \forall g \in X' : gImb\}$. It loops through all context attributes and conditions and verifies if all objects of the provided object set have the attribute and conditions. If all objects have the attribute and condition, it is selected to be a part of the returned set. In Line 1, the algorithm

Algorithm 1: Function Extent - Returns the set of objects containing the attributes x conditions in X

```
Require: Formal Context (G,M,B,Y) and a set of attributes x conditions $X \subset (M,B)$
Ensure : Set of objects containing the attributes and conditions in X
1  for $(all\ \text{M} \in M)$ & $(all\ \text{B} \in B)$ do
2      $HasAttributesConditions = true$
3      $X' = \emptyset$
4      for $all$ G $\in G$ do
5          if $!g$ I $(m,b)$ then
6              $HasAttributesConditions = false$
7      if $HasAttributesConditions = true$ then
8          $X' = X' \cup g$
9      $G = X'$
10 $return\ X'$
```

checks if each object in X', passed as a parameter, has the attributes and conditions in set M and B - Line 5. In Line 8, only attributes and conditions present in an object are passed forward and tested in the next object. The computational complexity of the algorithm is $\theta(|M|\text{x}|B|\text{x}|X'|)$.

Algorithm 2: Function Intent - Returns a set of attributes and conditions shared by all objects of object set X′

```
Require: Formal Context (G,M,B,Y) and a set of attributes and conditions
         $X' \subset (M,B)$
Ensure : Set of objects containing the attributes and conditions in X′
1  for $all$ G $\in X'$ do
2      $HasAttributeConditions = true$
3      $X'' = \emptyset$
4      for $(all\ \text{M} \in M)$ & $(all\ \text{B} \in B)$ do
5          if $!g$ I $(m,b)$ then
6              $HasAttributeConditions = false$
7      if $HasAttributesConditions = true$ then
8          $X'' = X'' \cup (m,b)$
9  $returns\ X''$
```

2.5 SCGaz - Context Synthetic Generator

Using a synthetic database for generating formal contexts becomes significant due to the complexity of the databases obtained from real scenarios. Real databases usually require preprocessing, a task that can, if not done correctly, directly interfere with the results. Considering that, using tools for database simulation becomes interesting and extremely useful in comparative analyzes between algorithms, as realized in (MORAES et al., 2016) (SANTOS et al., 2018).

The *SCGaz* tool proposed in (RIMSA; SONG; ZÁRATE, 2013) is a random synthetic generator of dyadic formal contexts with density control. Through *SCGaz* it is possible to specify the amount of objects and attributes desired in a formal context, as well as density, to generate irreducible contexts. Density values for a given context vary according to their dimensions and/or can be specified in advance. The generated context is irreducible, that is, there are no attributes that are not shared by at least one object or attributes that are shared by all objects. The same occurs with objects that do not have any attributes or objects that share all the attributes of the context. Objects that share the same attributes, in FCA, are considered redundant and therefore are not inserted into the context.

3 RELATED WORK

The identification of data behaviors and their expression through rules has been a topic of interest in many research studies. The FCA provided a formal framework for data representation and extraction rules, such as implication and association rules.

Studies on the use of FCA suggest several algorithms for extracting different types of rules (NOVAIS et al., 2021). The Next-Closure algorithm, proposed by Ganter (GANTER, 2002)(GANTER, 2010), is used to find a minimal implication bases (Stem-Base or Duquenne-Guigues base) suggested by Duquenne and Guigues (DUQUENNE; GUIGUES, 1986). The minimal implication base provides a complete set of implications. In other words, any valid implication for the database can be obtained by combining the non-redundant and implication base rules. Therefore, removing a rule of the minimum base makes the base no longer complete.

Although it provides a minimum number of rules, the problem with using the minimum base for practical purposes is the difficulty of deriving all valid data rules by combining the base of implication rules. Carpineto and Romano (CARPINETO; ROMANO, 2003) discussed this problem and presented an algorithm to find a more readable (i.e. more easily understandable to end users) implications base. The proposed algorithm finds implication bases with a reduced number of attributes both in the antecedent and consequent of an implication. Taouil and Bastide (TAOUIL; BASTIDE, 2001) presented another proposal for the extraction of implication bases. Their proposed algorithm finds a set of rules with a minimum antecedent and a consequent containing only one element.

Implications can also be used to identify functional dependencies which describe relationships between attributes that are valid for all elements of the universe. Functional dependencies are implications involving multivalued attributes (attributes that can take on many values) such as the color of a car and the gender of a person. Therefore, functional dependencies are implications that are independent of attribute values. The following example illustrates what functional dependencies are.

The problem of identifying functional dependencies through FCA can be reduced to the problem of identifying implications. You can transform a multi-valued attribute set into a univalent attribute set so that the valid implications for univalent attributes are valid functional dependencies for multi-valued attributes. This approach was initially presented by Wille (WILLE, 1992). Alternatively, the framework proposed by Jaume

Baixeries (BAIXERIES, 2004) can be used. This method proposes new closure operators for the construction of the conceptual lattice. Built the lattice, it is used to discover implications equivalent to the functional dependencies of the conceptual lattice built with traditional operators.

In some situations are necessary to describe facts that partially occur, for example, if a person is over sixteen years old when he or she has a voter card. Note that there are people over sixteen who do not have a voter card; thus the facts are partially valid. This type of relationship can be described through association rules. They describe data behaviors that are valid for specific groups (for this example, for the group of people over sixteen and voter status) and not necessarily for all elements such as the implication rules. Thus, association rules can be considered generalizations of implication rules.

Several papers (STUMME; WILLE; WILLE, 1998)(PASQUIER et al., 1999)(BASTIDE et al., 2000b)(WILLE, 2001)(STUMME et al., 2002) discussed the use of FCA the association rules extraction. Additionally, it demonstrated the possibility of using closed sets to find frequent attributes in databases, Pasquier and others (PASQUIER et al., 1999) present the AClose algorithm. The algorithm was compared with the traditional and recognized Apriori algorithm (AGRAWAL; SRIKANT, 1994)(AGRAWAL et al., 1996) and generally presented better results. According to Pasquier and his coauthors (PASQUIER et al., 1999), the solution space is reduced by using conceptual lattices for the association rules extraction, which allowed AClose to perform better than Apriori. In addition to AClose, the main FCA-based algorithms include Pascal (BASTIDE et al., 2000a), Titanic (STUMME et al., 2002), Galicia (VALTCHEV et al., 2002) and Frequent Next Neighbors (CARPINETO; ROMANO, 2004).

Another approach known as 3WFCA (Three-way formal concept analysis) includes the combination of inclusion method (acceptance) and exclusion method (rejection) which are not supported by classical FCA. In 3WFCA, the intention and extension of a concept must be an orthopair (QI; WEI; Y, 2014).

Different from FCA, in 3WFCA the intension or extension of a concept has two parts: positive expresses the semantics of jointly possessed and negative expresses not jointly possessed. In (SHIVHARE; CHERUKURI, 2017), it was extended the FCA based BAM to 3WFCA to perform the negative recall along with the positive recall. The focus was on recalling the input pattern from memory.

In the literature, there are various works and some of them date back to 1995 (WILLE, 1995)(LEHMANN; WILLE, 1995a) have focused on triadic context analysis, concepts, diagrams and algorithms.

In (KUMAR et al., 2016), it was proposed an approach to models role based access control (RBAC) using Triadic FCA without transforming the triadic access control matrix into dyadic formal contexts. In (SUBRAMANIAN; CHERUKURI; CHELLIAH, 2018), they proposed a model to represent RBAC policy through 3WFCA.

In (KIS; TROANCA, 2016), the authors combined the triadic exploration technique proposed in (RUDOLPH; SACAREA; TROANCA, 2015) with other resources for manipulating dyadic and triadic contexts. Among these functionalities is possible to apply basic scale operations (GANTER; WILLE, 1999b) and context projections, as well as calculation of dyadic and triadic concepts, creation, and manipulation of dyadic lattices.

In (TRABELSI; JELASSI; YAHIA, 2012b), the authors compared the results from three different triadic algorithms: TRICONS, TRIAS and DATA-PEELER. The same approach was used in (IGNATOV et al., 2015). The authors presented several definitions of optimal patterns for triadic data and results of experimental comparison of three triadic algorithms on real-world and synthetic dataset - including TRIAS.

There are many papers about applications of TCA, especially for analyzing data such as Folksonomy. In 2009, Cerf et al. introduced an algorithm DATA-PEELER to compute closed patterns from n-ary relations (IGNATOV et al., 2011)(KAVTOUE et al., 2011)(GNATYSHAK et al., 2012)(TRABELSI; JELASSI; YAHIA, 2012a). Tang et al. proposed the notion of a triadic decision context by combining triadic contexts and studied a rule acquisition method (TANG; FAN; LI, 2016). Analogously to FCA, many problems in TCA can be solved, such as, acquisitions of triadic concepts, mining rules and extraction of triadic association and implication rules.

In 1997, Biedermann proposed triadic implications. Based on studies of triadic implications proposed by Biedermann, Ganter et al. (GANTER; OBIEDKOV, 2004) gave a new definition of implication. According to Ganter, in a study suggested by Biederman (BIEDERMANN, 1998) was not clear what an implication should be. Biedermann called triadic implications the following representation $(R \to S)_C$, which should be interpreted as: *If an object has all attributes from R under all conditions from C, then it also has all attributes from S under all conditions from C.*

Ganter started by considering a class of implications that were more restricted - studied expressions of the form $R \overset{C}{\to} S$, where $R,S \subseteq M, C \subseteq B$. Then, it was called such an expression a *conditional attribute implication* and read it as: *R implies S under all conditions from C.* A conditional attribute implication $R \overset{C}{\to} S$ holds in a triadic context K iff the following is satisfied: *for each condition $c \in C$ it holds that if an object $g \in G$ has all the attributes in R then it also has all the attributes in S.*

Afterward, Missaoui and Kwuida obtained the way to mining triadic association rules from a ternary relationship (MISSAOUI; KWUIDA, 2011) based on the relationship between implications and association rules. It is important to note the algorithm proposed, implemented in the Lattice Miner tool, used the same dyadic projection that was implemented in TRIAS which is objects, attributes x conditions.

In (SALLEB; MAAZOUZI; VRAIN, 2002), BDDs were used to store transaction logs as a truth table and to find common patterns in large transactional data sets. In this paper, the use of BDDs allowed authors to load all transactions into the main memory, avoiding database processing on the disk.

In (MARQUEZ et al., 2017), a fault tree dynamic analysis was carried out by BDDs to obtain the system failure probability over time and using different time increments to evaluate the system. According to the authors, this analysis allows critical electrical and electronic components of the converters to be identified in different conditions. The results were used to develop a scheduled maintenance that improves the decision making and reduces the maintenance costs.

BDDs were employed as a suitable and operational method to facilitate an analysis of the behavior of the wind turbine system under certain conditions and to get an analytical expression by the Boolean functions. The size of the binary decision diagram, i.e., the computational cost for solving the problem, has an important dependence on the order of the components or events considered (MARQUEZ et al., 2020).

Related to the use of BDD in FCA, different works were identified . An algorithm to extract formal concepts using BDD was proposed by (YEVTUSHENKO, 2002). The BDD was used to represent the list of concepts. However, the authors used contexts with 900 objects and 50 attributes, which proved to be efficient only for dense contexts.

Different from (YEVTUSHENKO, 2002), in (RIMSA; ZÁRATE; SONG, 2009) the authors used BDD for extracting formal concepts, but their focus was on using different BDD libraries to check which library best could be used in FCA. In (RIMSA; ZÁRATE; SONG, 2009), they used brute force methods to obtain the set of intentions, and consequently the BDDs that represented the extensions. In all experiments performed in the study, the algorithms with BDD were more efficient when compared to the original algorithms.

In (NETO; ZÁRATE; SONG, 2018) the authors used the binary decision diagram to deal with high-dimensional dyadic contexts in order to extract concepts. The authors proposed modifications in the *In-Close2* through BDD to manipulate objects. However, it was applied only for a dyadic formal context.

In (SANTOS et al., 2018) the authors proposed modifications in the algorithm to extract implications *ProperIm* in a dyadic formal context, adding BDDs in the structure in order to manipulate and extract proper rules from dyadic contexts. The *ProperImplicBDD* algorithm presented a significantly better runtime. The tests varied the number of attributes and their density for a total of 120,000 objects.

Another approach known as 3WFCA (Three-way formal concept analysis) includes the combination of exclusion method (rejection) and inclusion method (acceptance) which are not supported by classical FCA. In 3WFCA, the extension and intension of a concept must be an orthopair (QI; WEI; Y, 2014).

Different from FCA, in 3WFCA the extension or intension of a concept has two parts: negative expresses not jointly possessed and positive expresses the semantics of jointly possessed. In (SHIVHARE; CHERUKURI, 2017), it was extended the FCA based BAM (bidirectional associative memory) to 3WFCA to perform the negative recall along with the positive recall. The focus was on recalling the input pattern from memory.

There are some differences between TCA and 3WFCA, the main differences are: in TCA the context is triadic, and in 3WFCA is dyadic. Moreover, in TCA the context is represented by the three-dimensional table containing the relationship of the objects, attributes and conditions, and in 3WFCA the context is represented by the two-dimensional binary table containing the relationship between objects and attributes. Also, the triadic concepts consist of the extent, intent and modus, and in 3WCA consists of only the extension and intension.

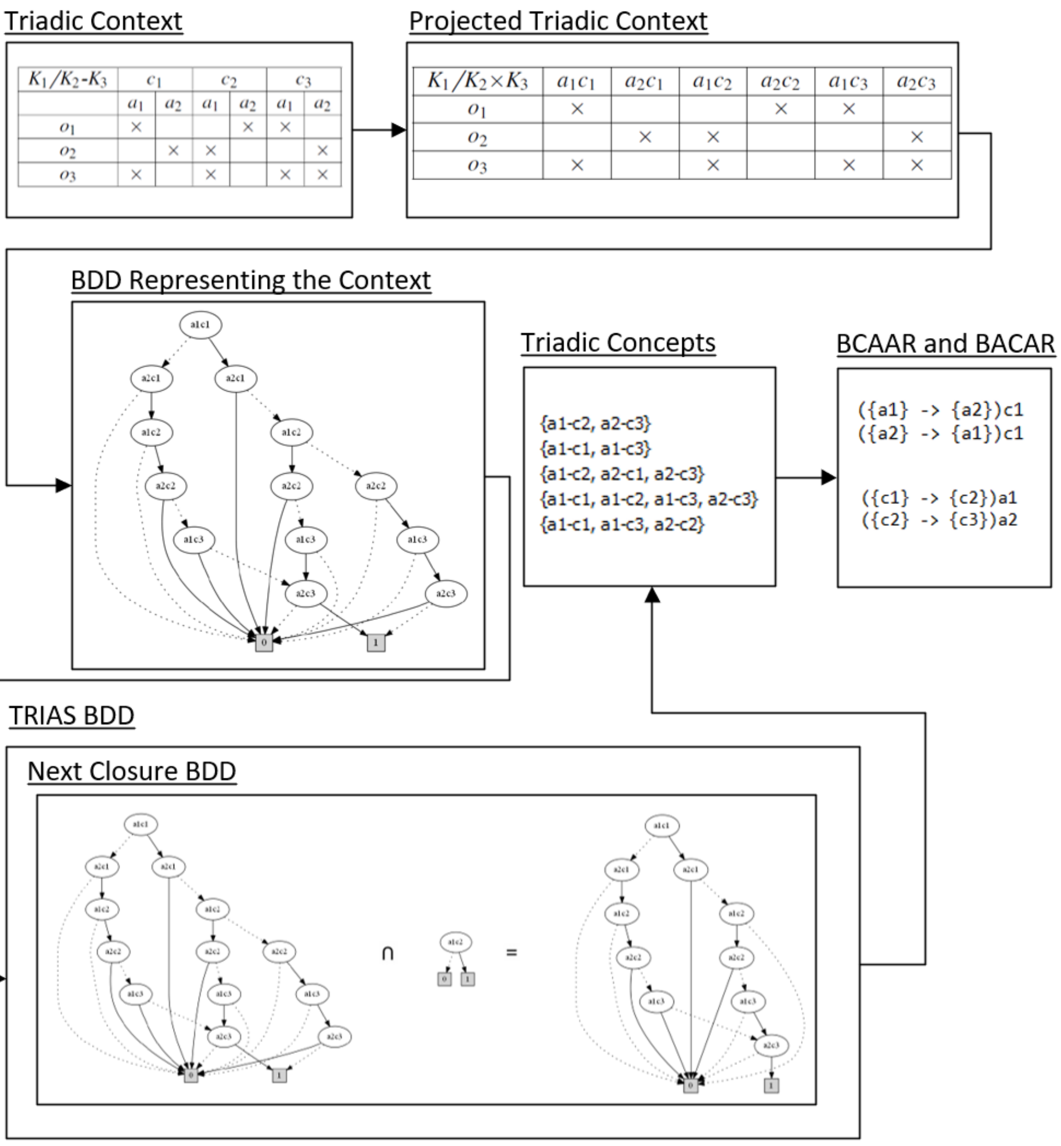
In (KUMAR et al., 2016), it was proposed an approach to model RBAC (role based access control) using Triadic FCA without transforming the triadic access control matrix into dyadic formal contexts. In (SUBRAMANIAN; CHERUKURI; CHELLIAH, 2018), they proposed a model to represent RBAC policy through 3WFCA.

Another important research is to handle the complexity of large contexts in FCA. There are some approaches in the literature that deal with those problems, such as: use proper implications extracted from reduced concept lattice to represent the behavior of the process being studied in a symbolic and qualitative form (DIAS et al., 2020); and selecting an appropriate reduction method to reduce the input data (LI et al., 2017).

However, no triadic approaches were found that use the data efficient manipulation provided by BDDs. Therefore, this thesis presents an approach for triadic contexts using BDDs through dyadic projections.

4 APPLYING BINARY DECISION DIAGRAM IN TCA

Figure 4 represents the flowchart used in this work. Initially, it shows the transformation of the triadic context into a projected dyadic context. Then, the representation of the context in a BDD structure. From this representation, operations are carried out on TRIAS BDD, within the NextClosure BDD function, in order to generate the triadic concepts. Finally, BCAAR and BACAR rules are generated based in dyadic and triadic concepts. The following sections will explain our approach in more detail.



K_1/K_2-K_3	c_1		c_2		c_3	
	a_1	a_2	a_1	a_2	a_1	a_2
o_1	×			×	×	
o_2		×	×			×
o_3	×		×		×	×

$K_1/K_2 \times K_3$	a_1c_1	a_2c_1	a_1c_2	a_2c_2	a_1c_3	a_2c_3
o_1	×			×	×	
o_2		×	×			×
o_3	×		×		×	×

{a1-c2, a2-c3}
{a1-c1, a1-c3}
{a1-c2, a2-c1, a2-c3}
{a1-c1, a1-c2, a1-c3, a2-c3}
{a1-c1, a1-c3, a2-c2}

```
({a1} -> {a2})c1
({a2} -> {a1})c1

({c1} -> {c2})a1
({c2} -> {c3})a2
```

Figure 4: TRIAS BDD Flowchart

4.1 Representation of Triadic contexts using BDD

Given a formal triadic context *(K_1, K_2, K_3, Y)* where K_1, K_2 and K_3 are called objects, attributes and condition respectively and Y the ternary relation between K_1, K_2 and K_3, a projection can be performed in the triadic context (Table 15) resulting in a dyadic context *(K_1, $K_2 \times K_3$, Y)* (Table 16).

Table 15: Triadic Context (K_1, K_2, K_3, Y)

K_1/K_2-K_3	c_1		c_2		c_3	
	a_1	a_2	a_1	a_2	a_1	a_2
o_1	×			×	×	
o_2		×	×			×
o_3	×		×		×	×

The projection results from the combination of attributes and conditions where each attribute is renamed according to the condition to which it belongs. The retrieval and manipulation of attributes and conditions can be done from the label assigned to each attribute. In the context represented by Table 16 the dyadic incidence given by the tuple (o_1, a_1c_1) is equivalent to the triadic incidence given by the triple (o_1, a_1, c_1) of the context presented in Table 15.

Table 16: Dyadic Context Projection (K_1, $K_2 \times K_3$, Y)

$K_1/K_2{\times}K_3$	a_1c_1	a_2c_1	a_1c_2	a_2c_2	a_1c_3	a_2c_3
o_1	×			×	×	
o_2		×	×			×
o_3	×		×		×	×

Once projected, the triadic context, now described by a dyadic context, can be represented by a binary decision diagram converting the context to a boolean formula used to generate the corresponding BDD. Table 16 describes the triadic context projected in a dyadic context and Equation 4.1 represents it through conjunctive and disjunctive operations between objects and attributes. The symbols with a slash over the letter represent that the attribute is false.

$$\begin{aligned} f(a_1c_1,a_2c_1,a_1c_2,a_2c_2,a_1c_3,a_2c_3) &= a_1c_1.\overline{a_2c_1}.\overline{a_1c_2}.a_2c_2.a_1c_3.\overline{a_2c_3}+ \\ &\overline{a_1c_1}.a_2c_1.a_1c_2.\overline{a_2c_2}.\overline{a_1c_3}.a_2c_3+a_1c_1.\overline{a_2c_1}.a_1c_2.\overline{a_2c_2}.a_1c_3.a_2c_3 \end{aligned} \tag{4.1}$$

Figure 5 represents the dyadic projection of the triadic context defined by Expression 4.1. This representation allows manipulating triadic contexts using a BDD,

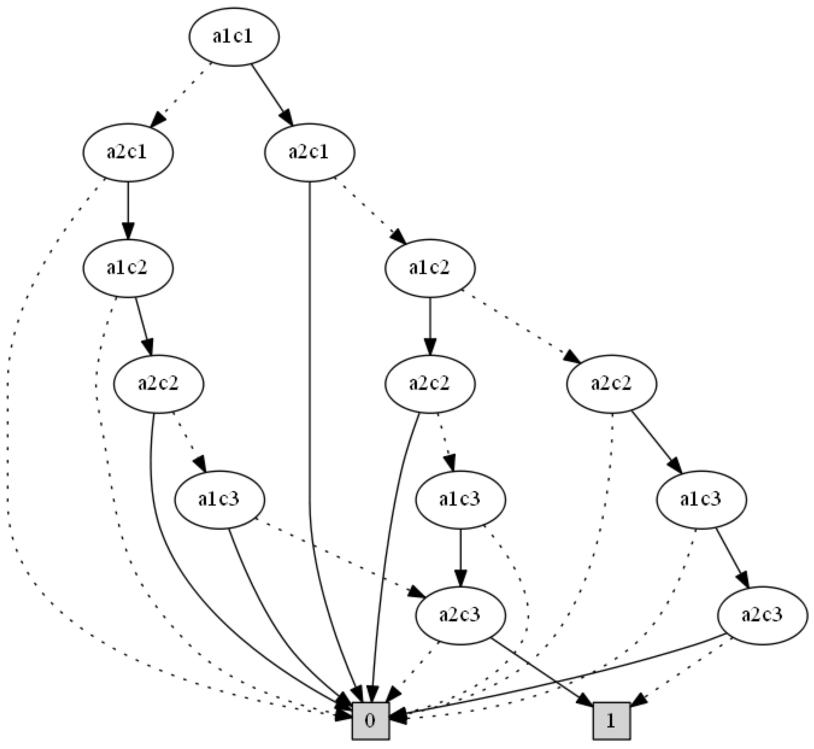


Figure 5: Context (K_1, $K_2 \times K_3$, Y) represented by a BDD.

providing efficient manipulation and storage (BRYANT, 1986b). Given a triadic context projected and represented by a BDD, it is possible to provide strategies for recovering objects, attributes and conditions, since any algorithm that uses this representation requires to recovery and efficient alteration of these elements.

Given the context presented in Table 16, retrieval of objects can be done, for example, from logical operations AND or OR under Equation 4.1 of the context. If it is necessary to obtain all the objects of the context which have the attribute a_1c_2, it can create a BDD that represents such an attribute and apply a logical operation AND between the BDDs. Figure 6 represents such an operation, returning in a new BDD with the objects o_2 and o_3 since both are sharing the attribute a_1c_2.

In some situations, if it is necessary to retrieve all the objects which have, for example, the attributes a_1c_1 and a_1c_3, the same operation presented previously can be performed (Figure 7).

As presented in our experimental results (Table 20), TRIAS did not return any concept when processing high-dimensional contexts. For example, in the context with 120,000 objects, 15 attributes and 5 conditions it did not return a concept within 14 days.

The proposed algorithm, TRIAS BDD, in this work includes the BDD structure in which is used to represent the triadic context. The **Algorithm 3** receives as an input

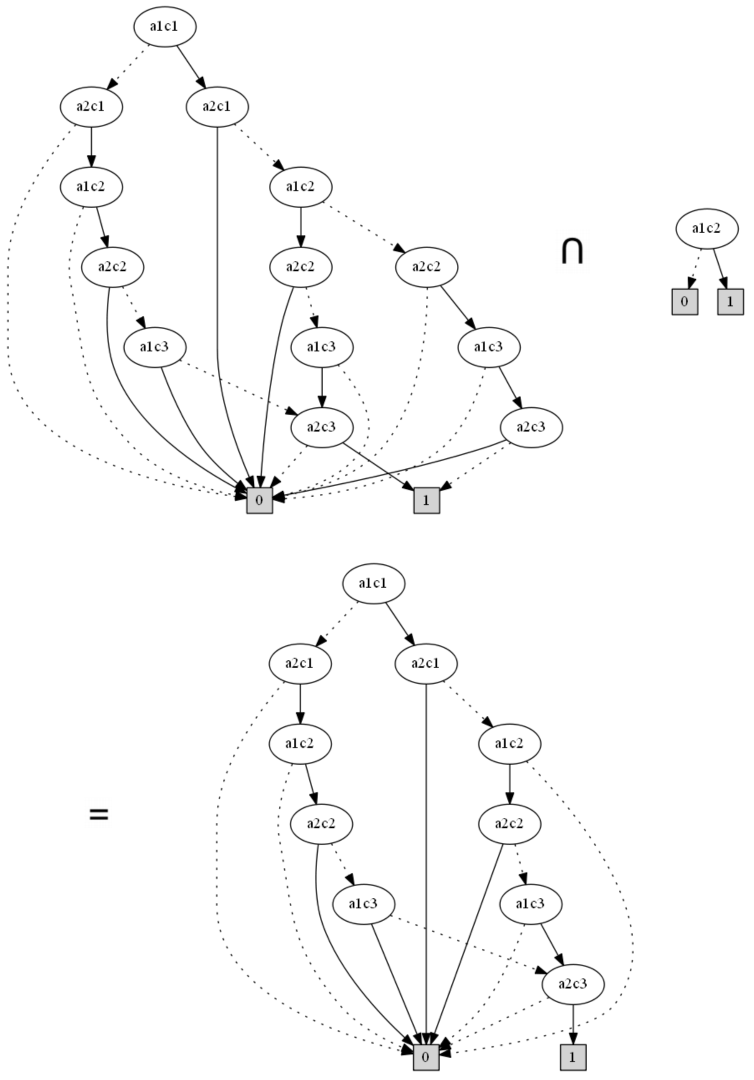


Figure 6: Logical operation between the attribute a_1c_2 and the BDD Context.

a file which the lines are the incidences that represent a formal triadic context. In other words, each line represents if that object has that attribute and also the condition. Line 1 initializes BDDTemp and Line 2 initializes ContextBDD as empty sets. Then, from Line 3 to 9, the algorithm loops through all objects, attributes and conditions and, if an object has the attribute and condition, the BDD variable indicated by $(g,m,b) \in I$ is included in the temporary BDD (true node). Otherwise, it will be added as a false node. Finally, in Line 10, it adds the BDD which represents the object (BDDTemp) to the BDD that represents the context (ContextBDD).

Once the BDD that represents the formal context has been created, some operations can be performed easily, taking advantage of an optimization inherited by the structure used as shown in **Algorithm 6**. The algorithm returns a BDD with the

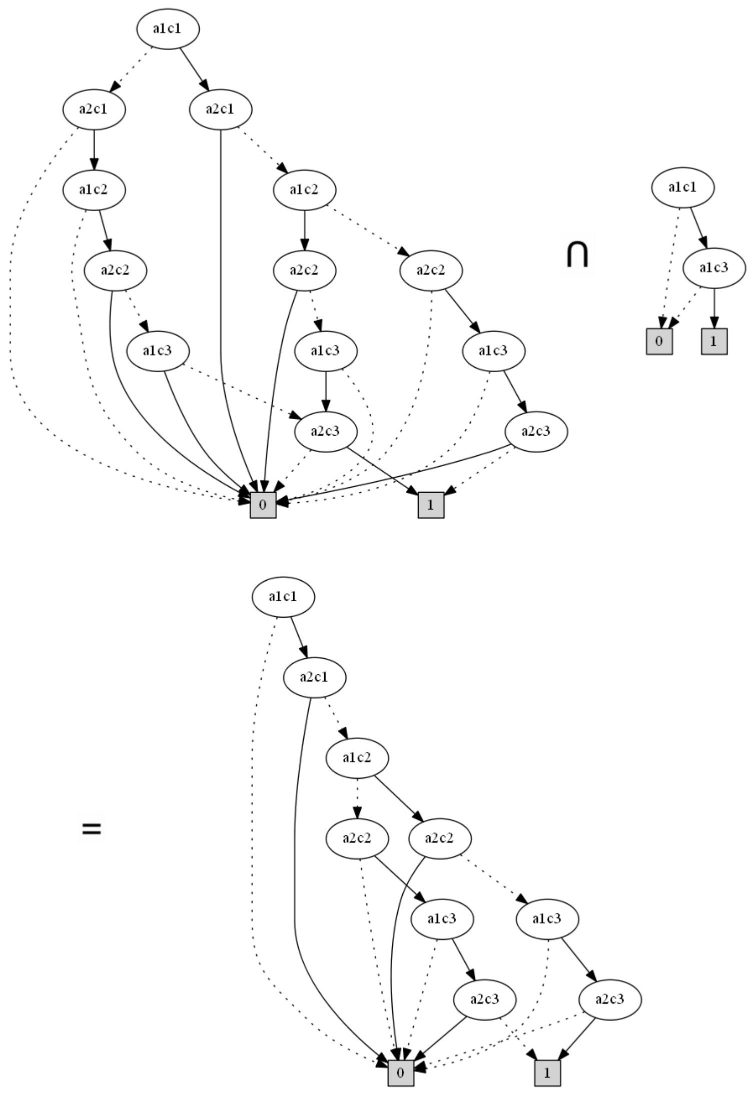


Figure 7: Logical operation between the attribute a_1c_1 and a_1c_3 and the BDD Context.

objects that have the desired attributes and conditions. In our example, Table 16, there are conditions and attributes such as c_1a_1, c_1a_2, c_1a_3, c_2a_1, c_2a_2, c_2a_3, c_3a_C1, c_3a_2 and c_3a_3. Note that it requires only a logical AND operation with BDD. As a result, the *"Objects"* variable will contain a BDD that represents objects that shares attributes and conditions c_1a_1, c_1a_2, c_1a_3, c_2a_1, c_2a_2, c_2a_3, c_3a_1, c_3a_2 and c_3a_3.

4.2 Extracting Triadic Concepts through BDD

As explained in subsection 2.4, TRIAS were implemented using the *NextClosure* structure (JASCHKE et al., 2006). The basic logic of Algorithm *NextClosure* was not modified. However, modifications have been implemented to this Algorithm to support the new structure (BDD) - it has been denominated *NextClosureBDD*. The difference is

Algorithm 3: LoadTCAContext() - Building a triadic formal context using BDD

```
Input  : Formal Triadic context $(G,M,B,I)$
Output: Formal Triadic context structured as a BDD (ContextBDD)
1  $BDDTemp = \emptyset$
2  $ContextBDD = \emptyset$
3  forall the $g \in G$ do
4      forall the $m \in M$ do
5          forall the $b \in B$ do
6              if $(g,m,b) \in I$ then
7                  $BDDTemp = BDDTemp.nodeTrue$
8              else
9                  $BDDTemp = BDDTemp.nodeFalse$ ;
10             ContextBDD = ContextBDD ∪ BDDTemp
11 return ContextBDD
```

the data type used to represent the formal context, objects, attributes and conditions. All of these structures are represented using BDDs.

The main algorithm for identifying and extracting formal concepts is given by the *Concepts Extraction()* function described in the Algorithm **4**.

Algorithm 4: Concepts Extraction

```
Require: Formal triadic context
Ensure : Set containing all formal concepts in the provided context
1  $LoadTCAContext(TXTFile, Context)$
2  $X = \theta$
3  while $X.size() < Context.NumAttributesConditions.size()$ do
4      $X = NextclosureBDD(X, ExtentBDD, IntentBDD)$
5      $ConceptList.add(ExtentBDD, IntentBDD)$
```

In Line 1, the module responsible for loading the formal context, *LoadTCAContext()*, takes as input a file in TXT format file which represents the formal context to be processed. A bit matrix stores the data. When an object has a certain attribute and condition, the corresponding attribute and condition bit is set to a *true* value. All other operations performed in the context also handle data in this format. In Line 2, it initializes the attribute and condition set (variable X, in **Algorithm 4**) with an empty set. The variable increases in lexicographical order until it has all the attributes and conditions of the context. Then, in Line 3, the Algorithm calls *NextClosureBDD* while the attributes and conditions set is not complete. Internally, Line 4 Algorithm *NextClosure* calls the function *DoublePrime* (**Algorithm 5**) which is responsible for the double derivation. The computational complexity of *NextClosure* algorithm is θ ($|2^G \mid G \mid M \mid B \mid$) (CARPINETO; ROMANO, 2005). Finally, in Line 5, it adds the concepts to the Concept List.

Algorithm 5 returns a set of attributes and conditions derived from the provided attribute set. Basically, the algorithm uses two operations. In the first one - Line 1 - the Extent of provided attributes and conditions are computed. In other words, all objects sharing the attributes and conditions provided as parameters are extracted from the formal context. In Line 2, which is the second function (IntentBDD) computes the final derivation, which returns the intents of the object set returned by the first derivation. The function identifies all attributes and conditions shared by the objects obtained from the extent. This attribute set represents the final derivation of the original attribute set.

Algorithm 5: DoublePrime Algorithm - ($''$) operator

```
Require: Attribute and Condition set $X \subset (M,B)$
Ensure : X$''$ = Attribute and Condition set derived from X
1 $X' = ExtentBDD(FormalContext, X)$
2 $X'' = IntentBDD(FormalContext, X')$
3 returns $X''$
```

An optimization in the Extent function **(Algorithm 1)** and Intent function **(Algorithm 2)** was implemented resulting in the **Algorithms 6 (ExtentBDD)** and **7 (IntentBDD)**. In order to obtain the extent, the **Algorithm 6** was implemented to manipulate BDDs. For each searched attribute and condition, the algorithm would process through all structure, eliminating objects that do not have the attribute and condition specified. Therefore, for each searched attribute and condition, the BDD is reduced proportionally to the incidence of that attribute and condition in the formal context. The computational complexity of the algorithm is $\theta(|G|$ x $|X|)$. Regarding the extraction of the intent, the **Algorithm 7** also manipulates attributes and conditions in BDD structure. The algorithm checks the attributes and conditions in the set of objects passed as a parameter, and only attributes and conditions present in an object are passed forward to be tested in the next object.

Algorithm 6: ExtentBDD - Returns the set of objects containing the attributes and conditions in X

```
Require: Formal Context (G,M,B,Y) and a set of attributes and conditions
         $X \subset (M,B)$
Ensure : Set of objects containing the attributes and conditions in X
1 $X' = BDDAnd(ContextBDD, BDDAttributesCondition)'$
2 returns $X'$
```

In Line 1, the function *BDDAnd()* is the responsible for computing the conjunction of two BDDs and returns a pointer to the resulting BDD if it is successful. In this algorithm, the first BDDs passed as a parameter is a formal context benefiting from structural optimizations due to BDD representation. The other BDD parameter represents the set of attributes and conditions that must be shared by the context objects. As described in **Algorithm 3**, the context BDD is formed by a sequence of BDDs representing objects and linked by OR operators, whose attributes and conditions are nodes of that BDD.

Algorithm 7: IntentBDD - Returns a set of attributes and conditions shared by all objects in input parameter X′

```
Require: A set of objects $X' \subseteq G$ in BDD structure
Ensure : Set of attributes and conditions shared by objects in X′, in BDD structure
1 for ($all$ M $\in M$) & ($all$ B $\in B$) do
2     $BDDAttributeCondition = (m,b)$
3     $BDDtmp = BDDAnd(X', (m,b))$
4     if $BDDtmp = X'$ then
5         $AttributeConditionList = (m,b) \cup AttributeConditionList$
6 for $all\ attributecondition \in AttributeConditionList$ do
7     $X'' = BDDAnd(X'', attributecondition)$
8 $returnsX''$
```

Algorithm 7 takes as an input a BDD representing a set of objects. From Lines 1 to 3, the algorithm loops through all attributes and conditions of the formal context, assembling BDDs for each one. In Line 4, if the conjunction of an attribute and condition BDD and an object BDD results in a BDD identical to the original object BDD, it means that all objects represented by that object BDD have the attribute and condition represented by that attribute and condition BDD. In Line 5, this attribute and condition are then inserted into a separate attribute and condition list. Then, in Lines 6 and 7, the algorithm creates a BDD containing all attributes and conditions shared by the objects. This logic is the same of the original algorithm, with the exception that the new algorithm does not test objects one by one. The computational complexity of the algorithm is $\theta(|M|\mathrm{x}|B|\mathrm{x}|X'|)$.

4.3 Extracting BCAAR and BACAR rules

TRIAS BDD was extended to extract BCAAR and BACAR rules based on the algorithms proposed and developed in (MISSAOUI; KWUIDA, 2011). **Algorithm 8** computes triadic concepts and generators. Lines 3 to 13 collect distinct attribute and condition values found in the intent of the current dyadic c in order to construct the sub-context $K_{aj,ak}$ and generate the corresponding concepts. In Line 3, Gen(c) is the set of generators associated with the intent of c. Line 4 collects distinct attribute values $a_j \in$ c and line 5 collect distinct conditions values $a_k \in$ c. In Line 8, initialize the sub-context

$K_{aj,ak}$ to 0 where rows and columns are attributes and conditions found in c. Line 10 extracts the attribute value a_j in e = $a_j \times ak$ and line 11 extracts the conditions value a_k in e. Line 12 constructs the sub-context $K_{aj,ak}$ from the intent of the concept c. Line 16 checks whether each computed triple is a triadic concept. The computation of generators is done through lines 18 to 23. For each class of triadic concepts and their associated generators. Line 20 collects distinct attribute values $a_j \in$ g and Line 21 collects distinct conditions values $a_k \in$ g.

Algorithm 8: Triadic concepts and generators

```
Input  : c: concept generated from TRIAS BDD
Output: C and G: a set of triadic concepts and a set of associated generators
1  $C = \emptyset$
2  $G = \emptyset$
3  $U \leftarrow$ Gen(c)
4  $A \leftarrow$ DistinctA(c)
5  $Cond \leftarrow$ DistinctCond(c)
6  forall the $a_j \in A$ do
7      forall the $a_k \in Cond$ do
8          $K_{aj,ak} \leftarrow 0$
9  forall the $e \in IntentBDD(c)$ do
10     $a_j \leftarrow$ Attribute(e)
11     $a_k \leftarrow$ Conditions(e)
12     $K_{aj,ak} \leftarrow 1$
13 $ACond \leftarrow$ AttributeConditions($K_{aj,ak}$)
14 forall the $e \in ACond$ do
15     $e_1 \leftarrow$ Derive(e) {$e_1$ is the (1)-derivation of $e$}
16     if $e_1$ = EXTENTBDD(C) then
17         Cond $\leftarrow$ Cond $\cup$ {($e_1$, ExtentBDD(e), IntentBDD(e))}
18 if $Cond \neq \emptyset$ then
19     forall the $g \in U$ do
20         A $\leftarrow$ DistinctA(g)
21         C $\leftarrow$ DistinctCond(g)
22         if $Size(A) \times Size(Cond) = Size(g)$ then
23             G $\leftarrow$ G $\cup$ {(A,Cond)}
24 return C,G
```

Algorithm 9 is responsible for retrieving the BCAAR and BACAR triadic association rules functions and compute them. In Line 2, it loops through all the classes, which contain triadic concepts together with associated generators and predecessors. In Line 3, it loops through all generators. In Line 6-8, it verifies if the class of the predecessor is empty, if it is not, it calls the BCAAR function (Line 7) and BACAR function (Line 8).

```
Algorithm 9: Function AssociationRules
Input  : T= a set of classes. Each class contains triadic concepts together with
         associated generators and predecessors.
Output: AR: A set of n-uples (L,R,C, s, c) representing association rules with left
         hand-side L, right hand-side R, condition C and quality measures.
1 $AR = \emptyset$
2 forall the $CL \in T$ do
3     forall the $generator \in Gen(CL)$ do
4         A ← Intent(generator)
5         D ← Conditions(generator)
6         if Pred(CL) $\neq \emptyset$ then
7             AR ← AR ∪ BCAAR(CL,A,D,E)
8             AR ← AR ∪ BACAR(CL,A,D,E)
9 return AR
```

Algorithm 10 is the function responsible for computing the BCAAR triadic association rules. It receives the class with the associated generator and predecessor; the intent A, the extent E and the condition of the current generator. It loops through all predecessors of the classes, computes the Intent(p) in B, the support s (Line 4) and confidence c (Line 5).

```
Algorithm 10: Function BCAAR
Input  : CL: A class of triadic concepts together with associated generators and
         predecessors; A and D are the intent and the condition of the current
         generator respectively while E is the extent of the current triadic concept
Output: BCAAR: a set of association rules whose confidence is less than 1.
1 $BCAAR = \emptyset$
2 forall the $p \in Predecessor(CL)$ do
3     $B$ ← Intent(p)
4     $s$ ← Extent(p)/$Size(K_1)$
5     $c$ ← Extent(p)/$Size(E)$
6     $BCAAR$ ← $BCAAR$ ∪ {(A,B\ A,D, s, c)}
7 return BCAAR
```

Algorithm 11 is the function responsible for computing the BACAR triadic association rules. It receives the class with the associated generator and predecessor; the intent A, the extent E and the condition of the current generator. It loops through all predecessors, computes the Condition(p) in F, the support s (Line 4) and confidence c (Line 5).

Algorithm 11: Function BACAR

```
Input  : CL: A class of triadic concepts together with associated generators and
         predecessors; A and D are the intent and the condition of the current
         generator respectively while E is the extent of the current triadic concept
Output: BACAR: a set of association rules whose confidence is less than 1.
1 $BACAR = \emptyset$
2 forall the $p \in Predecessor(CL)$ do
3     $F \leftarrow$ Conditions(p)
4     $s \leftarrow$ Extent(p)$/Size(K_1)$
5     $c \leftarrow$ Extent(p)$/Size(E)$
6     $BACAR \leftarrow BACAR \cup$ {(D,F\ D,A, s, c)}
7 return BACAR
```

5 EXPERIMENTS AND RESULT ANALYSIS

The contexts used, in our experiments, correspond to two types of triadic contexts - synthetic (subsection 5.1) and real contexts (subsection 5.2). The results (time execution) were compared for both TRIAS and TRIAS BDD Algorithms.

The experiments were run on an Intel Core i7-4790 3.60GHz with 8 cores, 16 threads, 32GB RAM and Ubuntu 18.04 LTS operating system. It is important to emphasize that our approach was developed to use sequential processing and the *top* Linux command was used to collect RAM usage.

5.1 Results for the synthetic triadic contexts

The random synthetic generator *SCGaz* was used to generate several contexts. The main objective was to evaluate the performance of the TRIAS and TRIAS BDD in the concept extraction. Initially, synthetic triadic contexts with an arbitrary number of objects, dimensions, conditions and densities were generated to be used in both algorithms. In this work, contexts with 500, 1,500, 3,000, 5,000 and 10,000 objects were created with 10, 15 and 20 attributes and 5 conditions. The density was fixed at 30%, 50% and 70% for all contexts.

In order to perform a better analysis, 10 different contexts were generated for each testing scenario randomly.

The results presented in the Table 17, Table 18, Table 19 and Table 20 correspond to the time execution average of the 10 contexts. In Table 17, Table 18 and Table 19, cells with the "-" symbol represent that the algorithm failed to complete the concepts extraction processing within 7 days.

The proposed algorithm (TRIAS BDD) consumed less memory and allowed the algorithm to extract concepts in larger and denser contexts. The cells in bold present the fastest times.

Initially, the number of objects of each context was increased while maintaining the number of attributes, conditions and density. Secondly, it was varied the density of each context. Lastly, it was increased the number of attributes to make our contexts more complex in order to test both algorithms and observe their behaviors in high-dimensional scenarios.

As presented in Table 17, which considers 10 attributes and 5 conditions, it can be seen that the TRIAS BDD algorithm showed a higher speedup than TRIAS, varying between 28% and 36%, for density 30% and between 12% and 22% for density 50%.

Considering the density of 30%, there was no relationship of proportionality between the increase the number of objects and the percentage variation of speedup. However, when the 50% density is observed, a reduction in speedup is directly proportional to the number of objects up to 5,000 objects. For 10,000 objects, TRIAS BDD's speedup percentage gain compared to TRIAS was exactly the same achieved for 500 objects (22%).

Analyzing the density of 70%, it can be observed that the TRIAS BDD obtained a speedup greater than the TRIAS for 500, 1,500 and 5,000 objects, presenting a variation between 6% and 19%. However, for 3,000 objects, TRIAS showed a speedup greater than TRIAS BDD by 5%.

It is important to note that TRIAS did not obtain results for 10,000 objects and a density of 70% in the same context of attributes and conditions, which did not allow the comparison between the algorithms.

Table 17: TRIAS x TRIAS BDD Algorithm Results in 10 attributes and 5 conditions

Context (G x M x B)	Density (%)	Incidences	TRIAS (Minutes)	TRIAS BDD (Minutes)	SpeedUp (%)
500 x 10 x 5	30	7,500	4.55	**3.57**	22%
1,500 x 10 x 5	30	22,500	19.36	**13.16**	32%
3,000 x 10 x 5	30	45,000	30.00	**21.70**	28%
5,000 x 10 x 5	30	75,000	55.18	**35.32**	36%
10,000 x 10 x 5	30	150,000	119.53	**82.48**	31%
500 x 10 x 5	50	12,500	7.09	**5.53**	22%
1,500 x 10 x 5	50	37,500	31.62	**25.61**	19%
3,000 x 10 x 5	50	75,000	54.85	**46.62**	15%
5,000 x 10 x 5	50	125,000	130.37	**114.72**	12%
10,000 x 10 x 5	50	250,000	292.19	**227.91**	22%
500 x 10 x 5	70	17,500	9.09	**7.37**	19%
1,500 x 10 x 5	70	52,500	39.52	**33.20**	16%
3,000 x 10 x 5	70	105,000	**62.22**	65.33	-5%
5,000 x 10 x 5	70	175,000	128.76	**121.03**	6%
10,000 x 10 x 5	70	350,000	-	**269.61**	-

As can be observed in Figure 8, considering the context of 10 attributes, 5 conditions and 30% density, for processing context with 500 objects TRIAS spent 4.55 minutes and TRIAS BDD spent 3.57 minutes, a difference of 0.98 minutes in favor of TRIAS BDD. For processing context with 1,500 objects, TRIAS spent 19.36 minutes and the proposed algorithm spent 13.16 minutes, which is a difference of 6.2 minutes less. For processing context with 3,000 objects, the original algorithm spent 30 minutes and the proposed algorithm spent 21.70 minutes. This difference corresponds to 8.3 minutes in favor of TRIAS BDD. For processing context with 5,000 objects, TRIAS spent 55.18 minutes and TRIAS BDD took 35.32 minutes, which corresponds to a difference of 19.86 minutes in favor of the proposed algorithm. For processing context with 10,000 objects, TRIAS spent 119.53 minutes while TRIAS BDD spent 82.48 minutes, corresponding to a difference of 37.05 minutes.

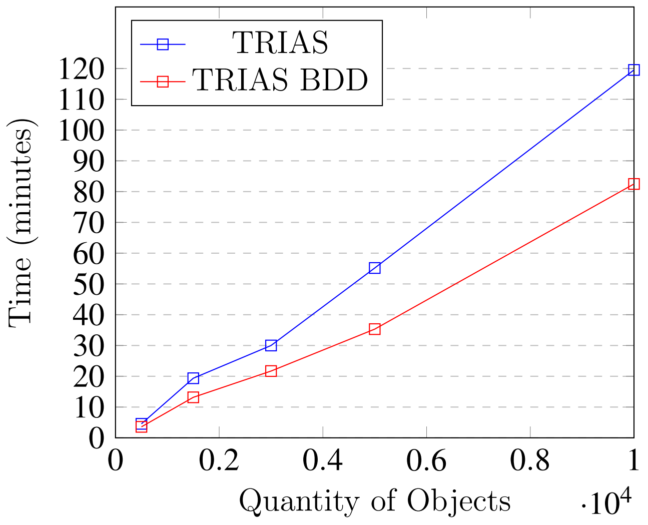


Figure 8: Contexts with 10 attributes, 5 conditions and 30% density

As indicated in Figure 9, considering the context of 10 attributes, 5 conditions and 50% density, for processing context with 500 objects TRIAS spent 7.09 minutes and TRIAS BDD spent 5.53 minutes, a difference of 1.56 minutes in favor of the proposed algorithm. For processing context with 1,500 objects, TRIAS spent 31.62 minutes and TRIAS BDD spent 25.61 minutes, which corresponds to a difference of 6.01 minutes less for the proposed algorithm. For processing context with 3,000 objects, TRIAS spent 54.85 minutes and TRIAS BDD spent 46.62 minutes. This difference corresponds to 8.23 minutes in favor of TRIAS BDD. For processing context with 5,000 objects, the original algorithm spent 130.37 minutes and the proposed algorithm spent 114.72, which corresponds to a difference of 15.65 minutes in favor of the proposed algorithm. For processing context with 10,000 objects, TRIAS spent 292.19 minutes while TRIAS BDD spent 227.91 minutes, corresponding to a difference of 64.28 minutes.

As it might be seen in Figure 10, considering the context of 10 attributes, 5 conditions and 70% density, for processing context with 500 objects, TRIAS spent 9.09 minutes and TRIAS BDD spent 7.37 minutes, a difference of 1.72 minutes in favor of TRIAS BDD. For processing context with 1,500 objects, the original algorithm spent 39.52 minutes and the proposed algorithm spent 33.20 minutes, which corresponds to a difference of 6.32 minutes less. For processing context with 3,000 objects, TRIAS

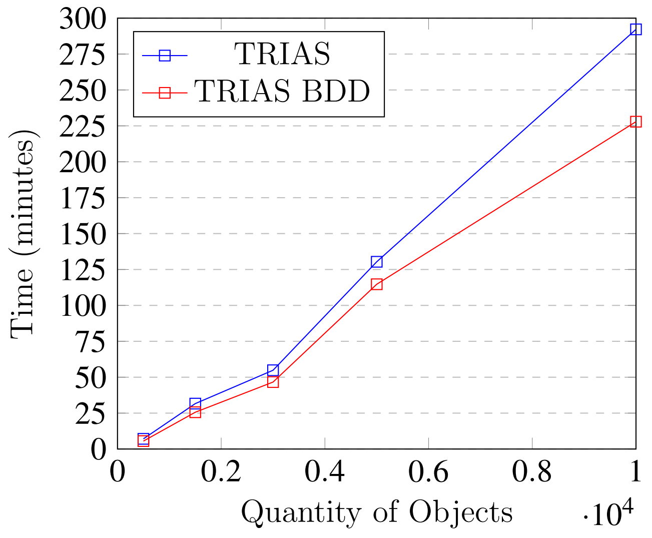


Figure 9: Contexts with 10 attributes, 5 conditions and 50% density

spent 62.22 minutes and TRIAS BDD spent 65.33 minutes. This difference corresponds to 3.11 minutes in favor of TRIAS. For processing context with 5,000 objects, the original algorithm spent 128.76 minutes and the proposed algorithm spent 121.03, which corresponds to a difference of 7.73 minutes in favor of the proposed algorithm. For processing context with 10,000 objects, the TRIAS BDD spent 269.61 minutes. TRIAS did not generate results within 7 days. Therefore, it was not possible to compare the algorithm's results.

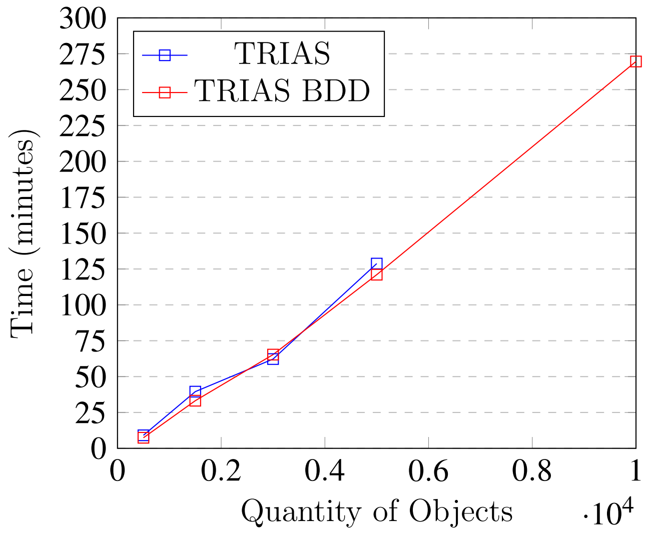


Figure 10: Contexts with 10 attributes, 5 conditions and 70% density

As presented in Table 18, it can be seen that the TRIAS BDD algorithm presented a higher speedup than TRIAS, varying between 19% and 22%, for density 30% and between 5% and 18% for density 50%. It is important to point out that TRIAS did not show results, at densities 30% and 50%, for 10,000 objects, considering 15 attributes and 5 conditions. Therefore, it was not possible to comparate the algorithm's results.

Analyzing the density of 70%, it is observed that the TRIAS BDD obtained results for the five contexts with the quantity of the following objects: 500, 1,500, 3,000, 5,000 and 10,000. However, TRIAS only presented results for 500 and 1,500 objects. For these values, TRIAS BDD showed a higher speedup than TRIAS, by 22% and 19%, respectively.

Table 18: TRIAS x TRIAS BDD Algorithm Results in 15 attributes and 5 conditions

Context (G x M x B)	Density (%)	Incidences	TRIAS (Minutes)	TRIAS BDD (Minutes)	SpeedUp (%)
500 x 15 x 5	30	11,250	42.68	**34.39**	19%
1,500 x 15 x 5	30	33,750	212.48	**166.60**	22%
3,000 x 15 x 5	30	67,500	376.20	**297.49**	21%
5,000 x 15 x 5	30	112,500	768.8	**595.97**	22%
10,000 x 15 x 5	30	225,000	-	**1,348.12**	-
500 x 15 x 5	50	18,750	53.20	**50.28**	5%
1,500 x 15 x 5	50	56,250	284.56	**256.36**	10%
3,000 x 15 x 5	50	112,500	575.94	**488.09**	15%
5,000 x 15 x 5	50	187,500	1,564.42	**1,282.31**	18%
10,000 x 15 x 5	50	375,000	-	**3,155.67**	-
500 x 15 x 5	70	26,250	122.35	**95.48**	22%
1,500 x 15 x 5	70	78,750	583.78	**474.26**	19%
3,000 x 15 x 5	70	157,500	-	**839.92**	-
5,000 x 15 x 5	70	262,500	-	**1,738.24**	-
10,000 x 15 x 5	70	525,000	-	**4,183.66**	-

In all contexts with 15 attributes, the TRIAS BDD was more efficient in all five object variations with speedup up to 22%. However, in contexts with 10 attributes and 5 conditions, the BDD implementation was faster for contexts with 30% and 50% densities. For density equal 70%, 3,000 objects, 10 attributes and 5 conditions the original algorithm was 5% faster. The main benefit that the BDD provided was to manipulate a more complex data structure.

As can be seen in Figure 11, considering the context of 15 attributes, 5 conditions and 30% density, for processing context with 500 objects TRIAS spent 42.68 minutes and TRIAS BDD spent 34.39 minutes, a difference of 8.29 minutes in favor of TRIAS BDD. For processing context with 1,500 objects, the original algorithm spent 212.48 minutes and the proposed algorithm spent 166.60 minutes, which corresponds to a difference of 45.88 minutes less. For processing context with 3,000 objects, TRIAS spent 376.20 minutes and TRIAS BDD spent 297.49 minutes. The difference corresponds to 78.71 minutes in favor of TRIAS BDD. For processing context with 5,000 objects, the original algorithm spent 768.8 minutes and the proposed algorithm spent 595.97, which corresponds to a difference of 172.83 minutes in favor of the proposed algorithm. For processing context with 10,000 objects, the TRIAS BDD spent 1,348.12 minutes. TRIAS did not generate results within 7 days. Therefore, it was not possible to compare the algorithm's results.

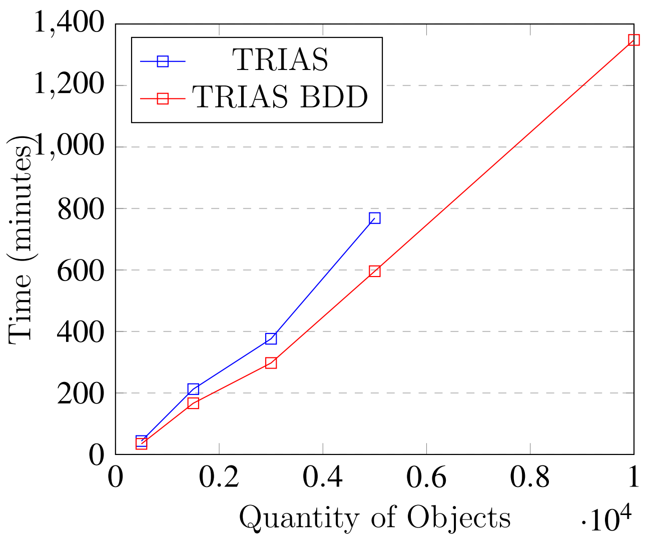


Figure 11: Contexts with 15 attributes, 5 conditions and 30% density

As presented in Figure 12, considering the context of 15 attributes, 5 conditions and 50% density, for processing context with 500 objects TRIAS spent 53.20 minutes and TRIAS BDD spent 50.28 minutes, a difference of 2.92 minutes in favor of TRIAS BDD. For processing context with 1,500 objects, the original algorithm spent 284.56 minutes and the proposed algorithm spent 256.36 minutes, which is a difference of 28.2 minutes less. For processing context with 3,000 objects, TRIAS spent 575.94 minutes and TRIAS BDD spent 488.09 minutes. The difference corresponds to 87.85 minutes in favor of TRIAS BDD. For processing context with 5,000 objects, the original algorithm spent 1,564.42 minutes and the proposed algorithm spent 1,282.31, which corresponds to a difference of 282.11 minutes in favor of the proposed algorithm. For processing context with 10,000 objects, the TRIAS BDD took 3,155.67 minutes. TRIAS did not generate results within 7 days. Therefore, it was not possible to compare the algorithm's results.

As can be observed in Figure 13, considering the context of 15 attributes, 5 conditions and 70% density, for processing context with 500 objects TRIAS spent 122.35 minutes and TRIAS BDD spent 95.48 minutes, a difference of 26.87 minutes in favor of TRIAS BDD. For processing context with 1,500 objects, the original algorithm spent 583.78 minutes and the proposed algorithm spent 474.26 minutes, which corresponds to a difference of 109.52 minutes less. For processing context with 3,000, 5,000 and 10,000 objects, TRIAS BDD spent 839.92, 1,738.24 and 4,183.66 minutes, respectively. TRIAS did not generate results within 7 days for 3,000, 5,000 and 10,000 objects. Therefore, it was not possible to compare the algorithm's results.

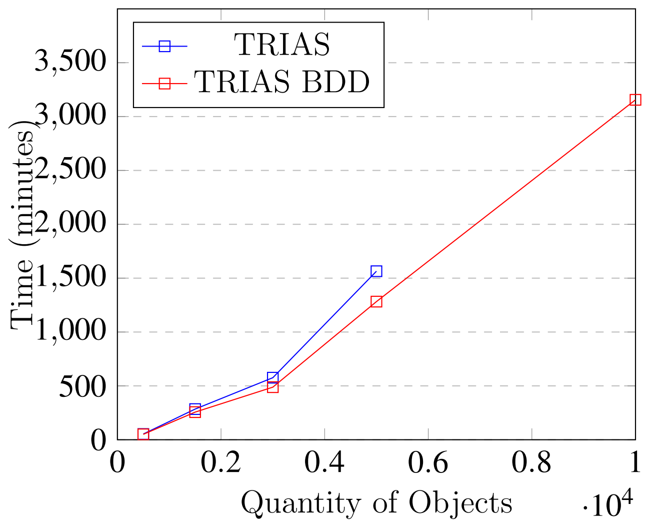


Figure 12: Contexts with 15 attributes, 5 conditions and 50% density

As presented in Table 19, it can be seen that the TRIAS BDD algorithm showed a higher speedup than the TRIAS for contexts with 500, 3,000 and 5,000 objects, corresponding 6%, 22% and 30%, for density equal to 30%. For context with 1,500 objects, TRIAS had a speedup 4% higher than TRIAS BDD. TRIAS did not obtain results for the processing context with 10,000 objects, with 30% density for 20 attributes and 5 conditions. Therefore, it was not possible to compare the algorithm's results.

Table 19: TRIAS x TRIAS BDD Algorithm Results in 20 attributes and 5 conditions

Context (G x M x B)	Density (%)	Incidences	TRIAS (Minutes)	TRIAS BDD (Minutes)	SpeedUp (%)
500 x 20 x 5	30	15,000	54.56	**51.29**	6%
1,500 x 20 x 5	30	45,000	**271.00**	281.84	-4%
3,000 x 20 x 5	30	90,000	479.95	**374.36**	22%
5,000 x 20 x 5	30	150,000	1,293.28	**903.89**	30%
10,000 x 20 x 5	30	300,000	-	**2,462.38**	-
500 x 20 x 5	50	25,000	70.93	**64.55**	9%
1,500 x 20 x 5	50	75,000	379.41	**318.70**	16%
3,000 x 20 x 5	50	150,000	767.93	**622.02**	19%
5,000 x 20 x 5	50	250,000	2,885.89	**1,279.88**	56%
10,000 x 20 x 5	50	500,000	-	**5,259.46**	-
500 x 20 x 5	70	35,000	223.56	**147.31**	34%
1,500 x 20 x 5	70	105,000	958.98	**614.45**	36%
3,000 x 20 x 5	70	210,000	-	**1,119.89**	-
5,000 x 20 x 5	70	350,000	-	**2,317.66**	-
10,000 x 20 x 5	70	700,000	-	**7,578.21**	-

Analyzing the results for 50% density, it is observed that the proposed algorithm presented a percentage of speedup higher than the original algorithm in 9% for context

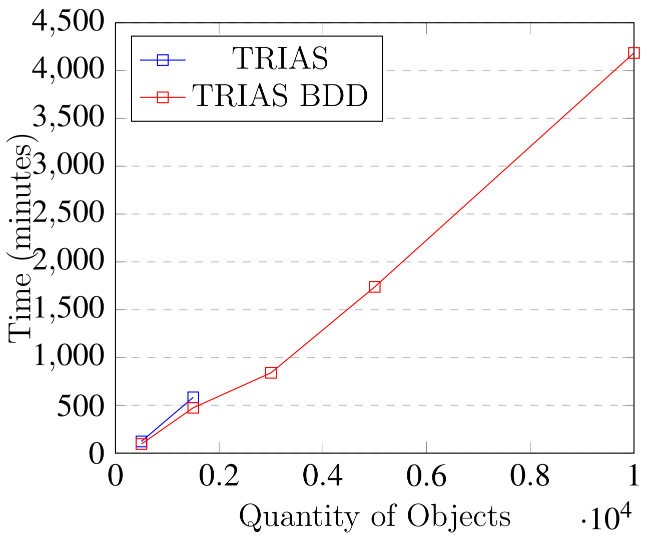


Figure 13: Contexts with 15 attributes, 5 conditions and 70% density

with 500, 16% for context with 1,500, 19 % for context with 3,000 and 56 % to context with 5,000 objects. It was not possible to perform comparative analysis for 10,000 objects because the original algorithm did not present any result.

Analyzing the same results for context with 70% density, it is observed that the TRIAS algorithm presented processing results only for contexts with 500 and 1,500 objects. For these results, TRIAS BDD showed a higher speedup than TRIAS, at 34% and 36%, respectively.

The concept extractions performed in contexts that have more than 250,000 incidences, TRIAS BDD was able to process contexts that the original implementation was not able to handle. Therefore, the use of the BDD was mandatory in these cases to enable concept extraction. As an example, contexts with 5,000 objects, 20 attributes, 5 conditions and 70% density - only TRIAS BDD was able to extract the concepts.

As a summary, the execution time for TRIAS become faster than TRIAS BDD in only two contexts (3,000 objects x 10 attributes x 5 conditions x 70% density and 1,500 objects x 20 attributes x 5 conditions x 30% density). However, the speedup in both cases was 5% and 4% when compared to TRIAS BDD. On the other hand, TRIAS BDD was faster in all remaining contexts (42 of 45 total).

As presented in Figure 14, considering the context of 20 attributes, 5 conditions and 30% density, for processing context with 500 objects, TRIAS spent 54.56 minutes and TRIAS BDD spent 51.29 minutes, a difference of 3.27 minutes in favor of TRIAS BDD. For processing context with 1,500 objects, TRIAS spent 271.00 minutes and TRIAS BDD spent 281.84 minutes, which corresponds to a difference of 10.84 minutes less for TRIAS. For processing context with 3,000 objects, the original algorithm required 479.95 minutes and the proposed algorithm required 374.36 minutes. This difference corresponds to 105.59 minutes in favor of the proposed algorithm. For processing context with 5,000 objects, TRIAS spent 1,293.28 minutes and TRIAS BDD spent 903.89, which corresponds to a difference of 389.39 minutes in favor of TRIAS BDD. For processing context with 10,000 objects, TRIAS BDD spent 2,462.38 minutes. TRIAS did not generate results within 7 days. Therefore, it was not possible to compare the algorithm's results.

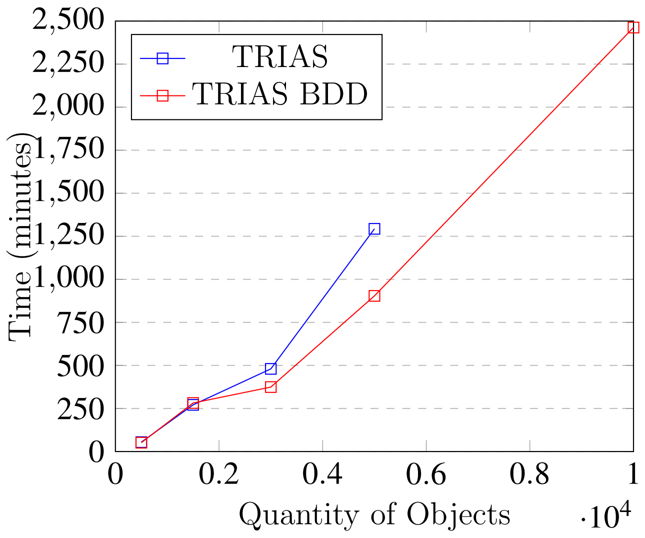


Figure 14: Contexts with 20 attributes, 5 conditions and 30% density

As can be seen in Figure 15, considering the context of 20 attributes, 5 conditions and 50% density, for processing context with 500 objects TRIAS spent 70.93 minutes and TRIAS BDD spent 64.55 minutes, a difference of 6.38 minutes in favor of TRIAS BDD. For processing context with 1,500 objects, the original algorithm spent 379.41 minutes and the proposed algorithm spent 318.70 minutes, which corresponds to a difference of 60.71 minutes less. For processing context with 3,000 objects, TRIAS spent 767.93 minutes and TRIAS BDD spent 622.02 minutes. This difference corresponds to 145.91 minutes in favor of TRIAS BDD. For processing context with 5,000 objects, the original algorithm spent 2,885.89 minutes and the proposed algorithm spent 1,279.88, which corresponds to a difference of 1,606.01 minutes in favor of the proposed algorithm. For processing context with 10,000 objects, TRIAS BDD spent 5,259.46 minutes. TRIAS did not generate results within 7 days. Therefore, it was not possible to compare the algorithm's results.

As presented in Figure 16, considering the context of 20 attributes, 5 conditions and 70% density, for processing context with 500 objects TRIAS spent 223.56 minutes and TRIAS BDD spent 147.31 minutes, a difference of 76.25 minutes in favor of TRIAS BDD. For processing context with 1,500 objects, the original algorithm spent 958.98 minutes and the proposed algorithm spent 614.45 minutes, which corresponds to a difference of 344.53 minutes less. For processing the contexts with 3,000, 5,000 and 10,000 objects,

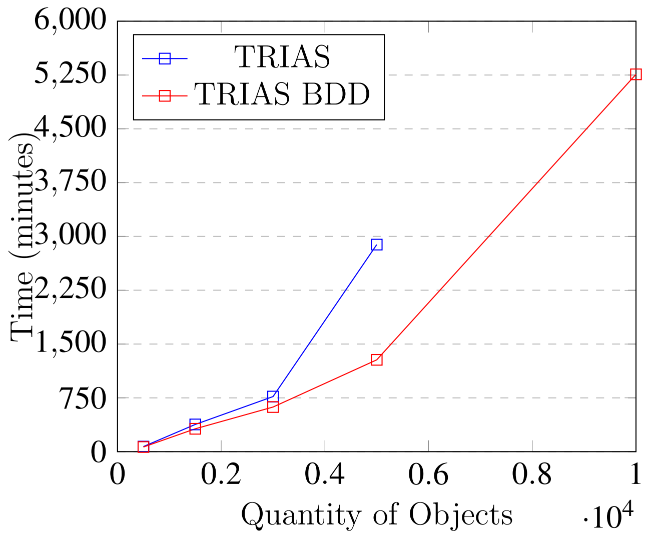


Figure 15: Contexts with 20 attributes, 5 conditions and 50% density

TRIAS BDD spent 1,119.89, 2,317.66 and 7,578.21 minutes, respectively. TRIAS did not generate results within 7 days for 3,000, 5,000 and 10,000. Therefore, it was not possible to compare the algorithm's results.

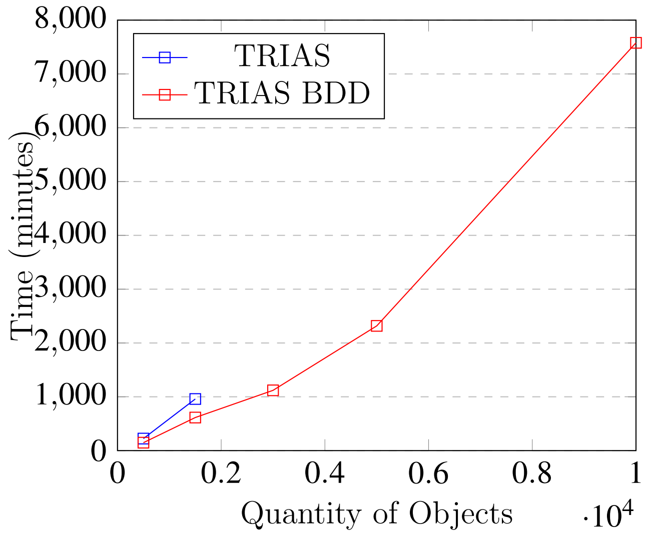


Figure 16: Contexts with 20 attributes, 5 conditions and 70% density

In 2006, the ICFCA (International Conference on Formal Concept Analysis) at Desdren (OLD; PRISS, 2006) discussed the main challenges of formal analysis. Since that date, the requirements to deal with dense and high-dimensional formal contexts have been discussed, such as contexts with 120,000 objects and 70,000 attributes, which are considerably larger than the experiments made here. In order to attend partially to the challenge, new synthetic triadic contexts were created using *SCGaz* (Table 20). Tests were performed and compared TRIAS and TRIAS BDD results for contexts with 120,000 objects, 5, 10 and 15 attributes, 5 and 10 conditions and 30%, 50% and 70% density. Initially, the number of objects and density of each context were fixed in 120,000 and 30% while changing the number of attributes and conditions. Secondly, it was varied the density of each context.

In Table 20, cells with the "-" symbol represent that the algorithm failed to complete the concepts extraction processing within 14 days. Note that in these scenarios the algorithms were dealing with contexts that have a high number of incidences, varying from 1,776,769 a 6,299,886.

Table 20: TRIAS x TRIAS BDD Algorithm Results for High-dimensional Contexts

Context (G x M x B)	Density	Incidences	TRIAS (Days)	TRIAS BDD (Days)
120,000 x 15 x 5	30%	2,699,984	-	12.79
120,000 x 10 x 5	30%	1,776,769	-	10.88
120,000 x 5 x 10	30%	1,776,769	-	10.46
120,000 x 15 x 5	50%	4,498,986	-	-
120,000 x 10 x 5	50%	2,999,948	-	13.16
120,000 x 5 x 10	50%	2,999,948	-	13.54
120,000 x 15 x 5	70%	6,299,886	-	-
120,000 x 10 x 5	70%	4,199,988	-	-
120,000 x 5 x 10	70%	4,199,988	-	-

As presented in Table 20, in none of the experiments, the TRIAS algorithm was able to extract concepts within 14 days. The algorithm was not able to deal with those contexts called high-dimensional. However, TRIAS BDD retrieved concepts before 14 days in all cases of contexts with 30% density. As an example, it completed the execution in 10.88 days for a context with a density of 30%, 120,000 objects, 10 attributes and 5 conditions. It was also able to extract concepts for contexts with a density of 50% considering the limit of 2,999,948 incidences. In these cases, with the use of BDDs, optimizing the representation, it confirmed to be an efficient solution. Therefore, the use of BDD is an interesting approach in high-dimensional contexts.

Table 20 shows the results obtained for high-dimensional contexts. Considering 120,000 objects, for densities 30%, 50% and 70%, three different combinations between the number of attributes and conditions were tested, such as: 15 attributes and 5 conditions, 10 attributes and 5 conditions and 5 attributes and 10 conditions.

According to Table 20, considering the density of 30%, it can be seen that TRIAS did not generate results for the three combinations of attributes and conditions. TRIAS BDD spent 12.79 days to process context with 120,000 objects, 15 attributes and 5 objects. The processing of the context with 120,000 objects, with 10 attributes and 5 conditions, TRIAS BDD spent 10.88 days. Also, for processing context with 120,000 objects, 5 attributes and 10 conditions, TRIAS BDD spent 10.46 days. A variation of 3.86% is perceived between the time spent by both.

Moreover, it can be observed in Table 20, considering the density of 50%, TRIAS did not generate results for the three combinations of attributes and conditions. TRIAS BDD did not generate results for context with 120,000 objects, 15 attributes and 5 conditions. A variation of 2.88% is perceived between the time spent by TRIAS BDD for processing context with 120,000 objects, 10 attributes and 5 conditions (13.16 days) and the time spent by TRIAS BDD for processing context with 120,000 objects, 5 attributes and 10 conditions (13.54 days).

Also, according to Table 20, it is observed that, considering the density 70%, context with 120,000 objects and the three combinations of attributes and conditions (15 attributes x 5 conditions, 10 attributes x 5 conditions and 5 attributes x 10 conditions), neither TRIAS nor TRIAS BDD were able to generate results.

However, note that even this new solution is not efficient when the number of incidences exceeds 3,000,000 - both algorithms were not able to complete the concept extractions for all contexts with 70% density within 14 days. But, for the contexts presented in Table 20, only the TRIAS BDD was able to extract the concepts.

5.2 Results for real datasets contexts

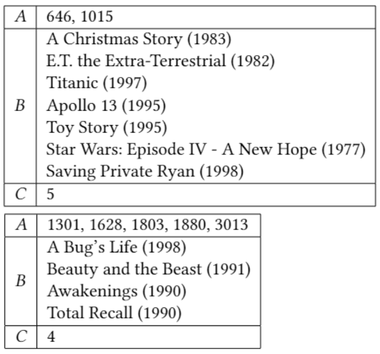
The use of synthetic databases is an interesting strategy to evaluate algorithms. However, understand the behavior of the algorithm in real scenarios is extremely important to understand its real efficiency. Considering that, it was applied both algorithms to an extensive database of movie ratings called *MovieLens*[1]. The database consists of ratings of more than 6,000 anonymous users in approximately 4,000 movies. It has more than 1,000,000 records with users rate movies from 1 to 5 ratings. This base is considered sparse, meaning despite the huge number of ratings, there is no guarantee that users rated the same movies, consequently generating sparse triadic contexts regarding the number of incidences. For example, it is not possible to confirm if user A has rated the same movies as user B. Additionally, in the movie range (attributes) that are required to be evaluated, it is not possible to confirm that the user rated (conditions) all movies. In other words, the user may have evaluated only a single movie within the possible movies available in the context.

The contexts generated from the database have as a set of objects the users who have made classifications, the attributes of the context are the classified movies and the conditions are the rates received. Then, the triadic context can be defined as $\mathscr{K}$ =

[1]https://grouplens.org

(K_1, K_2, K_3, Y) where K_1 are the set of users in the dataset, K_2 are set of movies, K_3 the set of rates given by users and Y are the relation between users and movies and their respective rates.

Figure 17 represents one example of a triadic concept extract from the dataset movie. Note that in the first concept users 646, 1015 rated the same movie with the same rate.



A	646, 1015
B	A Christmas Story (1983) E.T. the Extra-Terrestrial (1982) Titanic (1997) Apollo 13 (1995) Toy Story (1995) Star Wars: Episode IV - A New Hope (1977) Saving Private Ryan (1998)
C	5

A	1301, 1628, 1803, 1880, 3013
B	A Bug's Life (1998) Beauty and the Beast (1991) Awakenings (1990) Total Recall (1990)
C	4

Figure 17: Example of triconcepts from *MovieLens* Dataset

In Table 21, cells with the "-" symbol represent that the algorithm failed to complete the concepts extraction processing within 7 days. It shows the results of the algorithms applied to contexts generated from the real dataset *MovieLens*. It was decided to fix the number of objects (users) and conditions (rates), using the maximum number of objects available in the dataset, and varied the number of attributes (movies) of the context.

Table 21: TRIAS x TRIAS BDD Algorithm Results for *MovieLens* dataset

Context (G,M,B) (G = Users $\times$ M = Movies $\times$ B = Rates)	Density	Incidences	TRIAS (Days)	TRIAS BDD (Days)
6,000 x 100 x 5	17.3%	105,986	-	5.88
6,000 x 150 x 5	16.4%	158,029	-	6.12
6,000 x 200 x 5	14.3%	203,898	-	6.85

As presented in Table 21, the TRIAS and TRIAS BDD algorithms were applied to the *MovieLens* dataset. Three different scenarios were considered, such as: 1) 6,000 users (objects), 100 movies (attributes) and a rates (conditions) equal to 5; 2) 6,000 users, 150 movies and 5 rates; 3) 6,000 users, 200 movies and 5 rates. It is noticeable, the number of users and the rate were maintained in all analyzes. Only the quantity of movies was varied between 100, 150 and 200. For the context of 100 movies, a density of 17.3% was considered. For 150 movies, a density of 16.4% was considered and, finally, for 200 movies, a density of 14.3% was considered.

Note that TRIAS Algorithm did not complete the computation of all formal concepts, during a period of seven days. TRIAS BDD obtained results for 100, 150 and 200 movies, it spent 5.88, 6.12 and 6.85 days, respectively.

The larger formal context created from MovieLens (6,000 x 200 x 5) has approximately only 20% of the full database, reinforcing that the triadic approach using the algorithm TRIAS was not able to deal with the dataset, even in small subsets of real data. However, TRIAS BDD was able to compute the results for the context with 100 and 150 attributes before 7 days.

5.3 RAM Memory Usage Analysis

Figures, 18, 19, 20, below, present the behavior of the RAM memory in three situations that were evaluated in this research. The purpose of these figures is to present a comparative analysis between the RAM memory usage by TRIAS and TRIAS BDD.

In order to have a better understanding of the behavior of the RAM memory, an analysis of three scenarios was evaluated. The first, which considers 5,000 objects, 20 attributes and 5 conditions, where TRIAS BDD presented a shorter processing time than TRIAS. And the second and third figures reflect the behavior of the memory where TRIAS BDD spent more processing time than TRIAS.

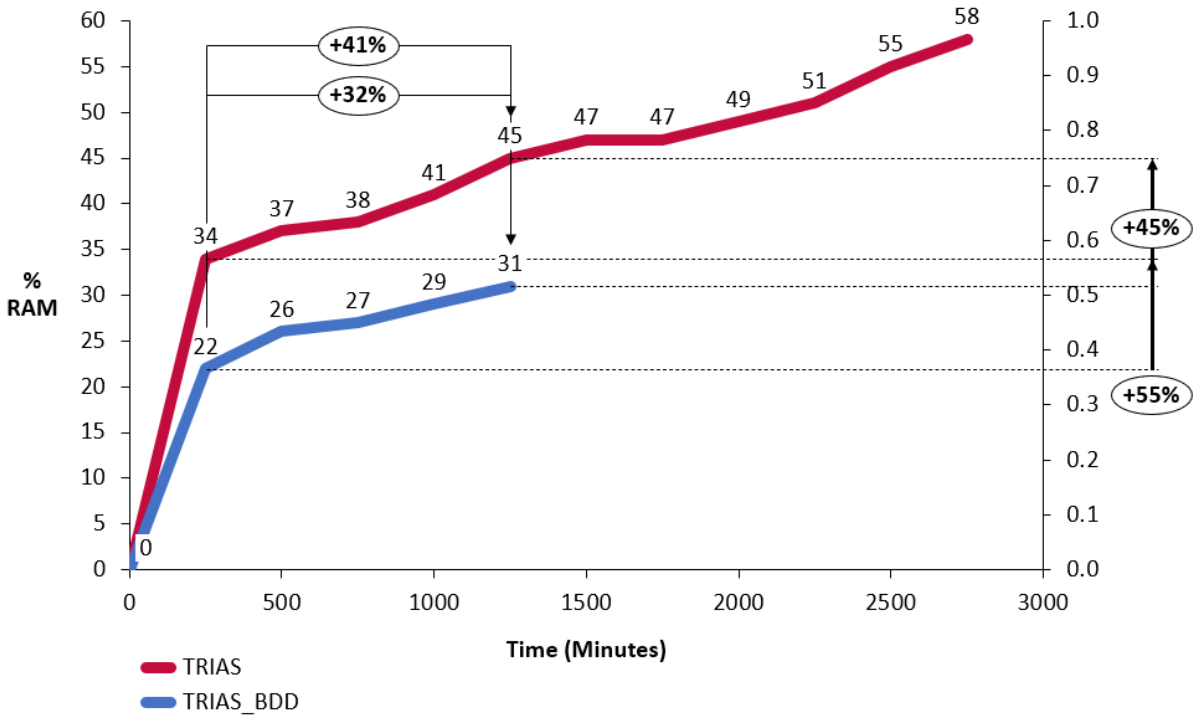


Figure 18: Memory Usage of the context with 5,000 objects, 20 attributes and 5 conditions - presented in Table 19

It can be seen in Figure 18 that in the first 250 minutes of processing, TRIAS

BDD had consumed 22% of RAM and TRIAS had consumed 34%, that is, in 250 minutes, TRIAS had consumed 55% more RAM memory than TRIAS BDD, for processing of 5,000 objects, 20 attributes and 5 conditions.

After 250 minutes, the memory consumption by TRIAS and TRIAS BDD showed a steady growth, but less accentuated. TRIAS BDD spent 1279.88 minutes to complete the entire processing of 5,000 objects, 20 attributes and 5 conditions. After 250 minutes, the RAM memory consumption of TRIAS BDD went from 22% to 26% in 500 minutes, 29% in 1,000 minutes and 31% in 1,279.88 minutes, when processing was completed, a 41% increase in memory consumption RAM from 250 minutes to 1,250 minutes.

In contrast, TRIAS spent 2,885.89 minutes to complete the processing of 5,000 objects, 20 attributes and 5 conditions. After 250 minutes, the RAM memory consumption went from 34% to 37% in 500 minutes, 41% in 1,000 minutes, 49% in 2,000 minutes and 58% in 2,885.89, when the processing was finished.

As presented in Figure 18, it can be said that TRIAS BDD consumed 22% of RAM in the first 250 minutes of processing and 31% at 1,279.88 minutes, when processing was finished. TRIAS consumed 34% of RAM in the first 250 minutes of processing and 58% at 2,885.89 minutes, when processing was completed.

A comparative analysis between TRIAS and TRIAS BDD processing enables us to understand that TRIAS, between 250 minutes and 1,250 minutes, had a 32% increase in memory consumption, going from 34% to 45% of RAM memory usage. In contrast, TRIAS BDD had a 41% increase in RAM memory consumption, from 22% to 31% in memory usage. It is important to note that, in the same time interval, TRIAS BDD demanded 9% more memory increase than TRIAS. However, TRIAS BDD finished processing 5,000 objects, 20 attributes and 5 conditions in 1,279.88 minutes, consuming 31% of memory, while TRIAS finished processing only at 2,885.89 minutes, with 58% of memory usage.

Another information that Figure 18 present is that, considering the processing time of 250 minutes, TRIAS BDD had consumed 22% of RAM, while TRIAS had consumed 34%, that is, 55% more than TRIAS BDD.

Finally, considering the final time of the comparative analysis, that is, the interval between 250 and 1,250 minutes, TRIAS BDD had consumed 31% of RAM memory and TRIAS 45%, 45% more of RAM memory consumption.

As presented in Figure 19, it can be observed that TRIAS BDD spent 65.33 minutes to perform all the processing of 3,000 objects, 10 attributes and 5 conditions and TRIAS took 62.22 minutes to perform the same processing, in other words, TRIAS was 5% faster. However, analyzing the consumption of RAM memory, during 60 minutes of processing, it is evident that TRIAS BDD consumed 26% of RAM memory, while TRIAS consumed

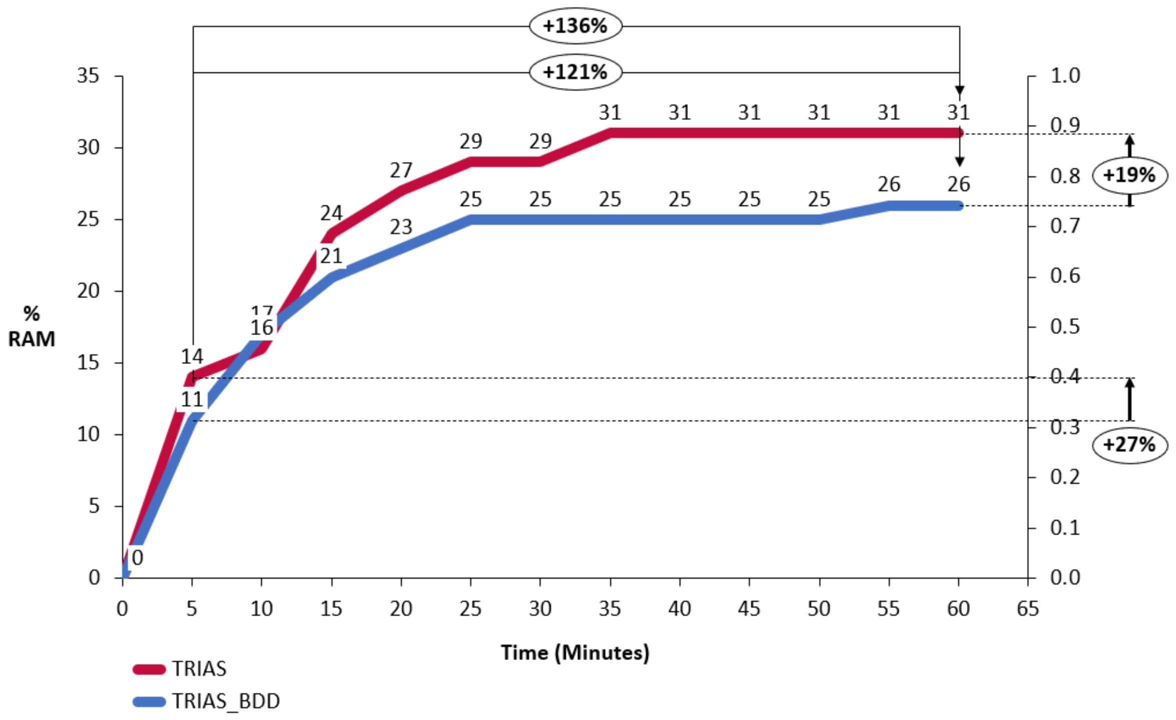


Figure 19: Memory Usage of the context with 3,000 objects, 10 attributes and 5 conditions - presented in Table 17

31%, that is, TRIAS consumed 19% more than the TRIAS BDD.

In the first 5 minutes of processing, TRIAS BDD had consumed 11% of RAM memory and TRIAS had consumed 14%, that is, in 5 minutes, TRIAS consumed 27% more RAM memory than TRIAS BDD, for the processing of 3,000 objects, 10 attributes and 5 conditions.

Evaluating the interval between 5 and 30 minutes of processing of TRIAS and TRIAS BDD, it can be observed that TRIAS BDD went from 11% to 25% of RAM memory, an increase of 127%. TRIAS increased from 14% to 29% of RAM memory usage, which corresponds to an increase of 107%. After 30 minutes of processing, there was a small variation between the percentage of RAM memory used by the two algorithms, tending to a constant. Conform can be seen in Figure 19, TRIAS BDD went from 25% to 26%, an increase of 4% and TRIAS went from 29% to 31%, which corresponds to 7%.

A comparative analysis between TRIAS and TRIAS BDD processing demonstrates us to understand that TRIAS that the TRIAS BDD, between 5 minutes and 60 minutes, had an increase of 136% in memory consumption, going from 11% to 26% of RAM memory usage, in this interval. TRIAS, on the other hand, had a 121% increase in RAM consumption, from 14% to 31% in memory usage. However, although TRIAS BDD has a greater memory variation than TRIAS in this interval, when evaluating the total amount of memory consumed in 60 minutes, it is evident that TRIAS BDD used 26% of the memory in its processing, while TRIAS used 31%, as previously presented. Finally, when evaluating the beginning of this period, in other words, 5 minutes of processing, it is evident that TRIAS used 14% of RAM memory and TRIAS BDD, 11%, that is, TRIAS BDD used 27% less RAM memory than TRIAS. At the end of the evaluated period, the memory consumption by TRIAS, 31%, was higher than that of TRIAS BDD, 26%, by 19%.

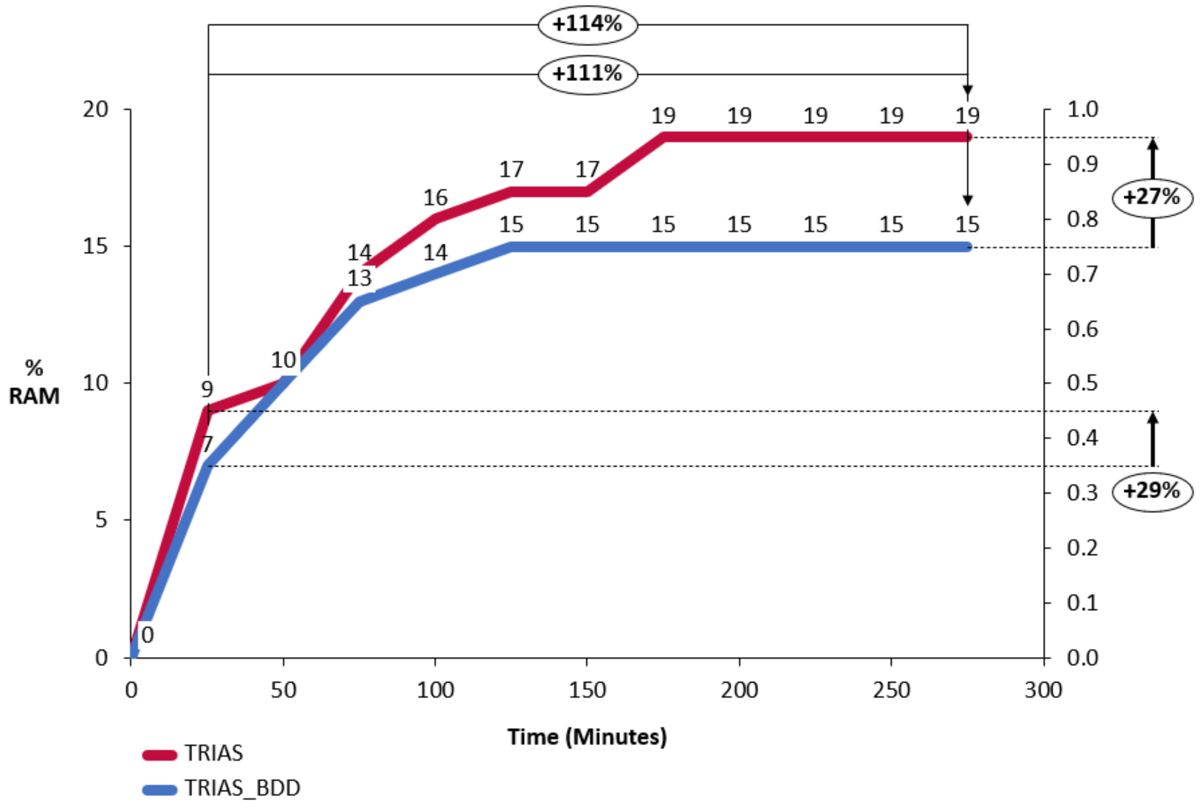


Figure 20: Memory Usage of the context with 1,500 objects, 20 attributes and 5 conditions - presented in Table 19

As presented in Figure 20, it can be seen that TRIAS BDD spent 281.84 minutes to perform all the processing of 1,500 objects, 20 attributes and 5 conditions and TRIAS took 271 minutes to perform the same processing, in other words, TRIAS was 4% faster. However, analyzing the consumption of RAM memory, during the 271 minutes of processing, it is evident that TRIAS BDD consumed 15% of RAM, while TRIAS consumed 19%, that is, TRIAS consumed 27% more than the TRIAS BDD.

In the first 25 minutes of processing, TRIAS BDD had consumed 7% of RAM and TRIAS had consumed 9%, that is, in 25 minutes, TRIAS had consumed 29% more RAM than TRIAS BDD, for the processing of 1,500 objects, 20 attributes and 5 conditions.

Evaluating the interval between 25 and 150 minutes of processing of TRIAS and TRIAS BDD, it can be observed that TRIAS BDD went from 7% to 15% of RAM memory usage, an increase of 114%. TRIAS increased from 9% to 17% of RAM memory usage, which corresponds to an increase of 89%. After 125 minutes of processing, TRIAS BDD kept the consumption of RAM memory constant, at 15%. TRIAS also showed constant memory consumption after 175 minutes, when the memory consumption remained at 19% until the end of processing.

A comparative analysis between the TRIAS and TRIAS BDD processing permits us to understand that the TRIAS BDD, between 25 minutes and 271 minutes, had an increase of 114% in memory consumption, going from 7% to 15% of RAM memory use, in this interval. TRIAS, on the other hand, had a 111% increase in RAM consumption, from 9% to 19% in memory usage. However, despite the fact that TRIAS BDD has a greater memory variation than TRIAS in this interval, when evaluating the total amount of memory consumed in 271 minutes, it is noticed that TRIAS BDD used 15% of the memory in its processing, while TRIAS used 19%, as previously presented. Finally, when evaluating the beginning of this period, that is, 25 minutes of processing, it is evident that TRIAS used 9% of RAM and TRIAS BDD, 7%, that is, 29% less than TRIAS. At the end of the evaluated period, memory consumption by TRIAS, 19%, was higher than that of TRIAS BDD, 15% by 27%.

5.4 Ordering the triadic context by attributes and conditions (column)

As described in section 2.3, the JavaBDD library was used in this research. One of its characteristics is the random selection of the input order of the variables. In other words, the variables are randomly ordered. However, it is extremely important to evaluate the order of entry of the variables, as it impacts the number of nodes generated in the BDD directly, the volume of data to be stored in RAM memory and, later, processed in the algorithm, changing the processing time.

For the contexts in which the results of the TRIAS BDD algorithm obtained less processing time results in relation to TRIAS. For example, in those contexts where the execution time of TRIAS BDD was greater than the execution time of TRIAS, a heuristic of prior ordering of the variables was created, manipulating an order of priority in the entry of variables in the BDD.

In order to analyze the impact of ordering variables on the data to be processed the following heuristics were implemented:

- Sorting per column (attribute x condition) from the higher to lower density, as presented in the Table 23;
- Sorting per column (attribute x condition) from the lower to higher density, as

presented in the Table 24

Table 22 presents an example of a triadic context used for this test. As it can be seen, the ordering of the input of the variables is random and a triadic to dyadic projection can be observed. Thus, according to the example, for eight objects, where K_1 represents the set of objects, K_2 the attributes and K_3 the conditions, the following occurrences were found: attribute 1 and condition 1, 4 occurrences; attribute 2 and condition 1, 5 occurrences; attribute 1 and condition 2, 3 occurrences; attribute 2 and condition 2, 7 occurrences; attribute 1 and condition 3, 6 occurrences; attribute 2 and condition 3, 3 occurrences.

Table 22: Dyadic Context Projection Example

$K_1/K_2{\times}K_3$	a_1c_1	a_2c_1	a_1c_2	a_2c_2	a_1c_3	a_2c_3
$object_1$	×	×	×	×	×	
$object_2$		×		×	×	
$object_3$	×		×	×		×
$object_4$		×			×	
$object_5$	×	×		×	×	
$object_6$				×	×	×
$object_7$		×	×	×		×
$object_8$	×			×	×	
Quantity	4	5	3	7	6	3

Tables 23 and 24 present the same objects, attributes and conditions. However, in a different ordering. In Table 23, objects were ordered from highest to lowest density, and in Table 24, objects were ordered from lowest to highest density, as presented below.

Table 23: Sorting per column (attribute x condition) from the higher to a lower density of the Table 22

$K_1/K_2{\times}K_3$	a_2c_2	a_1c_3	a_2c_1	a_1c_1	a_1c_2	a_2c_3
$object_1$	×	×	×	×	×	
$object_2$	×	×	×			
$object_3$	×			×	×	×
$object_4$		×	×			
$object_5$	×	×	×	×		
$object_6$	×	×				×
$object_7$	×		×		×	×
$object_8$	×	×		×		
Quantity	7	6	5	4	3	3

As presented in Table 25, it can be observed a comparative analysis between the processing time spent by TRIAS, TRIAS BDD, TRIAS BDD ordered from the highest density to the lowest (Descending order) and in Table 26 TRIAS BDD ordered from the lowest density to the highest density (Ascending order).

Table 24: Sorting per column (attribute x condition) from the lower to a higher density of the Table 22

$K_1/K_2{\times}K_3$	a_1c_2	a_2c_3	a_1c_1	a_2c_1	a_1c_3	a_2c_2
$object_1$	×		×	×	×	×
$object_2$				×	×	×
$object_3$	×	×	×			×
$object_4$				×	×	
$object_5$			×	×	×	×
$object_6$		×			×	×
$object_7$	×	×		×		×
$object_8$			×		×	×
Quantity	3	3	4	5	6	7

It is noticed that, for the context of 1,500 objects, 20 attributes and 5 conditions, TRIAS spent 271 minutes to perform all the processing and TRIAS BDD spent 281.84, that is, TRIAS BDD spent 4% more time to finish processing.

Analyzing the information regarding the processing performed by TRIAS BDD ordered from highest to lowest density, it can be observed that the TRIAS BDD ordered from highest to lowest density spent 246.21 minutes to perform the processing, that is, 9% less than TRIAS and 13% less than TRIAS BDD.

Observing the processing data performed by TRIAS BDD ordered from the lowest to the highest density, it can be seen that the TRIAS BDD ordered from the lowest to the highest density spent 254.79 minutes to perform the processing, that is, 6% less than TRIAS and 10% less than TRIAS BDD.

Evaluating the context of 3,000 objects, 10 attributes and 5 conditions, where TRIAS spent 62.22 minutes and TRIAS BDD 65.33 minutes, that is, TRIAS BDD spent 5% more than TRIAS for processing the context.

Analyzing the information referring to the processing performed by TRIAS BDD ordered from highest to lowest density, it is observed that the TRIAS BDD ordered from highest to lowest density took 53.21 minutes to perform the processing, that is, 14% less than TRIAS and 19% less than TRIAS BDD.

Verifying the processing data performed by the TRIAS BDD ordered from the lowest to the highest density, it can be seen that the TRIAS BDD ordered from the lowest to the highest density spent 56.92 minutes to perform the processing, that is, 9% less than TRIAS and 13% less than TRIAS BDD.

5.4.1 TRIAS × TRIAS BDD Descending and Ascending Order Analysis

In Table 25, TRIAS BDD Descending order represents the behavior of the TRIAS BDD ordered from the highest to the lowest density, and in Table 26 TRIAS BDD

Ascending order represents the behavior of the TRIAS BDD ordered from the lowest to the highest density.

Table 25: TRIAS, TRIAS BDD and TRIAS BDD Descending Order

Context	TRIAS	TRIAS BDD	TRIAS BDD Descending Order	Descending Order x TRIAS	Descending Order x TRIAS BDD
1,500x20x5 (30%)	271.00	281.84	246.21	9%	13%
3,000x10x5 (70%)	62.22	65.33	53.21	14%	19%

Table 26: TRIAS, TRIAS BDD and TRIAS BDD Ascending Order

Context	TRIAS	TRIAS BDD	TRIAS BDD Ascending Order	Ascending Order x TRIAS	Ascending Order x TRIAS BDD
1,500x20x5 (30%)	271.00	281.84	254.79	6%	10%
3,000x10x5 (70%)	62.22	65.33	56.92	9%	13%

In the following, the behavior of RAM memory will be presented, using the ordering from highest to lowest density, for the two contexts that TRIAS BDD presented a longer processing time than TRIAS: 1- 3,000 objects, 10 attributes and 5 conditions; 2- 1,500 objects, 20 attributes and 5 conditions.

The Figure 21 shows the behavior of RAM memory for 3,000 objects, 10 attributes and 5 conditions, with ordering the entry of variables from higher to lower density.

Through a comparative analysis between Figures 21 and 19, it can be seen that, ordering the input of the variables from the highest to the lowest density, the consumption of RAM memory by TRIAS BDD, at the end of the processing, was equal at 26%, while TRIAS BDD from higher to lower density consumed 24%, a reduction of 8%.

Analyzing the first five minutes of processing, TRIAS BDD had consumed 11% of RAM and TRIAS had consumed 14%, that is, TRIAS used 27% more RAM.

Between 5 and 30 minutes of processing, both showed an increase in RAM consumption. TRIAS BDD went from 11% to 21%, an increase of 91% and TRIAS went from 14% to 29%, which corresponds to 107%.

From 30 minutes of processing to 53.21 minutes, TRIAS BDD went from 21% to 24%, an increase that corresponds to 14%. TRIAS went from 29% to 31%, which corresponds to an increase of 7%.

In Figure 22, a comparative analysis was performed between the behavior of memory consumption by TRIAS BDD ordered from highest to lowest density and TRIAS in the interval between 5 minutes and 50 minutes of processing. At the end of the processing, TRIAS BDD had consumed 24% of RAM memory and TRIAS 31%, that is, TRIAS BDD consumed 29% less RAM memory than TRIAS.

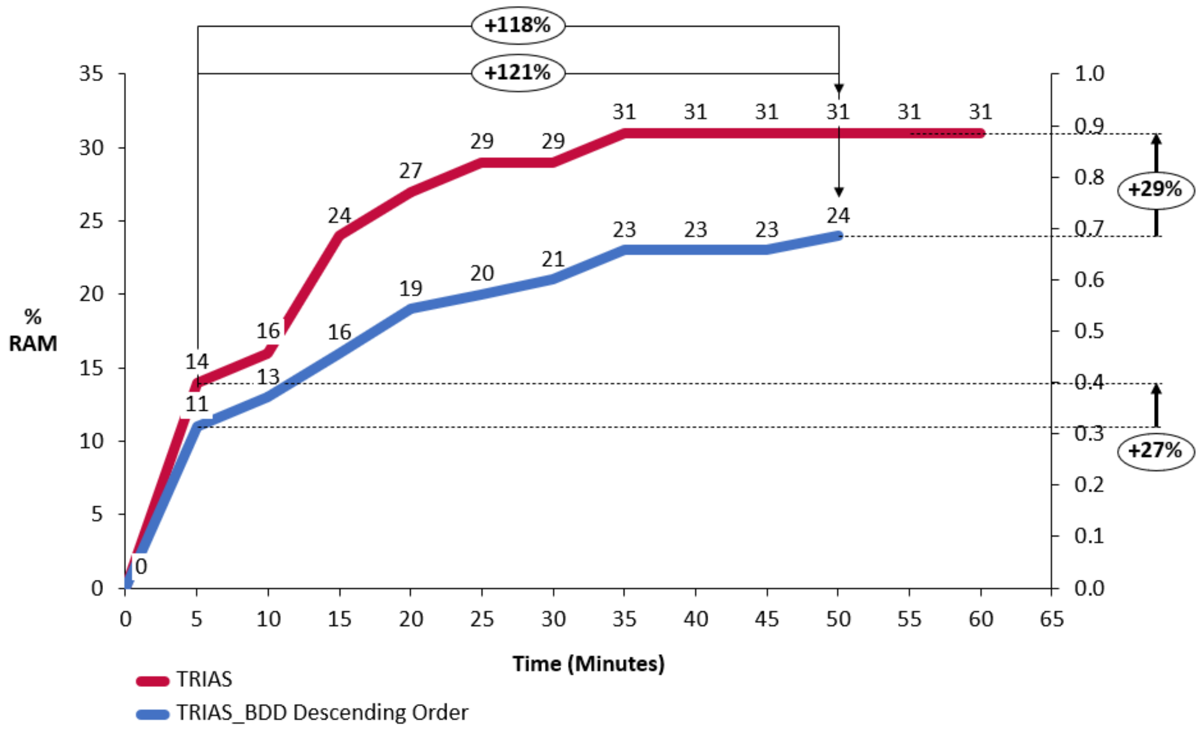


Figure 21: Descending Ordering - Memory Usage of the context with 3,000 objects, 10 attributes and 5 conditions - presented in Table 19

In the same interval, memory consumption by TRIAS increased from 14% to 31%, an increase of 121%. TRIAS BDD went from 11% to 24%, which corresponds to a variation of 118%.

From the results obtained, it can be seen that the usage of TRIAS BDD from lower to higher density, for 3,000 objects, 10 attributes and 5 conditions, generated a reduction in processing time and in RAM usage.

The Figure 22 presents the behavior of RAM memory for 3,000 objects, 10 attributes and 5 conditions, with ordering the entry of attributes x condition from lower to higher density.

Through a comparative analysis between graph 22 and graph 19, it can be seen that, by ordering the entry of the attributes x conditions from the lowest to the highest density, the consumption of RAM memory by TRIAS BDD, at the end of processing, was equal at 26%, while TRIAS BDD from lower to higher density consumed 25%, a reduction of 4%.

Analyzing the first five minutes of processing, TRIAS BDD had consumed 12% of RAM and TRIAS had consumed 14%, that is, TRIAS used 17% more of RAM.

Between 5 and 30 minutes of processing, both showed an increase in the RAM

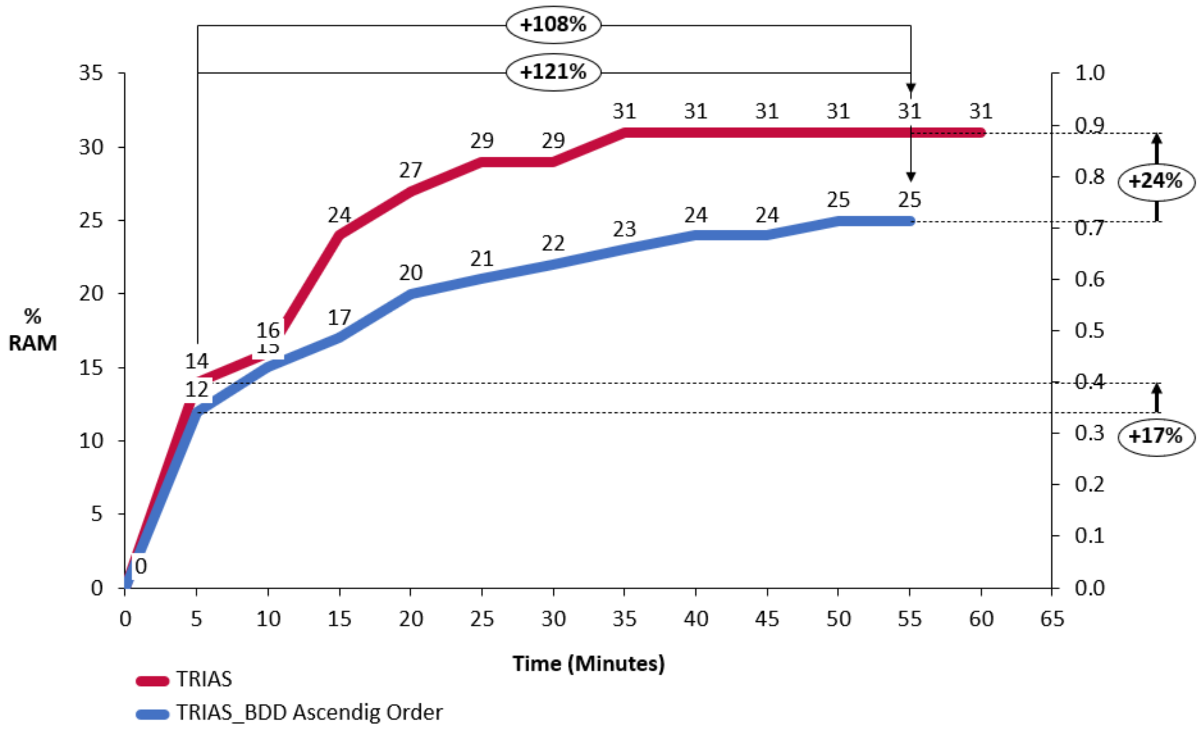


Figure 22: Ascending Ordering - Memory Usage of the context with 3,000 objects, 10 attributes and 5 conditions - presented in Table 19

consumption. TRIAS BDD went from 12% to 22%, an increase of 83% and TRIAS went from 14% to 29%, which corresponds to 107%.

From 30 minutes of processing to 56.92 minutes, TRIAS BDD went from 22% to 25%, an increase that corresponds to 14%. TRIAS went from 29% to 31%, that is, an increase of 7%.

Observing Figure 22, a comparative analysis was performed between the behavior of memory consumption by TRIAS BDD ordered from lowest to highest density and TRIAS in the interval between 5 minutes and 55 minutes of processing. At the end of the processing, TRIAS BDD had consumed 25% of RAM memory and TRIAS 31%, that is, TRIAS BDD consumed 24% less RAM memory than TRIAS.

In the same interval, memory consumption by TRIAS increased from 14% to 31%, an increase of 121%. TRIAS BDD went from 12% to 25%, which corresponds to a variation of 108%.

From the results obtained, it can be seen that the usage of TRIAS BDD from higher to lower density, for 3,000 objects, 10 attributes and 5 conditions, generated a reduction in processing time and in RAM usage.

Figure 23 shows the behavior of RAM memory for 1,500 objects, 20 attributes and 5 conditions, with ordering the entry of variables from higher to lower density.

Through a comparative analysis between Figures 23 and 19, it can be seen that, by ordering the input of the attributes x conditions from the highest to the lowest density, the RAM memory consumption by TRIAS BDD, at the end of the processing, was equal at 15%, while TRIAS BDD from higher to lower density consumed 13%, a reduction of 13%.

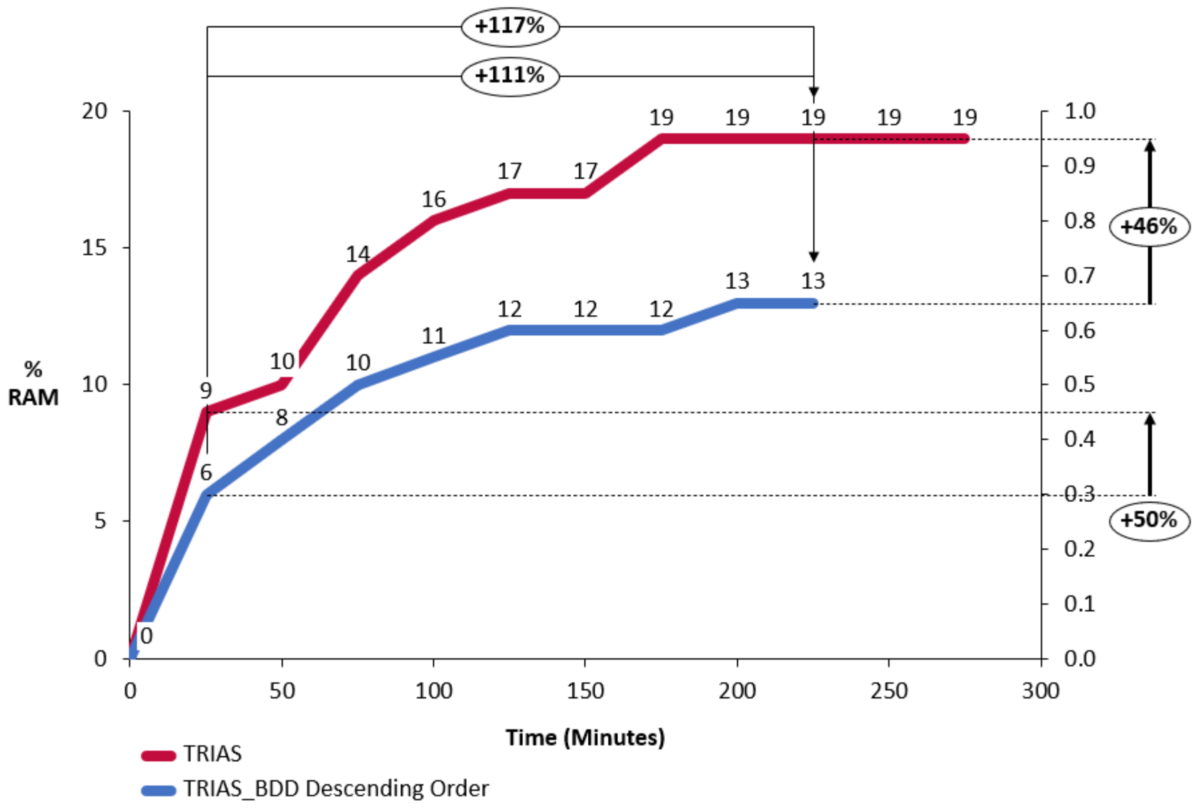


Figure 23: Descending Order - Memory Usage of the context with 1,500 objects, 20 attributes and 5 conditions - presented in Table 19

Analyzing the first 25 minutes of processing, TRIAS BDD had consumed 6% of RAM and TRIAS had consumed 9%. In other words, TRIAS used 50% more of RAM memory.

Between 25 and 125 minutes of processing, both presented an increase in the RAM consumption. TRIAS BDD went from 6% to 12%, an increase of 100% and TRIAS went from 9% to 17%, which corresponds to 89%.

From 125 minutes of processing to 246.21 minutes, TRIAS BDD went from 12% to 13%, an increase that corresponds to 8%. TRIAS went from 17% to 19%. In other words, an increase of 12%.

Observing Figure 23, a comparative analysis was performed between the behavior of memory consumption by TRIAS BDD ordered from highest density to lowest density and TRIAS in the interval between 25 minutes and 225 minutes of processing. At the end of the processing, TRIAS BDD had consumed 13% of RAM memory and TRIAS 19%. In other words, TRIAS BDD consumed 46 % less RAM memory than TRIAS.

In the same interval, memory consumption by TRIAS increased from 9% to 19%, an increase of 111%. TRIAS BDD went from 6% to 13%, which corresponds to a variation of 117%.

From the results obtained, it can be seen that the use of TRIAS BDD from higher to lower density, for 1,500 objects, 20 attributes and 5 conditions, generated a reduction in processing time and in RAM usage.

Figure 24 presents the RAM memory behavior for 1,500 objects, 20 attributes and 5 conditions, with ordering the entry of attributes x conditions from lower to higher density.

Through a comparative analysis between Figures 23 and 19, it can be seen that, by ordering the entry of the attributes x conditions from the lowest to the highest, the RAM memory consumption by TRIAS BDD, at the end of processing, was equal at 15%, while TRIAS BDD from lower to higher density consumed 14%, a reduction of 7%.

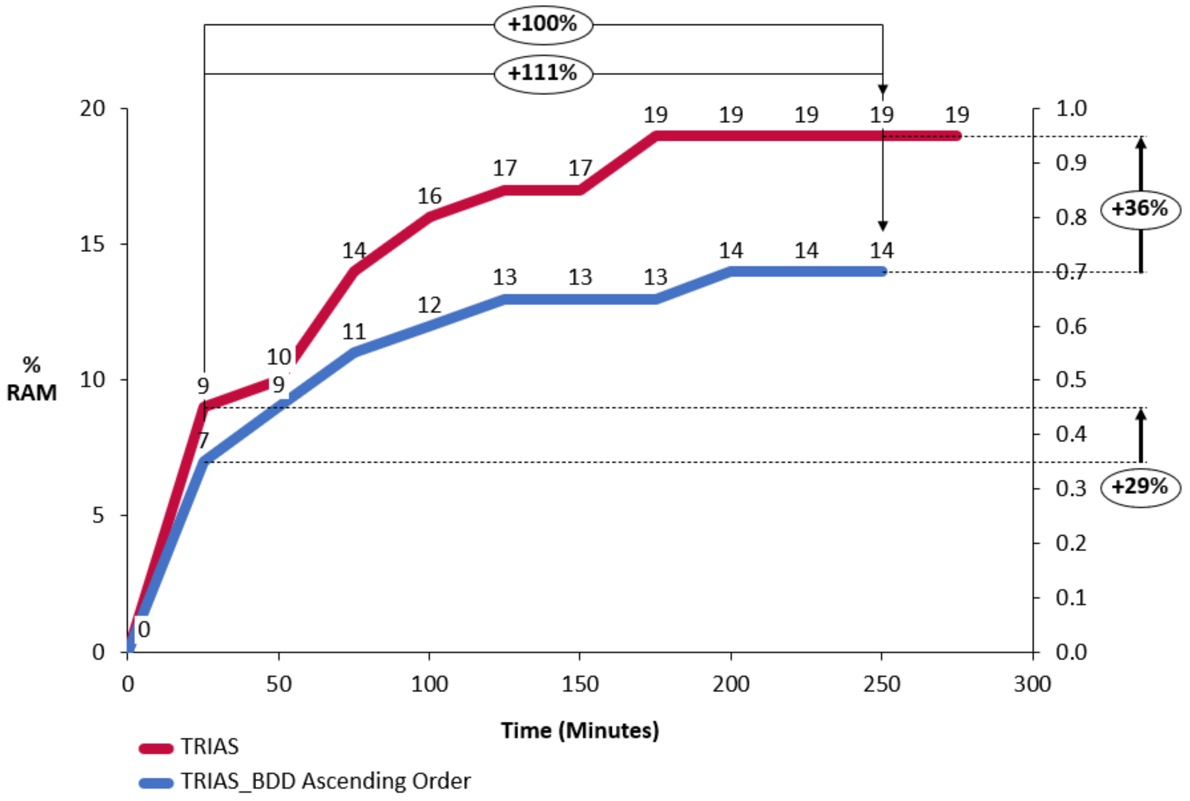


Figure 24: Ascending Order - Memory Usage of the context with 1,500 objects, 20 attributes and 5 conditions - presented in Table 19

Analyzing the first 25 minutes of processing, TRIAS BDD had consumed 7% of RAM and TRIAS had consumed 9%. In other words, TRIAS used 29% more of RAM.

Between 25 and 125 minutes of processing, both presented an increase in the RAM consumption. TRIAS BDD went from 7% to 13%, an increase of 86% and TRIAS went from 9% to 17%, which corresponds to 89%.

From 125 minutes of processing to 254.79 minutes, TRIAS BDD went from 13% to 14%, an increase that corresponds to 8%. TRIAS went from 17% to 19%, that is, an increase of 12%.

Observing 24, a comparative analysis was performed between the memory consumption behavior by TRIAS BDD ordered from lowest to highest density and TRIAS in the interval between 25 minutes and 250 minutes of processing. At the end of the processing, TRIAS BDD had consumed 14% of RAM memory and TRIAS 19%. In other words, TRIAS BDD consumed 36% less RAM memory than TRIAS.

In the same interval, memory consumption by TRIAS increased from 9% to 19%, an increase of 111%. TRIAS BDD went from 7% to 14%, which corresponds to a 100% variation.

From the results obtained, it can be seen that the usage of TRIAS BDD from lower to higher density, for 1,500 objects, 20 attributes and 5 conditions, generated a reduction in processing time and in RAM usage.

5.4.2 TRIAS BDD × TRIAS BDD Descending and Ascending Order Analysis

This section analyzes the two contexts, where the processing time of TRIAS BDD was greater than the processing time of TRIAS, that is, 3,000 objects, 10 attributes and 5 conditions and 1,500 objects, 20 attributes and 5 conditions. This analysis aims to understand the behavior of the TRIAS BDD algorithm, after ordering the density obtained, from highest to lowest (Descending order) and lowest to highest (Ascending order).

The first context evaluated comprises the processing of 3,000 objects, 10 attributes and 5 conditions, where the processing time spent by TRIAS was 62.22 minutes and the time spent by TRIAS BDD was 65.33, 5% more.

In order to have a better understanding of the behavior of the TRIAS BDD algorithm, an analysis of the RAM memory behavior was performed, comparing TRIAS BDD and TRIAS BDD in decreasing and increasing order, according to their density, as shown in Figures 25 and 26.

Figure 25 shows the memory behavior of the TRIAS BDD and TRIAS BDD algorithms in Decreasing order. It is observed that, in the first five minutes of processing, the memory consumption by the algorithms was the same, 11%.

Between 5 minutes and 53.21 minutes of processing, both increase memory consumption. In this interval, which ends with the processing by the TRIAS BDD Descending order, the TRIAS BDD went from 11% of memory consumption to 25%, an increase of 127%.

The TRIAS BDD Descending order consumed 118 % in the same range, going from 11% to 24%. It can be said that the TRIAS BDD Descending order consumed 9% less RAM than the TRIAS BDD in the interval. TRIAS BDD completed the processing at 65.33 minutes, reaching 26% of RAM memory usage.

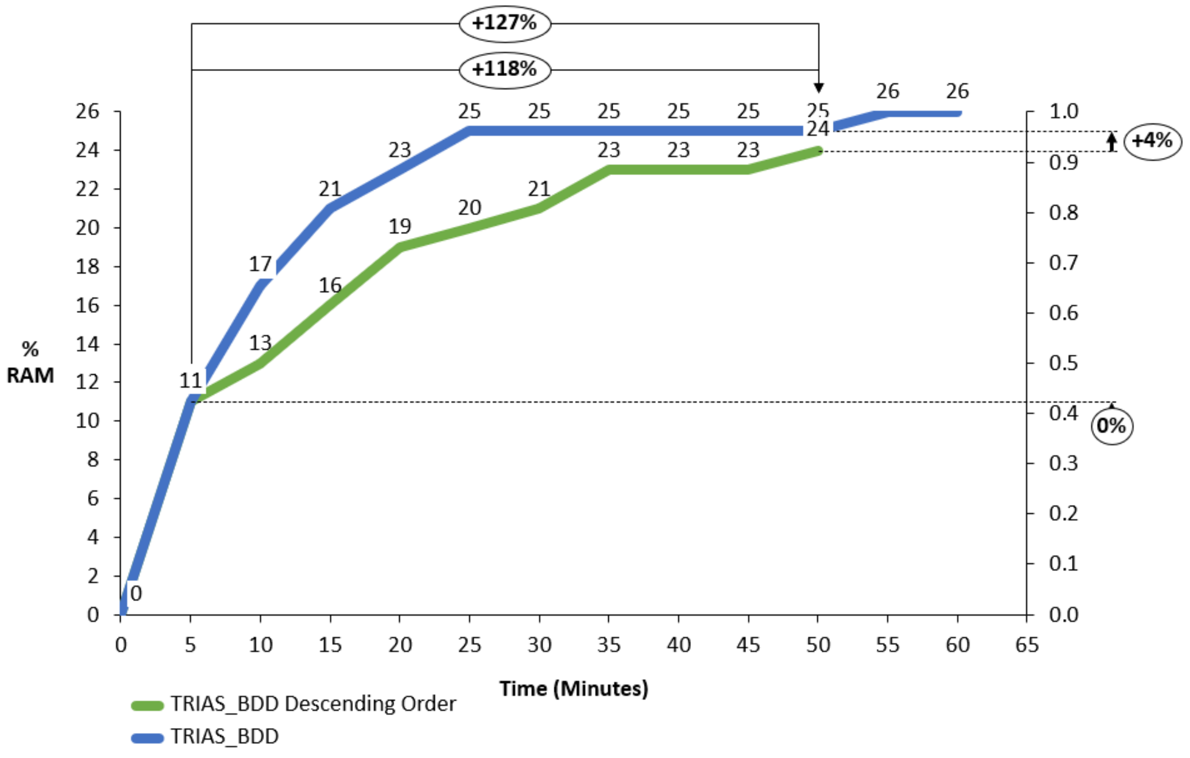


Figure 25: Descending Order - Memory Usage of the context with 3,000 objects, 10 attributes and 5 conditions - presented in Table 19

Figure 26 shows the behavior of the memory of the TRIAS BDD and TRIAS BDD algorithms in Ascending order. In the first five minutes of processing, there is a variation of 9% between the consumption of RAM by TRIAS BDD, 11%, and TRIAS BDD Ascending order, 12%.

Between 5 minutes and 56.92 minutes of processing both increase memory consumption. In this interval, which ends with the processing by the TRIAS BDD Ascending order, the TRIAS BDD went from 11% of memory consumption to 26%, an increase of 136%.

The TRIAS BDD Ascending order consumed 108% in this same interval, going

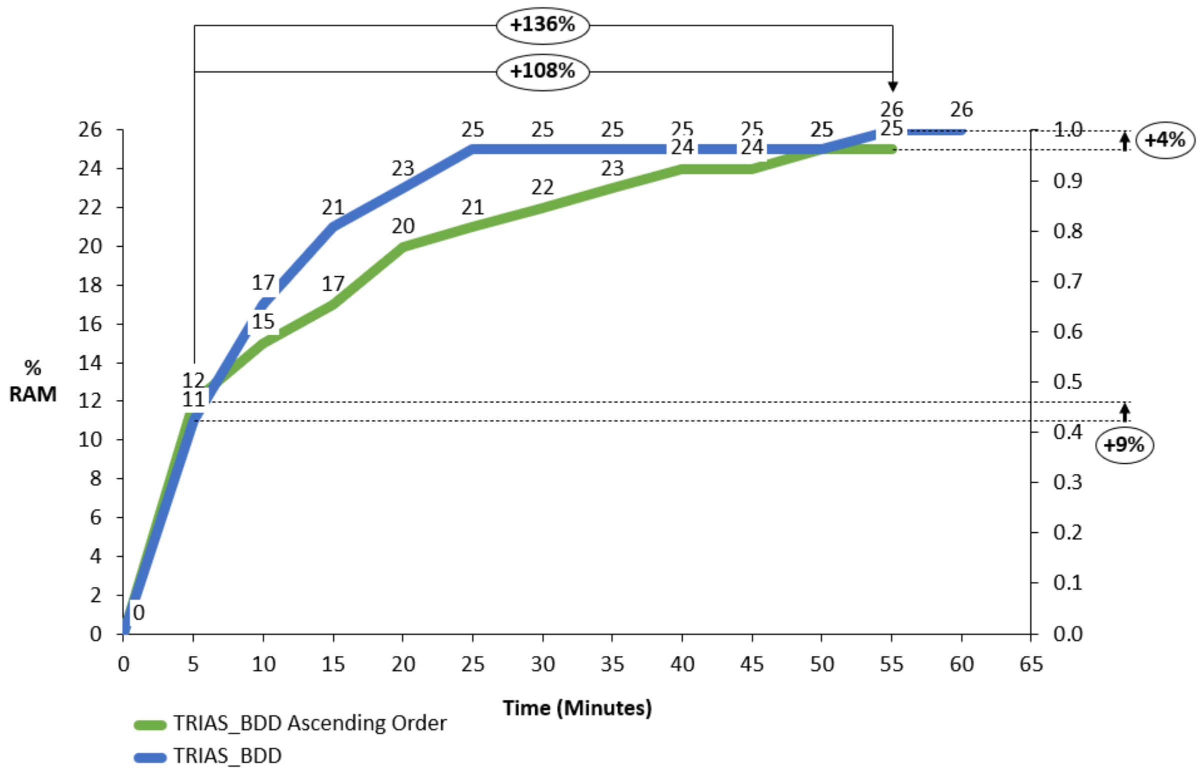


Figure 26: Ascending Order - Memory Usage of the context with 3,000 objects, 10 attributes and 5 conditions - presented in Table 19

from 12% to 25%. It can be said that the TRIAS BDD Ascending order consumed 28% less RAM than the TRIAS BDD in the interval. TRIAS BDD finished processing at 65.33 minutes, reaching 26% of RAM memory usage.

The second context evaluated comprises the processing of 1,500 objects, 20 attributes and 5 conditions, where the processing time spent by TRIAS was 271 minutes and the time spent by TRIAS BDD was 281.84, 4% more.

In order to have a better understanding of the behavior of the TRIAS BDD algorithm, an analysis of the RAM memory behavior was performed, comparing TRIAS BDD and TRIAS BDD in Decreasing and Increasing order, according to their density, as shown in Figures 27 and 28.

Figure 27 shows the memory behavior of the TRIAS BDD and TRIAS BDD algorithms in decreasing order. It is observed that, in the first twenty five minutes of processing, there is a variation of 17% in the memory consumption by the algorithms. TRIAS BDD consumed 7% and TRIAS descending order 6%.

In the interval from 25 minutes and 125 minutes, both increase memory consumption. TRIAS BDD went from 7% to 15%, an increase of 114% and TRIAS BDD descending went from 6% to 12%, growth of 100%.

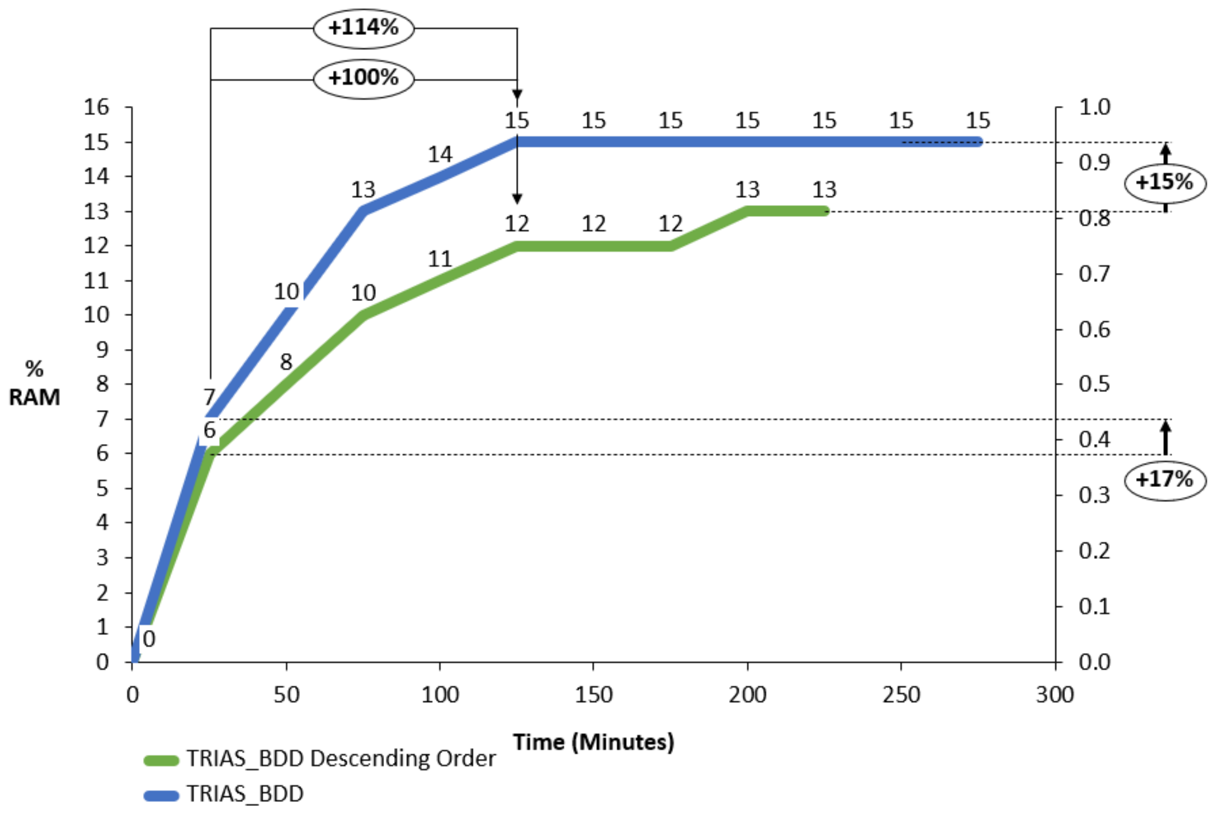


Figure 27: Descending Order - Memory Usage of the context with 1,500 objects, 20 attributes and 5 conditions - presented in Table 19

From 125 minutes to 246.21, when TRIAS BDD descending order finishes processing, TRIAS BDD descending goes from 12% to 13%, varying 8% and TRIAS BDD remains at 15% of memory consumption RAM up to 281.84 minutes, when processing is complete.

Figure 28 shows the behavior of the memory of the TRIAS BDD and TRIAS BDD algorithms in increasing order. It is observed that, in the first twenty five minutes of processing, there is no variation between the processing time by the algorithms. Both consumed 7% of RAM.

In the interval from 25 minutes and 125 minutes, both increase memory consumption. TRIAS BDD went from 7% to 15%, an increase of 114% and TRIAS BDD ascending went from 7% to 13%, an increase of 86%.

From 125 minutes to 254.79, when the TRIAS BDD ascending order finishes processing, the TRIAS BDD ascending goes from 13% to 14%, varying 8% and the TRIAS BDD remains at 15% of memory consumption RAM up to 281.84 minutes, when processing is complete.

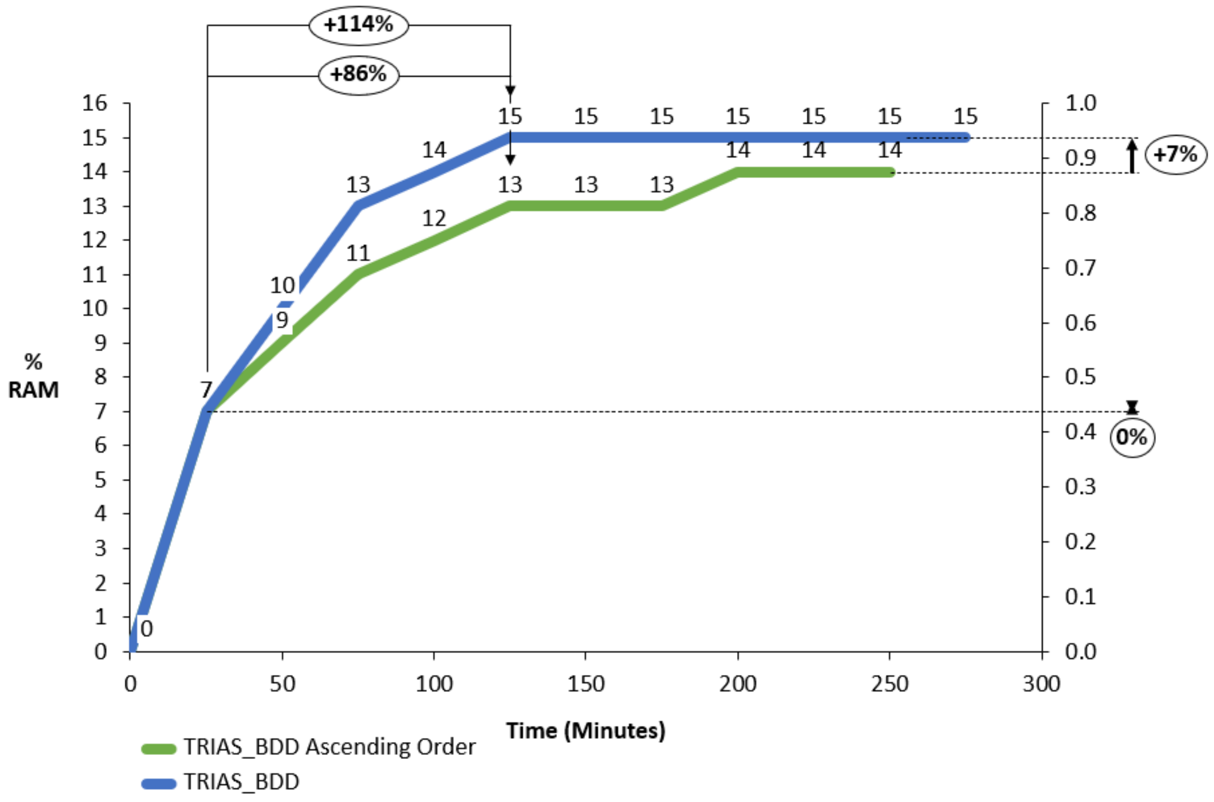


Figure 28: Ascending Order - Memory Usage of the context with 1,500 objects, 20 attributes and 5 conditions - presented in Table 19

5.5 BCAAR and BACAR triadic rules of the real context *MovieLens*

As described in the previous sections, synthetic and real contexts were used in this research. Rules are comprehensible information that contains data from a specific dataset and using synthetic contexts to perform some analysis should not be an interesting approach. Therefore, in this section, only results from the *MovieLens* context described in Section 5.2 will be presented. Also, the dataset used contains 8,545 objects (users), 50 movies (attributes) and 5 ratings (condition).

Table 27 shows the BCAAR rules for movies {Star Wars, Toy Story, I.Q., The Net}. To extract all rules, they were passed as parameters those rules that have support and confidence greater than 15%.

Given the first rule (Star Wars $\rightarrow$ Toy Story) 5 presented in Table 27 with 31.11% support and 72.28% confidence. In this example, it can be said that in 31.11% (support) of cases (2,659 objects of 8,545 total in the context) where it has the attribute *Star Wars* under the condition 5, it also has the attribute *Toy Story* under the same condition (rate 5). It occurs 72.28% (confidence) in the sub dataset where, at least, the movie *Star Wars* was rated with 5 and from the sub dataset where the movie *Toy Story* was also rated with

5 (1,922 of 2,659 objects).

Table 27: Biedermann Conditional Attribute Association Rule (BCAAR) of the MovieLens sub dataset

Rule	Support	Confidence
(Star Wars → Toy Story) 5	31.11%	72.28%
(Star Wars → The Net) 3	26.74%	53.65%
(I.Q. → The Net) 5	18.91%	44.33%
(I.Q. → Toy Story) 3	15.43%	34.22%
(Toy Story → Star Wars) 1	22.33%	34.22%
(Star Wars → The Net) 2	29.47%	49.55%

Table 28 shows the BACAR rules for movies {Free Willy 2, Batman Forever, Doom Generation, Mad Love, Toy Story, Star Wars, Crimson Tid, Brothers McMullen, Quiz Show}. To extract all rules, they were passed as parameters those rules that have support greater than 0% and confidence greater than 50%.

Given the first rule (1 → 3) Free Willy 2, Batman Forever presented in Table 28 with 0.00% support and 100.00% confidence. In this example, it can be said that in 00.00% (support) of cases (0 objects of 6,985 total) where it has the condition 1 under the attribute Free Willy 2 and *Batman Forever*, it also has the condition 3 under the same attributes, and this occurs 100.00% (confidence) in the dataset.

Table 28: Biedermann Attributional Condition Association Rule (BACAR) of the MovieLens sub dataset

Rule	Support	Confidence
(1 → 3) Free Willy 2, Batman Forever	0.0%	100.00%
(5 → 4) Doom Generation, Mad Love	0.0%	100.00%
(5 → 4) Toy Story	0.0%	100.00%
(3 → 4) Star Wars	0.0%	100.00%
(5 → 2) Crimson Tid	0.0%	100.00%
(4 → 3) Brothers McMullen	0.0%	100.00%
(5 → 4) Quiz Show	0.0%	100.00%

5.6 BCAAR and BACAR triadic rules of the real context for network access control

In this section, only results from the Role-Based Access Control (RBAC) will be presented. RBAC is an access control method that manages permissions to end-users based on their role within your organization. It provides fine-grained control, offering a simple, manageable approach to access management that is less error-prone than individually assigning permissions. The dataset used contains 6,985 objects (roles), 3 types of permissions (attributes) and 50 document types (condition).

Table 29: Biedermann Conditional Attribute Association Rule (BCAAR) of the RBAC dataset

Rule	Support	Confidence
(Read→ Write) installation guide	21.31%	66.42%
(Read→ Write) template	45.54%	69.35%
(Write→ Approve) template	15.51%	39.57%
(Write → Read) design	18.22%	25.55%
(Approval→ Read) budget	15.22%	19.22%
(Approval → Read) design document	19.77%	39.52%

Table 29 shows some of the BCAAR rules for permissions {Read, Write, Approve}. To extract all rules, they were passed as parameters those rules that have support and confidence greater than 15%.

Given the first rule (Write → Approve)installation guide presented in Table 29 with 21.31% support and 66.42% confidence. In this example, it can be said that in 21.31% (support) of cases (1,489 objects of 6,985 total in the context) where it has the attribute *Write* under the condition installation guide, it also has the attribute *Approve* under the same condition. It occurs 66.42% (confidence) in the sub dataset where, at least, the permission *Write* was defined as installation guide and from the sub dataset where the permission *Approve* was also classified with installation guide (989 of 1,489 objects). In other words, in 989 objects denominated *Write* was classified as installation guide of 1,489 total where the permission *Write* was also installation guide.

Table 30: Biedermann Attributional Condition Association Rule (BACAR) of the RBAC dataset

Rule	Support	Confidence
(marketing document → user manual, external technical interface document)Read	25.52%	76.22%
(marketing document → term of use document, rating entry)Read	19.37%	65.14%
(rating entry → marketing document, term of use document)Read	28.41%	56.74%
(installation guide → design document)Read	47.16%	57.56%
(term of use document → installation guide)Read	49.32%	66.29%

Table 30 shows some of the BACAR rules extracted from the RBAC context. To extract the rules, they were passed as parameters those rules that have support greater and confidence greater than 15%.

Given the first rule (marketing document → user manual, external technical interface document)Read presented in Table 30 with 25.52% support and 76.25%

confidence. In this example, it can be said that in 25.52% (support) of cases (1,783 objects of 6,985 total) where it has the condition marketing document under the attribute Read, it also has the condition user manual, external technical interface document under the same attribute, and this occurs 76.22% (confidence) in the dataset (1,359 objects of 1,783 total).

6 CONCLUSIONS AND FUTURE WORK

Due to the third dimension, the process of extracting concepts from a triadic context is much more complex than in the classic approach of FCA (dyadic). The representation of the data in three dimensions leads to the high-dimensional of the databases. Considering that and due to the growth of contexts, techniques like the TRIAS algorithm might not be the best approach to extract information.

The usage of BDD has been shown to be a better option for many contexts. The BDD enables the execution of the algorithm in contexts that would be difficult to analyze. As presented previously, the proposed algorithm was faster in all contexts used in the experiments. In some cases, the algorithm can be up to 56% times faster than the original - see Table 17.

In synthetic contexts with 120,000 objects, which is close to the challenge defined in (OLD; PRISS, 2006), the TRIAS BDD algorithm was able to extract concepts before 14 days - see Table 20. However, the original TRIAS algorithm did not return any results within 14 days.

In the real dataset, presented in Table 21, the TRIAS, within 7 days, was not able to extract concepts in contexts that contain more than 100,000 incidences. However, using TRIAS BDD the algorithm retrieved triadic concepts before 7 days in two scenarios. In other words, it was able to find concepts that the original algorithm was not.

This research presents the behavior of the TRIAS and TRIAS BDD algorithms in different contexts. Among 45 triadic synthetic contexts used, between 500 and 10,000 objects, only two TRIAS BDD has a longer processing time than the original algorithm (TRIAS). They are: context with 3,000 objects, 10 attributes and 5 conditions and context with 1,500 objects, 20 attributes and 5 conditions. In the context of synthetic studies, results were also obtained from algorithms for high-dimensional contexts. In addition, contexts of real databases were studied and in the three contexts obtained, TRIAS BDD presented a better processing time than TRIAS.

A study was also carried out on the use of RAM memory for both algorithms, in addition to the ordering of the variables of the triadic context by attributes and conditions to analyze the impact of this ordering on BDD's creation. In the two cases that TRIAS BDD presented the longest processing time, for synthetic triadic contexts. Despite demanding more data processing time, the two TRIAS BDD contexts studied in relation to the behavior of RAM memory showed a lower percentage of memory utilization.

In order to aim the complementation of the studies, the variables ordering of the triadic context was carried out by attributes and conditions, from higher density to lower density, and from lower density to higher density, in the two cases that TRIAS BDD presented a longer processing time than the TRIAS, in synthetic triadic contexts. In all cases, TRIAS BDD reduced the processing time, having a better time than the time spent by TRIAS.

Finally, an analysis was made of the use of RAM memory between the TRIAS BDD, TRIAS BDD ordered from the highest density to the lowest (Descending) and TRIAS BDD ordered from the lowest density to the highest (Ascending), in the two cases that the TRIAS BDD presented a longer processing time than the TRIAS.

In the two contexts evaluated (3,000 objects, 10 attributes and 5 conditions and for 1,500 objects, 20 attributes and 5 conditions), TRIAS BDD required more processing time than the ordered models. Besides, TRIAS BDD also consumed more RAM than the TRIAS BDD descending order and TRIAS BDD ascending order approaches. It is important to note that, in both contexts, the TRIAS BDD descending order consumed less RAM and less time to perform the processing.

As future work, in the Parallel Computing area we intend to reduce the processing time by distributing the workload, paralleling the generation of the BDD and/or the concurrent extraction of the concepts.

Also, using the concepts of the Computer Architecture, we would like to implement the parallelization of the algorithms via threads or GPU (Graphics Processing Unit), besides the possibility to distribute the execution of this in computers within the same network. The main idea is to distribute the algorithm execution on different computers. Another important implementation would be using FPGA (Field Programmable Gate Array) to have a better execution time performance. Finally, on the performance aspect, there are other topics that we should consider, such as: page fault, cache miss and etc.

Also, we intend to include the BDD structure which was used in this work into DATA-PEELER algorithm (CERF et al., 2008) (CERF et al., 2009), recognized as state-of-art to extracts all closed n-sets from a n-ary relation, to evaluate its performance.

We also intend to implement the function to extract the rules based on the triadic concepts directly. Additionally, we suggest that TRIAS BDD would be evaluated in contexts of the real-world problems with a large number of attributes and conditions.

REFERENCES

AGRAWAL, R. et al. Fast discovery of association rules. In: PRESS, A. (Ed.). Advances in knowledge discovery and data mining. [S.l.: s.n.], 1996. p. 307–328.

AGRAWAL, R.; SRIKANT, R. Fast algorithms for mining association rules. In: 20th Very Large Data Bases Conference. [S.l.: s.n.], 1994. p. 487–499.

AKERS, S. B. Binary decision diagrams. IEEE Transactions on Computers, C-27, n. 6, p. 509–516, June 1978. ISSN 0018-9340.

ANANIAS, K. et al. Manipulating triadic concept analysis contexts through binary decision diagrams. Proceedings of the 21st International Conference on Enterprise Information Systems - ICEIS, p. 182–189, 2019.

ANANIAS, K. et al. Análise formal de conceitos triádicos através da utilização de diagramas binários de decisão. Proceedings of the 6th Symposium on Knowledge Discovery, Mining and Learning, p. 20–27, 2018.

ANDREWS, S. In-close2, a high performance formal concept miner. ICCS: Conceptual Structures for Discovering Knowledge, v. 6828, p. 5–062, 2011.

ASHFAQ, R.; HE, Y.; GHEN, D. Toward an efficient fuzziness based instance selection methodology for intrusion detection system. International Journal Machine Learning Cybernet, v. 8, p. 1767–1776, 2017.

ASHFAQ, R. et al. Fuzziness based semi-supervised learning approach for intrusion detection system. Information Science, v. 378, p. 484–497, 2017.

BAIXERIES, J. A formal concept analysis framework to mine functional dependencies. In: Mathematical Methods for Learning. [S.l.: s.n.], 2004.

BASTIDE, Y. et al. Mining minimal non-redundant association rules using frequent closed item sets. In: Computational Logic. [S.l.: s.n.], 2000. p. 972–986.

BASTIDE, Y. et al. Mining frequent patterns with counting inference. SIGKDD Exploration Newsletter, p. 66–75, 2000.

BERNHARD, G.; RUDOLF, W. Formal concept analysis: mathematical foundations. [S.l.]: Springer, 1999.

BERTET, K.; MONJARDET, B. The multiple facets of the canonical direct unit implicational basis. In: THEORETICAL COMPUTER SCIENCE, 411(22–24):2155 – 2166. [S.l.], 1999.

BIEDERMANN, K. How triadic diagrams represent conceptual structures. ICCS, p. 304–317, 1997.

BIEDERMANN, K. A foundation of the theory of trilattices. ISBN 3-8265-4028-X, p. Aachen, 1998.

BIRKHOFF, G. Lattice theory. American Mathematical Society, 1940.

BRYANT, R. E. Graph-based algorithms for boolean function manipulation. Computers, IEEE Transactions on, IEEE, v. 100, n. 8, p. 677–691, 1986.

BRYANT, R. E. Graph-based algorithms for boolean function manipulation. In: IEEE TRANSACTIONS ON, VOL. 100, NO. 8, PP. 677–691. [S.l.], 1986.

CARPINETO, C.; ROMANO, G. Mining short-rule covers in relational databases. Computational Intelligence, p. 215–234, 2003.

CARPINETO, C.; ROMANO, G. Concept data analysis: Theory and applications. 2004.

CARPINETO, C.; ROMANO, G. Concept data analysis: Theory and applications. John Wiley and Sons, England, 2005.

CERF, L. et al. Data peeler: Contraint-based closed pattern mining in n-ary relations. SDM, p. 37–48, 2008.

CERF, L. et al. Closed patterns meet n-ary relations. ACM Transactions on Knowledge Discovery from Data (TKDD), v. 3, 2009.

CERF, L.; BESSON J.AND ROBARDET, C.; BOULICAUT, J. Closed patterns meet n-ary relations. In: ACM (Ed.). [S.l.: s.n.], 2009. p. 1–36.

DAVENPORT, T. H.; PRUSAK, L. Conhecimento empresarial: como as organizações gerenciam o seu capital intelectual. 1998.

DAVEY, B. A.; PRIESTLEY, H. A. Introduction to lattices and order. 2001.

DIAS, S. et al. Extraction of qualitative behavior rules for industrial processes from reduced concept lattice. Intelligent Data Analysis, p. 643–663, 2020.

DUQUENNE, V.; GUIGUES, J. L. Familles minimales d'implications informatives resultant d'un tableau de données binaires. p. 5–18, 1986.

FAN, F. et al. A visualization method for chinese medicine knowledge discovery based on formal concept analysis. ICIC Express Letters, v. 4, p. 801–808, 2013.

GANTER, B. Formal concept analysis: algorithmic aspects. Lecture notes, 2002.

GANTER, B. Two basic algorithms in concept analysis. Springer, 2010.

GANTER, B.; OBIEDKOV, S. Implications in triadic formal contexts. ICCS, p. 186–195, 2004.

GANTER, B.; STUMME, G.; WILLE, R. Formal concept analysis: foundations and applications. In: VOL. 3626, SPRINGER SCIENCE & BUSINESS MEDIA. [S.l.], 2005.

GANTER, B.; WILLE, R. Formal concept analysis: Mathematical foundations. 1999.

GANTER, B.; WILLE, R. Formal concept analysis: Mathematical foundations. In: SPRINGER-VERLAG, GERMANY. [S.l.], 1999.

GNATYSHAK, D. et al. Gaining insight in social networks with biclustering and triclustering. BIR 2012: Perspectives in Business Informatics Research, p. 162–171, 2012.

HAMROUNI, T.; VALTCHEV, P.; YAHIA, S. About the lossless reduction of the minimal generator family of a context. ICFCA, p. 1305–150, 2007.

HE, Y. et al. Owa operator based link prediction ensemble for social network. Expert System Application, p. 21–50, 2015.

HE, Y.; WANG, X.; HUANG, J. Fuzzy nonlinear regression analysis using a random weight network. Information Science, p. 364–365, 2016.

IGNATOV, D. et al. Triadic formal concept analysis and triclustering: searching for optimal patterns. Machine Learning - Springer, p. 271–302, 2015.

IGNATOV, D. et al. From triconcepts to triclusters. RSFDGrC: International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, p. 257–264, 2011.

JASCHKE, R. et al. Trias–an algorithm for mining iceberg tri-lattices. In: IEEE. Data Mining, 2006. ICDM'06. Sixth International Conference on. [S.l.], 2006. p. 907–911.

JAVABDD. Javabdd algorithm. In: . [s.n.], 2019. Available on: <https://javabdd.sourceforge.net/index.html>.

K., W.; W, Y. W. turing machine represention in temporal concept analysis. In: IN THE PROCEEDINGS OF THE 3RD INTERNATIONAL CONFERENCE ON FORMAL CONCEPT ANALYSIS (CFCA) OTAWA, P. 263-314. [S.l.], 2005.

KAVTOUE, M. et al. Minging biclusters of similar values with triadic concept analysis. CLA, p. 175–190, 2011.

KIS, C. S. L.; TROANCA, D. Towards a navigation paradigm for triadic concepts. FCA4AI@ ECAI, p. 42–50, 2016.

KODAGODA, N.; ANDREWS, S.; PULASINGHE, K. A parallel version of the in-close algorithm. NCTM: Proceedings of the 6th National Conference on Technology and Management, p. 1–5, 2017.

KOESTER, B. Conceptual knowledge retrieval with fooca: Improving web search engine results with contexts and concept hierarchies. Industrial Conference on Data Mining, p. 176–190, 2006.

KRYSZKIEWICZ, M.; GAJEK, M. Concise representation of frequent patterns based on generalized disjunction-free generators. Proceedings of the 6th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining, p. 159–171, 2002.

KUMAR, C. A. et al. Role based access control design using triadic concept analysis. JOURNAL OF CENTRAL SOUTH UNIVERSITY, p. 3183–3191, 2016.

LEHMANN, F.; WILLE, R. A triadic approach to formal concept analysis. ICCS, p. 32–43, 1995.

LEHMANN, F.; WILLE, R. A triadic approach to formal concept analysis. CONCEPTUAL STRUCTURES: APPLICATIONS, IMPLEMENTATION AND THEORY, Springer, p. 32–43, 1995.

LI, J. et al. Comparison of reduction in formal decision contexts. INTERNATIONAL JOURNAL OF APPROXIMATE REASONING, p. 100–122, 2017.

LI, S.; TSAI, F. A fuzzy conceptualization model for text mining with application in opinion polarity classification. KNOWLEDGE-BASED SYSTEMS, v. 39, p. 23–33, 2013.

LUXENBURGER, M. Implication partielles dans un contexte. ISBN 3-8265-4028-X, p. 35–55, 1991.

MAIO, C. D. et al. Formal and relational concept analysis for fuzzy-based automatic semantic annotation. APPLIED INTELLIGENCE, v. 40, p. 153–174, 2014.

MARQUEZ, F. P. G. et al. Optimal dynamic analysis of electrical/electronic components in wind turbines. JOURNAL OF ENERGIES, p. 1111–1129, 2017.

MARQUEZ, F. P. G. et al. Reliability dynamic analysis by fault trees and binary decision diagrams. JOURNAL OF INFORMATION, p. 324–339, 2020.

MISSAOUI, R.; KWUIDA, L. Mining triadic association rules from ternary relations. INTERNATIONAL CONFERENCE ON FORMAL CONCEPT, Springer, p. 204–218, 2011.

MO, Y. et al. Choosing a heuristic and root node for edge ordering in bdd-based network reliability analysis. RELIABILITY ENGINEERING AND SYSTEM SAFETY, v. 131, p. 83 – 93, 2014. ISSN 0951-8320. Available on: <http://www.sciencedirect.com/science/article/pii/S0951832014001550>.

MORAES, N. R. de et al. Parallelization of the next closure algorithm for generating the minimum set of implication rules. ARTIFICIAL INTELLIGENCE RESEARCH, v. 5, n. 2, p. 40, 2016.

NETO, S. M.; ZÁRATE, L. E.; SONG, M. A. Handling high dimensionality contexts in formal concept analysis via binary decision diagrams. INFORMATION SCIENCES, Elsevier, v. 429, p. 361–376, 2018.

NEVES, J. et al. Applying binary decision diagram to extract concepts from triadic formal context. PROCEEDINGS OF THE 35TH ANNUAL ACM SYMPOSIUM ON APPLIED COMPUTING, p. 466–469, 2020.

NEVES, J. et al. Exploring different paradigms to extract proper implications from high dimensional formal contexts. IEEE ACCESS, v. 8, p. 134161 – 134175, 2020.

NICOLAS, P. Data mining: algorithms for extracting and reducing rules of association in databases. PHD THESIS, UNIVERSITÉ BLAISE PASCAL CLERMONT-FERRAND, 2000.

NOVAIS, J. P. P. et al. An open computing language-based parallel brute force algorithm for formal concept analysis on heterogeneous architectures. CONCURRENCY AND COMPUTATION - PRACTICE AND EXPERIENCE, p. e6220, 2021.

OLD, J.; PRISS, U. Some open problems in formal concept analysis. problems presented at international conference on formal concept analysis (icfca). In: . [S.l.: s.n.], 2006.

PASQUIER, N. et al. Discovering frequent closed item sets for association rules. In: SPRINGER-VERLAG (Ed.). ICDT '99: PROCEEDING OF THE 7TH INTERNATIONAL CONFERENCE ON DATABASE THEORY. [S.l.: s.n.], 1999. p. 398–416.

PEI, J. et al. Closet: An efficient algorithm for mining frequent closed itemsets. In: ACM SIGMOD WORKSHOP ON RESEARCH ISSUES IN DATA MINING AND KNOWLEDGE DISCOVERY. [S.l.: s.n.], 2000. v. 4, n. 2, p. 21–30.

QI, J.; WEI, L.; Y, Y. Three-way formal concept analysis. INTERNATIONAL CONFERENCE ON ROUGH SETS AND KNOWLEDGE TECHNOLOGY, SPRINGER INTERNATIONAL PUBLISHING, p. 732–741, 2014.

R., N.; L., Z. Handling large formal contexts with support of distributed systems. In: IN PROCEEDINGS OF IADIS INTERNATIONAL CONFERENCE, V. 22, P. 1-15, 2011. [S.l.], 2011.

RIMSA, A.; SONG, M. A.; ZÁRATE, L. E. Scgaz-a synthetic formal context generator with density control for test and evaluation of fca algorithms. In: IEEE. SYSTEMS, MAN, AND CYBERNETICS (SMC), 2013 IEEE INTERNATIONAL CONFERENCE ON. [S.l.], 2013. p. 3464–3470.

RIMSA, A.; ZÁRATE, L.; SONG, M. A. Evaluation of different bdd libraries to extract concepts in fca–perspectives and limitations. In: COMPUTATIONAL SCIENCE–ICCS 2009, PP. 367–376. [S.l.], 2009.

RUDOLPH, S.; SACAREA, C.; TROANCA, D. Fca tools bundle - a tool that enables dyadic and triadic conceptual navigation. INTERNATIONAL CONFERENCE ON FORMAL CONCEPT ANALYSIS, p. 252–267, 2015.

SALLEB, A.; MAAZOUZI, Z.; VRAIN, C. Mining maximal frequent itemsets by a boolean based approach. In: EUROPEAN CONF. ON ARTIFICIAL INTELLIGENCE, LYON FRANCE (JULY 2002). [S.l.: s.n.], 2002. p. 285–289.

SANTOS, P. et al. An approach to extract proper implications set from high-dimension formal contexts using binary decision diagram. PROCEEDINGS OF THE 20TH INTERNATIONAL CONFERENCE ON ENTERPRISE INFORMATION SYSTEMS, v. 1, p. 50–57, 2018.

SELMANE, S. et al. Mining triadic association rules. COMPUTER SCIENCE & INFORMATION TECHNOLOGY, v. 3, p. 305–316, 2013.

SENATORE, S.; PASI, G. Lattice navigation for collaborative filtering by means of (fuzzy) formal concept analysis. PROCEEDINGS OF THE 28TH ANNUAL ACM SYMPOSIUM ON APPLIED COMPUTING, p. 920–926, 2013.

SHIVHARE, R.; CHERUKURI, A. K. Three-way conceptual approach for cognitive memory functionalities. INTERNATIONAL JOURNAL OF MACHINE LEARNING AND CYBERNETICS, p. 21–34, 2017.

STUMME, G. Formal concept analysis on its way from mathematics to computer science. ICCS - LECTURE NOTES IN COMPUTER SCIENCE, p. 2–19, 2002.

STUMME, G. et al. Computing iceberg concept lattices with titanic. DATA KNOWLEDGE ENGINEERING, p. 189–222, 2002.

STUMME, G.; WILLE, R.; WILLE, U. Conceptual knowledge discovery in databases using formal concept analysis methods. PROCEEDINGS OF THE SECOND EUROPEAN SYMPOSIUM ON PRINCIPLES OF DATA MINING AND KNOWLEDGE DISCOVERY, p. 450–458, 1998.

SUBRAMANIAN, C. M.; CHERUKURI, A. K.; CHELLIAH, C. Role based access control design using three-way formal concept analysis. INTERNATIONAL JOURNAL OF MACHINE LEARNING AND CYBERNETICS, p. 1807–1837, 2018.

TANG, P.; HUIA, S.; FONG. A lattice-based approach for chemical structural retrieval. ENGINEERING APPLICATIONS OF ARTIFICIAL INTELLIGENCE, v. 39, p. 215–222, 2015.

TANG, Y.; FAN, M.; LI, J. An information fusion technology for triadic decision contexts. INTERNATIONAL JOURNAL OF MACHINE LEARNING AND CYBERNET, p. 13–24, 2016.

TAOUIL, R.; BASTIDE, Y. Computing proper implications. ICCS-2001 INTERNATIONAL WORKSHOP ON CONCEPT LATTICE-BASED THEORY, METHODS AND TOOLS FOR KNOWLEDGE DISCOVERY IN DATABASES, p. 49–61, 2001.

TRABELSI, C.; JELASSI, N.; YAHIA, S. Scalable mining of frequent tri-concepts from folksonomies. THE PACIFIC-ASIA CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING, p. 231–244, 2012.

TRABELSI, C.; JELASSI, N.; YAHIA, S. B. Scalable mining of frequent tri-concepts from folksonomies. In: SPRINGER. PACIFIC-ASIA CONFERENCE ON KNOWLEDGE DISCOVERY AND DATA MINING. [S.l.], 2012. p. 231–242.

VALTCHEV, P. et al. Generating frequent itemsets incrementally: two novel approaches based on galois lattice theory. JOURNAL OF EXPERIMENTAL AND THEORETICAL ARTIFICIAL INTELLIGENCE, p. 115–142, 2002.

WANG, X. Learning from big data with uncertainty. JOURNAL INTEL FUZZY SYSTEM, p. 2329–2330, 2015.

WANG, X.; ASHFAQ, R.; FU, A. Fuzziness based sample categorization for classifier performance improvement. JOURNAL INTEL FUZZY SYSTEM, p. 1185–1196, 2015.

WILLE, R. Restructuring lattice thoery: An approach based on hierarchies of concepts. p. 445–470, 1982.

WILLE, R. Concept lattices and conceptual knowledge systems. COMPUTERS AND MATHEMATICS WITH APPLICATIONS, p. 493–515, 1992.

WILLE, R. The basic theorem of triadic concept analysis. ORDER, Springer, v. 12, n. 2, p. 149–158, 1995.

WILLE, R. Restructuring mathematical logic: an approach based on peirce's pragmatism. LECTURE NOTES IN PURE AND APPLIED MATHEMATICS, p. 267–282, 1996.

WILLE, R. Why can concept lattices support knowledge discovery in databases? PROCEEDINGS OF THE CONCEPT LATTICES BASED KNOWLEDGE DISCOVERY IN DATABASES WORKSHOP, p. 7–20, 2001.

WU, W. Attribute reduction based on evidence theory in incomplete decision systems. INFORMATION SCIENCE, p. 1355–1371, 2008.

WU, W.; LEUNG, Y.; MI, J. Granular computing and knowledge reduction in formal contexts. IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, p. 1461–1474, 2009.

YEVTUSHENKO, S. Bdd-based algorithms for the construction of the set of all concepts. foundations and applications of conceptual structures. In: FOUNDATIONS AND APPLICATIONS OF CONCEPTUAL STRUCTURES. CONTRIBUTIONS TO ICCS. [S.l.: s.n.], 2002. v. 2002, p. 61–73.

ZERARGA, L.; DJOUADI, Y. Interval-valued fuzzy extension of formal concept analysis for information retrieval. NEURAL INFORMATION PROCESSING, v. 7663, p. 608–615, 2013.