Propositional Fuzzy Logics: Decidable for Some (Algebraic) Operators, Undecidable for More Complicated Ones

Mai Gehrke[1], Vladik Kreinovich[2,3], and Bernadette Bouchon-Meunier[3]

[1]Department of Mathematical Sciences
New Mexico State University
Las Cruces, NM 88003, USA
email mgehrke@nmsu.edu

[2]Department of Computer Science
University of Texas at El Paso
El Paso, TX 79968, USA
email vladik@cs.utep.edu

[3]LAFORIA-IBP
University Paris VI, Case 169
4, place Jussieu
75252 Paris Cedex 05, France
email bouchon@laforia.ibp.fr

Abstract

If we view fuzzy logic as a logic, i.e., as a particular case of a multi-valued logic, then one of the most natural questions to ask is whether the corresponding propositional logic is decidable, i.e., does there exist an algorithm that, given two propositional formulas F and G, decides whether these two formulas always have the same truth value. It is known that the simplest fuzzy logic, in which $\& = \min$ and $\vee = \max$, is decidable. In this paper, we prove a more general result: that all propositional fuzzy logics with *algebraic* operations are decidable, We also show that this result cannot be generalized further: e.g., no deciding algorithm is possible for logics in which operations are algebraic with constructive (non-algebraic) coefficients.

1 Formulation of the Problem (in Informal Terms)

1.1 Multi-Valued Logics are Very Helpful in Describing Human Reasoning

The main idea behind fuzzy logic is to replace the classical two-valued logic, as a means of describing human reasoning, by a more general *multi-valued* logic, in which, in addition to "true" ($= 1$), and "false" ($= 0$), other truth values are possible that correspond to expert's opinions like "to some extent". Usually, the interval $[0, 1]$ is used to represent different truth values. Logical operations from classical logic are then extended to the new (enlarged) set of truth values.

If we choose an interpretation of such basic operations as $\&$, $\vee$, $\neg$, etc., then we can compute the truth value of arbitrarily complicated *propositional formulas* (i.e., expressions of the type $(a \vee \neg b)\&c$).

1.2 Positive and Negative Consequences of a Multi-Valued Extension

Such an extension has both positive and negative aspects:

- On one hand, the extension to multi-valued logic is clearly *advantageous* (*positive*) because it enables us to formalize the fine distinctions between human's degrees of assuredness that classical logic is unable to capture.

- On the other hand, two-valued logic is, usually, not just formulated as an arbitrary formalism. It is based on natural *axioms* (see, e.g., [27, 13, 3]), usually, axioms of the type $F \leftrightarrow G$, meaning that some formulas F and G are *equivalent* (in the sense that for all truth values of the input variables, the resulting values of F and G coincide). For example, in classical logic, $a \vee \neg a$ is equivalent to "true". All these (seemingly natural) equivalences uniquely determine the classical logic. Thus, no matter how we extend the 2-valued operations, at least some of these "axioms" will be false. So, for each multi-valued extension of classical logic, there always exists pairs of formulas F and G that are equivalent in classical logic, but that are no longer equivalent in the new extended logic. This fact was emphasized, in effect, by Zadeh himself: in his pioneer papers, starting from [33], he emphasizes that in fuzzy logic, e.g., $a \vee \neg a$ is no longer equivalent to "true".

  Since the axioms (equivalences) of classical logic have been chosen as natural (intuitively correct) statements, the fact that in multi-valued logic, we have to violate some of them can be viewed as a *negative* feature of fuzzy logic. (It not only *can* be viewed as a negative feature, it actually *is* viewed by some researchers as a negative feature; see, e.g., [12].)

Zadeh emphasized that this "negative" consequence of the extension may not be so negative after all, in the sense that for some pairs of previously equivalent formulas, their non-equivalence may be intuitively even more acceptable than their equivalence. For example, for "fuzzy" properties like $Young(x)$, $Young(x) \vee \neg Young(x) =$"true" would mean that for every person x, either we are absolutely sure that this person is young, or we are absolutely sure that this person is not young; this is, of course, intuitively, not the case: we view many people as not being absolutely young and as not being absolutely not young (this is a typical view that a person has of him/herself).

In this sense, Zadeh's original non-equivalence is not such a negative feature after all. On the other hand, there are other non-equivalences whose violation may be much less intuitive and will be, thus, rather a negative feature of the resulting logic.

1.3 It is Desirable to Keep the Balance Between Positive and Negative Consequences

In view of this possibility, it is desirable to choose a multi-valued extension that keeps a *balance* between the positive and negative consequences of the extension: i.e., only those equivalence should be violated for which this violation can be (at least to some extent) intuitively justified.

1.4 For Keeping This Balance, Decidability is Desirable

To be able to judge whether in a given extension, all violations of classical equivalences are indeed intuitively justified, we need to be able, first, to tell which equivalences are violated and which are not.

In other words, it is desirable to have an algorithm that, given an extension and a pair of formulas, would tell:

- whether the given two formulas F and G are indeed *equivalent* in this extension (i.e., their truth values coincide for all possible truth values of the inputs), or
- the two given formulas are *not equivalent* in the sense that for some truth values of inputs, the truth values of F and G are different.

If it is not possible to have such a universal algorithm, we would at least like to have such an algorithm for some specific extensions (i.e., for some specific logics).

In logical terms, the existence of such an algorithm for a given logic is called its *decidability*. So, in these terms, our problem is to analyze whether propositional fuzzy logics are decidable.

1.5 For Decidable Logics, It Is Important to Know the Running Time of the Deciding Algorithm

If a logic is, in principle, decidable (i.e., if such an algorithm exists after all), then the next natural question is: how practical is the deciding algorithm? If an algorithm takes years to decide a simple problem, then it is better than having no algorithm at all, but still not of much practical use. From this viewpoint, it is desirable to analyze the running time of the deciding algorithms.

1.6 What is Known and What we Are Planning to Do

1.6.1 What is Known

It is known [15] that the simplest propositional fuzzy logic, in which $\&$ = min and $\vee$ = max, is decidable. The algorithm proposed in [15] requires 3^n computational steps for formulas with n variables.

In [30], it is shown that this problem belong to the class of complicated problems (called *NP-hard*; see, e.g., [14]), about which most computer scientists believe that no algorithm can solve all these problems faster than in exponential time. In this sense, the 3^n time algorithm seems to be asymptotically the best possible.

A similar algorithm has been proposed in [24] for a natural interval-valued generalization of the simplest propositional fuzzy logic. This algorithm requires time 4^n.

1.6.2 What We Are Planning to Do

In this paper, we *describe* a class of *extensions* for which the *deciding algorithm* still *exists*. As a particular case, we will get decidability of propositional fuzzy logics described by most operations that have been actually used. We will show that *our result is practically final* in the sense that for a slightly more general class of extensions, no deciding algorithm is possible.

1.6.3 A word of warning

Although in principle, decidability of the simplest propositional fuzzy logic and of its interval extension follow from our general decidability result, our general algorithm will not replace the algorithms from [15] and [24], because the running time of our general algorithm is mush worse (at least doubly exponential for our algorithm as opposed to exponential time for algorithms from [15, 24]).

2 The Main Decidability Result

2.1 Definitions Necessary for Formulating the Problem: Language, Formulas, Logics, etc.

Definition 1. *By a propositional language (or simply language, for short), we will mean a triple (V, O, ar), where:*

- *$V = \{v_1, \ldots, v_n, \ldots\}$ is a denumerable set. Its elements will be called variables.*
- *$O = \{o_1, \ldots, o_{\text{nop}}\}$ is a finite set. Its elements will be called symbols for propositional logical operations, or simply operation symbols, for short.*
- *$ar : O \to N$ is a function that assigns to every operation symbol $o \in O$ a non-negative number $ar(o)$ called its arity. Operation symbols with arity 0 are will also be called symbols for constants.*

Comments.

- In traditional propositional logic, the set O consists of the operation symbols &, $\vee$, and $\neg$, for which $ar(\&) = ar(\vee) = 2$ and $ar(\neg) = 1$. In principle, we can add implication $\to$ with $ar(\to) = 2$, and/or constant symbols T ("true") and F ("false") for which $ar(T) = ar(F) = 0$.
- For each propositional language, we can recursively define the notion of a *propositional formula* (since in this paper, we will only consider propositional formulas, we will also call them simply *formulas*, for short):

Definition 2. *Let a propositional language $L = (V, O, ar)$ be given. Then, the notion of a propositional formula (or simply formula, for short) is defined as follows:*

- *every variable $v \in V$ is a formula;*
- *if $F_1, \ldots, F_n$ are formulas, and o is an operation symbol with $ar(o) = n$, then the expression $o(F_1, \ldots, F_n)$ is a formula.*

Comments.

- If the list O of logical operations contains standard logical operations (&, $\vee$, $\neg$, $\to$, etc.), then we get the standard definition of a propositional formula,
- If the language L contains symbols for constants, then, according to the the second part of the definition, each constant symbol is a formula.

Definition 3. *By a propositional multi-valued logic (or simply propositional logic, for short), we mean a triple $\mathcal{L} = (L, T, \hat{})$, where:*

- *$L = (V, O, ar)$ is a propositional language.*
- *T is a set; elements of this set will be called truth values.*
- *$\hat{}$ is mapping that maps every operation symbol $o \in O$ with arity $n = ar(o)$ into a function $\hat{o} : T^n \to T$. The resulting function $\hat{o} : T^n \to T$ maps every tuple of n values $(t_1, \ldots, t_n)$, $t_i \in T$, into a value $\hat{o}(t_1, \ldots, t_n) \in T$; this function will be called a logical operation corresponding to the operation symbol o.*

Comments.

- In 2-valued logic, $T = \{0, 1\}$, and the logical operations $\hat{\&}$, $\hat{\vee}$, and $\hat{\neg}$ corresponding to operation symbols $\&$, $\vee$, and $\neg$ are defined by their standard truth tables.
- In traditional fuzzy logic, $T = [0, 1]$. As logical operation corresponding to the operation symbols $\&$, $\vee$, and $\neg$, we can take, e.g., $\hat{\&} = \min$, $\hat{\vee} = \max$, and $\hat{\neg}(a) = 1 - a$ (for other possible operations, see, e.g, [20, 25]).
- In interval-valued fuzzy logic, T is the set of all subintervals of the interval $[0, 1]$.
- If the language L contains a constant o, then $\hat{o}$ is simply a constant $\hat{o} \in T$.

Definition 4. *Let a propositional logic $\mathcal{L}$ be given.*

- *By a truth assignment, we mean a mapping $t : V \to T$ that maps every variable $v \in V$ into a truth value $t(v)$.*
- *For every formula F and for every truth assignment t, we can define the truth value $t(F)$ of this formula F by induction over the length of the formula:*

  Induction base: For formulas of length 1, truth value is defined as follows:
  - *if F is a variable v, then $t(F) = t(v)$.*
  - *if F is a constant o, then $t(F) = \hat{o}$;*

  Induction step. If $t(F)$ is defined for all formulas of length k, and F is a formula of length k, then this formula F is of the type $o(F_1, \ldots, F_n)$ for some operation symbol f, and for some formulas F_i of length $< k$. Then, by induction assumption, for all formulas F_i, the truth value $t(F_i)$ is already defined, and we can define $t(F)$ as $\hat{o}(t(F_1), \ldots, t(F_n))$.

Definition 5. *We say that two propositional formulas F and G are equivalent in a given propositional logic $\mathcal{L}$, if for every truth assignment t, these formulas get the same truth value: $t(F) = t(G)$.*

2.2 Definitions Necessary for Formulating our Main Result

Comment. Our goal is to show that for a large class of operations, there exists an algorithm that, given a propositional logic $\mathcal{L}$ and two formulas F and G, decides whether the two given formulas are equivalent in the given logic or not. To describe this result, let us describe the corresponding class of logics.

Definition 4.

- *Let n be a positive integer, and R^n be an $n-$dimensional vector space. A set $S \subseteq R^n$ is called semi-algebraic (or, simply algebraic) if this set is a union of finitely many sets $S_1 \ldots, S_p$, each of which consists of all tuples that satisfy one or several conditions of the types $P_j(x_1, \ldots, x_n) = Q_j(x_1, \ldots, x_n)$, $P_k(x_1, \ldots, x_n) > Q_k(x_1, \ldots, x_n)$, or $P_l(x_1, \ldots, x_n) \geq Q_l(x_1, \ldots, x_n)$, for some polynomials P_i and Q_i with rational coefficients.*
- *Let $S \subseteq R^n$ be an algebraic set, and let $f : S \to R$ be a function.*
  - *By a graph of the function f, we mean the set*
  $$\{(x_1, \ldots, x_n, f(x_1, \ldots, x_n)) \,|\, (x_1, \ldots, x_n) \in S\}.$$
  - *A function $f : S \to R$ is called algebraic if its graph is an algebraic set.*
- *A propositional logic $\mathcal{L}$ is called algebraic if the following two conditions hold:*
  - *its set of truth values T is an algebraic subset of R^l (for some l), and*
  - *all logical operations $\hat{o}$, $o \in O$, are algebraic functions.*

2.3 Our Definitions are Sufficiently General (i.e., Cover Most Fuzzy Logical Operations)

Let us show that most propositional logics that have been used in fuzzy approach are indeed algebraic logics.

2.3.1 Standard Sets of Truth Values are Algebraic

- The set $T = [0, 1]$ consists of all real numbers that satisfy two inequalities $t \geq 0$ and $1 \geq t$. Thus, $[0, 1]$ is an algebraic set.
- The set I of all subintervals of the interval $[0, 1]$ is also an algebraic set, because it can be described as
$$\{(\underline{a}, \overline{a}) \,|\, 0 \leq \underline{a} \leq \overline{a} \leq 1\}.$$

2.3.2 Most Frequently Used t-Norms and t-Conorms (i.e., "And" and "Or" Operations) are Algebraic

- The graph of the function $a \hat{\&} b = \min(a, b)$ is a union of two sets:

  $$S_1 = \{(x_1, x_2, x_3) \,|\, x_2 \geq x_1 \,\&\, x_3 = x_1\}$$

  and

  $$S_2 = \{(x_1, x_2, x_3) \,|\, x_1 \geq x_2 \,\&\, x_3 = x_2\}.$$

  Thus, min is an algebraic function.

- Similarly, maximum is an algebraic function.

- Clearly, operations $a \hat{\&} b = a \cdot b$ and $a \hat{\vee} b = a + b - a \cdot b$, proposed by Zadeh in [33] under the names of "algebraic product" and "algebraic sum", are also algebraic functions.

- The graphs of the "bounded difference" ("bold intersection")

  $$a \hat{\&} b = \max(0, a + b - 1)$$

  and "bounded sum" ("bold union")

  $$a \hat{\vee} b = \min(a + b, 1)$$

  [16] are also algebraic functions: e.g., the graph of the bold intersection can be represented as a union of two sets:

  $$S_1 = \{(x_1, x_2, x_3) \,|\, x_1 + x_2 - 1 \geq 0 \,\&\, x_3 = x_1 + x_2 - 1\}$$

  and

  $$S_2 = \{(x_1, x_2, x_3) \,|\, x_1 + x_2 - 1 \leq 0 \,\&\, x_3 = 0\}.$$

- Hamacher's operations

  $$a \hat{\&} b = \frac{a \cdot b}{k + (1 - k)(a + b - a \cdot b)} \text{ and } a \hat{\vee} b = \frac{(1 - k)a \cdot b + k(a + b)}{a \cdot b + k},$$

  proposed originally in [17, 18], are algebraic for algebraic numbers k: e.g., the graph of the first operation can be represented as

  $$\{(x_1, x_2, x_3) \,|\, [k + (1 - k)(x_1 + x_2 - x_1 \cdot x_2)] \cdot x_3 = x_1 \cdot x_2\}.$$

- Operations

  $$a \hat{\&} b = \frac{a \cdot b}{\max(a, b, e)} \text{ and } a \hat{\vee} b = \frac{a + b - a \cdot b - \min(a, b, 1 - e)}{\max(1 - a, 1 - b, e)},$$

  proposed by Dubois and Prade [10, 11], are algebraic for algebraic numbers e.

- Yager's operations
$$a\hat{\&}b = \min(1, (a^p + b^p)^{1/p})$$
and
$$a\hat{\vee}b = 1 - \min(1, ((1-a)^p + (1-b)^p)^{1/p})$$
from [32] are algebraic for rational p: e.g., for integer p, the graph of the first operation can be represented as the union of two sets:
$$S_1 = \{(x_1, x_2, x_3) \,|\, x_1^p + x_2^p \geq 1 \,\&\, x_3 = 1\}$$
and
$$S_2 = \{(x_1, x_2, x_3) \,|\, x_1^p + x_2^p \leq 1 \,\&\, x_3^p = x_1^p + x_2^p\}.$$
- Dombi's operations [9]
$$\frac{1}{1 + [(a^{-1} - 1)^p + (b^{-1} - 1)^p]^{1/p}}$$
are algebraic for rational p.
- Operations
$$a\hat{\vee}b = \max(0, a^p + b^p - 1)^{1/p}$$
and
$$a\hat{\&}b = 1 - \max[0, (1-a)^{-p} + (1-b)^{-p} - 1]^{1/p},$$
that were originally proposed in [28] for non-fuzzy purposes, and used for fuzzy logic in [19], are algebraic for rational p.

2.3.3 Other Fuzzy Logical Operations are Also Algebraic

- Traditional fuzzy *negation* $\hat{\neg}(a) = 1 - a$ is an algebraic operation.
- Most frequently used *fuzzy implication* operations $\hat{\rightarrow}$ [19] are algebraic:
  - $\min(1, b/a)$ (proposed by Gaines);
  - $\min(1, b/a, (1-a)/(1-b))$ (a modification of Gaines' proposal);
  - $\min(1, 1 - a + b)$ (Lukaciewicz);
  - $1 - a + a \cdot b$ (Kleene – Dienes – Lukaciewicz);
  - $\max(1 - a, b)$ (Kleene –Dines);
  - $\max(1 - a, \min(a, b))$ (Zadeh);
  - $\min(\max(1 - a, b), \max(1 - b, a, \min(1 - a, b)))$ (Willmot).
- *Modifiers* a^2 and $\sqrt{a}$, proposed originally by Zadeh [34] to represent "very" and "slightly", are also algebraic operations:
  - the graph of a^2 is clearly algebraic;
  - the graph of $\sqrt{a}$ can be represented as $\{(x_1, x_2) \,|\, x_2^2 = x_1\}$.
- *Interval-valued operations* [24] are algebraic.

2.4 Formulation of the Main Result

PROPOSITION 1. *There exists an algorithm that, given an arbitrary algebraic propositional logic $\mathcal{L}$ and arbitrary two propositional formulas F and G, checks whether the given formulas F and G are equivalent in the given logic $\mathcal{L}$.*

2.5 Proof of the Main Result

Let $\mathcal{L}$, F, and G be given. Let us show how to check whether the formulas F and G are equivalent in the logic $\mathcal{L}$.

Let n be the total number of variables in the formulas F and G. Without loss of generality, we can name these variables $v_1, \ldots, v_n$. Then, the condition that the formulas F and G are equivalent can be represented as

$$\forall v_1 \ldots \forall v_n (t(F) = t(G)).$$

By definition of $t(F)$ and $t(G)$, each of these terms is a composition of several logical operations. Since every logical operation is an algebraic function, each of the terms $t(F)$ and $t(G)$ is a composition of algebraic functions. Thus, the formula that we want to check is obtained from the elementary formula (equality of two algebraic terms) by using quantifiers that run over all real numbers.

There exists an algorithm (proposed originally by Tarski [31]) that checks, for each such formula, whether it is true or not (see also [29] and [2]). Thus, by applying Tarski's algorithm to the resulting formula, we will be able to check whether the formulas F and G are equivalent or not. Q.E.D.

2.6 Corollaries of the Main Result

Definition 5. *A propositional logic $\mathcal{L}$ is called decidable if there exists an algorithm that, given arbitrary two formulas F and G, checks whether the given formulas F and G are equivalent (in this logic $\mathcal{L}$).*

COROLLARY. *Every algebraic propositional logic $\mathcal{L}$ is decidable.*

Comment. As particular cases of this Corollary, we can conclude that:

- the simplest propositional fuzzy logic (with min and max) is decidable;
- the corresponding interval-valued logic is also decidable.

Thus, *decidability results from* [15] *and* [24] *follow from our main result.*

Our main result is, however, *more general* that results from [15] and [24], for two reasons:

- First, our result is applicable not only to the simplest propositional fuzzy logics, but also to other versions of fuzzy logic such as algebraic sum and union, bold sum and union, and to other operations described above.

- Second, our result is applicable not only to the formulas that only contain $\&$, $\vee$, and $\neg$: we also get decidability for formulas that may contain implication, modifiers, and other operations (as long as these other operations are algebraic).

3 Our Main Result is the Most General Decidability Result for Propositional Fuzzy Logic

3.1 Motivations

We have shown that all propositional logics with *algebraic* operations are decidable. Is it possible to further extend this class of operations and still preserve decidability?

In order to find out how our class can be extended, let us recall how we restricted ourselves to this class. Our definition of an algebraic operation was slightly different from the standard definitions from calculus: e.g., according to our definition, not all linear functions $y = kx$ (and even, not all linear functions with computable coefficients) are algebraic: only those linear functions for which the coefficient k is an algebraic number. In view of this remark, we can envision two consequent extensions of the above class:

- First, we can have a natural (tiny) extension: namely, we can allow algebraic functions with computable (and not necessarily algebraic) coefficients.
- After that, if we still get decidability, we can try to add different non-algebraic functions.

It turns out that even the first (tiny) extension — to algebraic functions with computable coefficients — makes decidability impossible. In this sense, our main result is the most general one.

Comment. This negative result also explains why we have chosen such a strange definition of algebraic function: because for a more general (and more natural) definition, we do not have decidability.

3.2 Definitions and the Non-Extension Result

Comment. A real number is called *computable* if there exists an algorithm that computes this number with an arbitrary accuracy (see, e.g., [5, 7, 21, 4, 6, 1]):

Definition 6. *A real number x is called constructive if there exists an algorithm (program) that transforms an arbitrary integer k into a rational number x_k that is $2^{-k}-$close to x. It is said that this algorithm computes the real number x.*

Comment. Let us now describe the notion of an algebraic function with computable coefficients.

The main idea behind our definition is that, e.g., the function $f(x) = kx$ with computable coefficient x can be viewed as an algebraic function of *two* variables, into which we have substituted a computable value instead of one of the variables.

In general, a function $f(x_1, \ldots, x_n)$ with coefficients $a_1, \ldots, a_m$ can be viewed as a function of $n + m$ variables in which we have substituted computable values a_i instead of m variables. So, we get the following definition:

Definition 7. *By an algebraic function with computable coefficients, we mean a function of the type $f(x_1, \ldots, x_n) = F(x_1, \ldots, x_n, a_1, \ldots, a_m)$, where $F(x_1, \ldots, x_n, \ldots, x_{n+m})$ is an algebraic function in the sense of Definition 4, and $a_1, \ldots, a_m$ are computable numbers.*

Definition 8. *A propositional logic $\mathcal{L}$ with the set of truth values $T = [0, 1]$ is called almost algebraic if all its operations are algebraic functions with computable coefficients.*

Comment. When we say that we are given an almost algebraic propositional logic, we mean that for each of its operation f, we are given both the corresponding algebraic function F, and the algorithms for computing the coefficients a_i.

PROPOSITION 2. *No algorithm is possible, that, given an arbitrary almost algebraic propositional logic $\mathcal{L}$ and arbitrary two formulas F and G, checks whether the given formulas F and G are equivalent in the given logic $\mathcal{L}$.*

3.3 Proof of the Non-Extension Result

We will prove this result by reduction to a contradiction. Namely, let us assume that such an algorithm is possible. We will show that the contradiction occurs when we apply this hypothetic algorithm to some reasonably simple operations $\hat{\&}$, $\hat{\vee}$, and $\hat{\neg}$.

Namely, let α be a non-negative computable real number. Let us define the following propositional logic:

- Its language consists of three operation symbols $O = (\{\&, \vee, \neg\})$.
- The set T of truth values is the interval $[0, 1]$.
- The operations $\hat{o}$ are as follows:
  - $a \hat{\&} b = \min(a, b)$;
  - $\hat{\neg}(a) = 1 - a$; and

- $a \hat{\vee} b = \max(a, b)$ for $a \geq \alpha$ or $b \geq \alpha$, and $\min(a + b, \alpha)$ for $a, b \leq \alpha$. (This operation can be described as $\max\{\min(a + b, \alpha), a, b\}$ and is thus algebraic in terms of a, b, and α, and is, therefore, almost algebraic for constructive α).

Let us take $a \vee b$ as F and $\neg(\neg a \& \neg b)$ as G. Then, as one can easily check, formulas F and G are equivalent if and only if $\alpha = 0$. Thus, if we have an algorithm for deciding whether formulas are equivalent in a given almost algebraic propositional logic, we would this get an algorithm for deciding whether a given computable real number is equal to 0 or not.

Such an algorithm is, however, impossible (see, e.g., [21]). Indeed, if such algorithm was possible, then for every algorithmic function f from natural numbers to natural numbers we would be able to form a computable real number x defined as follows:

- $x_k = 0$ if $\forall n_{\leq k}(f(n) = 0)$ and
- $x_k = 2^{-n_{\min}}$ if $\exists n_{n \leq k}(f(n) \neq 0)$, where $n_{\min}$ is the smallest $n \leq k$ for which $f(n) \neq 0$.

It is easy to show that this sequence x_k indeed computes a real number, and that for this real number x, $x = 0$ if and only if $\forall n(f(n) = 0)$.

- Thus, if we were able to check whether a given computable real number is equal to 0 or not, we would thus be *able to check*, for a given computable function f from natural numbers to natural numbers, *whether* this *function is always 0* or not.
- However, it is known (see, e.g., [22, 23, 26]) that there exists *no* algorithm for deciding whether a program (to be more precise, a program that always finishes its computations) always returns 0. In other words, *there exists no algorithm*, that, given an algorithmic (everywhere defined) function $f(n)$ from natural numbers to natural numbers would *check whether* $\forall n(f(n) = 0)$.

This contradiction shows that our initial assumption — that there exists an algorithm that decides, for a given almost algebraic propositional logic $\mathcal{L}$ and for two given formulas F and G, whether the given formulas F and G are equivalent in $\mathcal{L}$ — is false. The proposition is proven.

4 A Word of Warning: The General Algorithm Does not Replace the Previously Known Deciding Algorithms

As we have mentioned, in principle, the decidability of the simplest propositional fuzzy logic and of its interval extension follows from our general decidability

result, However, this does not mean that our general algorithm can replace the algorithms from [15] and [24]:

- Indeed, our algorithm uses Tarksi's algorithm, and it is known that for the decidability problem solved by Tarski's algorithm, for some cases, at least doubly exponential time is necessary (i.e., time $\geq 2^{2^n}$; see, e.g., [8]). Thus, the running time of our algorithm is at least doubly exponential.
- On the other hand, algorithms from [15] and [24] require at worst exponential time (3^n or, correspondingly, 4^n).

Acknowledgments. This work was partially supported by NASA Grant No. NAG 9-757. This work was done when V. Kreinovich was an invited professor at Laforia, University Paris VI, Summer 1996.

References

[1] O. Aberth, *Precise numerical analysis*, Wm. C. Brown Publishers, Dubuque, Iowa, 1988.

[2] V. I. Arnold, *Geometrical methods in the theory of ordinary differential equations*, Springer-Verlag, N.Y., 1983.

[3] J. Barwise (ed.), *Handbook of Mathematical Logic*, North-Holland, Amsterdam, 1977.

[4] M. J. Beeson, *Foundations of constructive mathematics*, Springer-Verlag, N.Y., 1985.

[5] E. Bishop, *Foundations of Constructive Analysis*, McGraw-Hill, 1967.

[6] E. Bishop, D. S. Bridges, *Constructive Analysis*, Springer, N.Y., 1985.

[7] D. S. Bridges, *Constructive Functional Analysis*, Pitman, London, 1979.

[8] J. H. Davenport, J. Heintz, "Real quantifier elimination is doubly exponential", *Journal of Symbolic Computations*, 1988, Vol. 5, No. 1/2, pp. 29–35.

[9] J. Dombi, "A general class of fuzzy operators, the De Morgan class of fuzzy operators and fuzziness measures induced by fuzzy operators", *Fuzzy Sets and Systems*, 1982, Vol. 8, pp. 149–163.

[10] D. Dubois and H. Prade, *Fuzzy sets and systems: theory and applications.* Academic Press, N.Y., London, 1980.

[11] D. Dubois and H. Prade, "New results about properties and semantics of fuzzy set-theoretic operators", In: *Fuzzy sets: theory and applications to policy analysis and information systems* (P.P. Wang, S.K. Chang, eds). Plenum Press, New York, 1980, pp. 59–75.

[12] C. Elkan, "The paradoxical success of fuzzy logic", in *Proceedings of AAAI-93, American Association for Artificial Intelligence 1993 Conference*, pp. 698–703.

[13] H. B. Enderton. *A mathematical introduction to logic*. Academic Press, N.Y., 1972.

[14] M. Garey and D. Johnson, *Computers and intractability: a guide to the theory of NP-completeness*, Freeman, San Francisco, 1979.

[15] M. Gehrke, C. Walker, and E. A. Walker, *A mathematical setting for fuzzy logic*, Preprint, New Mexico State University, Department of Mathematical Sciences, 1995 (available from the authors at `elbert@nmsu.edu`).

[16] R. Giles, "Lukaciewicz logic and fuzzy set theory", *Intern. J. Man–Machine Studies*, 1976, Vol. 8, pp. 313–327.

[17] H. Hamacher, "Uber logische Verpunfungen Unscarfer Aussagen unde Deren Zugenhorige Bewertungs-funktionen", In: *Progress in Cybernetics and Systems Research*, 1975, Vol. 3 (R. Trappl, G. J. Klir and L. Ricciardi, Eds.) Hemisphere, N. Y., pp. 276–287.

[18] H. Hamacher. *Uber logische Aggregationen nicht-binar expliziter Entscheidungskrieten*. Frankfurt/Main, 1978.

[19] G. J. Klir and T. A. Folger. *Fuzzy sets, uncertainty and information*. Prentice Hall, Englewood Cliffs, NJ, 1988.

[20] G. Klir and B. Yuan, *Fuzzy sets and fuzzy logic: theory and applications*, Prentice Hall, Upper Saddle River, NJ, 1995.

[21] B. A. Kushner, *Lectures on Constructive Mathematical Analysis*, Translations of Mathematical Monographs, Vol. 60, American Mathematical Society, Providence, RI, 1984.

[22] L. R. Lewis and C. H. Papadimitriou, *Elements of the theory of computation*, Prentice-Hall, Englewood Cliffs, NJ, 1981.

[23] J. C. Martin, *Introduction to languages and the theory of computation*, McGraw-Hill, N.Y., 1991.

[24] H. T. Nguyen, O. M. Kosheleva, and V. Kreinovich, "Is the success of fuzzy logic really paradoxical? Or: Towards the actual logic behind expert systems", *International Journal of Intelligent Systems*, 1996, Vol. 11, No. 5, pp. 295–326.

[25] H. T. Nguyen and E. A. Walker, *A First Course in Fuzzy Logic*, CRC Press, Boca Raton, Florida, 1996 (to appear).

[26] C. H. Papadimitriou, *Computational Complexity*, Addison Wesley, San Diego, 1994.

[27] J. R. Schoenfield. *Mathematical logic*. Addison-Wesley, 1967.

[28] B. Schweizer and A. Sklar, "Associative functions and statistical triangle inequalities", *Publicationes Mathematicae Debrecen*, 1961, Vol. 8, pp. 169–186.

[29] A. Seidenberg, "A new decision method for elementary algebra", *Annals of Math.*, 1954, Vol. 60, pp. 365–374.

[30] S. Sudarsky, "Fuzzy satisfiability", *Proc. of the Third International Conference on Industrial Fuzzy Control and Intelligent Systems, Dec. 1-3, 1993*, Houston, TX, pp. 228-231.

[31] A. Tarski, *A decision method for elementary algebra and geometry*, 2nd ed., Berkeley and Los Angeles, 1951.

[32] R. R. Yager, "On a general class of fuzzy connectives", *Intl. Journal of General Systems*, 1980, Vol. 4, pp. 235–242.

[33] L. A. Zadeh, "Fuzzy Sets", *Inform. and Control*, 1965, Vol. 8, pp. 338–353.

[34] L. A. Zadeh, "Outline of a new approach to the analysis of complex systems and decision processes", *IEEE Transactions on Systems*, Man and Cybernetics, 1973, Vol. 3, pp. 28–44.

[35] L. A. Zadeh, "The concept of a linguistic variable and its application to approximate reasoning, Part 1", *Information Sciences*, 1975, Vol. 8, pp. 199–249.