Chapter 1

A Study of Self-Referential Systems, Knights and Knaves Using Fuzzy Logic and Global Optimization

Athanasios Kehagias

Department of Electrical and Computer Engineering,
Aristotle University, Thessaloniki, Greece,
kehagiat@ece.auth.gr

In this work we study *self-referential sentences*, i.e., sentences which talk about themselves. It is well known that self-referential sentences may generate logical paradoxes, which have been studied since the time of the ancient Greeks. We show that every system of self-referential sentences generates a corresponding system of numerical equations, with the unknowns being the *truth-values* of the original sentences; this numerical system is the *truth-value system*. We then prove that, under appropriate conditions, the truth-value system possesses at least one solution in *fuzzy* truth values (i.e., values in the real numbers interval $[0, 1]$). Then we concentrate on a class of self-referential systems which are genreated, from Smullyan's *Knights and Knaves puzzles*. We conclude with a *global optimization* approach to the solution of the truth value system; this approach is evaluated by numerical experiments.

1. Introduction

We introduce a formalism for studying *self-referential sentences*, i.e., sentences which talk about themselves. Sentences of this type often generate logical paradoxes, the study of which goes back to the ancient Greeks. We show how to obtain, for each system of self-referential sentences a corresponding system of numerical equations, where the unknowns are the *truth-values* of the original sentences. Accordingly, the numerical system is called the *truth-value system*. We show that, under mild conditions, the truth-value system posseses at least one solution in *fuzzy* truth values, each of which lies in the real numbers interval $[0, 1]$. We pay special attention to a particular class of self-referential systems, obtained from Smullyan's *Knights and Knaves puzzles*. Finally we present a *global optimization* approach to the solution of the truth value system and evaluate this approach by numerical experiments.

The prototypical example of a self-referential sentence is the *Liar Sentence* (supposedly stated the Cretan philosopher Epimenides):

"All Cretans are Liars".

Assuming that "All Cretans are Liars" means that everything a Cretan says is not true, we obtain the following paradox. Since Epimenides is a Cretan, his statement that "All Cretans are Liars" is not true; if we take "not true" to mean "false", then the opposite of "All Cretans are Liars" must be true; and if we take this opposite to be "All Cretans are truth-tellers", then what Epimenides says must be true, i.e., it is true that "All Cretans are Liars"; but then it must be the case that what Epimenides says is false. It seems that we have entered a vicious circle, concluding first that Epimenides' statement is false, then that it is true, then again that it is false and so on ad infinitum. As already mentioned, there is an extensive literature on self-referential sentences [3, 6, 9, 16, 19–22, 26–30, 32, 34, 37]. The application of fuzzy logic to the Liar paradox goes back to a paper by Zadeh [41]; in summary, he resolves the paradox by assigning to the Liar sentence the *fuzzy truth value* 0.5. Following Zadeh's paper, several authors have analyzed self-reference using fuzzy logic [8, 11, 12, 23, 34].

A particular type of self-referential sentences appears in the family of *Knights and Knaves* logic puzzles, introduced by Raymond Smullyan in *What Is the Name of This Book* [35]. Typically such a puzzle is set on a fictional island where each inhabitant is either a *knight* (who only makes true statements) or a *knave* (who only makes false statements). In the simplest form of the puzzle a visitor meets some islanders and he must deduce their *types* (knight or knave) from statements they make about the type (knight or knave) of themselves. For example, the visitor meets two islanders and one of them says

"At least one of us is a knave".

The visitor can deduce from the above that the speaker is a knight and the other islander is a knave[a]. As the example illustrates, knight and knave puzzles usually result in a consistent truth value assignment to the islanders' statements rather than in a paradox[b]. Fuzzy versions of such puzzles have been studied in [4].

In this work we consider the problem of assigning consistent fuzzy truth values to collections of self-referential sentences (including knight and knave sentences) and show that the problem can be reduced to the solution of a system of nonlinear equations. Furthermore, we prove that, under mild conditions, such a system always has a solution (i.e. a consistent truth value assignment).

This chapter is structured as follows. In Section 2 we present some preliminaries regarding Boolean and fuzzy logic. In Section 3 we introduce the subject of self-reference and in Section 4 we present a framework for the formulation of self-referential sentence systems. In Section 5 we show how to reduce a system of self-referential sentences to a system of numerical truth value equations and study some properties of such systems. Section 6 is devoted to the presentation of some

[a]The reader may at this point try to solve the puzzle; the solution is presented in Section 6.2.1.
[b]However, it should be emphasized that Smullyan [35, 36] and several other authors [2, 17, 25, 33] have produced a great variety of Knights and Knaves puzzles of increasing complexity and sophistication.

specific examples of our approach. In Section 7 we introduce our approach to the solution of truth-value systems by global optimization and in Section 8 we present numerical examples to evaluate the approach. Finally, in Section 9 we summarize our work and present some future research directions.

2. Preliminaries

2.1. *Boolean Logic*

We use *sentence* as a synonym for *statement*; both terms mean a syntactically well formed structure which makes some claim. On the other hand, a *proposition* is a statement which has a *truth value*; as will be seen, in certain contexts not all sentences have truth values. Initially we will only consider two truth values, **TRUE** and **FALSE**, but this will be extended in the sequel. *Propositional logic* is the study of (i) the ways in which propositions/sentences can be joined and/or modified and (ii) of the resulting truth values. The *basic logical operations* by which new sentences can be formed are the conjunction **AND**, the disjunction **OR** and the negation **NOT**. Thus, given logical sentences X_1, X_2 we can form new sentences X_1 **AND** X_2, X_1 **OR** X_2, **NOT** X_1 as well more complicated combinations.

A convenient calculus for the study of propositional logic is *Boolean logic* or *Boolean algebra* [13]. In its simplest form [c], this is a formalism for operating on sentences X_1, X_2 or, more conveniently, on their respective truth values x_1, $x_2 \in \{0, 1\}$, where 0 is shorthand for **FALSE**, and 1 is shorthand for **TRUE**. The logical operations can be conveniently denoted as follows: **AND** is denoted by $\wedge$, **OR** is denoted by $\vee$ and **NOT** is denoted by $'$. These can be understood as algebraic operations, defined by the following table

x_1	x_2	$x_1 \wedge x_2$	$x_1 \vee x_2$	x_1'
0	0	0	0	1
0	1	0	1	1
1	0	0	1	0
1	1	1	1	0

Table 1.: Definition of Boolean conjunction, disjunction and negation.

It is easily checked that the above table corresponds to the intuitively expected results of **TRUE** and **FALSE** combinations. For example, it is reasonable that, when either X_1 or X_2 are **FALSE**, then X_1 **AND** X_2 is also **FALSE**. Given sentences (i.e., logical formulas) X_1, X_2, X_3, ... we can form new sentences $X_1 \wedge X_2$ (i.e., X_1 **AND** X_2), $X_1 \vee X_2$ (i.e., X_1 **OR** X_2), X_1' (i.e., **NOT** X_1) as well as more complex combinations, e.g., $(X_1' \vee X_2) \wedge (X_2 \vee X_3 \vee X_5)$ etc. and easily compute their truth values. Additional logical operations can be defined in terms of the above

[c]It should be added that, more generally, a Boolean algebra is defined to be a *complemented distributive lattice* [7].

three (actually all binary logical operations can be defined in terms of negation and either one of conjunction or disjunction) and various Boolean identities between combinations of such operations can be proved; the reader is referred to [13] for a detailed exposition. In what follows we take the informal point of view that precedence of operators, grouping of terms etc. are well understood from the context and do not require special explanation. Similarly, we treat the use of parentheses in an informal manner.

It is worth emphasizing that in the above paragraphs the symbols 0 and 1, are understood as shorthand for **FALSE** and **TRUE**, rather than as real numbers. However, as will be seen in the sequel, we can also define $\wedge$, $\vee$, $'$ as *numerical* functions operating on the real numbers 0 and 1.

2.2. *Fuzzy Logic*

According to the previous remarks, in Boolean logic sentences can only have the truth values 0 or 1. In *fuzzy logic* [15, 24], on the other hand, it is assumed that the truth value of a sentence may be any *real number* between 0 (completely false) and 1 (completely true); *partially* true or false sentences are assigned truth values from the interval $(0, 1)$. In other words, fuzzy logic is a many-valued (actually infinite-valued) logic [31]; while the term "fuzzy logic" itself was introduced in the 1960's by Zadeh [40], the subject has actually been studied since the 1920s [5, 18].

Hence the building blocks of fuzzy logic are (i) the set of truth-values, which is the real unit interval $[0, 1]$ (with its natural ordering $\leq$) and (ii) the logical operators, among which the fundamental ones are (as in Boolean logic) conjunction, disjunction and negation. These are now understood as numerical functions with domain either $[0, 1] \times [0, 1]$ (for conjunction and disjunction) or $[0, 1]$ (for negation) and range $[0, 1]$. There is considerable latitude in the choice of such numerical functions; the usual approach is to introduce a family of functions called *t-norms* (which generalize conjunction) and another family of functions called *t-conorms* (which generalize disjunction); these families are required to satisfy the following conditions.

(1) A t-norm is a function $T : [0, 1] \times [0, 1] \to [0, 1]$ that satisfies the following properties:
   (a) Commutativity: $T(a, b) = T(b, a)$
   (b) Monotonicity: $(a \leq c \text{ and } b \leq d) \Rightarrow T(a, b) \leq T(c, d)$.
   (c) Associativity: $T(a, T(b, c)) = T(T(a, b), c)$
   (d) The number 1 acts as identity element: $T(a, 1) = a$.

(2) A t-conorm is a function $S : [0, 1] \times [0, 1] \to [0, 1]$ that satisfies the following properties:
   (a) Commutativity: $S(a, b) = S(b, a)$
   (b) Monotonicity: Monotonicity: $(a \leq c \text{ and } b \leq d) \Rightarrow S(a, b) \leq S(c, d)$.

(c) Associativity: $S(a, S(b, c)) = S(S(a, b), c)$
(d) The number 0 acts as identity element: $S(a, 0) = a$.

(3) A negation is a function $N : [0, 1] \to [0, 1]$ that satisfies the following properties:

(a) Monotonicity: $a \leq b \Rightarrow N(a) \geq N(b)$.
(b) $N(0) = 1$ and $N(1) = 0$.

It is easily proved that, when the above conditions are satisfied, $T(\cdot)$, $S(\cdot)$ and $N(\cdot)$ behave "classically" when they operate on the Boolean truth values 0 and 1. In other words, $T(\cdot)$, $S(\cdot)$ and $N(\cdot)$ yield the same values as the Boolean operators $\wedge$, $\vee$ and $'$ do in Table 1. In what follows we will often also denote a t-norm T by the symbol $\wedge$, a t-conorm S by the symbol $\vee$ and a negation N by the symbol $'$.

There is a great number of t-norms and t-conorms (i.e., functions which satisfy the above conditions). Some of the most popular ones are presented in the following table.

	Standard	**Algebraic**	**Lukasiewicz**	**Drastic**	**Hamacher/Einstein**
$x \wedge y$	$\min(x, y)$	xy	$\max(0, x + y - 1)$	$\begin{array}{ll} y & \text{iff } x = 1 \\ x & \text{iff } y = 1 \\ 0 & \text{otherwise} \end{array}$	$\begin{array}{ll} 0 & \text{iff } x = y = 0 \\ \frac{xy}{x+y-xy} & \text{otherwise} \end{array}$
$x \vee y$	$\max(x, y)$	$x + y - xy$	$\min(x + y, 1)$	$\begin{array}{ll} y & \text{iff } x = 0 \\ x & \text{iff } y = 0 \\ 1 & \text{otherwise} \end{array}$	$\frac{x+y}{1+xy}$

Table 2.: Some popular t-norms and t-conorms.

In addition there exist several possible negation functions; for example we can use $N(x) = 1 - x^p$, for any $p > 0$. In this paper we always use $N(x) = 1 - x$.

3. Self-Reference

Self-referential sentences are sentences which talk about themselves. By a *system of self-referential sentences* or, for short, a *self-referential system* we mean a finite collection of sentences which talk about each other. Sentences of this type *may* generate logical paradoxes, the study of which goes back to the ancient Greeks. The paradox generally reduces to the fact that a sentence, or a system of sentences, may fail to have consistent truth values (that is why we speak of *sentences* rather than *propositions*). The Liar and related systems have been investigated exhaustively and many approaches have been introduced to "neutralize" the paradoxes [10, 28–30, 32, 34, 37–39].

3.1. *Liar-like Examples*

Consider the following statement supposedly made by the Cretan philosopher Epimenides: "*All Cretans are Liars*". This statement (Liar Sentence) is paradoxical in the sense that it cannot be either true or false. Because, assuming the statement means that everything a Cretan says is not true then (since Epimenides is a Cretan) the statement itself is not true, i.e., it is false. Consequently the opposite is

true, i.e., "All Cretans are truth-tellers" and hence the statement "All Cretans are Liars" is also true. So we we have a vicious circle in which we first conclude that Epimenides' statement is false, then that it is true, then again that it is false and so on ad infinitum.

3.1.1. *The Liar*

The above reasoning is not rigorous. First, "All Cretans are Liars" does not necessarily mean that all Cretans *always* make false statements. Second, the negation of "All Cretans are Liars" is "*Some* Cretans are truth-tellers". But the paradox persists under more exact reasoning. Consider the following sentence:

$$\text{"This sentence is false."} \tag{1}$$

or, in the notation which we will use throughout this paper,

$$X = \text{"}X \text{ is false"} \tag{2}$$

This is the classic example of a self-referential sentence which leads to paradox, namely the so-called "*Liar Paradox*". The name originates from the previously presented Epimenides formulation. The paradox can be seen by the following reasoning.

(1) If we assume the sentence to be true, then what it says must hold, i.e. the sentence must be false. Consequently its opposite must be true, i.e. it must be true that "This sentence is true". But then it is true that "This sentence is false" and we have again entered an oscillation between two conclusions: first that the sentence is false, then that it is true.
(2) It can be easily checked that we obtain a similar oscillation if we assume the sentence to be false: then we would conclude that the sentence is true, which would mean that the sentence is false etc.

The *Liar paradox* is that the sentence (2) cannot be either true or false, i.e., it cannot have a (Boolean) truth value.

3.1.2. *The Inconsistent Dualist*

A similar paradox can be obtained using two sentences. The following pair is the so-called *inconsistent dualist*:

$$X_1 = \text{"}X_2 \text{ is true"}$$
$$X_2 = \text{"}X_1 \text{ is false"}$$

If X_1 is true, then X_2 is also true; but then X_1 must be false and so X_2 must be false and so on ad infinitum.

3.1.3. *The Consistent Dualist*

Self-reference does not necessarily lead to paradox. The following pair is the so-called *consistent dualist*:

$$X_1 = \text{“}X_2 \text{ is true”}$$
$$X_2 = \text{“}X_1 \text{ is true”}$$

If X_1 is true, then X_2 is also true, which confirms that X_1 is true. In short, accepting that X_1 and X_2 are both true is perfectly consistent. However, note that we could equally well assume that X_1 is false, which would mean that X_2 is also false, which confirms that X_1 is false. In short, we can accept equally well that either both X_1 and X_2 are true or both are false or, in other words, the above self-referential system does not have a *unique* truth value assignment. Is this a paradox?

3.1.4. *A More Complicated Example*

In each of the previous examples we have a collection of sentences which talk about the truth or falsity of each other. Using this idea we can create more complicated examples. Consider the following self-referential system.

$$X_1 = \text{“}X_2 \text{ is true”} \ \wedge \ \text{“}X_3 \text{ is false”}$$
$$X_2 = \text{“}X_1 \text{ is true”} \ \wedge \ \text{“}X_3 \text{ is false”}$$
$$X_3 = \text{“}X_1 \text{ is false”}$$

The reader is encouraged to *resolve* this example, i.e., to find consistent assignments of truth and falsity for X_1, X_2, X_3.

3.2. *Knights and Knaves Examples*

A particular type of self-referential sentence systems has been popularized by the logician R. Smullyan in the form of *Knights and Knaves* (henceforth KK) puzzles [35, 36]. The original setting of these puzzles is that a logician visits an island in which every inhabitant is either a *Knight* (always tells truths) or a *Knave* (always tells lies) and meets some islanders, each of which makes one or more statements regarding himself and his compatriots; the logician (actually the reader) must deduce from these statements the *type* (knighthood or knavehood) of one or more islanders. [d] Note that KK puzzles usually do not lead to paradox; in other words, usually there exists one or more consistent knighthood/knavehood assignments. Nevertheless, KK puzzles are a special case of self-referential systems, as will become clear in the following examples.

[d]The above is the *basic* setup. Smullyan has presented various modifications which result in puzzles of increasing complexity.

3.2.1. *First Smullyan Example*

Here is one of the simplest KK puzzles; it appears as Example 28 in the book *What is the Name of This Book?* [35]. It involves islanders K_1 and K_2, each of whom is either a knight or a knave. K_1 makes the statement:

"At least one of us is a knave"

and it is required to determine the *types* (knight or knave) of K_1 and K_2.

If K_1 were a knave, then "At least one of us is a knave" would be false (since knaves only make false statements); consequently both K_1 and K_2 would be knights. In other words, if K_1 were a knave he would also be a knight. Hence K_1 is not a knave, which means he is a knight and his statement is true. So at least one of the islanders is a knave; since K_1 is a knight, K_2 is a knave.

The problem can be reduced to the following self-referential system

$$X_1 = \text{"}K_1 \text{ is a knight"} \tag{3}$$

$$X_2 = \text{"}K_2 \text{ is a knight"} \tag{4}$$

$$Y_1 = X_1' \vee X_2' \tag{5}$$

Keeping in mind that Y_1 is the sentence stated by K_1, we see that (3)-(5) is equivalent to the original statement of the puzzle.

Now, we can reason as follows. On the one hand, Y_1 will be true iff at least one of X_1, X_2 is false. On the other hand, Y_1 *is stated by* K_1 hence it must have the same truth value as X_1 (i.e., it will be true if K_1 is a knight and false if K_1 is a knave). We illustrate these facts in the following table.

X_1	X_2	$Y_1 = X_1' \vee X_2'$	$X_1 \equiv Y_1$
0	0	1	0
0	1	1	0
1	**0**	**1**	**1**
1	1	0	0

Table 3.: Truth table for (3)-(5).

The first two columns of the above table list the $2^2 = 4$ possible combinations of Boolean truth values for X_1 and X_2 (one per row); the third column lists the corresponding truth values of $Y_1 = X_1' \vee X_2'$; the final column lists the truth values of $X_1 \equiv Y_1$, which takes the value 1 (resp. 0) iff the truth values of X_1 and Y_1 are equal (resp. different). Note that $\equiv$ can be understood as Boolean operation combining X_1 and Y_1; in fact it is equivalent to the double implication $X_1 \Leftrightarrow Y_1$ which can be rewritten in terms of the elementary logical operations as $(X_1 \wedge Y_1) \vee (X_1' \wedge Y_1')$. Now, a necessary and sufficient condition for a truth value assignment to be a solution of this particular puzzle is that X_1 and Y_1 have the same truth values. More explicitly, we must have (i) either that K_1 is a knight and his stated sentence Y_1 is true (ii) or that K_1 is a knave and his stated sentence Y_1 is false. We see in the

table that this only happens in the third row. So the puzzle has a unique solution: K_1 is a knight (X_1 is true) and he states a true sentence "K_1 is a knight and K_2 is a knave" (Y_1 is true).

This approach can be generalized (as will be seen in the following examples) and can also be reformulated as a numerical problem (as will be seen in Section 5.2).

3.2.2. *Second Smullyan Example*

This appears as Example 33 in *What is the Name of This Book?* [35]. We have two islanders; suppose K_1 says, "I am a knave, but K_2 isn't". Now, K_1 cannot be a knight (because then his statement would be true and he would be a knave) hence he is a knave and his statement is false. K_2 cannot be a knight (because then K_1's statement would be true) hence K_2 is also a knave. So K_1, K_2 are both knaves.

We can write the problem as the following self-referential system

$$X_1 = \text{“}K_1 \text{ is a knight”} \tag{6}$$

$$X_2 = \text{“}K_2 \text{ is a knight”} \tag{7}$$

$$Y_1 = X_1' \wedge X_2 \tag{8}$$

From the above system we get the following table,

X_1	X_2	$Y_1 = X_1' \wedge X_2$	$X_1 \equiv Y_1$
0	**0**	**0**	**0**
0	1	1	0
1	0	0	1
1	1	0	0

Table 4.: Truth table for (6)-(8).

The solution is given by the first row, i.e., both "K_1 is a knight" and "K_2 is a knight" are false. Hence both K_1 and K_2 are knaves; this is the same solution we got by logical reasoning.

3.2.3. *Third Smullyan Example*

This appears as Example 34 in *What is the Name of This Book?* [35]. We have three islanders; K_1 says "K_2 is a knave" and K_2 says "K_1 and K_3 are of the same type"[e]. What is K_3?

(1) If K_1 were a knight, his statement that K_2 is a knave would be true. Then K_2's statement that K_1 and K_3 are of the same type would be false and K_1 and K_3 would be of different types. Thus, since K_1 is a knight, K_3 must be a knave.
(2) If K_1 were a knave, his statement that K_2 is a knave would be false and K_2 would be a knight. Thus K_2 's statement that K_1 and K_3 are of the same type would be true. Since K_1 is a knave, so is K_3.

[e]Two islanders are said to be *of the same type* if they are both knights or both knaves.

In other words, we have shown that, whether K_1 is a knight or a knave, K_3 is a knave.

We can write the problem as the following self-referential system

$$X_1 = \text{“}K_1 \text{ is a knight”} \tag{9}$$
$$X_2 = \text{“}K_2 \text{ is a knight”} \tag{10}$$
$$X_3 = \text{“}K_3 \text{ is a knight”} \tag{11}$$
$$Y_1 = X_2' \tag{12}$$
$$Y_2 = (X_1 \wedge X_3) \vee (X_1' \wedge X_3') \tag{13}$$

From the above system we get the following table,

X_1	X_2	X_3	$Y_1 = X_2'$	$Y_2 = (X_1 \wedge X_3) \vee (X_1' \wedge X_3')$	$X_1 \equiv Y_1$	$X_2 \equiv Y_2$
0	0	0	1	1	0	0
0	0	1	1	0	0	1
0	**1**	**0**	**0**	**1**	**1**	**1**
0	1	1	0	0	1	0
1	**0**	**0**	**1**	**0**	**1**	**1**
1	0	1	1	1	1	0
1	1	0	0	0	0	0
1	1	1	0	1	0	0

Table 5.: Truth table for (9)-(13).

The only acceptable truth value assignments are the ones in which both $X_1 \equiv Y_1$ and $X_2 \equiv Y_2$. There are two of these, in the third and fifth row of the table. In both of these X_3 is false, i.e., K_3 is a knave. Hence in this problem, while we cannot determine the type of K_1 and K_2, we know that K_3 is a knave; this is the same conclusion we reached by logical reasoning.

3.2.4. *The Arageorgis Example*

We conclude with an interesting KK variant; it is a slightly modified version of a puzzle presented by A. Arageorgis [1]. The puzzle concerns the investigation of a murder committed in the island of Knights and Knaves. There exist three suspects: K_1, K_2 and K_3 and it is certain that one or more of them is the murderer. The investigator A questions them and they make the following statements.

(1) K_1 says: I am guilty iff K_3 is a Knight, and you can prove my guilt if and only if you can prove K_3 is a knight.
(2) K_2 says: Neither me nor K_1 is a knight.
(3) K_3 says: You cannot prove that I am a knight.

The following additional information is provided: A is a “*perfect reasoner*”; in other words, he can only prove true statements and will never prove a false statement.

Let us set up the logical equation system corresponding to the above situation. We start with the standard statements

$$X_1 = \text{“}K_1 \text{ is a knight”}$$
$$X_2 = \text{“}K_2 \text{ is a knight”}$$
$$X_3 = \text{“}K_3 \text{ is a knight”}$$

In addition we introduce the statements

$$P = \text{“}K_1 \text{ is guilty”}$$
$$Q = \text{“}A \text{ can prove } P\text{”}$$
$$R = \text{“}A \text{ can prove } X_3\text{”}$$

It follows from the nature of the above three statements that they take truth values in $\{0, 1\}$. Now we can write the suspects' statements as follows:

$$Y_1 = (P \Leftrightarrow X_3) \wedge (Q \Leftrightarrow R) \tag{14}$$
$$Y_2 = X_1' \wedge X_2' \tag{15}$$
$$Y_3 = R'. \tag{16}$$

Finally, since A is a perfect reasoner, the following facts are fully true:

$$\text{“}A \text{ can prove } P\text{”} \Rightarrow P \tag{17}$$
$$\text{“}A \text{ can prove } X_3\text{”} \Rightarrow X_3 \tag{18}$$

We omit the Boolean solution of the system; we will present the fuzzy solution in Section 6.2.4.

3.3. *Two Important Remarks*

From the above analysis it is clear that KK puzzles can be formulated (and solved) as self-referential systems. In fact, the only distinguishing feature of KK problems is their *semantics*, i.e., the back-story involving the Island of Knights and Knaves.

Note that in all the previously presented examples we have adopted a *Boolean* point of view; in other words, we have implicitly assumed that *each sentence must be either completely true or completely false*. However, *in the remainder of the paper we will adopt a fuzzy point of view*. In other words we will assume that *the sentences which we study can have any truth value in the interval* $[0, 1]$. By extending the set of permissible truth values we also extend the set of possible solutions of a self-referential system. Consequently:

(1) Self-referential systems which are paradoxical in the Boolean context may become non-paradoxical in the fuzzy context (in other words, they will admit one ore more consistent truth value assignments);
(2) KK puzzles will, in general, admit a larger number of solutions in the fuzzy context than in the Boolean one.

4. A Logic Framework for Self-Referential Sentences

We will now introduce a framework which can be used to study self-referential systems.

(1) We start with a finite set V_1 , the set of *first-level elementary sentences* (also called first level variables):

$$V_1 = \{X_1, X_2, ..., X_M\}.$$

(2) From the first level variables we recursively build S_1 , the set of *first level sentences*:

$$\text{If } X_m \in V_1 \text{ then } X_m \in S_1$$
$$\text{If } Y_1, Y_2 \in S_1 \text{ then } Y_1 \vee Y_2,\ Y_1 \wedge Y_2,\ Y_1' \in S_1$$

Obviously $V_1 \subseteq S_1$.

(3) Next we build V_2 , the *set of second level elementary sentences*:

$$V_2 = \{\text{“}Tr(Y) = y\text{”} : Y \in S_1, y \in [0,1]\}$$

where $Tr(\cdot)$ is shorthand for “The truth value of ...”; in other words, “$Tr(Y) = y$” means “The truth value of Y is y”.

(4) Next we recursively build S_2, the set of *second level sentences*:

$$\text{If } Z \in S_1 \cup V_2 \text{ then } Z \in S_2$$
$$\text{If } Z_1, Z_2 \in S_2 \text{ then } Z_1 \vee Z_2,\ Z_1 \wedge Z_2,\ Z_1' \in S_2$$

Note that $V_2 \subseteq S_2$.

(5) We continue in this manner, and obtain the sets of sentences V_3, S_3, V_4, S_4, ...

(6) The *set of all sentences* (with which we are concerned) is $\overline{S} = \bigcup_{m=1}^{\infty} S_m$.

Here are some examples of elements from V_1, S_1, V_2, S_2.

$V_1 : X_1, X_2, ..., X_M.$
$S_1 : X_1 \vee X_3,\ X_2',\ (X_2 \wedge X_4) \wedge X_5',\ ...$
$V_2 : \text{“}Tr(X_1) = 1\text{”},\ \text{“}Tr\left(X_7'\right) = 0\text{”},\ \text{“}Tr\left(X_1 \vee \left(X_4 \wedge X_5'\right)\right) = 0.3\text{”},\ ...$
$S_2 : \text{“}Tr(X_1) = 1\text{”} \wedge \text{“}Tr\left(X_7'\right) = 0\text{”},\ \text{“}Tr\left(X_4 \wedge X_5'\right) = 0.3\text{”} \wedge \text{“}X_1 \wedge (Tr(X_7) = 0)\text{”},\ ...$

5. Systems of Truth Value Equations

5.1. *Explicit Truth Value Assignment*

Suppose that all members of the set of elementary sentences $V_1 = \{X_1, ..., X_M\}$ have known truth values; let

$$\forall m \in \{1, ..., M\} : x_m = Tr(X_m) \in [0,1].$$

Now take any $Y \in S_1$; it is a *logical* formula containing the variables $X_1, ..., X_M$. If we replace every occurrence of X_m with x_m and interpret $\vee$, $\wedge$, $'$ as numerical (rather than logical) operations then we obtain a *numerical* formula containing the variables $x_1, ..., x_M$ and the operations $\vee$, $\wedge$, $'$. To obtain the truth value of Y we choose a particular numerical "*implementation*" of $(\vee, \wedge, ')$; for example we can select $\vee$, $\wedge$ from Table 1, let $Tr(X') = 1 - Tr(X)$ and perform the numerical calculations. Then

$$Y = F_Y(X_1, ..., X_m) \Rightarrow Tr(Y) = f_Y(Tr(X_1), ..., Tr(X_M)) = f_Y(x_1, ..., x_M).$$

Here $F_Y : V_1 \to S_1$ is a logical formula and $f_Y : [0,1]^M \to [0,1]$ is the corresponding numerical formula obtained by replacing X_m with x_m and now understanding the symbols $\vee$, $\wedge$, $'$ as the corresponding *numerical* t-norm, t-conorm and negation functions. In this manner, the truth value assignment originally defined on V_1 has been extended to S_1.

Now we can use the truth values of first level sentences to assign truth values to second level elementary sentences as follows. Given a $Z \in V_2$, which has the form

$$Z = \text{“}Tr(Y) = y\text{”}$$

with $Y \in S_1, y \in [0,1]$, we define the *truth function* $Tr(\cdot)$ by

$$Tr(Z) = 1 - |Tr(Y) - y|. \tag{19}$$

$Tr(Y)$ in (19) has already been defined, since $Y \in S_1$. Note that, according to (19), the maximum truth value of Z is 1 and it is achieved when $Tr(Y) = y$; the latter is exactly what Z says. More generally, the truth value of Z is a decreasing function of the absolute difference between $Tr(Y)$ and y. In this manner we can compute the truth value of every $Z \in V_2$.

Next, we can extend truth values from V_2 to S_2 in exactly the same manner as we extended truth values from V_1 to S_1. We can proceed doing this to assign truth values to any element of V_m, S_m for any $m \in \mathbb{N}$.

This *explicit* assignment of truth values does not involve any self-reference or circularity: starting with the initial specification of the truth values of X_1,..., X_M as x_1, ..., x_M, the truth values of all sentences are obtained as functions of x_1, ..., x_M.

5.2. *Implicit Truth Value Assignment*

Suppose we are given M elementary sentences X_1, ..., $X_M \in V_1$ and K sentences $Y_1, ..., Y_K \in \overline{S}$ of the form

$$\begin{aligned} Y_1 &= F_1(X_1, ..., X_M) \\ Y_2 &= F_2(X_1, ..., X_M) \\ &... \\ Y_K &= F_M(X_1, ..., X_M). \end{aligned} \tag{20}$$

In other words, each Y_k is a logical formula involving X_1, X_2, ..., X_M in various combinations obtained by use of $\wedge, \vee, '$ and the truth function $Tr(\cdot)$.

Now suppose that, for each $k \in \{1, ..., K\}$, we identify X_k with Y_k. Then (20) becomes

$$\begin{aligned} X_1 &= F_1(X_1, ..., X_M) \\ X_2 &= F_2(X_1, ..., X_M) \\ &... \\ X_K &= F_K(X_1, ..., X_M). \end{aligned} \tag{21}$$

Each sentence X_k makes a claim about the truth value of the other sentences (possibly also about its own truth value) and (21) is a self-referential system. Also, (21) implies that

$$\begin{aligned} Tr(X_1) &= f_1(Tr(X_1), ..., Tr(X_M)) \\ Tr(X_2) &= f_2(Tr(X_1), ..., Tr(X_M)) \\ &... \\ Tr(X_K) &= f_K(Tr(X_1), ..., Tr(X_M)) \end{aligned} \tag{22}$$

where $f_k : [0,1]^M \to [0,1]$ is the numerical formula obtained from the logical formula F_k. Letting $x_k = Tr(X_k)$, for $k \in \{1, ..., K\}$, we can rewrite (22) in simpler form as follows:

$$\begin{aligned} x_1 &= f_1(x_1, ..., x_M) \\ x_2 &= f_2(x_1, ..., x_M) \\ &... \\ x_K &= f_K(x_1, ..., x_M). \end{aligned} \tag{23}$$

We will call (23) (a system of K numerical equations in M unknowns) the system of *truth value equations*.

In general, the truth value equations will be nonlinear and, depending on the particular $(\wedge, \vee, ')$ implementation, they may have none, one or more than one solutions in $[0,1]^M$. By specifying a particular $(\vee, \wedge, ')$ implementation, we obtain a set of solutions of (23), i.e., a set of possible consistent truth value assignments for X_1, ..., X_M. It is conceivable that the set of solutions of (23) is empty, i.e. that there is no consistent truth value assignment. However, we will now show that, under mild conditions, every $(\vee, \wedge, ')$ implementation produces at least one consistent truth value assignment. This is the subject of Proposition 3. We first need two auxiliary propositions.

Proposition 1. If the $(\vee, \wedge, ')$ implementations are, respectively, a continuous t-conorm, a continuous t-norm and a continuous negation, then $f_1, f_2, ..., f_K$ in (23) are continuous functions of $(x_1, x_2, ..., x_M)$.

Proof. For the sake of brevity we give only a sketch of the proof. Suppose that $\vee$, $\wedge$, $'$ are a continuous t-conorm, t-norm and negation, respectively. We are given a self-referential system of the form (21). Take any $k \in \{1, 2, ..., K\}$ and the corresponding logical equation

$$X_k = F_k (X_1, ..., X_M) . \tag{24}$$

First, take any $F_k (X_1, ..., X_M) \in S_1$. Corresponding to (24 we have the numerical equation

$$x_k = f_k (x_1, x_2, ..., x_M)$$

where f_k is a finite combination of $\vee$, $\wedge$, $'$ and $x_1, ..., x_M$; hence f_k is clearly a continuous function of the vector $(x_1, x_2, ..., x_M)$.

Second, take any $F_k (X_1, ..., X_M) \in V_2$. Then $F_k (X_1, ..., X_M) =$"$Tr (Y) = y$", where $Y \in S_1$. Since $Y \in S_1$, we will have $Tr (Y) = g (x_1, ..., x_M)$, where $g (x_1, ..., x_M)$ is a finite combination of $\vee$, $\wedge$, $'$ and $x_1, ..., x_M$. Hence $g (x_1, ..., x_M)$ will be a continuous function and so will be

$$f_k (x_1, x_2, ..., x_M) = Tr (\text{“}Tr (Y) = y\text{”}) = 1 - |g (x_1, ..., x_M) - y| .$$

Next, take any $F_k (X_1, ..., X_M) \in S_2$. The corresponding $f_k (x_1, ..., x_M)$ will be a finite combination of $\vee$, $\wedge$, $'$ and continuous functions of $(x_1, ..., x_M)$; hence it will be a continuous function.

Continuing in this manner, for every $F_k (X_1, ..., X_M) \in \overline{S}$ there will exist some l such that $F_k (X_1, ..., X_M) \in S_l$ and hence we can prove that $f_k (x_1, x_2, ..., x_M)$ is continuous by extending the previous argument for l inductive steps. □

Proposition 2. Suppose that $\mathbb{X}$ is a nonempty, compact, convex subset of $\mathbb{R}^M$. If the function $\mathbf{f} : \mathbb{X} \to \mathbb{X}$ is continuous, then there exists at least one point $\mathbf{x} \in \mathbb{X}$ satisfying $\mathbf{x} = \mathbf{f} (\mathbf{x})$ (i.e., $\mathbf{f} (\cdot)$ has at least one *fixed point*).

Proof. This a special case of the well-known Brouwer's Fixed Point Theorem. Its proof can be found in a number of standard texts, for instance in [14]. □

Now we can easily prove the existence of consistent truth value assignments.

Proposition 3. If $K \leq M$ and the $(\vee, \wedge, ')$ implementations are, respectively, a continuous t-conorm, a continuous t-norm and a continuous negation, then (23) has at least one solution $\mathbf{x} = (x_1, x_2, ..., x_M) \in [0, 1]^M$.

Proof. First suppose that $K = M$. We define the vector function $\mathbf{f} (x_1, x_2, ..., x_M)$ as follows:

$$\mathbf{f} (x_1, x_2, ..., x_M) = (f_1 (x_1, x_2, ..., x_M) , f_2 (x_1, x_2, ..., x_M) , ..., f_M (x_1, x_2, ..., x_M))$$

where, for $m \in \{1, 2, ..., M\}$, $f_m (x_1, x_2, ..., x_M)$ is the function appearing in (23). Since $f_m (x_1, x_2, ..., x_M)$ computes a truth value, we have $f_m : [0, 1]^M \to [0, 1]$ and

hence $\mathbf{f} : [0,1]^M \to [0,1]^M$. Furthermore, by Proposition 3 each f_m is a continuous function and so $\mathbf{f}$ is also a continuous function. Now we can apply Proposition 2 with $\mathbb{X} = [0,1]^M$.

If $K < M$ then, using arbitrary constants $y_1, ..., y_{M-K} \in [0,1]$, we can augment (23) by $M - K$ equations as follows

$$\begin{aligned} x_1 &= f_1(x_1, ..., x_M) \\ &... \\ x_K &= f_K(x_1, ..., x_M) \\ x_{K+1} &= y_1 \\ &... \\ x_M &= y_{M-K}. \end{aligned} \tag{25}$$

Now we can apply Proposition 2 to the system (25). □

In short, the assignment of consistent truth values to a self-referential system has been reduced to solving the (algebraic) system of truth value equations (23). Proposition 3 shows that, subject to some mild continuity conditions, the truth value equations admit at least one solution. Consequently, the Liar and related self-referential systems cease to be paradoxical in the context of fuzzy logic. From the mathematical point of view the situation is rather straightforward. The truth value equations may have no solution in $\{0,1\}^M$; by expanding the set in which we seek solutions to $[0,1]^M$, we are assured that the system always has at least one solution. Of course, this result is achieved at the price of admitting fuzzy solutions, i.e., truth values which fall short of certainty.

From the computational point of view, the next question is how to solve the truth value equations. Many approaches can be used to this end; in this paper we will concentrate on a global optimization approach (in Section 7) and will briefly mention several other possibilities (in Section 9).

We conclude this section with the following remark. Every $(\wedge, \vee, ')$ implementation results in a *different* system of truth value equations, and each such system will, in general, have a different solution set. However, the solution set will always (i.e., for every $(\vee, \wedge, ')$ implementation) contain the *Boolean solutions* (if any such exist), since all $(\wedge, \vee, ')$ implementations give the same values when applied to Boolean truth values $x_1, ..., x_M \in \{0,1\}^M$.

6. Examples

In this section we continue the analysis of the examples presented in Section 3. For each such example we present the system of logical equations, the corresponding system of truth value equations and the solutions to the latter.

6.1. *Liar-like Examples*

6.1.1. *The Liar Continued*

Recall that this example involves a single self-referential sentence:

$$X = \text{“}X \text{ is false”} \tag{26}$$

Hence we have

$$Tr(X) = Tr(\text{“}X \text{ is false”}) \Rightarrow Tr(X) = 1 - |Tr(X) - 0|$$

and, letting $x = Tr\left(X\right)$, we get

$$x = 1 - |x - 0| \Rightarrow x = 1 - x$$

which has the unique solution $x = \frac{1}{2}$. So the paradox is resolved in the context of fuzzy logic.

6.1.2. *The Inconsistent Dualist Continued*

This example involves the following self-referential system:

$$X_1 = \text{“}X_2 \text{ is true”}$$
$$X_2 = \text{“}X_1 \text{ is false”}$$

Letting $x_1 = Tr\left(X_1\right)$ and $x_2 = Tr\left(X_2\right)$ we obtain the system of truth value equations

$$x_1 = x_2$$
$$x_2 = 1 - x_1$$

which has the unique solution is: $x_1 = \frac{1}{2}$, $x_2 = \frac{1}{2}$. So the paradox is resolved in the context of fuzzy logic.

6.1.3. *The Consistent Dualist Continued*

This example involves the following self-referential system:

$$X_1 = \text{“}X_2 \text{ is true”}$$
$$X_2 = \text{“}X_1 \text{ is true”}$$

Letting $x_1 = Tr\left(X_1\right)$ and $x_2 = Tr\left(X_2\right)$ we obtain the system of truth value equations

$$x_1 = x_2$$
$$x_2 = x_1$$

which has an infinite number of solutions

$$\left\{\left(x_1, x_2\right) = \left(z, z\right),\, z \in [0, 1]\right\}.$$

Once again, the paradox is resolved.

6.1.4. *A More Complicated Example Continued*

This example involves the system

$$X_1 = \text{“}X_2 \text{ is true”} \ \wedge \ \text{“}X_3 \text{ is false”}$$
$$X_2 = \text{“}X_1 \text{ is true”} \ \wedge \ \text{“}X_3 \text{ is false”}$$
$$X_3 = \text{“}X_1 \text{ is false”}$$

Let $x_1 = Tr(X_1)$, $x_2 = Tr(X_3)$ and $x_3 = Tr(X_3)$. Since two of the above equations involve a conjunction, we have a choice regarding its numerical implentation.

(1) If we use $x \wedge y = \min(x, y)$ (standard t-norm) then we obtain the truth value system

$$x_1 = \min(1 - |x_2 - 1|, 1 - |x_3 - 0|)$$
$$x_2 = \min(1 - |x_1 - 1|, 1 - |x_3 - 0|)$$
$$x_3 = 1 - |x_1 - 0|$$

which reduces to

$$x_1 = \min(x_2, 1 - x_3)$$
$$x_2 = \min(x_1, 1 - x_3)$$
$$x_3 = 1 - x_1$$

This has an infinity of solutions:

$$\{(x_1, x_2, x_3) = (z, z, 1 - z)\}.$$

(2) If we use $x \wedge y = xy$ (algebraic t-norm) then we obtain the truth value system

$$x_1 = (1 - |x_2 - 1|)(1 - |x_3 - 0|)$$
$$x_2 = (1 - |x_1 - 1|)(1 - |x_3 - 0|)$$
$$x_3 = 1 - |x_1 - 0|$$

This reduces to

$$x_1 = x_2(1 - x_3)$$
$$x_2 = x_1(1 - x_3)$$
$$x_3 = 1 - x_1$$

which has three solutions: $(x_1, x_2, x_3) = (0, 0, 1)$, $(x_1, x_2, x_3) = (1, 1, 0)$ and the inadmissible $(x_1, x_2, x_3) = (-1, 1, 2)$.

6.2. *Knights and Knaves Examples*

Now we study some KK puzzles. Our analysis is based on a modification of Smullyan's basic setup. Namely, we assume that the puzzles play out in the island of *Fuzzy* Knights and Knaves; in this island each islander is characterized by a “*Knighthood Degree*” (KD) which takes values in $[0, 1]$ and *is the truth value of all statements made by the specific islander*. Hence, unlike classic Smullyan KK puzzles, the island of Fuzzy Knihts and Knaves is populated by a continuum of islander types.

(1) We have "full-knights" with KD equal to 1; they correspond to Smullyan's Knights.
(2) We also have "null-knights" with KD equal to 0; they correspond to Smullyan's Knaves.
(3) But also have "half-knights" with KD equal to $\frac{1}{2}$, "quarter-knights" with KD equal to $\frac{1}{4}$ etc.

We repeat that each islander's KD is fixed and consequently *all his statements have the same truth value*, equal to his KD.

In all the KK examples presented in the sequel, we will use the following shortcut, in the interest of brevity. In a situation involving M islanders $K_1, ..., K_M$ we will have M "primary" sentences of the form

$$X_m = \text{“}K_m \text{ is a knight”}$$

and N "secondary sentences" $Y_1, ..., Y_N$ which are stated by the islanders. According to our previous notation, let these sentences have corresponding truth values $x_1, ..., x_M$ and $y_1, ..., y_N$. Now, whenever K_m is stating Y_n we will set $y_n = x_m$. In this way we immediately eliminate the y_n's from the truth value equations.

Note also that it is no longer appropriate to substitute expressions of the form "K_i and K_j are of the same type" with the formula "$(X_i \wedge X_j) \vee \left(X_i' \wedge X_j'\right)$". This is the case because in fuzzy logic the principle of excluded middle does not hold; i.e., the formula "$(X_i \wedge X_j) \vee \left(X_i' \wedge X_j'\right)$" does not necessarily take values exclusively in $\{0, 1\}$. Hence, when formulating our truth value equations, we will take

$$Tr\left(\text{“}K_i \text{ and } K_j \text{ are of the same type”}\right)$$

to have the truth value

$$1 - \left|Tr\left(X_i\right) - Tr\left(X_j\right)\right|.$$

6.2.1. *First Smullyan Example Continued*

This has been introduced in Section 3.2.1. Recall that it involves the following self-referential system:

$$X_1 = \text{“}K_1 \text{ is a knight”}$$
$$X_2 = \text{“}K_2 \text{ is a knight”}$$
$$Y_1 = X_1' \vee X_2'$$

We let $x_1 = Tr\left(A_1\right)$, $x_2 = Tr\left(A_2\right)$.

(1) Using the standard t-norm $x \vee y = \max\left(x, y\right)$ we obtain the truth value equation

$$x_1 = \max\left(1 - x_1, 1 - x_2\right)$$

This has the solution set

$$\left\{\left(x_1, x_2\right) = \left(\frac{1}{2}, z\right), \ z \in \left[\frac{1}{2}, 1\right]\right\} \bigcup \left\{\left(x_1, x_2\right) = \left(1 - z, z\right), \ z \in [0\frac{1}{2})\right\}.$$

We have an infinity of admissible solutions, but the only Boolean one is $(x_1, x_2) = (1, 0)$ which means that "K_1 is a knight" is true and "K_2 is a knight" is false.

(2) If we use the algebraic t-norm $x \vee y = x + y - xy$, we get the truth value equation

$$x_1 = (1 - x_1) + (1 - x_2) - (1 - x_1)(1 - x_2)$$

with solution set

$$\left\{ (x_1, x_2) = \left(\frac{1}{1+z}, z \right), \ z \in [0, 1] \right\}.$$

Again we have an infinity of admissible solutions, but the only Boolean one is $(x_1, x_2) = (1, 0)$ which means that "K_1 is a knight" is true and "K_2 is a knight" is false.

In short, using either standard or algebraic t-norm, we obtain an infinity of solutions to the puzzle but the only Boolean solution is the same one we have previously obtained by either logical reasoning or the truth table method.

6.2.2. *Second Smullyan Example Continued*

This has been introduced in Section 3.2.2. It involves the following self-referential system:

$$X_1 = \text{"}K_1 \text{ is a knight"}$$
$$X_2 = \text{"}K_2 \text{ is a knight"}$$
$$Y_1 = X_1' \wedge X_2$$

We let $x_1 = Tr(A_1)$, $x_2 = Tr(A_2)$, $x_3 = Tr(A_3)$.

(1) If we use $x \wedge y = \min(x, y)$, we obtain the truth value equation

$$x_1 = \min(1 - x_1, x_2)$$

The solution set is

$$\left\{ (x_1, x_2) = (z, z), \ z \in [0, \frac{1}{2}) \right\} \bigcup \left\{ (x_1, x_2) = \left(\frac{1}{2}, z \right), \ z \in (\frac{1}{2}, 1] \right\}.$$

We have an infinity of admissible solutions, but the only Boolean one is $(x_1, x_2) = (0, 0)$ which means that both "K_1 is a knight" and "K_2 is a knight" are false.

(2) If we use $x \wedge y = xy$, we obtain the truth value equation

$$x_1 = (1 - x_1) x_2$$

with solution set

$$\left\{ (x_1, x_2) = \left(\frac{z}{1+z}, z \right), \ z \in [0, 1] \right\}.$$

We have an infinity of admissible solutions, but the only Boolean one is $(x_1, x_2) = (0, 0)$ which means that both "K_1 is a knight" and "K_2 is a knight" are false.

Again we obtain an infinity of solutions to the puzzle but the only Boolean solution is the same one we have previously obtained by either logical reasoning or the truth table method.

6.2.3. *Third Smullyan Example Continued*

This has been introduced in Section 3.2.3. It involves the following self-referential system:

$$\begin{aligned} X_1 &= \text{“}K_1 \text{ is a knight”} \\ X_2 &= \text{“}K_2 \text{ is a knight”} \\ X_3 &= \text{“}K_3 \text{ is a knight”} \\ Y_1 &= X_2' \\ Tr\left(Y_2\right) &= 1 - \left|Tr\left(X_1\right) - Tr\left(X_3\right)\right| \end{aligned}$$

We let $x_1 = Tr\left(A_1\right)$, $x_2 = Tr\left(A_2\right)$, $x_3 = Tr\left(A_3\right)$ and get

$$\begin{aligned} x_1 &= 1 - x_2 \\ x_2 &= 1 - \left|x_1 - x_3\right| \end{aligned}$$

The solution set is:

$$\left\{\left(x_1, x_2, x_3\right) = \left(1 - z, z, 0\right) : z < 1\right\} \bigcup \left\{\left(x_1, x_2, x_3\right) = \left(1 - z, z, 2\left(1 - z\right)\right) : z \leq 1\right\}$$

We have two Boolean solutions: $\left(x_1, x_2, x_3\right) = \left(1, 0, 0\right)$ and $\left(x_1, x_2, x_3\right) = \left(0, 1, 0\right)$. So, no matter what the truth values of X_1 and X_2 are, we always have $Tr\left(X_3\right) = 0$, i.e., K_3 is a knave; in other words, once again the only Boolean solution is the same one as obtained by logical reasoning and the truth tables method.

6.2.4. *The Arageorgis Example Continued*

We translate the logical equations of Section 3.2.4 to numerical ones. First, let us translate (17)-(18) in terms of our truth value function.

$$1 = Tr\left(Q \Rightarrow P\right) = Tr\left(Q' \vee P\right) \tag{27}$$

$$1 = Tr\left(R \Rightarrow X_3\right) = Tr\left(R' \vee X_3\right) \tag{28}$$

Using our standard notation and the fact that the truth value of a person's statements equals his KD, we obtain $x_1 = y_1$, $x_2 = y_2$, $x_3 = y_3$. Then, using the standard t-norm and t-conorm implementation, we get the truth value system

$$x_1 = \min\left(1 - \left|p - x_3\right|, 1 - \left|q - r\right|\right) \tag{29}$$

$$x_2 = \min\left(1 - x_1, 1 - x_2\right) \tag{30}$$

$$x_3 = 1 - r \tag{31}$$

$$1 = \max\left(1 - q, p\right) \tag{32}$$

$$1 = \max\left(1 - r, x_3\right) \tag{33}$$

Here (29)-(31) have been obtained from (14)-(16) and (32)-(33) have been obtained from (27)-(28).

We solve (29)-(33) analytically as follows. From (31) and (33) we immediately have

$$1 = \max(x_3, x_3) = x_3$$
$$1 = \max(1-r, 1-r) = 1-r \Rightarrow r = 0.$$

Substituting we get

$$x_1 = \min(1 - |p-1|, 1 - |q-0|) = \min(p, 1-q)$$
$$x_2 = \min(1-x_1, 1-x_2)$$
$$1 = \max(1-q, p)$$

Now we examine cases.

(1) If $x_2 = 1 - x_1 \leq 1 - x_2$, then $x_2 \leq \frac{1}{2} \leq x_1$.

(a) If in addition $1 = 1 - q$, then $q = 0$ and $x_1 = p$. But $x_1 \geq \frac{1}{2}$ and $p \in \{0, 1\}$ imply $x_1 = p = 1$. Then we also have $x_2 = 0$. So we finally have the solution

$$(x_1, x_2, x_3, p, q, r) = (1, 0, 1, 1, 0, 0).$$

(b) If however $1 = p$, then $x_1 = 1 - q$. Since $q \in \{0, 1\}$ and $x_1 \geq \frac{1}{2}$, we have $q = 0$ and $x_1 = 1$, $x_2 = 0$. So get the same solution as in the previous case.

(2) If $x_2 = 1 - x_2$ then $x_2 = \frac{1}{2}$ and $x_2 \leq 1 - x_1 \Rightarrow x_1 \leq \frac{1}{2}$.

(a) If in addition $1 = 1 - q$, then $q = 0$ and $x_1 = p$. But $x_1 \leq \frac{1}{2}$ and $p \in \{0, 1\}$ imply $x_1 = p = 0$. So we have a solution

$$(x_1, x_2, x_3, p, q, r) = \left(0, \frac{1}{2}, 1, 0, 0, 0\right).$$

(b) If on the other hand we have $1 = p$, then $1 - q = x_1 \leq \frac{1}{2} \Rightarrow 1 - \frac{1}{2} \leq q \Rightarrow q = 1$.Then $x_1 = \min(1, 1-1) = 0$. So we get the solution

$$(x_1, x_2, x_3, p, q, r) = \left(0, \frac{1}{2}, 1, 1, 1, 0\right).$$

In short, the solution set of the truth value equations is:

$$\left\{(1, 0, 1, 1, 0, 0), \left(0, \frac{1}{2}, 1, 0, 0, 0\right), \left(0, \frac{1}{2}, 1, 1, 1, 0\right)\right\}.$$

Only the first solution is Boolean.

An interesting phenomenon results from the use of sentences of the form "A can prove ...". Compare the purely Boolean solution $(1, 0, 1, 1, 0, 0)$ to the fuzzy one $\left(0, \frac{1}{2}, 1, 1, 1, 0\right)$. In both cases we have $p = 1$, i.e., K_1 is the murderer. But in the purely Boolean solution, $q = 0$, i.e., A *cannot* prove K_1's guilt (he *can* prove in the fuzzy solution, in which $x_2 = \frac{1}{2}$, i.e., K_2 is a half-Knight). The following metaquestion is also interesting: couldn't A apply the resoning presented above and thus actually prove K_1 *is* the murderer? The reader is encouraged to furnish his own answer!

7. Solving Self-Referential Systems by Global Optimization

In the previous sections we have set up the machinery for translating self-referential systems to systems of truth value equations

$$x_1 = f_1(x_1, ..., x_M), \quad ..., \quad x_K = f_M(x_1, ..., x_M). \tag{34}$$

The variables x_1, ..., x_M in (34) are the truth values of the M sentences X_1, ..., X_M and the numerical functions f_1, ..., f_M are obtained by the numerization of the original system of logical sentences. We will use global optimization methods to solve truth value systems. First we define the cost function

$$J(x_1, ..., x_M) = \sum_{k=1}^{K} (x_k - f_k(x_1, ..., x_M))^2 \tag{35}$$

which achieves its global minimum, equal to zero, at every solution of (34) and then we apply various global optimization algorithms to the problem of $J(x_1, ..., x_M)$ minimization. We evaluate the algorithms by performing a suite of experiments, using truth value equations obtained from self-referential systems and KK puzzles.

Let us first present the optimization algorithms which we have used in our experiments. These can be separated into three groups.

7.1. *Boolean Solutions by Exhaustive Search*

We can find the *Boolean* solutions of (34) by checking, for each element of $\mathbf{z} = \{0,1\}^M$, the value of $J(\mathbf{z})$. When $J(\mathbf{z}) = 0$, $\mathbf{z}$ is one of the Boolean solutions. This simple-minded method is our first choice for a quick exploration of the truth value equations, and is computationally viable for values of M up to about 15.

7.2. *Steepest Descent*

Steepest descent is perhaps the simplest numerical optimization algorithm. For the purposes of the current work, we have implemented a continuous steepest descent version based on the numerical integration of the following system of differential equations

$$\begin{aligned} \frac{dx_1}{dt} &= -\kappa \frac{\partial J}{\partial x_1} x_1 (1 - x_1) \\ &... \\ \frac{dx_M}{dt} &= -\kappa \frac{\partial J}{\partial x_M} x_M (1 - x_M) \end{aligned} \tag{36}$$

Note the inclusion of the term $x_m(1 - x_m)$ in the m-th differential equation; these terms are introduced to ensure that the trajectories of (36) remain inside $[0,1]^M$ (as is required for $\mathbf{x}$ to represent a vector of truth values). The set of equilibria of (36) contains all stationary points of $J(\mathbf{x})$; hence we can numerically integrate (36) and check, whenever an equilibrium point $\overline{\mathbf{x}}$ is reached, whether $J(\overline{\mathbf{x}}) = 0$. We have

performed the integration of (36) using a naive Euler ODE solver which we have coded in the Matlab language. As illustrated in the examples of Section 6, we can in general expect that the truth value equations have more than one solutions. To recover as many solutions as possible, we use a *multistart* approach, i.e., we restart the ODE solver from various initial conditions (in particular, we select the initial conditions randomly and uniformly from $[0,1]^M$).

7.3. *Matlab Global Optimization Toolbox*

We have also used the following standard optimization algorithms as provided in the Matlab *Global Optimization Toolbox*.

`fmincon`	local minimum of a constrained function
`patternsearch`	global minimum of a function by pattern search
`ga`	global minimum of a function by genetic optimization
`particleswarm`	global minimum of a function by particle swarm optimization
`surrogateopt`	global minimum of a function by surrogate optimization

Table 6.: Matlab Global Optimization functions used in our experiments.

To obtain as many solutions as possible, we have used the Matlab-provided methods `GlobalSearch` and `MultiStart`.

8. Global Optimization Experiments

8.1. *Experiment Group A*

We start with the KK example discussed in Sections 3.2.1 and 6.2.1. This is a simple example in which we can solve the truth value equations analytically; it is presented here to illustrate the basics of our global optimization approach. Recall that the logical equations are

$$\begin{aligned} X_1 &= \text{“}K_1 \text{ is a knight”} \\ X_2 &= \text{“}K_2 \text{ is a knight”} \\ Y_1 &= X_1' \vee X_2' \end{aligned}$$

Depending on the t-norm/t-conorm implementation we have the following truth value equations.

(1) Using the standard t-conorm implementation we obtain the truth value equation

$$x_1 = \max\left(1 - x_1, 1 - x_2\right) \tag{37}$$

It is easy to obtain the solution set analytically; it is

$$\left\{(x_1, x_2) = \left(\frac{1}{2}, z\right), \;\; z \in \left[\frac{1}{2}, 1\right]\right\} \bigcup \left\{(x_1, x_2) = (1 - z, z), \;\; z \in [0\frac{1}{2})\right\}.$$

The corresponding cost function is

$$J_s(x_1, x_2) = (x_1 - \max(1 - x_1, 1 - x_2))^2 .$$

(2) Using the algebraic t-conorm implementation we obtain the truth value equation

$$x_1 = (1 - x_1) + (1 - x_2) - (1 - x_1)(1 - x_2) \tag{38}$$

Once again, we can easily find the solution set to be

$$\left\{ (x_1, x_2) = \left(z, \frac{1 - z}{z} \right), \ \ z \in [0, 1] \right\}.$$

The corresponding cost function is

$$J_a(x_1, x_2) = (x_1 - ((1 - x_1) + (1 - x_2) - (1 - x_1)(1 - x_2)))^2 .$$

(3) Using the Lukasiewicz implementation we obtain the truth value equation

$$x_1 = \min((1 - x_1) + (1 - x_2), 1) \tag{39}$$

with solution set

$$\{(x_1, x_2) = (1, 0)\} \bigcup \{(x_1, x_2) = (z, 2 - 2z), \ \ z \in [0, 1)\}.$$

The corresponding cost function is

$$J_l(x_1, x_2) = (x_1 - \min((1 - x_1) + (1 - x_2) - 1, 1))^2 .$$

(4) Finally, using the Hamacher/Einstein implementation we obtain the truth value equation

$$x_1 = \frac{(1 - x_1) + (1 - x_2)}{1 + (1 - x_1)(1 - x_2)} \tag{40}$$

with solution set

$$\left\{ (x_1, x_2) = \left(z, \frac{z^2 - 3z + 2}{z^2 - z + 1} \right), z \in \left[\frac{1}{2}, 1 \right] \right\}.$$

The corresponding cost function is

$$J_l(x_1, x_2) = \left(x_1 - \frac{(1 - x_1) + (1 - x_2)}{1 + (1 - x_1)(1 - x_2)} \right)^2 .$$

Now we proceed to obtain the solution sets of each one of (37)-(40). First we use exhaustive enumeration to determine the Boolean solutions by. It turns out that there is a unique such solution: $(x_1, x_2) = (1, 0)$.

Next we run the previously mentioned optimization algorithms to obtain fuzzy solutions. All algorithms are run with a random multistart mechanism, repeated 100 times. All algorithms succeed in obtaining a "good sample" of the true solution sets (with a few exceptions which will be noted in the sequel). For example, our multi-start steepest descent algorithm produces the results plotted in Figure 1.

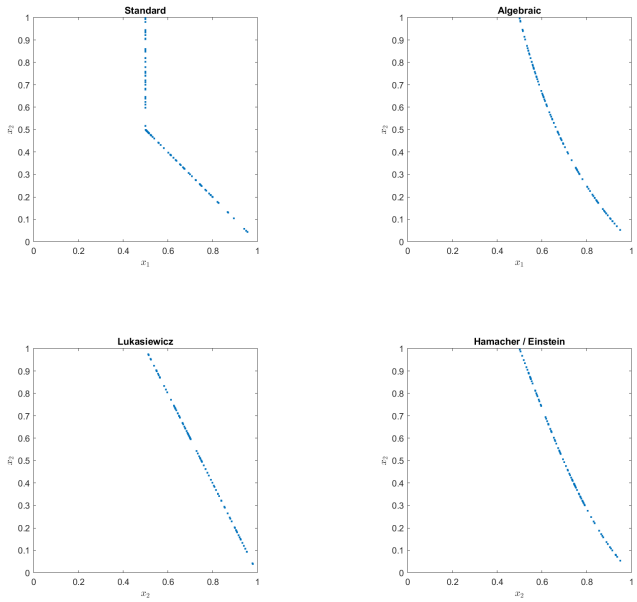
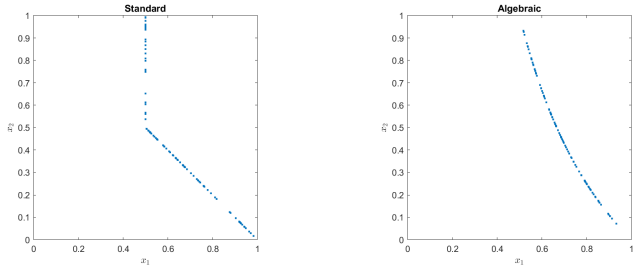


Fig. 1.: Solutions of equations (37)-(40), obtained by the Steepest Descent algorithm: (a) standard implementation, (b) algebraic implementation, (c) Lukasiewicz implementation,(d) Hamacher/Einstein implementation.

Each figure corresponds to one of the implementations (standard, algebraic, Lukasiewicz and Hamacher/Einstein). In Figure 1 we are plotting the 100 solutions (x_1, x_2) obtained by the 100 starting initial conditions. All the obtained solutions achieve the global minimum, namely zero. As can be seen, the solutions "cover" the corresponding solutions sets of (37)-(40).

The results obtained from the Matlab optimization functions are similar. For example, in Figure 2 we plot the results obtained with multi-start `fmincon`.



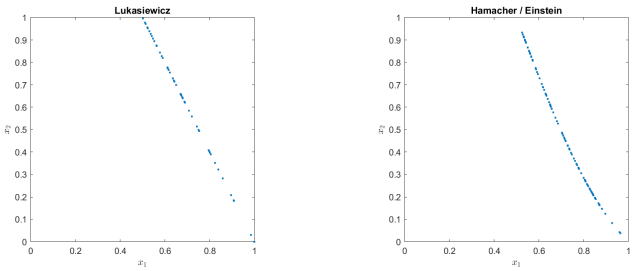


Fig. 2.: Solutions of equations (37)-(40), obtained by the `fmincon` algorithm: (a) standard implementation, (b) algebraic implementation, (c) Lukasiewicz implementation,(d) Hamacher/Einstein implementation.

It is easily seen that Figure 1 is very similar to Fig. 2. In general all algorithms yield very similar results, with the following exceptions.

(1) The Matlab `particleswarm` optimization algorithm locates a single solution, $(x_1, x_2) = (1, 0)$, for all t-norm/t-conorm implementations.
(2) The Matlab `surrogateopt` algorithm locates fewer solutions than the remaining algorithms. To illustrate this we present the obtained solutions for the algebraic and the Hamacher/Einstein implementations in Fig 3. Comparing this to Figures 1 and 2 we see the sparser coverage achieved by surrogate optimizaton.

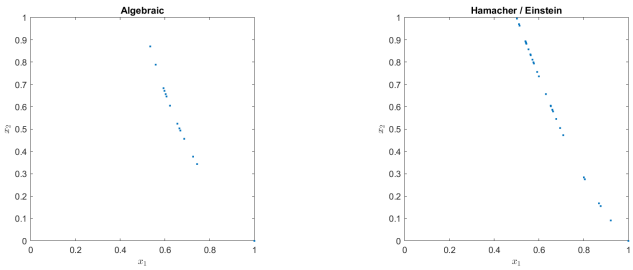


Fig. 3.: Solutions of equations (38) and (40), obtained by the `surrogateopt` algorithm: (a) algebraic implementation, (b) Hamacher/Einstein implementation.

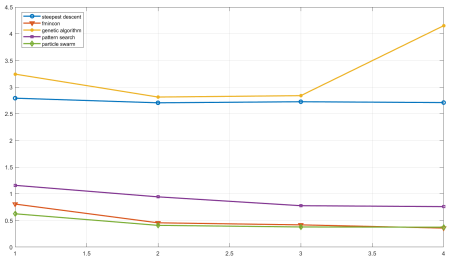


Fig. 4.: Execution times of the optimization algorithms in Experiment Group A, for the various t-norm/t-conorm implementations. The horizontal axis corresponds to t-norm/t-conorm implementations (1→ standard, 2→ algebraic, 3→ Lukasiewicz, 4→ Hamacher/Einstein) and the vertical axis corresponds to execution time (per 100 runs of each algorithm) in seconds.

We complete the comparison of the used algorithms by plotting their execution times in Figure 4. Each line corresponds to one optimization algorithm. The horizontal axis lists t-norm/t-conorm implementations and the vertical axis execution time (for the total of 100 runs of each algorithm) in seconds. Results for `surrogateopt` are not listed, but they are about 50 times higher than the average of the other algorithms.

In conclusion, we see that the application of optimization algorithms to the solution of this particular system of truth value equations has been succesful. In particular: Steepest Descent, `fmincon`, `ga` and `particleswarm` achieve a good coverage of the solution set with execution times in the order of a few seconds.

8.2. *Experiment Group B*

The second example we present is the three-dimensional self-referential system, presented in Sections 3.1.4 and 6.1.4. Recall that the logical equations are

$$\begin{aligned} X_1 &= \text{“}X_2 \text{ is true”} \ \wedge \ \text{“}X_3 \text{ is false”} \\ X_2 &= \text{“}X_1 \text{ is true”} \ \wedge \ \text{“}X_3 \text{ is false”} \\ X_3 &= \text{“}X_1 \text{ is false”} \end{aligned}$$

Depending on the t-norm/t-conorm implementation we have the following truth value equations.

(1) Using the standard implementation of the t-norm we obtain

$$\begin{aligned} x_1 &= \min\left(x_2, 1 - x_3\right) \\ x_2 &= \min\left(x_1, 1 - x_3\right) \\ x_3 &= 1 - x_1 \end{aligned}$$

and the cost function

$$J_s\left(x_1, x_2, x_3\right) = \left(x_1 - \min\left(x_2, 1 - x_3\right)\right)^2 + \left(x_2 - \min\left(x_1, 1 - x_3\right)\right)^2 + \left(x_3 - \left(1 - x_1\right)\right)^2$$

The exact solution set is easily found to be

$$\{(x_1, x_2, x_3) = (z, z, 1-z)\}.$$

(2) Using the algebraic implementation we get

$$\begin{aligned} x_1 &= x_2\,(1-x_3) \\ x_2 &= x_1\,(1-x_3) \\ x_3 &= 1-x_1 \end{aligned}$$

and the cost function

$$J_a\,(x_1, x_2, x_3) = (x_1 - x_2\,(1-x_3))^2 + (x_2 - x_1\,(1-x_3))^2 + (x_3 - (1-x_1))^2$$

There exist only two solutions, both of them Boolean:

$$(x_1, x_2, x_3) = (0, 0, 1) \text{ and } (x_1, x_2, x_3) = (1, 1, 0).$$

(3) Using the Lukasiewicz implementation we get

$$\begin{aligned} x_1 &= \max(0, x_2 + 1 - x_3 - 1) \\ x_2 &= \max(0, x_1 + 1 - x_3 - 1) \\ x_3 &= 1 - x_1 \end{aligned}$$

and the cost function

$$\begin{aligned} J_a\,(x_1, x_2, x_3) = (x_1 - \max(0, x_2 + 1 - x_3))^2 + (x_2 - \max(0, x_1 + 1 - x_3))^2 \\ + (x_3 - (1-x_1))^2. \end{aligned}$$

Again, the only two solutions are

$$(x_1, x_2, x_3) = (0, 0, 1) \text{ and } (x_1, x_2, x_3) = (1, 1, 0).$$

(4) Using the Hamacher/Einstein implementation we get

$$\begin{aligned} x_1 &= \frac{x_2(1-x3)}{(x_2 + 1 - x_3 - x_2(1-x_3))} \\ x_2 &= \frac{x_1(1-x3)}{(x_1 + 1 - x_3 - x_1(1-x_3))} \\ x_3 &= 1 - x_1 \end{aligned}$$

and the cost function

$$\begin{aligned} J_a\,(x_1, x_2, x_3) = &\left(x_1 - \frac{x_2(1-x3)}{(x_2 + 1 - x_3 - x_2(1-x_3))}\right)^2 \\ &+ \left(x_2 - \frac{x_1(1-x3)}{(x_1 + 1 - x_3 - x_1(1-x_3))}\right)^2 \\ &+ (x_3 - (1-x_1))^2. \end{aligned}$$

Again, the only two solutions are

$$(x_1, x_2, x_3) = (0, 0, 1) \text{ and } (x_1, x_2, x_3) = (1, 1, 0).$$

Similarly to Experiment Group A, we perform 100 multi-start runs of each previously mentioned algorithm on the above cost functions. The results are quite similar to the ones of Experiment Group A. Namely all runs achieve the global minimum equal to zero and locate either all the solutions (for algebraic, Lukasiewicz and Hamacher/Einstein implementations) or a good sample of the solution set (for standard implementation). The execution times are as plotted in Figure 5.

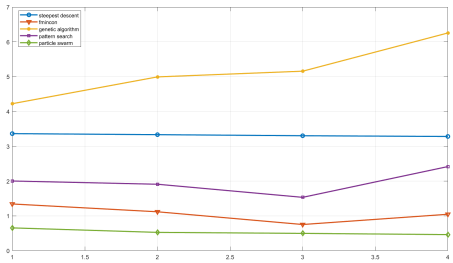


Fig. 5.: Execution times of the optimization algorithms in Experiment Group B, for the various t-norm/t-conorm implementations. The horizontal axis corresponds to t-norm/t-conorm implementations (1→ standard, 2→ algebraic, 3→ Lukasiewicz, 4→ Hamacher/Einstein) and the vertical axis corresponds to execution time (per 100 runs of each algorithm) in seconds.

8.3. *Experiment Group C*

We now move to a significantly more complex problem, the solution of the Arageorgis puzzle discussed in Sections 3.2.4 and 6.2.4. Recall that in this problem the logical equations are

$$\begin{aligned} X_1 &= (P \Leftrightarrow X_3) \wedge (Q \Leftrightarrow R) \\ X_2 &= X_1' \wedge X_2' \\ X_3 &= R' \\ 1 &= Tr\,(Q' \vee P) \\ 1 &= Tr\,(R' \vee X_3) \end{aligned}$$

which yield the truth value equations

$$\begin{aligned} x_1 &= (1 - |p - x_3|) \wedge (1 - |q - r|) \\ x_2 &= (1 - x_1) \wedge (1 - x_2) \\ x_3 &= 1 - r \\ 1 &= (1 - q) \vee p \\ 1 &= (1 - r) \vee x_3 \end{aligned}$$

with the additional constraint that $p, q, r \in \{0, 1\}$.

This example is considerably harder than the ones of Sections 8.1 and 8.2. First, it obviously has a higher dimensionality. In addition, as will be seen presently, the truth value equations are more complicated than the previously encountered ones. Finally there is the additional constraint that p, q, r must take Boolean values.

(1) Using the standard implementation we obtain

$$\begin{aligned} x_1 &= \min\left(1 - |p - x_3|, 1 - |q - r|\right) \\ x_2 &= \min\left(1 - x_1, 1 - x_2\right) \\ x_3 &= 1 - r \\ 1 &= \max\left(1 - q, p\right) \\ 1 &= \max\left(1 - r, x_3\right). \end{aligned}$$

Recall from Section 6.2.4 that the truth value equations possess three solutions (only the first one is Boolean):

$$\begin{aligned} (x_1, x_2, x_3, p, q, r) &= (1, 0, 1, 1, 0, 0) \\ (x_1, x_2, x_3, p, q, r) &= \left(0, \frac{1}{2}, 1, 0, 0, 0\right) \\ (x_1, x_2, x_3, p, q, r) &= \left(0, \frac{1}{2}, 1, 1, 1, 0\right). \end{aligned}$$

The cost function is

$$\begin{aligned} J_s\left(x_1, x_2, x_3, p, q, r\right) = &\left(x_1 - \min\left(1 - |p - x_3|, 1 - |q - r|\right)\right)^2 \\ &+ \left(x_2 - \min\left(1 - x_1, 1 - x_2\right)\right)^2 + \left(x_3 - (1 - r)\right)^2 \\ &+ \left(1 - \max\left(1 - q, p\right)\right)^2 + \left(1 - \max\left(1 - r, x_3\right)\right)^2. \end{aligned}$$

This must be minimized with respect to $x_1, x_2, x_3 \in [0, 1]$ and $p, q, r \in \{0, 1\}$. We handle the Boolean constraint by minimizing the augmented cost function

$$\widetilde{J}_s\left(x_1, x_2, x_3, p, q, r\right) = J_s\left(x_1, x_2, x_3, p, q, r\right) + p^2\left(1 - p\right)^2 + q^2\left(1 - q\right)^2 + r^2\left(1 - r\right)^2$$

which obviously achieves the global minimum 0 for the solutions of the truth value system which respect the Boolean constraint (and only for these solutions).

(2) Using the algebraic implementation we obtain the truth value system

$$\begin{aligned} x_1 &= \left(1 - |p - x_3|\right)\left(1 - |q - r|\right) \\ x_2 &= \left(1 - x_1\right)\left(1 - x_2\right) \\ x_3 &= 1 - r \\ 1 &= 1 - q + p - \left(1 - q\right)p \\ 1 &= 1 - r + x_3 - \left(1 - r\right)x_3, \end{aligned}$$

We have not been able to solve the truth value equations analytically. But we can obtain at least some solutions by using the cost function

$$\begin{aligned} J_a(x_1,x_2,x_3,p,q,r) &= (x_1-(1-|p-x_3|)(1-|q-r|))^2 \\ &+ (x_2-(1-x_1)(1-x_2))^2+(x_3-(1-r))^2 \\ &+ (1-(1-q+p-(1-q)p))^2 \\ &+ (1-(1-r+x_3-(1-r)x_3))^2 \end{aligned}$$

and minimizing the augmented cost function

$$\widetilde{J}_a(x_1,x_2,x_3,p,q,r) = J_a(x_1,x_2,x_3,p,q,r)+p^2(1-p)^2+q^2(1-q)^2+r^2(1-r)^2$$

(3) Using the Lukasiewicz implementation we obtain the truth value system

$$\begin{aligned} x_1 &= \max(0,1-|p-x_3|+1-|q-r|-1) \\ x_2 &= \max(0,1-x_1+1-x_2-1) \\ x_3 &= 1-r \\ 1 &= \min(1-q+p,1) \\ 1 &= \min(1-r+x_3,1). \end{aligned}$$

Again, we have not been able to solve the truth value equations analytically, so we will obtain solutions by using the cost function

$$\begin{aligned} J_l(x_1,x_2,x_3,p,q,r) &= (x_1-\max(0,1-|p-x_3|+1-|q-r|-1))^2 \\ &+ (x_2-\max(0,1-x_1+1-x_2-1))^2 \\ &+ (x_3-(1-r))^2+(1-\min(1-q+p,1))^2 \\ &+ (1-\min(1-r+x_3,1))^2 \end{aligned}$$

and minimizing the augmented cost function

$$\widetilde{J}_l(x_1,x_2,x_3,p,q,r) = J_l(x_1,x_2,x_3,p,q,r)+p^2(1-p)^2+q^2(1-q)^2+r^2(1-r)^2.$$

We emphasize that we have not been able to solve the truth value equations analytically.

(4) Finally, using the Hamacher/Einstein implementation we obtain the truth value system

$$\begin{aligned} x_1 &= \frac{(1-|p-x_3|)(1-|q-r|)}{(1-|p-x_3|)+(1-|q-r|)-(1-|p-x_3|)(1-|q-r|)} \\ x_2 &= \frac{(1-x_1)(1-x_2)}{(1-x_1)+(1-x_2)-(1-x_1)(1-x_2)} \\ x_3 &= 1-r \\ 1 &= \frac{(1-q)+p}{1+(1-q)p} \\ 1 &= \frac{(1-r)+x_3}{1+(1-r)x_3}. \end{aligned}$$

Since this system also appears analytically intractable, we use the cost function

$$
\begin{aligned}
&J_e\left(x_1,x_2,x_3,p,q,r\right)= \\
&\left(x_1-\frac{\left(1-\left|p-x_3\right|\right)\left(1-\left|q-r\right|\right)}{\left(1-\left|p-x_3\right|\right)+\left(1-\left|q-r\right|\right)-\left(1-\left|p-x_3\right|\right)\left(1-\left|q-r\right|\right)}\right)^2 \\
&+\left(x_2-\frac{\left(1-x_1\right)\left(1-x_2\right)}{\left(1-x_1\right)+\left(1-x_2\right)-\left(1-x_1\right)\left(1-x_2\right)}\right)^2 \\
&+\left(x_3-\left(1-r\right)\right)^2+\left(1-\frac{\left(1-q\right)+p}{1+\left(1-q\right)p}\right)^2 \\
&+\left(1-\frac{\left(1-r\right)+x_3}{1+\left(1-r\right)x_3}\right)^2
\end{aligned}
$$

and minimize the augmented cost function

$$\widetilde{J}_e\left(x_1,x_2,x_3,p,q,r\right)=J_a\left(x_1,x_2,x_3,p,q,r\right)+p^2\left(1-p\right)^2+q^2\left(1-q\right)^2+r^2\left(1-r\right)^2$$

The above-presented augmented cost functions are quite complex and the corresponding optimization problems can be expected to be harder than the ones treated in Section 8.1 and 8.2. This is corroborated by the obtained results, which can be summarized as follows.

We apply the usual suite of optimization algorithms. As in the previous Experiment Groups, 100 repetitions of each algorithm are run. One important difference from the previous two Experiment Groups is that now we have some algorithm runs which fail to locate the global minimum[f]. More precisely, we have set a threshold $h=10^{-3}$ and considered that the global maximum is achieved by some $(\widehat{x}_1,\widehat{x}_2,\widehat{x}_3,\widehat{p},\widehat{q},\widehat{r})$ iff $\widetilde{J}_a\left(x_1,x_2,x_3,p,q,r\right)<h$. Using this criterion, we define (for each algorithm) the *success rate* s_h to be the number of returned $(\widehat{x}_1,\widehat{x}_2,\widehat{x}_3,\widehat{p},\widehat{q},\widehat{r})$ which satisfy $\widetilde{J}_a\left(x_1,x_2,x_3,p,q,r\right)<h$, divided by 100 (the number of runs of each algorithm). These results are plotted for each algorithm in Figure 6.

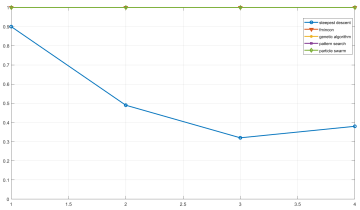


Fig. 6.: Success rate of the optimization algorithms for the various t-norm/t-conorm implementations. The horizontal axis corresponds to t-norm/t-conorm implementations (1→ standard, 2→ algebraic, 3→ Lukasiewicz, 4→ Hamacher/Einstein) and the vertical axis to s_h.

[f]Actually this only happens with the Steepest Descent Algorithm.

It can be seen that all algorithms except the Steepest Descent achieve the global minimum in every run; even the Steepest Descent algorithm achieves the global minimum in *some* runs. These conclusions hold for every implementation.

The next question is how well each algorithm "covers" the solution set. This can not be answered with complete certainty because we do not have an analytic expression of the full solution set (except for the standard impementation). However we can obtain some additional information. Consider again the three solutions of the truth value system corresponding to the standard implementation:

$$\begin{aligned}
\mathbf{x}^{(1)} &= (x_1, x_2, x_3, p, q, r) = (1, 0, 1, 1, 0, 0) \\
\mathbf{x}^{(2)} &= (x_1, x_2, x_3, p, q, r) = \left(0, \frac{1}{2}, 1, 0, 0, 0\right) \\
\mathbf{x}^{(3)} &= (x_1, x_2, x_3, p, q, r) = \left(0, \frac{1}{2}, 1, 1, 1, 0\right).
\end{aligned}$$

It is not implausible to assume that the above also solve the truth value systems corresponding to the remaining implementations (algebraic, Lukasiewicz, Einstein/Hamacher). This assumption is easily verified by checking that $J_a\left(\mathbf{x}^{(n)}\right) = J_l\left(\mathbf{x}^{(n)}\right) = J_e\left(\mathbf{x}^{(n)}\right) = 0$, for $n \in \{1, 2, 3\}$.

Hence, while we do not know how much of the "full" solution sets is covered by our algorithms, we can at least check which of the three known solutions $\mathbf{x}^{(1)}$, $\mathbf{x}^{(2)}$, $\mathbf{x}^{(3)}$ are discovered by each algorithm and also in what proportions. These results are summarized in the following Tables 7-11, where we list the proportion of solutions $\widehat{\mathbf{x}}$ discovered by each algorithm which are "close" (specifically, such that $\left\|\mathbf{x}^{(n)} - \widehat{\mathbf{x}}\right\| < 10^{-3}$) to each of the three known solutions.

	ST	AL	LU	HE
$\mathbf{x}^{(1)}$	0.29	0.08	0.06	0.04
$\mathbf{x}^{(2)}$	0.34	0.19	0.07	0.12
$\mathbf{x}^{(3)}$	0.27	0.22	0.19	0.22

Table 7.: Steepest Descent Algorithm

	ST	AL	LU	HE
$\mathbf{x}^{(1)}$	0.54	0.50	0.43	0.58
$\mathbf{x}^{(2)}$	0.24	0.24	0.38	0.23
$\mathbf{x}^{(3)}$	0.22	0.26	0.19	0.19

Table 8.: `fmincon`

	ST	AL	LU	HE
$\mathbf{x}^{(1)}$	0.29	0.08	0.06	0.04
$\mathbf{x}^{(2)}$	0.34	0.19	0.07	0.12
$\mathbf{x}^{(3)}$	0.27	0.22	0.19	0.22

Table 9.: `ga`

	ST	AL	LU	HE
$\mathbf{x}^{(1)}$	0.54	0.50	0.43	0.58
$\mathbf{x}^{(2)}$	0.24	0.24	0.38	0.23
$\mathbf{x}^{(3)}$	0.22	0.26	0.19	0.19

Table 10.: `patternsearch`

	ST	AL	LU	HE
$\mathbf{x}^{(1)}$	0.29	0.08	0.06	0.04
$\mathbf{x}^{(2)}$	0.34	0.19	0.07	0.12
$\mathbf{x}^{(3)}$	0.27	0.22	0.19	0.22

Table 11.: `particleswarm`

Note that in our computation of proportions we only consider algorithm results which are actual solutions of the truth value system. That is why in the Steepest Descent results the proportion columns do not add up to one; some results are not solutions at all (compare with Figure 6).

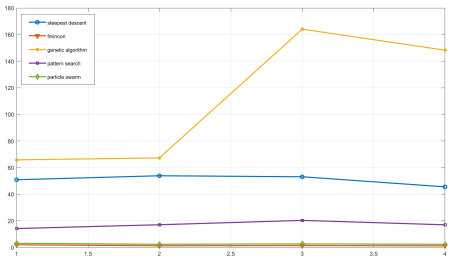


Fig. 7.: Execution times of the optimization algorithms for the various t-norm/t-conorm implementations. The horizontal axis corresponds to t-norm/t-conorm implementations (1→ standard, 2→ algebraic, 3→ Lukasiewicz, 4→ Hamacher/Einstein) and the vertical axis corresponds to execution time (per 100 runs of each algorithm) in seconds.

We complete the evaluation of the algorithms with a plot of execution times in Figure 7. It can be seen that the fastest algorithms are `fmincon` and `particleswarm`, while the slowest one is `ga`. Also, the execution times of all algo-

rithms are now significantly higher than they were for the previous two Experiment Groups; this reflects the fact that the current problem is significantly harder.

8.4. *Experiment Group D*

We now present a more extensive suite of experiments. We will study a large number of randomly generated logical systems of high dimension (i.e., involving a large number of sentences) and solve the corresponding truth value systems by global optimization. All the logical systems used in this section have the same general form; a typical example is the following system:

$$\begin{aligned} X_1 &= X_1 \wedge X_3' \wedge X_5 \\ X_4 &= X_2' \wedge X_3 \\ X_5 &= X_1 \wedge X_3 \wedge X_4' \wedge X_5'. \end{aligned} \tag{41}$$

More generally, we start with N "basic sentences" $X_1, ..., X_N$ and, having chosen some $K \leq N$, we use these to form K logical equations of the form

$$\forall k \in \{1, ..., K\} : X_{n_k} = Y_{n_k 1} \wedge ... \wedge Y_{n_k M_k}. \tag{42}$$

where $n_k \in \{1, ..., N\}$, $M_k \leq N$ and $\{Y_{n_k 1}, ..., Y_{n_k M_k}\} \subseteq \{X_1, ..., X_N, X_1', ..., X_N'\}$. In other words, each equation equates one basic sentence to a conjunction of some of the basic sentences and some of their negations.

We can form a one-to-one correspondence between systems of the form (42) and $N \times N$ matrices A with elements from $\{1, -1, 0\}$ as follows: the n-th row of A corresponds to the n-th equation of the system; $a_{nm} = 1$ means that X_m is included in the conjunction, $a_{nm} = -1$ means that X_m' is included in the conjunction and $a_{nm} = 0$ means that neither X_m nor X_m' is included; an all-zero row corresponds to a "null equation" which is omitted from the final system. As an example, the A matrix corresponding to the system (41) is

$$A = \begin{bmatrix} 1 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 1 & 1 & 0 & -1 & -1 \end{bmatrix}.$$

Consequently, to generate systems of the form (42) it suffices to generate the corresponding A matrices. To this end we sample A matrices from the following distribution:

$$\forall m, n \in \{1, ..., N\} : A_{mn} = \begin{cases} 1 \text{ with prob. } p \\ -1 \text{ with prob. } p \\ 0 \text{ with prob. } 1 - 2p \end{cases}.$$

Given an A matrix, we generate the corresponding logical system of the form (42) by the following rules:

(1) If $\forall m : A_{nm} = 0$, then no equation is generated.
(2) If $\exists m : A_{nm} \neq 0$, then the system includes the equation

$$X_n = Z_{m_1} \wedge Z_{m_2} \wedge ... \wedge Z_{m_N}$$

where

$$\forall i \in \{1, ..., N\} : Z_{m_i} = \begin{cases} X_i \text{ iff } A_{nm_i} = 1 \\ X_i' \text{ iff } A_{nm_i} = -1 \\ 1 \text{ iff } A_{nm_i} = 0 \end{cases}.$$

Of course, the 1's can be omitted from the equation.

The above remarks provide an algorithm, involving the parameters N and p, by which we can generate A matrices, corresponding logical systems and, ultimately, corresponding truth value systems.

Now, for various (N, p) combinations, we generate 100 A matrices and solve the corresponding truth value systems by the previously mentioned four algorithms (`fmincon`, `patternsearch`, `particleswarm`, `ga` – each with 100 multistarts) and the four t-norm implementations (standard, algebraic, Lukasiewicz, Hamacher/Einstein). In every case we generate 100 A matrices. Our results can be summarized as follows. First, every algorithm always finds the global minimum for every t-norm used. In addition, in Figures 8 – 11 we plot the averages (over the 100 truth value systems used) of (i) execution times and (ii) $\widetilde{R}_h$, a measure of "*root diversity*", i.e., of the number of distinct roots discovered.

Let us explain how $\widetilde{R}_h$, the root diversity measure, is computed. Suppose that we are given a specific truth value system (involving the truth values $x_1, ..., x_N$) and after T runs of an optimization algorithm we obtain the solutions $\mathbf{x}^{(1)}$, $\mathbf{x}^{(2)}$, ..., $\mathbf{x}^{(T)}$. Now some of the $\mathbf{x}^{(t)}$'s discovered will either be exactly identical, or very close to each other, simply because they are numerical estimates of the same true root of the system; on the other hand there will also exist pairs $\mathbf{x}^{(t)}$ and $\mathbf{x}^{(t')}$ which will be significantly different because they correspond to different true roots. We want to estimate the number of "essentially distinct" roots discovered by the optimization algorithm. A factor which complicates the problem is that in many cases we will have true roots which are very close to each other (this is the case when the true roots lie on a manifold of nonzero dimension).

Consequently, we cannot obtain an exact root diversity measure, especially since we do not know the true solution set. Hence we resort to an approximate measure, which is computed as follows. For a given truth value system and algorithm we perform the following computation.

$\widetilde{r}_h \leftarrow 1$
`For` $t = 2$ `to` T
  `if` $\min_{1 \leq s < t} \left\| \mathbf{x}^{(t)} - \mathbf{x}^{(s)} \right\| > h$ `then` $\widetilde{r}_h \leftarrow \widetilde{r}_h + 1$
`Next` t

After T iterations, $\widetilde{r}_h$ equals the number of roots which are at least h distant from all other roots. This is a relative measure and depends on the "resolution" parameter h. However, $\widetilde{r}_h$ gives two kinds of useful information.

(1) First, for a given truth value system, the $\widetilde{r}_h$ values obtained for different algorithms show how many roots of the same system each algorithm picks. For example, comparing Figure 1 to Figure 2, we see that the Steepest Descent Algorithm gives a denser coverage of the solution set than `fmincon` and this holds for every t-norm implementation. We cannot perform such a visual inspection in the current experiment group (since the solution set is both unknown and six-dimensional) but we can get analogous information from the $\widetilde{r}_h$ measures.
(2) Secondly, for a fixed algorithm, comparing the $\widetilde{r}_h$ values obtained for the truth value systems corresponding to different t-norm implementations, we get an idea of which implementations give a bigger solution set. For example, comparing Figure 3.a to Figure 3.b, we see that `so` achieves a denser coverage of the solution set for the Hamacher/Einstein implementation than for the algebraic one. Once again, we cannot perform such a visual inspection in the current experiment group but we can get analogous information from the $\widetilde{r}_h$ measures.

In Figures 8-11 we plot $\widetilde{R}_h$, which is the average of the $\widetilde{r}_h$ values over the 100 logical systems which we have used in the current Experiment Group. In this way we get a "condensed" view of the discovered roots diversity over all the truth value equations used.

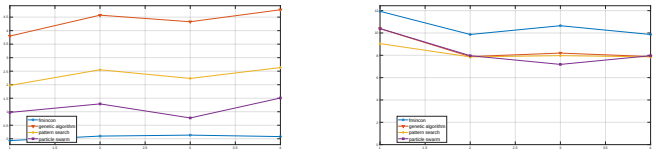


Fig. 8.: Results for $(N, p) = (7, 0.1)$: (a) Logarithm of execution time (in seconds), (b) Root diversity $\widetilde{R}_h$. In both cases the horizontal axis indicates t-norm implementation: $1 \rightarrow$ standard implementation, $2 \rightarrow$ algebraic implementation, $3 \rightarrow$ Lukasiewicz implementation, $4 \rightarrow$ Hamacher/Einstein implementation.

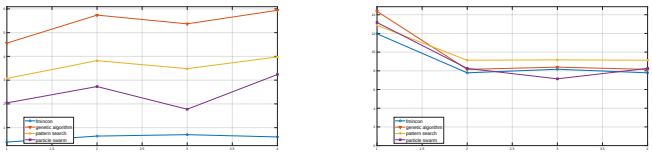


Fig. 9.: Results for $(N, p) = (10, 0.1)$: (a) Logarithm of execution time (in seconds), (b) Root diversity $\widetilde{R}_h$. In both cases the horizontal axis indicates t-norm implementation: $1 \rightarrow$ standard implementation, $2 \rightarrow$ algebraic implementation, $3 \rightarrow$ Lukasiewicz implementation, $4 \rightarrow$ Hamacher/Einstein implementation.

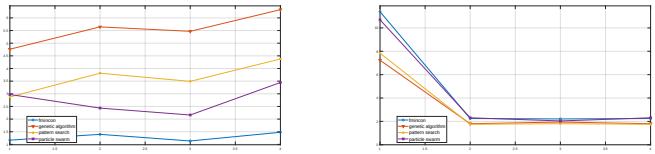


Fig. 10.: Results for $(N, p) = (10, 0.2)$: (a) Logarithm of execution time (in seconds), (b) Root diversity $\widetilde{R}_h$. In both cases the horizontal axis indicates t-norm implementation: $1 \rightarrow$ standard implementation, $2 \rightarrow$ algebraic implementation, $3 \rightarrow$ Lukasiewicz implementation, $4 \rightarrow$ Hamacher/Einstein implementation.

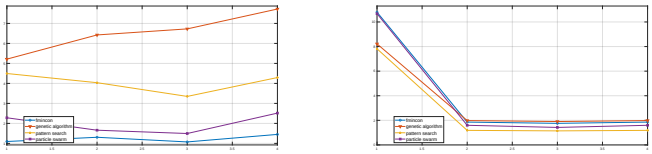


Fig. 11.: Results for $(N, p) = (10, 0.25)$: (a) Logarithm of execution time (in seconds), (b) Root diversity $\widetilde{R}_h$. In both cases the horizontal axis indicates t-norm implementation: $1 \rightarrow$ standard implementation, $2 \rightarrow$ algebraic implementation, $3 \rightarrow$ Lukasiewicz implementation, $4 \rightarrow$ Hamacher/Einstein implementation.

8.5. *Experiment Group E*

This Experiment Group is very similar to the previous one. The only difference is that we now solve logical systems of the form

$$\forall k \in \{1, ..., K\} : X_{n_k} = Y_{n_k 1} \vee ... \vee Y_{n_k M_k}. \tag{43}$$

where $n_k \in \{1, ..., N\}$, $M_k \leq N$ and $\{Y_{n_k 1}, ..., Y_{n_k M_k}\} \subseteq \{X_1, ..., X_N, X'_1, ..., X'_N\}$. In other words, each equation equates one basic sentence to a *disjunction* of some of

the basic sentences and some of their negations. The logical systems are obtained from randomly generated A matrices, in exactly the same manner as in Experiment Group D, except for the fact that we now form disjunctive equations. Our results are summarized in Figures 12 – 15.

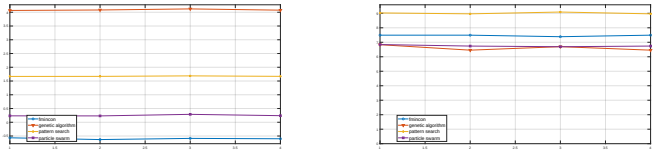


Fig. 12.: Results for $(N, p) = (7, 0.1)$: (a) Logarithm of execution time (in seconds), (b) Root diversity $\widetilde{R}_h$. In both cases the horizontal axis indicates t-norm implementation: 1 → standard implementation, 2 → algebraic implementation, 3 → Lukasiewicz implementation, 4 → Hamacher/Einstein implementation.

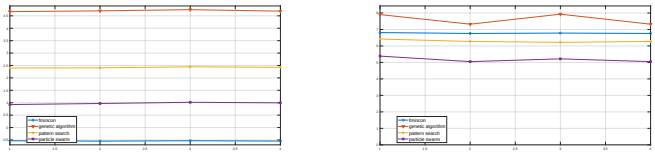


Fig. 13.: Results for $(N, p) = (10, 0.1)$: (a) Logarithm of execution time (in seconds), (b) Root diversity $\widetilde{R}_h$. In both cases the horizontal axis indicates t-norm implementation: 1 → standard implementation, 2 → algebraic implementation, 3 → Lukasiewicz implementation, 4 → Hamacher/Einstein implementation.

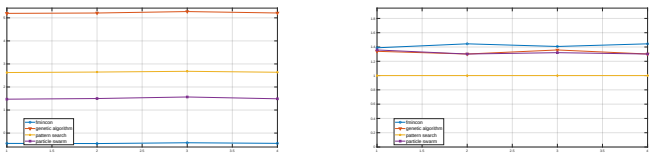


Fig. 14.: Results for $(N, p) = (10, 0.2)$: (a) Logarithm of execution time (in seconds), (b) Root diversity $\widetilde{R}_h$. In both cases the horizontal axis indicates t-norm implementation: 1 → standard implementation, 2 → algebraic implementation, 3 → Lukasiewicz implementation, 4 → Hamacher/Einstein implementation.

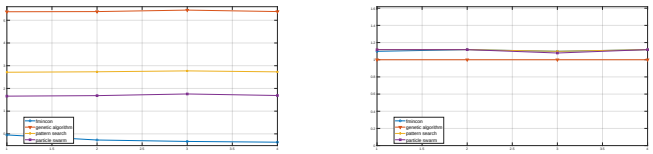


Fig. 15.: Results for $(N, p) = (10, 0.25)$: (a) Logarithm of execution time (in seconds), (b) Root diversity $\widetilde{R}_h$. In both cases the horizontal axis indicates t-norm implementation: $1 \rightarrow$ standard implementation, $2 \rightarrow$ algebraic implementation, $3 \rightarrow$ Lukasiewicz implementation, $4 \rightarrow$ Hamacher/Einstein implementation.

9. Conclusion

We have presented a framework for the formulation and study of systems of self-referential sentences and we have shown how to reduce each such system to a corresponding system of numerical truth value equations. We have proved that every such system with no more equations than unknowns and continuous $(\wedge, \vee, ')$ implementations has at least one solution in fuzzy truth values.

Furthermore we have addressed the problem of solving the truth-value system by global optimization techniques and we have established, by numerical experiments, that this approach yields useful results, even for truth-value systems of relatively high dimension.

The current work is only a first step into the study of self-referential systems. We conclude by listing some issues which we intend to investigate in the future.

(1) **Properties of the Truth-Value System**

- (a) Under what conditions does the system of truth value equations possess a unique solution?
- (b) Are the solutions (partially) ordered? Is the solution space a lattice?
- (c) What is the influence of the particular t-norm / t-conorm implementation on the structure of the solution space?
- (d) Are some solutions "better" than other? For instance, solutions on the corners of $[0, 1]^M$ are "crisper" (i.e., less fuzzy) than solutions on the boundary, which in turn are crisper than solutions in the interior. What are existence and uniqueness conditions for corner or boundary solutions?

(2) **Computational Issues**

- (a) Assuming that the solution of the truth value equations is effected by global optimization, can we introduce a bias towards crisper solutions? For example one could augment the cost function with a *fuzzy entropy* term. Conversely, one could attempt to minimize the fuzzy entropy of the solutions under the constraint that the truth value equations must be satisfied.

(b) What methods, other than global optimization, can be used to solve the truth-value system?

i. For example one can set up a dynamical system with its equilibria being the solutions of the truth-value system (this is essentiall the steepest descent approach). What are the properties of this dynamical system? For example, can we find appropriate Lyapunov functions to determine the basins of attraction of each solution?

ii. For certain t-norm/t-conorm implementations (e.g., the algebraic one) the truth-value equations are polynomial; hence their solutions can be found by homotopy methods.

References

[1] A. Arageorgis, Paradoxes and Games (in Greek), *Cogito*, vol.1, 1-23 (2004).
[2] L. Aszalós. The logic of knights, knaves, normals and mutes. *Acta Cybernetica*, **14**, 533-540, (2000).
[3] J. Barwise and J. Etchemendy. *The Liar*. Oxford Univ. Press, Oxford (1987).
[4] Jennifer Beineke and Jason Rosenhouse. *The Mathematics of Various Entertaining Subjects: The Magic of Mathematics*, **3**. Princeton University Press, Princeton (2019).
[5] Giles, Robin. "Łukasiewicz logic and fuzzy set theory." *International Journal of Man-Machine Studies*, **8**, 313-327, (1976).
[6] L. Goldstein. "This statement is not true". *Analysis*, **52**, 1-5,(1992).
[7] Grätzer, George. *General lattice theory*. Birkhauser, Basel (2002).
[8] P. Grim. Self-reference and chaos in fuzzy logic. *IEEE Trans. on Fuzzy Ssystems*, **1**, 237-253, (1993).
[9] A. Gupta. Truth and paradox. *J. of Phil. Logic*, **11**, 1-60, (1982).
[10] A. Gupta and N. Belnap. *The Revision Theory of Truth*. MIT Press, Boston (1993).
[11] P. Hajek, J. Paris and J. Shepherdson. The Liar paradox and fuzzy logic. *Preprint*, (1998).
[12] P. Hajek and J. Shepherdson. A note on the notion of truth in fuzzy logic. *An. Of Pure and Appl. Logic*, **109**, 65-69, (2001).
[13] Hanf, William. The Boolean algebra of logic. *Bulletin of the American Mathematical Society*, **81**, 587-589 (1975).
[14] Karamardian, S., ed. *Fixed Points: Algorithms and Applications*. Academic Press, New York (1977).
[15] G.I. Klir and B. Yuan. *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Prentice Hall, New Jersey (1995).
[16] S. Kripke. Outline of a theory of truth. *J. of Philosophy*, **72**, 690-716, (1975).
[17] Adam Kolany. A general method of solving Smullyan's puzzles. *Logic and Logical Philosophy*, **4**, 97–103, (1996).
[18] Lukasiewicz J., 1920, On three-valued logic. In L. Borkowski (ed.), *Selected works by Jan Łukasiewicz*, 87-88, North–Holland, Amsterdam, (1970).
[19] G. Mar and P. St. Denis. What the Liar taught Achilles. *J. of Phil. Logic*, **28**, 29-46, (1999).
[20] R.L. Martin. *The paradox of the Liar*, Ridgeview, New Haven (1970).
[21] R.L. Martin. *Recent Essays on Truth and the Liar Paradox*. Oxford Univ. Press, Oxford (1984).
[22] V. McGee. *Truth, vagueness and paradox*. Hackett, Indianapolis (1991).

[23] M. Navara. Satisfiability in fuzzy logics. *Neural Network World* , **10**, 845-858, (2000).
[24] H.T. Nguyen and E.A. Walker. *A First Course in Fuzzy Logic*, 2nd Edition, Chapman and Hall, Boca Raton, (1999).
[25] Jeff Pelletier. Using tableaux to solve knight-knave-normal problems, *Preprint*, (2011).
[26] D. Perlis. Languages with self reference I: foundations. *Art. Int.*, **25**, 301-322, (1985).
[27] D. Perlis. Languages with self reference II: knowledge, belief and modality. *Art. Int.*, **34**, 179-212, (1988).
[28] G. Priest. The logic of paradox revisited. *J. of Phil. Logic*, **13**, 153-179, (1984).
[29] A. Prior. Epimenides the Cretan. *J. of Symb. Logic*, **23**, 261-266, (1958).
[30] A. Prior. On a family of paradoxes. *Notre Dame J. of Formal Logic*, **2**, 16-32, (1961).
[31] N. Rescher. *Multi-valued Logic*. McGraw-Hill, New York (1969).
[32] A. Rieger. The Liar, the strengthened Liar and bivalence. *Erkenntnis*, **54**, 195-203, (2001).
[33] Lance J. Rips. The psychology of knights and knaves. *Cognition*, **31**, 85-116, (1989).
[34] B. Skyrms. Return of the Liar: three-valued logic and the concept of truth. *Amer. Philosoph. Quart.*, **7**, 153-161, (1970).
[35] R.M. Smullyan. *What is the Name of this Book?*. Prentice Hall, Englewood Cliffs (1981).
[36] R.M. Smullyan. *The Lady or the Tiger?*. Times Books, New York (1982).
[37] J. Tappenden. The Liar and sorites paradoxes: toward a unified treatment. *J. of Philosophy*, **90**, 551-577, (1993).
[38] B. van Fraassen. Singular terms, truth-value gaps and free logic. *J. of Philosophy*, **63**, 481- 495, (1966).
[39] B. van Fraassen. Presupposition, implication and self-reference. *J. of Philosophy*, **65**, 136-152, (1968).
[40] L.A. Zadeh. Fuzzy sets. *Information and Control*, **8**, 338-353, (1965).
[41] L.A. Zadeh. Liar's paradox and truth-qualification principle. *ERL Memo M79/34*, (1979).