Tampere University

Alexander Bakhtin

PARACONSISTENT MANY-VALUED LOGIC IN GUHA DATA MINING FRAMEWORK

Literature review

Bachelor Thesis
Faculty of Engineering and Natural Sciences
Examiner: Prof. Esko Turunen
March 2021

ABSTRACT

Alexander Bakhtin: Paraconsistent many-valued logic in GUHA Data mining framework
Bachelor Thesis
Tampere University
International Bachelor Programme in Science and Engineering
March 2021

While Classical Logic is a cornerstone of Mathematics, modern Machine Learing and Data Mining systems need another approache to help deal with situations where contradictory and inconsistent information is unavoidable. We give an outline of MV-algebras, and in particular, Lukasiewicz L-structure, which make it possible to create Fuzzy (Many-valued) and Paraconsistent logic settings. We also find an applied case of using Paraconsistent logic with evidence couples in another non-classical logic setting, the GUHA data mining method, to produces novel hypotheses.

Keywords: paraconsistent logic, many-valued logic, GUHA, LISpMiner, Data Mining, KDD

The originality of this thesis has been checked using the Turnitin OriginalityCheck service.

PREFACE

The idea of this thesis came while I was taking the course "Introduction to Data Mining: The B-Course and the GUHA Method" by prof. Esko Turunen. That course taught basics of theory behind GUHA Method for KDD and usage of LISpMiner software.

After contacting the professor with plea to supervise my bachelor thesis, I recieved an accepting reply and proposition of the topic of Paraconsitent Logic in GUHA Framework, literature review of which is presented in this work.

Tampere, 15th March 2021

Alexander Bakhtin

CONTENTS

LIST OF SYMBOLS AND ABBREVIATIONS

$\mathcal{FOUR}$	Belnap's 4-valued Logic FOUR
GUHA	General Unary Hypothesis Automaton
KDD	Knowledge Discovery in Databases
MV-algebra	Multi-Valued Algebra

1 INTRODUCTION

As any paper with keywords "Paraconsistent Logic" or similar in its title, this thesis shall begin with the following observation:

In classical logic, there holds the rule "Ex contradictione [sequitur] quodlibet".

Also know as Principle of Explosion, it states that if a contradiction is assumed to be `True`, then any statements (and thus other contradictions) will follow to be `True`.

While classical logic is well-established and is the cornerstone of Mathematics, modern world, with its focus on Big Data and Machine Learning, needs another approach. It needs to deal with potential contradictions, or, in other words, situations where both a statement and its negation seem to be probable, and where decisions need to be made bearing this in mind.

Also classical logic gives us only two ways to judge a statement about some set of objects - either 'All objects satisfy A' or 'At least one object satisfies A'. While such quantifiers are essential in Proof Theory, for real world they are pretty useless - due to noise in any big data set, the first kind of statement is doomed to be `False`, the second kind will almost always be `True` and seems trivial.

For these reasons other logic systems are invented, trying to solve one or both of these drawbacks of classical logic. Solving the problem of Ex contradictione quodlibet led to the development of Paraconsistent Logic(s) - approaches where contradictions are treated formally and assuming contradicting statements to be `True` does not lead to the explosion into triviality.

Many-valued Logics are an important tool in achieving this - if the truth value of a statement is allowed to be continuous instead of binary, we can interpret it as a degree of belief in that statement. The foundations for Many-valued logics discussed in this work are MV-algebras - special kind of algebras developed by Chang [1] to use as a structure for Łukasiewicz infinite-valued logic. In particular we follow the approach of Turunen ([2], [3]), and how he relates this mathematical machinery to Paraconsistency and Belnap's logic $\mathcal{FOUR}$, inspiration from which allows to use MV-algebra to create a logic system with evidence matrices containing values *true*, *false*, *contradictory*, *unknown* ([4]).

To solve the problem of usefulness of quantifiers, a logic system GUHA (General Unary

Hypothesis Automaton) was proposed by Petr Hajek and Tomas Havranek in 1966 [5]. It deals with this problem by introducing four-fold table for each pair of unary predicates (statements about a certain object) and using values from the table to define Truth conditions for quantifiers. Such approach allows to define many families of quantifiers by using those counts from the tables as well as introducing some threshold parameters. The purpose of this work is to explore with literature review how paraconsistent ideas can be expressed in this GUHA setting.

One such solution is identified [6]. Indeed, a recent systematic analysis done by Zamansky [7] shows that most applications of Paraconsistent logic come from the system called 'annotated logics', which is different from the one studied here. We study the development of Belnap's Logic $\mathcal{FOUR}$, and even though his work is mentioned in [7], no application of this logic was discovered there.

In the following chapter we present the mathematical foundations of Many-valued Logic with MV-algebras (section 2.2), and the Paraconsistent setting developed by Turunen; chapter 3 explains the GUHA framework and showcases its possible applications. Finally, we show how developments of Turunen can be applied in the GUHA framework.

2 MANY-VALUED AND PARACONSISTENT LOGIC

We begin by reviewing the history of Many-valued Logic and considering the mathematical foundations behind many Many-valued and Paraconsistent logics, which also serve as the foundation for the paraconsistent logic discussed further in this work.

2.1 History of Many-valued Logic

One of the people considered to be the founders of Many-valued Logic is Jan Łukasiewicz (1878-1956). Polish mathematician, logician and philosopher, he is responsible for introducing mathematical logic in Poland and establishing the so-called *Warsaw school of Logic* [8]. He work is closely related to works of Tarski, Gödel, Wajsberg and was a big influence on them.

The system that we now consider to be the first in Many-valued Logic was published by Jan Łukasiewicz in 1920 ([9], developed since 1917), and it is his Three-valued logic. To the values `True` and `False` of the classical logic he added a third value, called ’unknown’ or ’possible’ and developed logical connectives and operators that worked with all the three values. This logic is usually denoted as Ł$_3$.

Later, he (together with Tarski) extended his idea and developed an infinite-valued logic [10], denoted as Ł$_{\aleph_0}$, which is the system this work is ultimately based on. That logic allows a statement to have any value on the interval $[0, 1]$.

Several successful attempts were made to find a model for such a logic, one by Wajsberg, Łukasiewicz’s student, and one by Chen Chung Chung [1], Tarski’s student, introducing, respectively, Wajsberg algebras and MV-algebras. The latter approach seems to be more famous, and the works we cover here and thus us are using that one, however, these algebras are shown to be equivalent (for example in [2]), so it ultimately does not matter, as they give the same explanation of the Łukasiewicz logic.

2.2 MV-algebras

We now demonstrate the basic notions of MV-algebras. Comprehensive literature about this topic is available in [2, 4, 11]. We use the notation from [4].

Definition 1. *Given a set L, operations $\oplus$, * and element $\mathbf{0}$, we say that* ***L*** = $\langle L, \oplus, ^*, \mathbf{0}\rangle$ *is an* MV-algebra *if the following axioms hold:*

$$x \oplus y = y \oplus x \tag{2.1}$$

$$x \oplus (y \oplus z) = (x \oplus y) \oplus z \tag{2.2}$$

$$x \oplus \mathbf{0} = x \tag{2.3}$$

$$x^{**} = x \tag{2.4}$$

$$x \oplus \mathbf{0}^* = \mathbf{0}^* \tag{2.5}$$

$$(x^* \oplus y)^* \oplus y = (y^* \oplus x)^* \oplus x \tag{2.6}$$

We also define an operation $x \odot y = (x^* \oplus y^*)^*$, then we also obtain following relations (deduced respectively from 2.1-2.3):

$$x \odot y = y \odot x \tag{2.7}$$

$$x \odot (y \odot z) = (x \odot y) \odot z \tag{2.8}$$

$$x \odot \mathbf{1} = x \tag{2.9}$$

It is easy to notice that both $\oplus$ and $\odot$ obey the same axioms. Another way to declare them would would be to simply state that $\langle L, \oplus, \mathbf{0}\rangle$ and $\langle L, \odot, \mathbf{1}\rangle$ need to be commutative monoids.

Note These were the definitions of MV-algebra taken from [6], a book [2] by the same author gives a more thorough look into the topic, using first Wajsberg algebras, so the path to defining MV-algebras is different.

We notice that conditions $x^* \oplus y = \mathbf{1}$, $x \odot y^* = \mathbf{0}$ are equivalent:

$$\begin{aligned} x^* \oplus y = \mathbf{1} = \mathbf{0}^* &\quad \text{definition of } \mathbf{1} \qquad (2.10)\\ (x^* \oplus y)^* = \mathbf{0}^{**} &\quad \text{negation of both sides} \qquad (2.11)\\ (x^* \oplus y^{**})^* = \mathbf{0} &\quad \text{double negation for } y \text{ and } \mathbf{0} \qquad (2.12)\\ x \odot y^* = \mathbf{0} &\quad \text{definition of } \odot, \qquad (2.13)\end{aligned}$$

and we may define a partial order:

$$x \leq y \text{ iff } x^* \oplus y = \mathbf{1} \text{ iff } x \odot y^* = \mathbf{0} \qquad (2.14)$$

We also define three more operations:

$$x \vee y = (x^* \oplus y)^* \oplus y \qquad (2.15)$$

$$x \wedge y = (x^* \odot y)^* \odot y = (x^* \vee y^*)^* \qquad (2.16)$$

$$x \rightarrow y = x^* \oplus y \qquad (2.17)$$

One can spend a great deal of time now deducing and verifying many relations (tautologies) relating all of the symbols $\oplus, \odot, \leq, \vee, \wedge, \rightarrow$, which would be a good exercise in algebra and abstract thinking, however such results are not essential to this work and thus are omitted. Some of them can be found in one of the key references to this work, [4]. We now instead show some examples and uses of MV-algebras to explain their importance.

If we only consider a set of two numbers $\{0, 1\}$, and define operations $\oplus$ as the classical logic disjunction and * as negation, we would actually get a two-valued Boolean algebra. Thus an idea for a certain generalization of classical Boolean logic into Many-valued logic comes to mind - consider instead of set $\{0, 1\}$ an interval $[0, 1]$ with operations slightly redefined.

What we obtain in this case is the standard MV-algebra, or Łukasiewicz structure L. It was shown that this MV-algebra is the structure of the infinite-valued Łukasiewicz logic [1]. Now the operations are defined as

$$x \oplus y = \min(x + y, 1) \qquad (2.18)$$

$$x^* = 1 - x \qquad (2.19)$$

and the order $\leq$ is the regular one for the real numbers. It is possible to deduce that remaining operations are:

$$x \odot y = \max(0, x + y - 1) \tag{2.20}$$
$$x \vee y = \max(x, y) \tag{2.21}$$
$$x \wedge y = \min(x, y) \tag{2.22}$$
$$x \to y = \min(1, 1 - x + y) \tag{2.23}$$

In this case the operation $\odot$ is know as "Łukasiewicz t-norm" and $\oplus$ as its "t-conorm". We notice that if we consider only the elements 0 and 1 in the standard MV-algebra, the result of all operations corresponds to their Boolean counterparts.

This particular MV-algebra is also important since it was shown by Chang [1, 12] that it generates all the other MV-algebras, and thus a proof of an equation in this algebra proves the same equation for all MV-algebras.

2.3 Evidence couples and matrices

We now begin to demonstrate how the foundations of Many-Valued Logics presented above relate to notions of Paraconsistency, however at first we present important theorems on MV-algebras.

Classical logic has two important tautologies:

$$x \wedge x^* = \mathbf{0} \ \texttt{(False)} \tag{2.24}$$
$$x \vee x^* = \mathbf{1} \ \texttt{(True)}, \tag{2.25}$$

where * is the classical negation. Many-valued Logics have a similar set of relations, namely:

Theorem 1. *In an MV-algebra, for any $x, y \in L$ holds the following:*

$$(x \odot y) \wedge (x^* \odot y^*) = \mathbf{0} \tag{2.26}$$
$$(x^* \wedge y) \oplus (x \odot y) \oplus (x^* \odot y^*) \oplus (x \wedge y^*) = \mathbf{1} \tag{2.27}$$

We prove this in the Łukasiewicz structure L. Start with the first equation. Since $\wedge$ is defined as a $\min$ function, and all operations are bounded to $[0, 1]$, if $x \odot y = 0$, then we are done. If $x \odot y \neq 0$ then necessarily $x \odot y > 0 \Rightarrow x + y - 1 > 0$:

$$
\begin{aligned}
x + y - 1 &> 0 && \text{multiply by } -1 && (2.28)\\
1 - x - y &< 0 && \text{add and substract } 1 && (2.29)\\
1 - x + 1 - y - 1 &< 0 && \text{group terms} && (2.30)\\
(1-x) + (1-y) - 1 &< 0 && \text{definition of }^{*} && (2.31)\\
x^* + y^* - 1 &< 0 && \text{write in } \min / \max \text{ language} && (2.32)\\
\max(0, x^* + y^* - 1) &= 0 && \text{definition of } \odot \text{ in L-structure} && (2.33)\\
x^* \odot y^* &= 0 && && (2.34)
\end{aligned}
$$

And thus $(x \odot y) \wedge (x^* \odot y^*) = \min(x \odot y > 0, x^* \odot y^* = 0) = 0$. This essentially shows that either of the expressions is always 0. Now we turn to the second equation. This is easy to show, indeed assume that $x \wedge y^* = \min(x, 1-y) = x$, thus by switching elements to different sides of the inequality:

$$
\begin{aligned}
x &\leq 1 - y && \text{move all to one side} && (2.35)\\
x + y - 1 &\leq 0 && \text{multiply by } -1 && (2.36)\\
0 &\leq 1 - x - y && \text{add and substract 1, group} && (2.37)\\
0 &\leq (1-x) + (1-y) - 1 && \text{multiply eq. 2.35 by } -1 && (2.38)\\
y &\leq 1 - x && && (2.39)
\end{aligned}
$$

Now we express eq. 2.35-2.39 in $\min / \max$ language and also in the language of MV-algebras (given in the same order):

$$
\begin{aligned}
&\text{eq.2.35:} && \min(x, 1-y) = x && x \wedge y^* = x && (2.40)\\
&\text{eq.2.36:} && \max(0, x+y-1) = 0 && x \odot y = 0 && (2.41)\\
&\text{eq.2.38:} && \max(0, (1-x)+(1-y)-1) = 1-x-y && x^* \odot y^* = 1-x-y && (2.42)\\
&\text{eq.2.39:} && \min(y, 1-x) = y && x^* \wedge y = y && (2.43)
\end{aligned}
$$

Thus $(x^* \wedge y) \oplus (x \odot y) \oplus (x^* \odot y^*) \oplus (x \wedge y^*) = x \oplus 0 \oplus (1-x-y) \oplus y = 1$. Here $\oplus$ is the regular summation capped at 1, so we may first take the sum of several operands, and only then compare to 1.
If we assumed the opposite, that $x \wedge y^* = \min(x, 1-y) = 1-y$, that would simply mean switching the sign $\leq$ to $\geq$ everywhere above, thus selecting the other argument of every min/max function, leading to $(x^* \wedge y) \oplus (x \odot y) \oplus (x^* \odot y^*) \oplus (x \wedge y^*) = (1-y) \oplus (x+y-1) \oplus 0 \oplus (1-x) = 1$, which is the same result, thus the proof is complete.

As mentoioned previously, this proof proves the same equation in a general MV-algebra.

The four expressions used in the second equation of the theorem are quite special, since there is an interesting result about uniqueness of their values:

Theorem 2. *In an MV-algebra, if for $x, y, a, b \in L$ it is known that:*

$$\begin{cases} x^* \wedge y = a^* \wedge b \\ x \odot y = a \odot b \\ x^* \odot y^* = a^* \odot b^* \\ x \wedge y^* = a \wedge b^*, \end{cases}$$

then $x = a$, $y = b$.

Again, we prove this in the L structure. By **Theorem 1**, either $x \odot y = a \odot b = 0$ or $x^* \odot y^* = a^* \odot b^* = 0$.
Assume that the first equation holds, then $\max(0, x + y - 1) = 0$, thus $x + y - 1 \leq 0$. Rearranging, we get $x \leq 1 - y$, $x \leq y^*$, so $x \wedge y^* = \min(x, y^*) = x$. Similarly $a \wedge b^* = a$, so we obtained:

$$x = x \wedge y^* = a \wedge b^* = a. \tag{2.44}$$

Rearranging further, $y \leq 1 - x$, $y \leq x^*$, thus $x^* \wedge y = y$, and similarly for a, b:

$$y = x^* \wedge y = a^* \wedge b = b. \tag{2.45}$$

We got $x = a, y = b$.

Now assume that the second equation holds, then $\max(0, (1 - x) + (1 - y) - 1) = 0$, so $1 - x - y \leq 0$. Rearrange to get $1 - x \leq y$, $x^* \leq y$, then $x^* \wedge y = x^*$ and similarly $a^* \wedge b = a^*$.

$$\begin{aligned} x^* = x^* \wedge y = a^* \wedge b = a^* & \qquad \text{keep the sides} & (2.46) \\ x^* = a^* & \qquad \text{negation on both sides} & (2.47) \\ x^{**} = a^{**} & \qquad \text{rule of double negation} & (2.48) \\ x = a & & (2.49) \end{aligned}$$

And again rearrange $1 - x \leq y$ into $1 - y \leq x$ to get $y^* \leq x$ and so $x \wedge y^* = y^*$, the same for a, b:

$$\begin{aligned} y^* = x \wedge y^* = a \wedge b^* = b^* & \qquad \text{keep the sides} & (2.50) \\ y^* = b^* & \qquad \text{negation on both sides} & (2.51) \\ y^{**} = b^{**} & \qquad \text{rule of double negation} & (2.52) \\ y = b & & (2.53) \end{aligned}$$

Thus in either case we got $x = a, y = b$.

This theorem means that any pair $x, y \in L$ through the operations shown above creates a unique quadruplet of elements of L. What is more, if we denote these for values as f, k, u, t (foreshadowed in the Introduction chapter), we can say that:

Theorem 3. *If for $x, y \in L$ we calculate:*

$$\begin{cases} x^* \wedge y = f \\ x \odot y = k \\ x^* \odot y^* = u \\ x \wedge y^* = t, \end{cases}$$

then the reverse calculation is

$$\begin{cases} x = t \oplus k \\ y = f \oplus k \end{cases}$$

and

$$\begin{cases} x = (f \oplus u)^* \\ y = (t \oplus u)^* \end{cases}$$

This is shown by direct observation of results of **Theorem 2**. In that proof we essentially demonstrated that only two cases are possible:

$$\overset{\textbf{I}}{\begin{cases} x^* \wedge y = f = y \\ x \odot y = k = 0 \\ x^* \odot y^* = u = 1 - x - y \\ x \wedge y^* = t = x, \end{cases}} \quad \text{and} \quad \overset{\textbf{II}}{\begin{cases} x^* \wedge y = f = 1 - x \\ x \odot y = k = x + y - 1 \\ x^* \odot y^* = u = 0 \\ x \wedge y^* = t = 1 - y, \end{cases}} \tag{2.54}$$

Now just observe the suggested formulas:

$$\overset{\textbf{I}}{\begin{cases} x = t \oplus k = x + 0 = x \\ y = f \oplus k = y + 0 = y \\ x = (f \oplus u)^* = 1 - (y + 1 - x - y) = x \\ y = (t \oplus u)^* = 1 - (x + 1 - x - y) = y \end{cases}} \qquad \overset{\textbf{II}}{\begin{cases} x = t \oplus k = 1 - y + x + y - 1 = x \\ y = f \oplus k = 1 - x + x + y - 1 = y \\ x = (f \oplus u)^* = 1 - (1 - x + 0) = x \\ y = (t \oplus u)^* = 1 - (1 - y + 0) = y \end{cases}} \tag{2.55}$$

The proof is then complete. *Note* We did not write out the $\min(..., 1)$ operation implied by $\oplus$ to save space, since all the sums inside end up being $x, y, 1 - x, 1 - y$, which all are less than 1 in L.

The above theorems together show that there is a one-to-one correspondence between pairs of elements of L (elements of the product set $L \times L$) and the quadruplet of values f, k, u, t. Thus we present the following definition:

Definition 2. *For the MV-algebra* $\mathbf{L} = \langle L, \oplus, ^*, \mathbf{0} \rangle$*, the product set* $L \times L$ *is called* the set of evidence couples*.*

The set of evidence couples turns into an MV-algebra $\mathbf{L}_{EC} = \langle L \times L, \otimes, ^\perp, \overline{\mathbf{0}} \rangle$ by setting:

$$\langle a, b \rangle \otimes \langle c, d \rangle = \langle a \oplus c, b \odot d \rangle \tag{2.56}$$

$$\langle a, b \rangle^\perp = \langle a^*, b^* \rangle \tag{2.57}$$

$$\overline{\mathbf{0}} = \langle \mathbf{0}, \mathbf{1} \rangle \tag{2.58}$$

Clearly the axioms 2.1-2.6 are satisfied since $\oplus$ and $\odot$ obey them:

$$\langle a, b \rangle \otimes \langle c, d \rangle = \langle a \oplus c, b \odot d \rangle = \langle c \oplus a, d \oplus b \rangle = \langle c, d \rangle \otimes \langle a, b \rangle \tag{2.59}$$

$$\begin{aligned} \langle a, b \rangle \otimes (\langle c, d \rangle \otimes \langle e, f \rangle) &= \langle a, b \rangle \otimes \langle c \oplus e, d \odot f \rangle \\ &= \langle a \oplus (c \oplus e), b \odot (d \odot f) \rangle = \langle (a \oplus c) \oplus e, (b \odot d) \odot f \rangle \\ &= \langle a \oplus c, b \odot d \rangle \otimes \langle e, f \rangle = (\langle a, b \rangle \otimes \langle c, d \rangle) \otimes \langle e, f \rangle \end{aligned} \tag{2.60}$$

$$\langle a, b \rangle \otimes \langle 0, 1 \rangle = \langle a \oplus 0, b \odot 1 \rangle = \langle a, b \rangle \tag{2.61}$$

$$(\langle a, b \rangle^\perp)^\perp = \langle a^*, b^* \rangle^\perp = \langle a^{**}, b^{**} \rangle = \langle a, b \rangle \tag{2.62}$$

$$\langle a, b \rangle \otimes \langle 1, 0 \rangle = \langle a \oplus 1, b \odot 0 \rangle = \langle 1, 0 \rangle = \overline{\mathbf{0}} \tag{2.63}$$

$$\begin{aligned} (\langle a, b \rangle^\perp \otimes \langle c, d \rangle)^\perp \otimes \langle c, d \rangle &= \langle a^* \oplus c, b^* \odot d \rangle^\perp \otimes \langle c, d \rangle \\ &= \langle (a^* \oplus c)^* \oplus c, (b^* \odot d)^* \odot d \rangle = \langle (c^* \oplus a)^* \oplus a, (d^* \odot b)^* \odot b \rangle \\ &= \langle c^* \oplus a, d^* \odot b \rangle^\perp \otimes \langle a, b \rangle = (\langle c, d \rangle^\perp \otimes \langle a, b \rangle)^\perp \otimes \langle a, b \rangle \end{aligned} \tag{2.64}$$

And also operations similar to 2.7-2.17 can be defined, further explanation can be found in [4]. We skip it since it is just an intermediate step to evidence matrices.

Definition 3. *The set of matrices* $\mathcal{M} = \left\{ \begin{bmatrix} f & k \\ u & t \end{bmatrix} = \begin{bmatrix} x^* \wedge y & x \odot y \\ x^* \odot y^* & x \wedge y^* \end{bmatrix} \middle| \langle x, y \rangle \in L \times L \right\}$ *induced by the set of evidence couples is called* the set of evidence matrices.

We have already shown that sets of evidence couples and evidence matrices are in one-to-one correspondence.

We use operations defined for evidence couples above to define operations on evidence matrices:
Let A and B be evidence matrices induced by evidence couples $\langle x, y \rangle$ and $\langle a, b \rangle$. Operation defined in (2.24) creates an evidence couples $\langle x \oplus a, y \odot b \rangle$, which in turn induces an evidence matrix $C = \begin{bmatrix} (x \oplus a)^* \wedge (y \odot b) & (x \oplus a) \odot (y \odot b) \\ (x \oplus a)^* \odot (y \odot b)^* & (x \oplus a) \wedge (y \odot b)^* \end{bmatrix}$, so we define an operation $\bigoplus$ on evidence matrices so that $A \bigoplus B = C$, and similarly other operations. If we write $\mathcal{M}(\langle a, b \rangle)$ for "evidence matrix induced by the evidence couple $\langle a, b \rangle$", we get the following definitions:

$$\mathcal{M}(\langle a, b \rangle) \bigoplus \mathcal{M}(\langle x, y \rangle) = \mathcal{M}(\langle a, b \rangle \otimes \langle x, y \rangle) \tag{2.65}$$

$$\mathcal{M}(\langle a, b \rangle)^{\perp} = \mathcal{M}(\langle a, b \rangle^{\perp}) \tag{2.66}$$

$$F = \mathcal{M}(\langle \mathbf{0}, \mathbf{1} \rangle) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \tag{2.67}$$

The same way for the operations for evidence couples and matrices that we skipped.

This is done of course because now the structure $\mathcal{M} = \langle \mathcal{M}, \bigoplus, ^{\perp}, F \rangle$ is an MV-algebra (of evidence matrices).

The evidence couple $\langle \mathbf{0}, \mathbf{1} \rangle$ was used to define the zero element of the sets of evidence couples and matrices, however similarly we may use couples $\langle \mathbf{1}, \mathbf{0} \rangle, \langle \mathbf{1}, \mathbf{1} \rangle, \langle \mathbf{0}, \mathbf{0} \rangle$ to induce the following matrices, respectively:

$$T = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, K = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, U = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix}. \tag{2.68}$$

These matrices F, T, K, U then correspond to the values *false*, *true*, *contradictory* and *unknown* from Belnap's Logic $\mathcal{FOUR}$, which inspired the original authors of [4] for the system explained here.

2.4 Practical application for Paraconsistent logic

The mathematical machinery described above can be used to create a system of Paraconsistent logics, which in turn can be applied to a real data mining system.

Generally, Paraconsistent logic is a novel field of Logic, which aims to deal with situations where inconsistencies are unavoidable. Classical logic has the "Law of excluded middle", thus either a statement or its negation is false, but both cannot be assumed to be true, since that leads to explosion into triviality, and any statement can be deduced to be true.

Different paraconsistent logics deal with this problem differently, and a general presentation of the setting is given in [13]. We present the setting developed in [4], which deals with the problem by introducing evidence couples to each statement such that we have a value indicating the degree of belief that it is true, and another value indicating the degree of belief that the statement is false. Making inferences now involves operating on this evidence couple, thus information about the potential inconsistency is preserved, and inferred staments also have a limited degree of belief.

In short, associate to each statement A an evidence couple $\alpha = \langle a, b\rangle, a, b \in [0, 1]$, and then calculate the values Truth, Falsehood, Contradictory and Unknown as $T(\alpha) = \min(a, 1 - b), F(\alpha) = \min(1 - a, b), K(\alpha) = \max(0, a + b - 1), U(\alpha) = \max(0, 1 - a - b)$, and thus also associate an evidence matrix $\mathcal{M} = \begin{bmatrix} F(\alpha) & K(\alpha) \\ U(\alpha) & T(\alpha) \end{bmatrix}$. Previously proven theorem guarantees that $T(\alpha) + F(\alpha) + K(\alpha) + U(\alpha) = 1.$

However, for the practical application, there is an evident question now - where would the evidence couples for the statements actually come from? Next chapter presents a particular Data Mining approach which makes it possible to create the evidence couples for a particular kind of statements from available data and use them to judge how justified those statements are.

3 THE GUHA METHOD

This chapter shortly deals with a particular Data Mining method - GUHA - and shows how results of the chapter 2 can be applied to create a new useful quantifier for the program, which seems to detect dependies that would be overwise undiscovered.

3.1 History of GUHA Method

General Unary Hypothesis Automaton (GUHA), is a method proposed in 1966 by Petr Hajek and Tomas Havranek. The original book was since then turned into electronic form and is now available under [5]. It is a rigorously defined logical formalism, which allows to add to the logic language so-called generalized quantifiers, ie quantifiers other than existential $\exists$ and universal $\forall$.

Generalized quantifiers are used to define notions such as 'In most cases $P(x)$' or '$P(x)$ is more common than...' in precise mathematical language, giving them a truth value, and form closed formulas with them, thus allowing to make conclusions in situations where classical quantifiers do not provide enough insight. The purpose of software implementations of GUHA method is to mine for *hypothesis* - systematically produce formulas yielding `True` value for a given quantifier in an optimized manner, making use of the rules of inference derived for generalized quantifiers. The most basic type of hypothesis that is possible to mine are the Associational rules - quantifiers of the form $\psi(x) \approx \phi(x)$, which mean that ψ and ϕ often either occur together or both do not occur.

A Windows software packaged called LISp-Miner is developed by Jan Raunch in Prague University of Economics [14], with contributions by several scientists for different mining procedures (including the one by Esko Turunen [6], on which with work is based). A historic overview of software implementations and basic definitions for GUHA is given by Jan Raunch in [15]. There is also a book by the same author dealing with mining Associational rules from the formal point of view - [16].

Certain applied studies with GUHA demonstrate its usefullness to produces hypothesis for possible reasons of certain phenomena. These include: reasons for traffic incidents [17], sleep disorders [18], contraception use [19], [6].

We now present the basic ideas required to undestand mining for hypothesis with GUHA and defining generalized quantifiers.

3.2 GUHA Formalism

In GUHA Data Mining, data is presented in a flat 2D table (matrix), where every row represents an object of study (ie a person), and every column represents some attribute of such objects (height, age, gender etc). Each object is then treated as a variable of a formal language, and every attribute as a logical predicate, which can only yield values `True` or `False`. Thus attributes need to be somehow quantized first - continuous attributes like age can be binned into intervals, and then each interval can have a corresponding predicate: "Is age 20 to 25?", "Is age 26 to 30?" etc. Categorical attributes can either have a predicate for each category or for certain sets of categories. It is up to researcher to define predicates that are sensible to analyze based on the research questions.

The predicates which are formed from the data directly can then be connected into formulas with familiar symbols $\neg$ for negation, $\wedge$ for AND, $\vee$ for OR; for example 'Age is 20 to 25 AND Income is low OR Income is none', 'Age is greater than 26 AND Income is NOT low'. Now let $\psi(x)$ and $\phi(x)$ be two such formulas, where x denotes the variable and thus ranges over all objects in our matrix. For these two formulas we can form a four-fold table (4ft) as follows:

4ft	$\psi(x)$	$\neg\psi(x)$	
$\phi(x)$	a	b	r
$\neg\phi(x)$	c	d	s
	k	l	m

where a stands for number of objects satisfying both ψ and ϕ, b stands for the number of objects satisfying ϕ but not ψ etc. k, l, s, r and column- and row-wise sums, m is the total amount of objects.

Now, to define a generalized quantifier $\phi(x) \approx \psi(x)$, we need to come up with a rule which gives value `True` or `False` based on the values a, b, c, d (or, equivalently, r, s, k, l, m). Now we give two simple examples of such quantifiers.

Quantifier of basic association One of the simplest way to define an associational rule that claims that two predicates either occur or do not occur together is to use 'coincidence prevails over difference' - compare the values a, d with b, c. This can be done in one simple step as

$$ad > bc, \tag{3.1}$$

thus, if we denote this quantifier as $\sim$, the truth value $v(\phi(x) \sim \psi(x))$ is defined to be `True` iff $ad > bd$.

Quantifier of founded implication We can also define a quantifier which represents the notion 'ϕ being true implies ψ also true with certain evidence', denoted as $\Rightarrow_{p,\mathrm{BASE}}$. This one uses two selected parameters, $p \in (0,1]$ and $\mathrm{BASE} \in \mathbb{N}$. The first parameter is used to compare the amount of objects for which ϕ is true, to the amount of objects for which also ψ is true, ie the proportion $\frac{a}{a+b}$, while the second one ensures that we have enough 'evidence' of coexistence of the predicates in the first place. The truth value of this implicational quantifier is thus defined as:

$$v(\phi(x) \Rightarrow_{p,\mathrm{BASE}} \psi(x)) = \mathtt{True} \Leftrightarrow \begin{cases} \frac{a}{a+b} > p \\ a \geq \mathrm{BASE} \end{cases} \tag{3.2}$$

An application of those and further basic quantifiers is demonstrated by the present author in [19].

By carefully considering the meaning of a, b, c, d and proportions involving them, it is possible to derive many other, more 'complicated' quantifiers, involving statistical significance tests (Fisher and Chi-squared quantifiers), difference between groups with different value of an attribute (SD4ft procedure, making two 4ft tables), changes in value of some attribute (Ac4ft procedure, also involes two 4ft tables) and many others. We further consider, how it is possible to implement the ideas of Paraconsistent logic into a quantifier.

3.3 Paraconsistent Separation Quantifier

After becoming familiar with basic GUHA procedures and running some tasks, one can come to the conclusion that most of generalized quantifiers aim to show that two phenomena somehow cause each other's presence - either that just co-occur, or one implies another, or presence of one phenomena makes another one stronger. But at the same time, another research question might be "What are phenomena that do *not* occur together?", ie are mutually exclusive.

There do not seem to be quantifiers that cater to this question in the original GUHA procedures. However, as noted previously, 4ft values a, b, c, d directly lead to evidence of phenomena either occuring together or not, thus it seems that it is possible to utilise Paraconstent logic setting discussed in the end of chapter 2. Indeed, this was made possible in [6]. We now present an overview of the quantifier described in that paper and how it naturally comes from notion of evidence for and against some statement.

To use Paraconsistent logic machinery described previously, one is required to 'find evidence' for and against some statement to form an evidence couple and thus an evidence matrix. Assume that we wish to say "ψ and ϕ occur mutually exclusively", then from our 4ft setting the natural evidence *against* that statement is the value a - the number of objects which satisfy both ϕ and ψ simultaneously. On the over hand, the value *in favor* of this statement are values b and c taken together, so the amount of objects for which one and only one of ϕ, ψ is true, that is $b + c$.

Since we want to show that phenomena are mutually exclusive, we need that $b + c$ be suitably larger than a. This can be enforced by condition

$$(1 + p)a \leq b + c, p \in [0, 1] \tag{3.3}$$

which stands for "there are at least $p \cdot 100\%$ more cases that ϕ and ψ are mutually exclusive than that they co-occur", while we might also want enough evidence that phenomena can occur mutually exclusively:

$$b + c \geq \mathrm{BASE} \tag{3.4}$$

This could be already enough to define a quantifier, however, to get more insight into the data we need to find an evidence couple. To do it we need to somehow normalize values a and $b + c$. There are two meaningful ways to do it, we could either divide by $a + b + c$ to obtain $\langle \frac{b+c}{a+b+c}, \frac{a}{a+b+c} \rangle$, or by $n = a + b + c + d$ to obtain a couple $\langle \frac{b+c}{n}, \frac{a}{n} \rangle$.

We notice that in the first case, $\frac{a}{a+b+c} + \frac{b+c}{a+b+c} = 1$, thus paraconsistent values for Contradictory and Unknown will both be zero, which is not very insightful. On the other hand, $1 - \frac{a}{n} - \frac{b+c}{n} = \frac{d}{n} \neq 0$, which can be seen as the paraconsistent value Unknown, since d denotes the cases when neigher ϕ nor ψ is present. Thus it makes more sense to use the latter way as the evidence couple, so that we obtain the evidence matrix

$$\mathcal{M}\Big(\langle \frac{b+c}{n}, \frac{a}{n} \rangle\Big) = \begin{bmatrix} \frac{b+c}{n} & 0 \\ \frac{d}{n} & \frac{a}{n} \end{bmatrix} \tag{3.5}$$

As noted in [6], actually the first evidence couple is used to determine the truth value of the quantifier, since it does not require determining d, which saves computation time, however after the truth value is determined to be `True`, the second evidence couple can be used to actually obtain the evidence matrix.

The authors of [6] came into contact with the developers of LISp-Miner software, so such a quantifier is now programmed into current versions of the package. Same paper also includes a demonstration of possible generated hypotheses based on 1987 National Indonesia Contraceptive Prevalence Survey ([20], some overview given in [19]). Hypotheses found by Paraconsistent separation seem to be novel in a sense that they were not achieved by previously implemented quantifiers.

4 CONCLUSION

In this thesis, we gave an outline of the foundations of MV-algebras and in particular, the Lukasiewicz L-structure, which can be used to create Fuzzy (Many-valued) and Paraconsistent Logic settings. The key result is the ability to define evidence couples and matrices as MV-algebras, since such a structure descends naturally from Lukasiewicz L-structure.

Even though these systems are new, there is a growing interest and array of literature, both theoretic and applied, about different definitions, uses and benefits of such logic systems. We managed to find a working case of combining two of the non-classical logic systems - Paraconsistent logic with evidence couples on one hand and General Unary Hypothesis Automaton on the other, and the two seem to nicely complement each other.

However, previous literature review shows that these particular logic systems are not that famous in the wide scientific community, where most of research has been on Annotated Logics. Indeed, all key modern papers discussed in this work have the same author in common - Esko Turunen, who seems to have a personal interest and dedication to this particular topic.

Further research in this direction could be based on determining the applicability and effiency of Paraconsistent Separation Quantifier of GUHA, when applied to research questions in different domains. Also other quantifiers based on evidence couples could be created by carefully considering the meaning of a, b, c, d coefficients of 4ft setting. In general, applicability of Paraconsistent Logic with evidence couples for different data mining methods could also be studied.

REFERENCES

[1] Chang, C. C. Algebraic analysis of many-valued logics. *Transactions of the American Mathematical Society* 58 (1958), 467–490.

[2] Turunen, E. *Mathematics Behind Fuzzy Logic*. Advances in Soft Computing. Physica-Verlag, 1999.

[3] Turunen, E. A well-defined fuzzy sentential logic. *Mathematical Logic Quarterly* 41 (1995), 236–248.

[4] Turunen, E., Ozturk, M. and Tsoukias, A. Paraconsistent sematics for Pavelka style fuzzy sentential logic. *Fuzzy sets and systems* 161 (2010), 1926–1940.

[5] Hajek, P. and Havranek, T. *Mechanizing Hypothesis Formation*. Springer-Verlag, online, 2002. URL: `http://www.cs.cas.cz/hajek/guhabook/guhabook.pdf`.

[6] Turunen, E. Paraconsistent Many-Valued Logic in GUHA Framework. *Acta informatica Pragensia* 7 (2018), 104–111.

[7] Zamansky, A. On recent applications of paraconsistent logic: an exploratory literature review. *Journal of Applied Non-Classical Logics* 29 (2019), 382–391.

[8] Simons, P. *Jan Łukasiewicz*. The Stanford Encyclopedia of Philosophy (Summer 2020 Edition). 2020. URL: `https://plato.stanford.edu/archives/sum2020/entries/lukasiewicz/`.

[9] Łukasiewicz, J. O logice trójwartościowej. *Ruch Filozoficzny* 5 (1920).

[10] Łukasiewicz, J. Interpretacja liczbowa teorii zdań. *Ruch Filozoficzny* 7 (1922).

[11] Nola, A. D., Grigolia, R. and Turunen, E. *Fuzzy Logic of Quasi-Truth: An Algebraic Treatment*. Studies in Fuzziness and Soft Computing. Springer, 2016.

[12] Chang, C. C. A New Proof of the Completeness of the Łukasiewicz Axioms. *Transactions of the American Mathematical Society* 93 (1959), 74–80.

[13] Carnielli et al. Logics of Formal Inconsistency. 2007.

[14] *LISpMiner software*. URL: `https://lispminer.vse.cz/` (visited on 10/25/2020).

[15] Hajek, P., Holena, M. and Raunch, J. The GUHA method and its meaning for data mining. *Journal of Computer and System Sciences* 76 (2010), 34–48.

[16] Rauch, J. *Observational Calculi and Association Rules*. Springer, 2013.

[17] Turunen, E. Using GUHA Data Mining Method in Analyzing Road Traffic Accidents Occured in the Years 2004-2008 in Finland. *Data Science and Engineering* 2 (2017), 224–231.

[18] Merlevede, J. Studying of sleep-disordered breathing, The GUHA Method. MA thesis. Tampere, Finland: Tampereen Teknillinen Yliopisto, May 2009.

[19] Bakhtin, A. *Determining patterns in contraception use with GUHA Framework. Methodology overview*. 2020. URL: `https://www.researchgate.net/publication/346081663_Determining_patterns_in_contraception_use_with_GUHA_Framework_Methodology_overview`.

[20] Statistics (Indonesia), C. B. of. *National Contraceptive Prevalence Survey 1987*. 1987. URL: `https://microdata.worldbank.org/index.php/catalog/1398`.