Applied Mathematical Sciences, Vol. 7, 2013, no. 103, 5129 - 5138
HIKARI Ltd, www.m-hikari.com
http://dx.doi.org/10.12988/ams.2013.36317

From Binary Logic Functions to Fuzzy Logic Functions

Omar Salazar, Jairo Soriano and Humberto Serrano

Universidad Distrital Francisco Jose de Caldas, Bogota, Colombia
osalazarm@correo.udistrital.edu.co, jairosoriano@udistrital.edu.co,
jhserrano@udistrital.edu.co

Abstract

It is known that any binary logic function (in binary logic) can be represented by at least one logic formula, however this is not always true for a ternary logic function (in ternary logic). A ternary logic function can be represented by at least one logic formula if and only if it satisfies a mathematical condition which is called regularity. Also, it is known that a fuzzy logic formula (in standard fuzzy logic) can be obtained by means of truth tables in ternary logic. This paper presents a method destined for extending a binary logic function into one (or more) ternary logic function(s) representable by at least one logic formula, and therefore, to obtain fuzzy logic formulas from a binary logic function.

Mathematics Subject Classification: 03B52, 06D72, 08A72, 94D05

Keywords: fuzzy logic; binary logic; ternary logic; logic formulas

1 Introduction

Logic formulas in binary logic are important when designing systems based on this type of logic (e.g. digital circuits). Since truth values are "0" (false) and "1" (true), there are practical methods to obtain, manipulate and optimize logic formulas by using truth tables, e.g., Karnaugh maps and Quine-McCluskey method. It is known [3] that a logic formula represents one logic

function, but a logic function can be represented by several logic formulas. *Boole's expansion theorem* [3, pp. 98] represents one way to guarantee that there exists at least one logic formula for a logic function. Therefore, binary logic functions are representable by logic formulas.

In fuzzy logic the process to elaborate a truth table is not a direct task. Mainly because its truth values are in $[0,1]$. Gehrke *et al.* [1] presented a practical method to elaborate truth tables in standard fuzzy logic, which uses "and" ($\wedge$), "or" ($\vee$) and "not" ($'$) operations given by $a \wedge b = \min\{a,b\}$, $a \vee b = \max\{a,b\}$ and $a' = 1 - a$ for all $a, b \in [0,1]$. Their method uses Kleene's ternary logic system [4] where truth tables for standard fuzzy logic are made by means of truth tables in ternary logic. Gehrke *et al.* demonstrated that propositional logics from standard fuzzy logic and Kleene's ternary logic are the same, and therefore their logic formulas are equivalent. Unlike binary logic, in ternary logic not all ternary logic functions are representable by a logic formula. Mukaidono [6] established necessary and sufficient conditions in order that a ternary logic function could be represented by at least one logic formula. The key concept is the *regularity*, which is an extension to several variables of the same concept given by Kleene [4], of a ternary logic function. The main purpose of this paper is to present a method destined for extending a binary logic function into one (or more) ternary logic function(s) representable by at least one logic formula, and therefore, to obtain fuzzy logic formulas from a binary logic function.

2 Binary Logic Functions and Ternary Logic Functions

A binary logic function β of n variables $x_1,\ldots,x_n$ is defined by an application $\beta : B^n \mapsto B$, $B = \{0,1\}$, where "0" is "false" and "1" is "true". The (total) order in B is $0 \leq 1$. Any β can be represented by at least one well-formed logic formula. In particular, *Boole's expansion theorem* [3, pp. 98] establishes that any β can be written for all $\langle x_1,\ldots,x_n\rangle \in B^n$ as follows:

$$\beta(x_1,\ldots,x_n) = (x_1' \wedge \beta(0,\ldots,x_n)) \vee (x_1 \wedge \beta(1,\ldots,x_n)) \tag{1}$$

$$= (x_1' \vee \beta(1,\ldots,x_n)) \wedge (x_1 \vee \beta(0,\ldots,x_n)). \tag{2}$$

Operations "and" ($\wedge$), "or" ($\vee$) and "not" ($'$) are given in Table 1(a). If (1) and (2) are applied recursively to β (see [3] for more details), we will get:

$$\begin{aligned}\mathrm{D}(\beta)(x_1,\ldots,x_n) &= (\beta(0,\ldots,0)\wedge x_1'\wedge\cdots\wedge x_n')\vee(\beta(0,\ldots,1)\wedge x_1'\wedge\cdots\wedge x_n)\vee\cdots\\ &\quad\vee(\beta(1,\ldots,0)\wedge x_1\wedge\cdots\wedge x_n')\vee(\beta(1,\ldots,1)\wedge x_1\wedge\cdots\wedge x_n),\\ \mathrm{C}(\beta)(x_1,\ldots,x_n) &= (\beta(0,\ldots,0)\vee x_1\vee\cdots\vee x_n)\wedge(\beta(0,\ldots,1)\vee x_1\vee\cdots\vee x_n')\wedge\cdots\\ &\quad\wedge(\beta(1,\ldots,0)\vee x_1'\vee\cdots\vee x_n)\wedge(\beta(1,\ldots,1)\vee x_1'\vee\cdots\vee x_n').\end{aligned}$$

Table 1: (a) Binary logic operations "or" ($\vee$), "and" ($\wedge$) and "not" ($'$). (b) Ternary logic operations "or" ($\vee$), "and" ($\wedge$) and "not" ($'$). Operations $\wedge$, $\vee$ and $'$ satisfy a complete set of properties. See [2], [3], [5], [7] for more details.

(a)

$\vee$	0	1
0	0	1
1	1	1

$\wedge$	0	1
0	0	0
1	0	1

$'$	
0	1
1	0

(b)

$\vee$	0	u	1
0	0	u	1
u	u	u	1
1	1	1	1

$\wedge$	0	u	1
0	0	0	0
u	0	u	u
1	0	u	1

$'$	
0	1
u	u
1	0

D(β) are C(β) are known as *disjunctive and conjunctive Boolean normal forms* of β. The 2^n values $\beta_0 = \beta(0,\ldots,0), \beta_1 = \beta(0,\ldots,1), \ldots, \beta_{2^n-1} = \beta(1,\ldots,1)$ are constants in B.

A ternary logic function ϕ of n variables $x_1,\ldots,x_n$ is defined by an application $\phi : V^n \mapsto V$, $V = \{0, u, 1\}$, where "u" is "uncertain 0 or 1", that is, "ambiguous", in contrast to "0" (false) and "1" (true) [6]. The (total) order in V is $0 \leq u \leq 1$. Operations "and" ($\wedge$), "or" ($\vee$) and "not" ($'$) are given in Table 1(b), which correspond to Kleene's ternary logic system [4]. Unlike binary logic functions, ternary logic functions cannot be represented always by logic formulas [6]. We consider ternary logic functions defined by the following condition.

Condition 2.1 [6] *The ternary functions which can be represented by well-formed logic formulas consisting of variables $x_1,\ldots,x_n$, constants 0, u, 1 and logic connectives "and" ($\wedge$), "or" ($\vee$) and "not" ($'$) defined in Table 1(b).*

It is defined a partial order $\preceq$ concerning ambiguity on V and V^n as follows:

Definition 2.2 [6] *$0 \preceq u$, $1 \preceq u$, $a \preceq a$, $a \in V$. In the relation $\preceq$, 0 and 1 are not comparable to each other. The relation can be extended among V^n as follows: For two elements $\vec{a} = \langle a_1,\ldots,a_n\rangle$ and $\vec{b} = \langle b_1,\ldots,b_n\rangle$ of V^n, $\vec{a} \preceq \vec{b}$ if and only if $a_i \preceq b_i$ for all values of i. If $\vec{a} \preceq \vec{b}$, then $\vec{a}$ is said to be less ambiguous than or equal to $\vec{b}$.*

As usual, if $\vec{a} \preceq \vec{b}$ and $\vec{b} \preceq \vec{a}$ are not satisfied then $\vec{a}$ and $\vec{b}$ are said to be *incomparable* (in symbols $\vec{a} \,||\, \vec{b}$).

Condition 2.3 (Regularity) [6] *If $\phi(\vec{b}) \in B = \{0,1\}$; then $\phi(\vec{a}) = \phi(\vec{b})$ for every $\vec{a}$ such as $\vec{a} \preceq \vec{b}$.*

Definition 2.4 [6] *A ternary function ϕ is called a regular ternary logic function if and only if ϕ satisfies the regularity Condition 2.3.*

Condition 2.5 (Monotonicity for ambiguity) [6] *If* $\vec{a} \preceq \vec{b}$*, then* $\phi(\vec{a}) \preceq \phi(\vec{b})$.

Mukaidono [6, Thm. 1, 2 and 4] proved Condition 2.1, Condition 2.3 and Condition 2.5 are equivalent to each other, that is, a ternary logic function ϕ satisfies Condition 2.1 if and only if it satisfies Condition 2.3 if and only if it satisfies Condition 2.5.

Definition 2.6 (Ambiguity level) *It is defined the function* $\lambda : V^n \mapsto N : \vec{a} \mapsto \lambda(\vec{a})$*, where* $N = \{0, 1, 2, \ldots, n\}$*, as the total number of* u *in* $\vec{a}$*. The value* $\lambda(\vec{a}) \in N$ *is said to be the* ambiguity level of $\vec{a}$

For example, if $\vec{a}, \vec{b} \in V^3$ such that $\vec{a} = \langle 0, u, u\rangle$ and $\vec{b} = \langle 0, 1, 0\rangle$, then $\lambda(\vec{a}) = 2$ and $\lambda(\vec{b}) = 0$, i.e., $\vec{a}$ has two u and $\vec{b}$ has none.

We group all the elements in V^n depending on their ambiguity levels. We say $\vec{a} \in V^n$ is *equivalent in ambiguity* to $\vec{b} \in V^n$ (in symbols $\vec{a} \sim \vec{b}$) if and only if $\lambda(\vec{a}) = \lambda(\vec{b})$. The relation $\sim$ is an equivalence relation and it creates a partition on V^n. The quotient set is

$$V^n/\sim = \{[\langle 0, 0, \ldots, 0\rangle], [\langle u, 0, \ldots, 0\rangle], [\langle u, u, \ldots, 0\rangle], \ldots, [\langle u, u, \ldots, u\rangle]\}.$$

The function $\lambda^* : V^n/\sim \mapsto N$ defined by $\lambda^*([\vec{a}]) = \lambda(\vec{a})$, where $[\vec{a}]$ is the equivalence class that contains $\vec{a} \in V^n$, is given explicitly by $[\langle 0, 0, \ldots, 0\rangle] \overset{\lambda^*}{\mapsto} 0$, $[\langle u, 0, \ldots, 0\rangle] \overset{\lambda^*}{\mapsto} 1$, $[\langle u, u, \ldots, 0\rangle] \overset{\lambda^*}{\mapsto} 2$, $\ldots$, $[\langle u, u, \ldots, u\rangle] \overset{\lambda^*}{\mapsto} n$. The function λ^* orders linearly the equivalence classes with respect to the ambiguity level. The number of elements in each equivalence class is as follows:

$$\left|\,[\langle 0, 0, \ldots, 0\rangle]\,\right| = 2^n = \binom{n}{0} 2^{n-0}, \tag{3}$$

$$\left|\,[\langle u, 0, \ldots, 0\rangle]\,\right| = \underbrace{2^{n-1} + \cdots + 2^{n-1}}_{\binom{n}{1} \text{ times}} = \binom{n}{1} 2^{n-1}, \tag{4}$$

$$\left|\,[\langle u, u, \ldots, 0\rangle]\,\right| = \underbrace{2^{n-2} + \cdots + 2^{n-2}}_{\binom{n}{2} \text{ times}} = \binom{n}{2} 2^{n-2}, \tag{5}$$

$$\vdots \tag{6}$$

$$\left|\,[\langle u, u, \ldots, u\rangle]\,\right| = 1 = \binom{n}{n} 2^{n-n}. \tag{7}$$

It is clear that $\sum_{i=0}^{n} \binom{n}{i} 2^{n-i} = 3^n$ holds, where 3^n is the number of elements in V^n. The preceding formula can be verified using Newton's theorem $(a+b)^n = \sum_{i=0}^{n} \binom{n}{i} a^i b^{n-i}$ with $a = 1$ and $b = 2$.

Theorem 2.7 *Let $\vec{a}, \vec{b} \in V^n$ with $\vec{a} \neq \vec{b}$. If $\vec{a} \sim \vec{b}$ then $\vec{a} \,||\, \vec{b}$.*

Proof. We prove the theorem by contradiction. It is supposed that $\vec{a} \sim \vec{b}$ and either $\vec{a} \preceq \vec{b}$ or $\vec{b} \preceq \vec{a}$ holds. Without losing generality, it is supposed that $\vec{a} \preceq \vec{b}$ holds (the case $\vec{b} \preceq \vec{a}$ is analogous), hence $a_i \preceq b_i$ for all values of i. Since $\vec{a} \neq \vec{b}$, k components in $\vec{a}$ and $\vec{b}$ are such that $a_i = b_i$, and the remaining $n - k > 0$ components are such that $a_i \neq b_i$. For the k components where $\vec{a}$ and $\vec{b}$ coincide, their contribution to $\lambda(\vec{a})$ and $\lambda(\vec{b})$ are the same, namely K. For the $n - k$ components where $\vec{a}$ and $\vec{b}$ differ, $a_i \in B = \{0, 1\}$ and $b_i = u$ hold (since $a_i \preceq b_i$), hence their contribution to $\lambda(\vec{a})$ and $\lambda(\vec{b})$ are zero and $n - k$ respectively. Therefore we have that $\lambda(\vec{a}) = K$ and $\lambda(\vec{b}) = K + n - k$, i.e., $\lambda(\vec{a}) < \lambda(\vec{b})$, which contradicts $\vec{a} \sim \vec{b}$. ■

Theorem 2.8 *Let $\vec{a}, \vec{b} \in V^n$ for which $a_i = b_i$ for all $i \in \{1, 2, \ldots, n\}$ with the exception of $i = j$ where $a_j \in B = \{0, 1\}$ and $b_j = u$. Then (1) $\vec{a} \preceq \vec{b}$ and, (2) there is no $\vec{c} \in V^n$, $\vec{c} \neq \vec{a}$ and $\vec{c} \neq \vec{b}$, such that $\vec{a} \preceq \vec{c} \preceq \vec{b}$.*

Proof. (1) For the j-th component where $a_j \in B$ and $b_j = u$, it is clear that $a_j \preceq b_j$. For all $i \in \{1, 2, \ldots, n\}$, $i \neq j$, since $a_i = b_i$, it is clear that $a_i \preceq b_i$, hence $\vec{a} \preceq \vec{b}$.

(2) We prove the second part by contradiction. It is supposed that there is $\vec{c} \in V^n$, $\vec{c} \neq \vec{a}$ and $\vec{c} \neq \vec{b}$, such that $\vec{a} \preceq \vec{c} \preceq \vec{b}$, that is $a_i \preceq c_i \preceq b_i$ for all $i \in \{1, 2, \ldots, n\}$. But $a_i = b_i$ for all $i \in \{1, 2, \ldots, n\}$, $i \neq j$, therefore $c_i = a_i = b_i$ for the same values of i. For the j-th component where $a_j \in B$ and $b_j = u$: (a) if $a_j = 0$ then $0 = a_j \preceq c_j \preceq b_j = u$ and therefore $c_j = 0$ or $c_j = u$, and in both cases we conclude $\vec{c} = \vec{a}$ or $\vec{c} = \vec{b}$, which is a contradiction; (b) if $a_j = 1$ then $1 = a_j \preceq c_j \preceq b_j = u$ and therefore $c_j = 1$ or $c_j = u$, and in both cases we conclude $\vec{c} = \vec{a}$ or $\vec{c} = \vec{b}$, which is a contradiction. ■

Theorem 2.9 *If $\vec{a} \in B^n$ then there is no $\vec{b} \in V^n$, $\vec{a} \neq \vec{b}$, such that $\vec{b} \preceq \vec{a}$.*

Proof. We prove the theorem by contradiction. Let $\vec{a} \in B^n$. It is supposed that there is $\vec{b} \in V^n$, $\vec{a} \neq \vec{b}$, such that $\vec{b} \preceq \vec{a}$, that is, $b_i \preceq a_i$ for all $i \in \{1, 2, \ldots, n\}$. Since $\vec{a} \neq \vec{b}$, there is at least one j such that $a_j \neq b_j$. For such a j, if $a_j = 0$ then $b_j = 0$ (since $b_i \preceq a_i$) which contradicts $a_j \neq b_j$; if $a_j = 1$ then $b_j = 1$ (since $b_i \preceq a_i$) which contradicts $a_j \neq b_j$. ■

3 From Binary Logic Functions to Regular Ternary Logic Functions

Let $\beta : B^n \mapsto B$ be a binary logic function, and let $\phi : V^n \mapsto V$ be a ternary logic function. It is clear that $B \subset V$ and $B^n \subset V^n$. Given β, the main problem

consists in finding a function ϕ which satisfies the following two conditions:

$$\phi(\vec{a}) = \beta(\vec{a}), \text{ for all } \vec{a} \in B^n. \tag{8}$$

$$\phi \text{ is a regular ternary logic function.} \tag{9}$$

Condition (8) expresses the way as ϕ is defined over B^n, but it does not say the way as ϕ is defined over $V^n - B^n$. In Section 4 we show a method to define ϕ over $V^n - B^n$ such that (9) is satisfied. Now we prove an existence theorem.

Theorem 3.1 *Let $\beta : B^n \mapsto B$ be a given binary logic function. Then there exists at least one ternary logic function $\phi : V^n \mapsto V$ which satisfies* (8) *and* (9)*.*

Proof. Given $\beta : B^n \mapsto B$, we prove the theorem finding a trivial ϕ which satisfies (8) and (9). It is defined $\phi : V^n \mapsto V$ for all $\vec{a} \in V^n$ as follows:

$$\phi(\vec{a}) = \begin{cases} \beta(\vec{a}), & \text{if } \vec{a} \in B^n, \\ u, & \text{if } \vec{a} \in V^n - B^n. \end{cases} \tag{10}$$

The first case in (10) satisfies (8). Now we will prove that (10) satisfies (9). According to (10) for all $\vec{a} \in B^n$ we have that $\phi(\vec{a}) = \beta(\vec{a}) \in B$, but according to Theorem 2.9 there is no $\vec{b} \in V^n$, $\vec{a} \neq \vec{b}$, such that $\vec{b} \preceq \vec{a}$, therefore $\vec{a}$ is the unique by itself which satisfies Condition 2.3 (regularity). According to (10) for all $\vec{a} \in V^n - B^n$ we have that $\phi(\vec{a}) = u \notin B$ and therefore Condition 2.3 (regularity) is true. ■

4 Proposed Method

1. Given a binary logic function $\beta : B^n \mapsto B$ of n variables $x_i \in B$, $i = 1, 2, \ldots, n$, make a truth table of 2^n rows and $n+1$ columns (n columns for $x_1,\ldots,x_n$ and 1 column for β). These rows will have the 2^n combinations of values for $x_1,\ldots,x_n$ and the corresponding values for β (which are represented as $\beta_i \in B$, $i = 0, 1, \ldots, 2^n - 1$) as we show in Table 2(a). This step guarantees (3).

2. To the table elaborated in numeral 1, add $3^n - 2^n$ new rows. First those $\binom{n}{1}2^{n-1}$ rows whose combination of values for the variables have only one u, next those $\binom{n}{2}2^{n-2}$ rows whose combination of values for the variables have two u, and so on. At the end, the table finishes with that $\binom{n}{n}2^{n-n} = 1$ row which has n u values, as we show in Table 2(b). Each group of rows added previously is separated by a horizontal line which represents the separation between different ambiguity levels λ. The ambiguity level is increasing. This step guarantees (4)–(7).

Table 2: (a) Initial binary logic function $\beta : B^n \mapsto B$ given as a truth table. (b) Extended ternary logic function(s) $\phi : V^n \mapsto V$ given as a truth table

(a)

	x_1	x_2	$\cdots$	x_n	β
2^n rows	0	0	$\cdots$	0	β_0
	0	0	$\cdots$	1	β_1
	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
	1	1	$\cdots$	1	β_{2^n-1}

(b)

	λ	x_1	x_2	$\cdots$	x_n	ϕ
2^n rows	0	0	0	$\cdots$	0	$\phi_0 = \beta_0$
		0	0	$\cdots$	1	$\phi_1 = \beta_1$
		$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
		1	1	$\cdots$	1	$\phi_{2^n-1} = \beta_{2^n-1}$
$\binom{n}{1}2^{n-1}$ rows	1	u	0	$\cdots$	0	ϕ_{2^n}
		u	0	$\cdots$	1	ϕ_{2^n+1}
		$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
		1	1	$\cdots$	u	
$\binom{n}{2}2^{n-2}$ rows	2	u	u	$\cdots$	0	
		u	u	$\cdots$	1	
		$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
		1	1	$\cdots$	u	
1 row $\cdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
1 row	n	u	u	$\cdots$	u	ϕ_{3^n-1}

3. Set $\phi_i = \beta_i$ for $i = 0, 1, \ldots, 2^n - 1$. This step guarantees (8).

4. For each group with ambiguity level $\lambda = 1, 2, \ldots, n$ do the following:

   (a) For the j-th considered row in the group with ambiguity level λ, $j = 1, 2, \ldots, \binom{n}{\lambda}2^{n-\lambda}$, find those 2λ rows in the immediately previous group with ambiguity level $\lambda - 1$ by changing one by one each u by 0 and 1 on the j-th row. These 2λ rows are immediately less ambiguous than or equal to the j-th row (Theorem 2.8).

   (b) For each one of the 2λ rows found in numeral 4a, find the set of values greater ambiguous than or equal to the respective ϕ value of that row. Write this set on a separate column.

   (c) The set of values that ϕ can take on the j-th considered row is the intersection of those sets found in numeral 4b. If ϕ can take more than one value, create as many new columns for new ϕ functions as possible. For each ϕ created, repeat numeral 4.

5 Example

Let us consider the binary logic function of $n = 2$ variables given in Table 3(a). This table has $2^2 = 4$ rows and $2+1 = 3$ columns. Now we create a new table by adding $3^2 - 2^2 = 5$ new rows as we show in Table 3(b). First those $\binom{2}{1}2^{2-1} = 4$

Table 3: Example. (a) Initial binary logic function. (b) Extension to the ambiguity level $\lambda = 1$

(a)

x	y	β
0	0	0
0	1	1
1	0	0
1	1	1

(b)

Row	λ	x	y	ϕ	
1	0	0	0	0	$\{0, u\}$
2		0	1	1	$\{1, u\}$
3		1	0	0	$\{0, u\}$
4		1	1	1	$\{1, u\}$
1	1	u	0	$\{0, u\}$	
2		u	1	$\{1, u\}$	
3		0	u	$\{u\}$	
4		1	u	$\{u\}$	
1	2	u	u		

rows which have one u, and finally that $\binom{2}{2}2^{2-2} = 1$ row which has two u. The $3^2 = 9$ rows are divided into 3 groups which are separated by horizontal lines according to their ambiguity levels. The first group is the initial binary logic function.

1. For the group with ambiguity level $\lambda = 1$:

   (a) For the row $j = 1$ (where $x = u$ and $y = 0$), we find those $2(1) = 2$ rows less ambiguous than or equal to the row $j = 1$ by changing $x = u$ by $x = 0$ and $x = 1$. We find rows 1 and 3 with ambiguity level $\lambda = 0$ and whose sets of values greater ambiguous than or equal to their respective ϕ values are $\{0, u\}$ and $\{0, u\}$. These sets are written on a separate column. The set of values that ϕ can take on the row $j = 1$ with ambiguity level $\lambda = 1$ is $\{0, u\} \cap \{0, u\} = \{0, u\}$.

   (b) For the row $j = 2$ (where $x = u$ and $y = 1$), we find those $2(1) = 2$ rows less ambiguous than or equal to the row $j = 2$ by changing $x = u$ by $x = 0$ and $x = 1$. We find rows 2 and 4 with ambiguity level $\lambda = 0$ and whose sets of values greater ambiguous than or equal to their respective ϕ values are $\{1, u\}$ and $\{1, u\}$. These sets are written on a separate column. The set of values that ϕ can take on the row $j = 2$ with ambiguity level $\lambda = 1$ is $\{1, u\} \cap \{1, u\} = \{1, u\}$.

   (c) For the row $j = 3$ (where $x = 0$ and $y = u$), we find those $2(1) = 2$ rows less ambiguous than or equal to the row $j = 3$ by changing $y = u$ by $y = 0$ and $y = 1$. We find rows 1 and 2 with ambiguity level $\lambda = 0$ and whose sets of values greater ambiguous than or equal to their respective ϕ values are $\{0, u\}$ and $\{1, u\}$. These sets are written on a separate column. The set of values that ϕ can take on the row $j = 3$ with ambiguity level $\lambda = 1$ is $\{0, u\} \cap \{1, u\} = \{u\}$.

Table 4: Example (continuation). (a) Extension to the ambiguity level $\lambda = 2$. (b) Final regular ternary logic functions

(a)

Row	λ	x	y	ϕ_1		ϕ_2		ϕ_3		ϕ_4	
1	0	0	0	0		0		0		0	
2		0	1	1		1		1		1	
3		1	0	0		0		0		0	
4		1	1	1		1		1		1	
1	1	u	0	0	$\{0,u\}$	u	$\{u\}$	0	$\{0,u\}$	u	$\{u\}$
2		u	1	1	$\{1,u\}$	1	$\{1,u\}$	u	$\{u\}$	u	$\{u\}$
3		0	u	u	$\{u\}$	u	$\{u\}$	u	$\{u\}$	u	$\{u\}$
4		1	u	u	$\{u\}$	u	$\{u\}$	u	$\{u\}$	u	$\{u\}$
1	2	u	u	$\{u\}$		$\{u\}$		$\{u\}$		$\{u\}$	

(b)

x	y	ϕ_1	ϕ_2	ϕ_3	ϕ_4
0	0	0	0	0	0
0	1	1	1	1	1
1	0	0	0	0	0
1	1	1	1	1	1
u	0	0	u	0	u
u	1	1	1	u	u
0	u	u	u	u	u
1	u	u	u	u	u
u	u	u	u	u	u

(d) For the row $j = 4$ (where $x = 1$ and $y = u$), we find those $2(1) = 2$ rows less ambiguous than or equal to the row $j = 4$ by changing $y = u$ by $y = 0$ and $y = 1$. We find rows 3 and 4 with ambiguity level $\lambda = 0$ and whose sets of values greater ambiguous than or equal to their respective ϕ values are $\{0, u\}$ and $\{1, u\}$. These sets are written on a separate column. The set of values that ϕ can take on the row $j = 4$ with ambiguity level $\lambda = 1$ is $\{0, u\} \cap \{1, u\} = \{u\}$.

(e) Since rows $j = 1$ and $j = 2$ with ambiguity level $\lambda = 1$ can take two different values on their respective ϕ values, we create four new ϕ functions (ϕ_1, ϕ_2, ϕ_3 and ϕ_4) as we show in Table 4(a).

2. For the group with ambiguity level $\lambda = 2$:

(a) For the row $j = 1$ (where $x = u$ and $y = u$), we find those $2(2) = 4$ rows less ambiguous than or equal to the row $j = 1$ by changing $x = u$ or $y = u$ (but not at the same time) by $x = 0$ and $x = 1$ or $y = 0$ and $y = 1$. We find rows 1–4 with ambiguity level $\lambda = 1$ and whose sets of values greater ambiguous than or equal to their respective ϕ values are shown in Table 4(a) for each ϕ_i ($i = 1, 2, 3, 4$). These sets are written on separate columns. The set of values that ϕ can take on the row $j = 1$ with ambiguity level $\lambda = 2$ for each ϕ_i ($i = 1, 2, 3, 4$) are $\{0, u\} \cap \{1, u\} \cap \{u\} \cap \{u\} = \{u\}$, $\{u\} \cap \{1, u\} \cap \{u\} \cap \{u\} = \{u\}$, $\{0, u\} \cap \{u\} \cap \{u\} \cap \{u\} = \{u\}$ and $\{u\} \cap \{u\} \cap \{u\} \cap \{u\} = \{u\}$ respectively.

(b) Therefore, there are four regular ternary logic functions only (Table 4(b)) which are the corresponding extension of the given binary logic function.

The logic formulas (disyuntive normal forms) for ϕ_1, ϕ_2, ϕ_3 and ϕ_4 obtained by means of the procedure presented by Gehrke *et al.* [1, pp. 39] are:

$$\begin{aligned}
\phi_1 &= y \\
\phi_2 &= (x \wedge x' \wedge y') \vee y \\
\phi_3 &= (x' \wedge y) \vee (x \wedge y) \\
\phi_4 &= (x' \wedge y) \vee (x \wedge y) \vee (x \wedge x' \wedge y')
\end{aligned}$$

According to the theory presented by Gehrke *et al.* [1], the logic formulas for ϕ_1, ϕ_2, ϕ_3 and ϕ_4 are valid for fuzzy logic.

References

[1] Mai Gehrke, Carol Walker, and Elbert Walker. Normal forms and truth tables for fuzzy logics. *Fuzzy Sets and Systems*, 138(1):25–51, August 2003.

[2] George Grätzer. *General Lattice Theory*. Pure and Applied Mathematics. A Series of Monographs and Textbooks. Academic Press Inc., New York, 1978.

[3] Gary D. Hachtel and Fabio Somenzi. *Logic Synthesis and Verification Algorithms*. Kluwer Academic Publishers, Dordrecht, 1996.

[4] Stephen Cole Kleene. *Introduction to metamathematics*, volume 1 of *Bibliotheca Mathematica. A Series of Monographs on Pure and Applied Mathematics*. North-Holland Publishing Co., Amsterdam, 1952.

[5] George J. Klir and Bo Yuan. *Fuzzy sets and fuzzy logic: theory and applications*. Prentice Hall PTR, New Jersey, 1995.

[6] Masao Mukaidono. Regular ternary logic functions—ternary logic functions suitable for treating ambiguity. *IEEE Transactions on Computers*, C-35(2):179–183, February 1986.

[7] Hung T. Nguyen and Elbert A. Walker. *A First Course in Fuzzy Logic*. Chapman & Hall/CRC, Boca Raton, Florida, 3 edition, 2006.

Received: June 11, 2013