DOI 10.1007/s10559-022-00497-4
Cybernetics and Systems Analysis, Vol. 58, No. 4, July, 2022

ALGEBRAIC OPERATIONS ON FUZZY SETS AND RELATIONS IN AUTOMATA INTERPRETATION IMPLEMENTED BY LOGICAL HARDWARE

S. L. Kryvyi,[1] V. M. Opanasenko,[2] and S. B. Zavyalov[3] UDC 516.813

Abstract. *Algebraic operations on fuzzy sets and relations and their implementation by hardware in automata interpretation are considered. Two ways of representing the values of membership functions of fuzzy sets and methods of transformation of such images are described. Appropriate estimates of the complexity of operations with such images are given and correctness of the algorithms is proved.*

Keywords: *fuzzy sets, fuzzy relations, algebraic operations, finite automata, FPGA.*

INTRODUCTION

Adaptation of hardware methods to the problem of partitioning vectors with integer coordinates, which was considered in [1–8], makes it possible to implement operations on fuzzy sets and relations. The approach we propose here to performing these operations is known as reconfigurable computing [2, 3] and its implementation in real projects became possible due to the advent of programmable logic integrated circuits (PLIC).

In particular, a method for solving the problem of hardware adaptation with a formalized justification of the corresponding algorithms based on adaptive logic networks (ALNs), focused on the implementation of the algorithms of splitting a set of vectors with integer coordinates, is considered in [4–8]. In the paper, we will use such partitioning algorithm based on the automatic approach and propose algorithms for performing operations on fuzzy sets (FS) and fuzzy relations (FR).

ALGEBRAIC OPERATIONS AND THEIR IMPLEMENTATION

Operations on FS and FR are divided into logical and algebraic. As was shown in [1], subtraction, maximum, and minimum operations are needed to implement logical operations on FS and FR, and addition, subtraction, and multiplication are required to implement algebraic operations. In what follows, we will only consider algebraic operations on FS, which include:

(i) calculating the product of FS A and B, whose result is FS $A \cdot B$ with the membership function $\mu_{A \cdot B} = \mu_A(x) \cdot \mu_B(x)$;

(ii) calculating the sum $A + B$ of the FS A and B, whose result is FS $A + B$ with the membership function $\mu_{A+B}(x) = \mu_A(x) + \mu_B(x) - \mu_A(x) \cdot \mu_B(x)$;

(iii) raising the FS A to a power α, whose result is FS A^{α} with the membership function $\mu_A(x)^{\alpha}$, where α is a positive rational number;

[1]Taras Shevchenko National University of Kyiv, Kyiv, Ukraine, *sl.krivoi@gmail.com*. [2]V. M. Glushkov Institute of Cybernetics, National Academy of Sciences of Ukraine, Kyiv, Ukraine, *vlopanas@ukr.net*. [3]LLC Radioniks, Kyiv, Ukraine, *radionix13@gmail.com*. Translated from Kibernetyka ta Systemnyi Analiz, No. 4, July–August, 2022, pp. 172–182. Original article submitted December 3, 2021.

 ©2022 Springer Science+Business Media, LLC

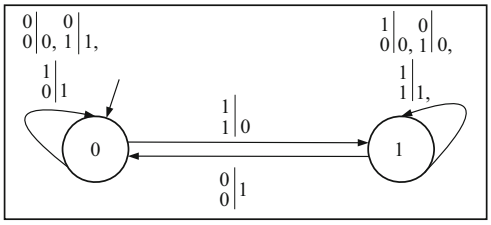

Fig. 1. Automaton A_+ of addition of binary numbers.

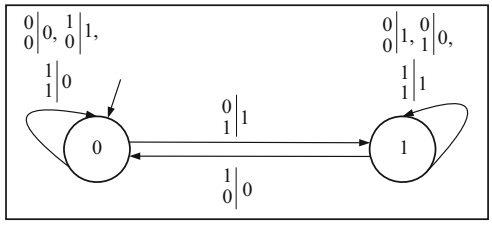

Fig. 2. Automaton A_- of subtraction of binary numbers x and y ($x \geq y$).

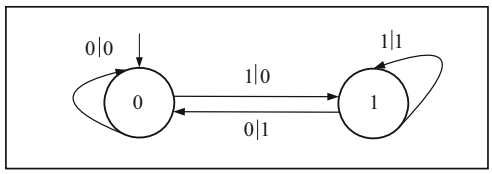

Fig. 3. Automaton A_2 of multiplication by 2.

(iv) multiplying the FS A by a number α, whose result is FS αA with the membership function $\alpha\mu_A(x)$, where $\alpha \cdot \max\limits_{x\in A} \mu_A(x) \leq 1$;

(v) finding the convex combination of FS $A_1, \ldots, A_n$, whose result is FS A with the membership function $\mu_A(x) = \alpha_1\mu_{A_1}(x) + \ldots + \alpha_n\mu_{A_n}(x)$, where $\alpha_1 \geq 0, \ldots, \alpha_n \geq 0$ and $\alpha_1 + \ldots + \alpha_n = 1$, $\alpha_i \in [0,1]$;

(vi) finding the Cartesian product of FS $A_1 \times A_2 \times \ldots \times A_n$, whose result is a FS with the membership function $\min(\mu_{A_1}(x_1), \ldots, \mu_{A_n}(x_n)) = \mu_A(x_1, \ldots, x_n)$;

(vii) obtaining a crisp set of level α, whose result is a crisp set $A_\alpha = \{x \mid \mu_A(x) \geq \alpha\}$, where $\alpha \in (0,1]$.

Sometimes one more operation is added to the above-mentioned ones:

(viii) operator Φ of increasing the fuzziness of the set A that is used to increase the fuzziness of the fuzzy set.

Let A be a fuzzy set, E be a universal set, and for all $x \in E$ fuzzy sets $K(x)$ be defined. The result of the action of operator Φ on the fuzzy set A is a fuzzy set

$$\Phi(A, K) = \bigcup_{x\in E} \mu_A(x)K(x),$$

where $\mu_A(x)K(x)$ is multiplication of a number by an FS.

According to the definitions of algebraic operations, their implementation requires automata that implement the operations of addition (Fig. 1), subtraction (Fig. 2), and multiplication by 2 (Fig. 3). Implementation of the multiplication operation requires a composition of these automata and of some control automaton, which will be described below.

Calculating the Product of Binary Numbers. Let us have binary numbers: $x = a_0a_1a_2a_3$ and $y = b_0b_1b_2b_3$, then the product of these numbers is defined by the expression

$$x \cdot y = x(b_0 + 2b_1 + 4b_2 + 8b_3) = xb_0 + 2xb_1 + 4xb_2 + 8xb_3. \quad (1)$$

It follows from (1) that for the correct calculation of the product of two binary numbers, the number of digits of the number $x = a_0a_1a_2a_3$ must be increased to nine (in the general case, to $2n+1$ digits for an n-digit number) by adding zeros to higher-order digits, i.e., $x = 00000a_0a_1a_2a_3$, and adding b_i into the lower-order digits y. Moreover, it follows from (1) that the value of b_i can be considered as a control signal whose value can be recognized by a control automaton A_{01} without outputs (Fig. 4).

In this automaton, state 0 is recognition of the value of b_i, from which the control is transferred by further calculations either to the Mealy machine A_0, which reacts to an arbitrary input signal of word x by the output of 0 (Fig. 5, state 1), when $b_i = 0$, or to the Mealy machine A_+, or A_2 when $b_i = 1$ (see Fig. 5, state 2). After that, a composition of such automata for calculating the multiplication operation is implemented by the scheme (see Fig. 5).

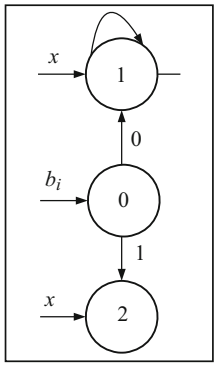

Fig. 4. Control automaton A_{01}.

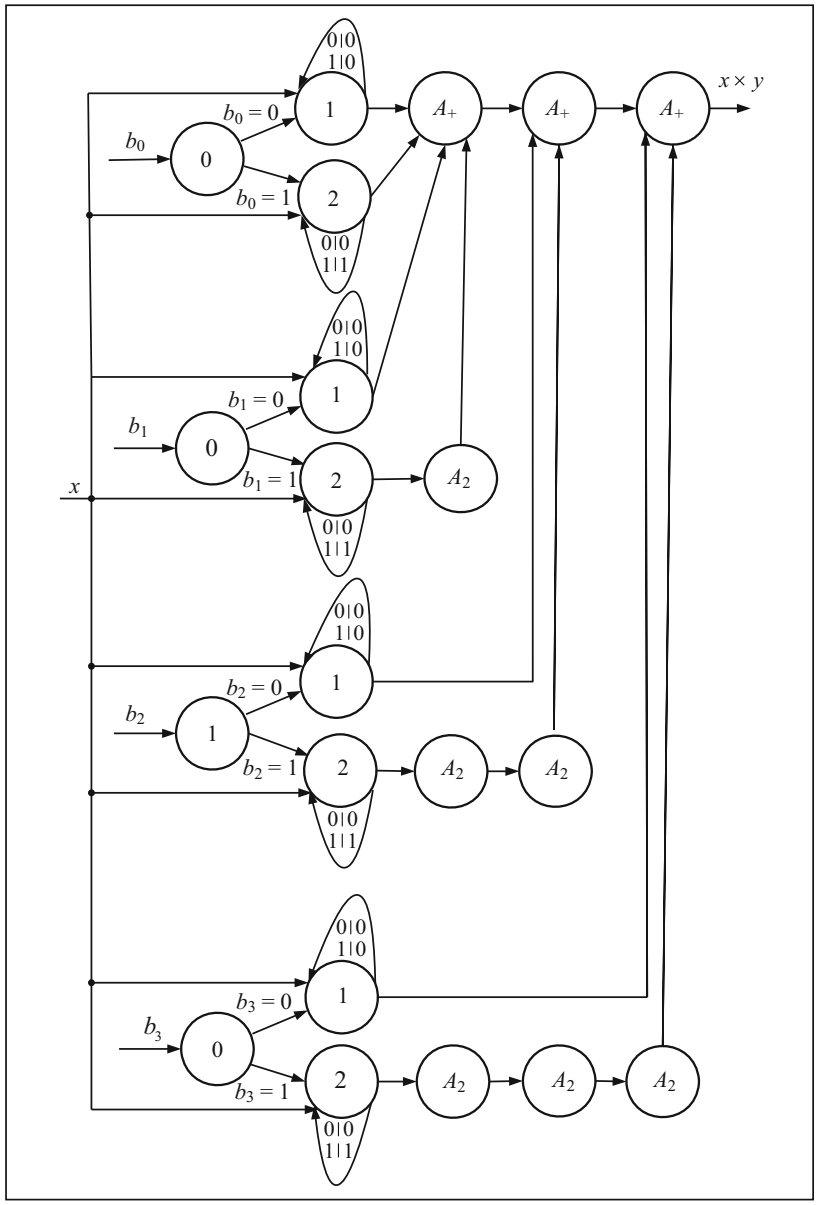

Fig. 5. Composition of automata $A_\times$.

As one can see, automaton $A_\times$, which implements the multiplication operation, is more complex as compared to automata for calculating logical operations [1].

While implementation of operations 1–5 and 7 require operations of addition, subtraction, and multiplication, a network of automata $A_{\min}$ is sufficient to calculate the Cartesian product operation and operator 8.

It follows from the aforesaid that to implement algebraic operations, it is necessary to have the implementation of automata A_{01}, $A_{\min}$, A_-, A_+, and $A_\times$, as well as networks of these automata and their compositions.

Further synthesis and configuration of the network (and its reconfiguration) are performed according to the structure of the algebraic expression (1) and membership function.

METHODS OF REPRESENTING THE ARGUMENTS OF OPERATIONS

First, let us consider some examples to illustrate what has been said.

Example 1. Calculate the FS

$$V = 2(A \cap B) + (A \cup C), \tag{2}$$

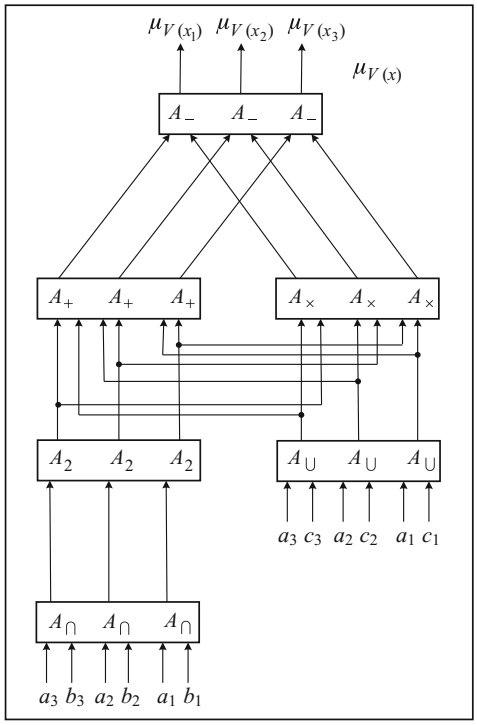

Fig. 6. Network for calculating FS V.

TABLE 1

x_1	x_2	x_3	Membership Functions
$\mu_A(x) \to 0.2$	0.5	0.4	
$\mu_B(x) \to 0.4$	0.6	0.1	
0.2	0.5	0,1	$\leftarrow \mu_{A \cap B}(x)$
0.4	1	0.2	$\leftarrow \mu_{2(A \cap B)}(x)$
$\mu_A(x) \to 0.2$	0.5	0.4	
$\mu_C(x) \to 0.1$	0	1	
0.2	0.5	1	$\leftarrow \mu_{A \cup C}(x)$
$\mu_{2(A \cap B)}(x) \to 0.4$	1	0.2	
$\mu_{A \cup C}(x) \to 0.2$	0.5	1	
0.6	1.5	1.2	$\leftarrow \mu_{(A \cup C)} + \mu_{2(A \cap C)}$
0.08	0.5	0.2	$\leftarrow \mu_{(A \cup C)} \cdot \mu_{2(A \cap C)}$
$\mu_{(A \cup C)} + \mu_{2(A \cap C)} \to 0.6$	1.5	1.2	
$\mu_{(A \cup C)} \cdot \mu_{2(A \cap C)} \to 0.08$	0.5	0.2	
0.52	1	1	$\leftarrow \mu_{(A \cup C)} + \mu_{2(A \cap C)} - \mu_{(A \cup C)} \cdot \mu_{2(A \cap C)}$

where

$$A = \{0.2 \,|\, x_1 + 0.5 \,|\, x_2 + 0.4 \,|\, x_3\};\ B = \{0.4 \,|\, x_1 + 0.6 \,|\, x_2 + 0.1 \,|\, x_3\};$$

$$C = \{0.1 \,|\, x_1 + 0 \,|\, x_2 + 1 \,|\, x_3\}.$$

Solution. It follows from (2) that calculating the FS V requires the automata $A_{\min} = A_\cap$, $A_{\max} = A_\cup A_-$, A_+, and $A_\times$ [1]. We synthesize the network according to the structure of expression (2) (Fig. 6).

Table 1 illustrates the calculations performed by this network.

As a result, we obtain

$$V = \{0.52 \,|\, x_1 + 1 \,|\, x_2 + 1 \,|\, x_3\}.$$

To implement addition, subtraction, and multiplication operations, it is necessary to describe the representation of their arguments, which are real numbers from the membership interval [0, 1], since their implementation has certain special features. In this paper, we consider two ways of representing such values:

(i) representation by a pair of integers;

(ii) representation by fractional binary numbers.

The first method is to represent the value of the membership function $\mu(x) = y$ as a pair of non-negative integers $(y; n)$, where y is an integer that represents the value of $\mu(x)$ and n is the number of decimal digits in the decimal representation of y. For example, for $x = 0.235$ the corresponding pair has the form (235; 3). Such a representation makes it possible to handle binary numbers that represent components of pairs. The introduced representation of the values of the membership function requires the substantiation of the rules for calculating the values of addition and subtraction. To calculate both operations, a simple transformation is required, which consists in "alignment" of components of the pairs. This alignment is implemented as follows: in the calculation of the values of operations on pairs $(x; n)$ and $(x'; n')$, when, for example, $n = n'+2$, the second pair should be converted into $(x'00; n)$. Thus, as a result of subtracting the pair (1; 2) from the pair (15; 3), the former becomes (10; 3), after that subtraction is performed, which yields (5; 3). Since addition and subtraction operations does not change the position of the decimal point, the calculation of these operations does not change the second components after alignment and no action is performed on them. To represent this situation, we introduce a special automaton A_i, which does not change the values of the input data.

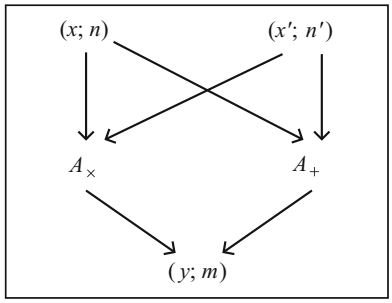

Fig. 7. Scheme of a network of two automata $(A_{\times}, A_{+})$.

In this representatin, let us consider the method of calculating the product of two real numbers from the membership interval [0, 1]. In performing the multiplication operation, decimal point shifts by a value that depends on the length of the mantissas of the arguments in the multiplication operation. Then to perform the operation of multiplying two numbers $(x; n)$ and $(x'; n')$, the first components are multiplied as integers (i.e., the decimal point is ignored), and the second components are added; therefore, we obtain the pair $(y; m)$, which represents the result of multiplication.

For example, the result of multiplying (21; 2) by (123; 3) is the pair (2583; 5), i.e., first, the value of 2583 is calculated as a result of multiplying 21 by 123 and the value 5 as a result of adding 2 and 3, and then the number 5 indicates that the value of 2583 should be converted into 0.02583. The second component in the pair determines the place of the decimal point. Therefore, the multiplication operation is implemented by a heterogeneous network of two automata $(A_{\times}, A_{+})$, with the pairs of arguments $(x; n)$ and $(x'; n')$ at the input and the pair $(y = xx'; m = n + n')$, whose scheme is shown in Fig. 7, at the output.

Example 2. Calculate the FS

$$X = ((A + B) \cdot C) \bigcap (A \cdot C), \tag{3}$$

where A, B, and C are the FS from Example 1.

Solution. The network of automata is synthesized according to the structure of the expression of the membership function, which specifies the FS (3) and is shown in Fig. 8.

If $xz \le z(x + y - xy)$, then the output is xz, k'; otherwise, the output is the pair $(z(x + y - xy, k = n'')$.

For illustration purposes, we will calculate each operation separately. Then the FS

$$A = \{0.2 \,|\, x_1 + 0.5 \,|\, x_2 + 0.4 \,|\, x_3\};\ B = \{0.4 \,|\, x_1 + 0.6 \,|\, x_2 + 0.1 \,|\, x_3\};$$

$$C = \{0.1 \,|\, x_1 + 0 \,|\, x_2 + 1 \,|\, x_3\}$$

becomes

$$A = \{(2;1) \,|\, x_1 + (5;1) \,|\, x_2 + (4;1) \,|\, x_3\};$$

$$B = \{(4;1) \,|\, x_1 + (6;1) \,|\, x_2 + (1;1) \,|\, x_3\};$$

$$C = \{(1;1) \,|\, x_1 + (0;0) \,|\, x_2 + (1;0) \,|\, x_3\}.$$

Applying the transformed values of the FS membership function A and B to the input of the network of automata $(A_{\times}, A_{+})$, we obtain

$$(A_{\times}(A, B), A_{+}(A, B)) = \{(8;2) \,|\, x_1 + (30;2) \,|\, x_2 + (4;2) \,|\, x_3\} = S.$$

Calculation of the sum of the values of the membership functions of the FS (A, B) does not require the alignment (since the second components are the same in these FS), and we obtain the following value for the network of automata $A_{\times}, A_i$ (automaton A_i keeps the second component unchanged):

$$(A_{\times}(A, B), A_i(A, B)) = \{(6;1) \,|\, x_1 + (11;1) \,|\, x_2 + (5;1) \,|\, x_3\} = S_1.$$

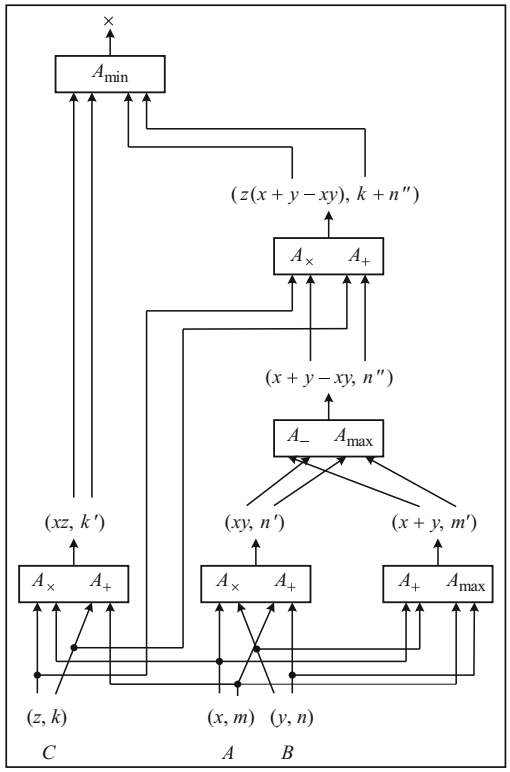

Fig. 8. Network of automata for implementation of expression X.

To perform the subtraction operation, we align the second components in the obtained set S_1, which yields the set $S_1 = \{(60;2)\,|\,x_1 + (110;2)\,|\,x_2 + (50;2)\,|\,x_3\}$. Now, the network of automata A_+, A_i yields the FS $(A+B)$:

$$A+B = A_+(S_1, S) = \{(52;2)\,|\,x_1 + (80;2)\,|\,x_2 + (46;2)\,|\,x_3\},$$

i.e., $A+B = \{0.52\,|\,x_1 + 0.8\,|\,x_2 + 0.46\,|\,x_3\}$. And the network of automata $A_\times, A_+$ yields the FS for the sub-expression $(A+B)\cdot C$:

$$(A_+((A+B)\cdot C),\ A_+((A+B)\cdot C)) = \{(52;3)\,|\,x_1 + (0;2)\,|\,x_2 + (46;2)\,|\,x_3\}.$$

Calculation of the product $(A\cdot C)$ yields the following FS after alignment:

$$(A_\times(A\cdot C),\ A_+(A\cdot C)) = \{(20;3)\,|\,x_1 + (0;2)\,|\,x_2 + (40;2)\,|\,x_3\}.$$

The final FS is calculated by the network:

$$(A_{\min}((A+B)\cdot C \cap A\cdot C),\ A_i((A+B)\cdot C \cap A\cdot C)) = \{(20;3)\,|\,x_1 + (0;2)\,|\,x_2 + (46;2)\,|\,x_3\},$$

i.e., $(A+B)\cdot C \cap A\cdot C = \{(0.02\,|\,x_1 + 0\,|\,x_2 + 0.46\,|\,x_3\}$.

The second method deals with binary fractional number representation:

$$x = 0.(a_1 2^{-1})(a_2 2^{-2})(a_3 2^{-3})(a_4 2^{-4})(a_5 2^{-5})(a_6 2^{-6})(a_7 2^{-7})(a_8 2^{-8}),$$

where $x \in [0.5\text{–}1)$, $a_i \in [0 \vee 1]$ and $\forall a_i$, $i = 1, \ldots, 8$,

$$a_1 2^{-1} = \begin{cases} 0.5 & \text{if } a_1 = 1, \\ 0 & \text{if } a_1 = 0, \end{cases} \quad a_2 2^{-2} = \begin{cases} 0.25 & \text{if } a_2 = 1, \\ 0 & \text{if } a_2 = 0, \end{cases}$$

$$a_3 2^{-3} = \begin{cases} 0.125 & \text{if } a_3 = 1, \\ 0 & \text{if } a_3 = 0, \end{cases} \quad a_4 2^{-4} = \begin{cases} 0.625 & \text{if } a_4 = 1, \\ 0 & \text{if } a_4 = 0, \end{cases}$$

$$a_5 2^{-5} = \begin{cases} 0.03125 & \text{if } a_5 = 1, \\ 0 & \text{if } a_5 = 0, \end{cases} \quad a_6 2^{-6} = \begin{cases} 0.0015625 & \text{if } a_6 = 1, \\ 0 & \text{if } a_6 = 0, \end{cases}$$

$$a_7 2^{-7} = \begin{cases} 0.00078125 & \text{if } a_7 = 1, \\ 0 & \text{if } a_7 = 0, \end{cases} \quad a_8 2^{-8} = \begin{cases} 0.000390625 & \text{if } a_8 = 1, \\ 0 & \text{if } a_8 = 0. \end{cases}$$

Note that the membership functions have inexact values due to the limited capacity of the grid of fractional numbers representation. For example, for

$$\mu = 0.1 \Rightarrow 2^{-4} + 2^{-5} + 2^{-7} + 2^{-8} = 0.06250_{10} + 0.03125_{10} + 0.0015625_{10}$$

$$+ 0.007825_{10} + 0.00390625_{10} = 0.096484375_{10} = 0.00011111_2;$$

$$\mu = 0.2 \Rightarrow 2^{-3} + 2^{-4} + 2^{-5} + 2^{-6} + 2^{-8} = 0.125 + 0.0625 + 0.03125$$

$$+ 0.0007825 + 0.00390625 = 0.20079375 = 0.00111011_2;$$

$$\mu = 0.3 \Rightarrow 2^{-2} + 2^{-4} = 0.25 + 0.0625 = 0.3125 = 0.010100_2;$$

$$\mu = 0.4 \Rightarrow 2^{-2} + 2^{-3} + 2^{-5} = 0.25 + 0.125 + 0.03125 = 0.40625 = 0.011010_2;$$

$$\mu = 0.5 \Rightarrow 2^{-1} = 0.5 = 0.100000_2;$$

$$\mu = 0.6 \Rightarrow 2^{-1} + 2^{-4} + 2^{-5} + 2^{-6} + 2^{-7} + 2^{-8} = 0.5 + 0.0625 + 0.03125$$

$$+ 0.0015625 + 0.00078125 + 0.000390625 = 0.596484375 = 0.10011111_2;$$

$$\mu = 0.7 \Rightarrow 2^{-1} + 2^{-3} + 2^{-4} + 2^{-5} + 2^{-7} + 2^{-8} = 0.5 + 0.125 + 0.0625$$

$$+ 0.03125 + 0.007825 + 0.00390625 = 0.703125 = 0.101101_2;$$

$$\mu = 0.8 \Rightarrow 2^{-1} + 2^{-2} + 2^{-4} + 2^{-5} + 2^{-6} = 0.5 + 0.125 + 0.03125$$

$$+ 0.015625 = 0.796875 = 0.110111_2;$$

$$\mu = 0.9 \Rightarrow 2^{-1} + 2^{-2} + 2^{-3} + 2^{-5} = 0.5 + 0.25 + 0.125 + 0.03125 = 0.903125 = 0.111010_2;$$

$$\mu = 1.0 \Rightarrow 2^{-1} + 2^{-2} + 2^{-3} + 2^{-4} + 2^{-5} + 2^{-6} = 0.5 + 0.25 + 0.125 + 0.0625 + 0.03125$$

$$+ 0.015625 + 0.00078125 + 0.00039125 = 0.98125 = 0.11111111_2.$$

Example 3. Calculate the sum $A + B$ of two FS, where

$$A = \{0.2 \,|\, x_1 + 0.5 \,|\, x_2 + 0.4 \,|\, x_3\},$$

$$A = \{(0.001111_2) \,|\, x_1 + (0.100000_2) \,|\, x_2 + (0.011010_2) \,|\, x_3\};$$

$$B = \{0.4 \,|\, x_1 + 0.6 \,|\, x_2 + 0.1 \,|\, x_3\},$$

$$B = \{(0.011010_2) \,|\, x_1 + (0.100110_2) \,|\, x_2 + (0.000111_2) \,|\, x_3\}.$$

1. Using the second method, calculate the value of the function μ^1_{A+B} for the element x_1: since $\mu^1_A(x_1)=0.2$; $\mu^1_B(x_1)=0.4$, we get

$$\mu^1_{(A+B)}(x_1)=\mu^1_A(x_1)+\mu^1_B(x_1)-\mu^1_A(x_1)\cdot\mu^1_B(x_1)=0.2+0.4-(0.2\times0.4)=0.52.$$

First, let us calculate the value of the product of the membership functions $\mu^1_A(x_1)\cdot\mu^1_B(x_1)$:

$$\mu^1_A(x_1)\cdot\mu^1_B(x_1)=(0.001111_2)\cdot(0.011010_2)=0.000110_2=0.09375_{10},$$

then the difference $\mu^1_B-\mu^1_A\cdot\mu^1_B$:

$$\mu^1_B-\mu^1_A\cdot\mu^1_B=0.011010_2-0.000110_2=0.010100_2=0.3125_{10}.$$

Therefore, the value of the membership function $\mu^1_{A+B}(x_1)$ is

$$\mu^1_{A+B}(x_1)=\mu^1_A(x_1)+\mu^1_B(x_1)-\mu^1_A(x_1)\cdot\mu^1_B(x_1)=0.001111_2+0.010100_2$$

$$=0.100011_2=0.546875_{10}.$$

2. Calculate the value of the function μ^2_{A+B} for the element x_2:

$$\mu^2_A(x_2)=0.5;\ \mu^2_B(x_2)=0.6;\ \mu^2_{A+B}(x_2)=0.5+0.6-(0.5\cdot0.6)=0.8.$$

First, let us calculate the value of the first operation, which is the product of the membership functions $\mu^2_A(x_2)\cdot\mu^2_B(x_2)$:

$$\mu^2_A(x_2)\cdot\mu^2_B(x_2)=(0.100000_2)\cdot(0.100111_2)=0.010011_2=0.296875_{10},$$

then the difference $\mu^2_B-\mu^2_A\cdot\mu^2_B$:

$$\mu^2_{A+B}(x_2)=\mu^2_B(x_2)-\mu^2_A(x_2)\cdot\mu^2_B(x_2)=0.100111_2-0.010011_2$$

$$=0.010100_2=0.296875_{10}.$$

Therefore, the value of the membership function $\mu^2_{A+B}(x_2)$ is

$$\mu^2_{A+B}(x_2)=\mu^2_A(x_2)+\mu^2_B(x_2)-\mu^2_A(x_2)\cdot\mu^2_B(x_2)$$

$$=0.100000_2+0.010100_2=0.110101_2=0.8125_{10}.$$

3. Calculate the value of the membership function μ^3_{A+B} for the element x_3:

$$\mu^3_A(x_3)=0.4;\ \mu^3_B(x_3)=0.1;\ \mu^3_{A+B}(x_3)=0.1+0.4-(0.1\times0.4)=0.46.$$

First, let us calculate the value of the first operation, which is the product of the membership functions $\mu^3_A(x_3)\cdot\mu^3_B(x_3)$:

$$\mu^3_A(x_3)\cdot\mu^3_B(x_3)=(0.011010_2)\cdot(0.000111_2)=0.000010_2=0.03125_{10},$$

then the difference $\mu_B^3 - \mu_A^3 \cdot \mu_B^3$:

$$\mu_{A+B}^3(x_3) = \mu_B^3(x_3) - \mu_A^3(x_3) \cdot \mu_B^3(x_3) = 0.000111_2 - 0.000010_2$$

$$= 0.000101_2 = 0.078125_{10}.$$

The final value of the membership function $\mu_{A+B}^3(x_3)$ is

$$\mu_{A+B}^3(x_3) = \mu_A(x_3) + \mu_B^3(x_3) - \mu_A^3(x_3) \cdot \mu_B^3(x_3)$$

$$= 0.011010_2 + 0.000100_2 = 0.011110_2 = 0.46875_{10}.$$

Operations on other elements of the FS are performed similarly.

Consider the complexity of representing numbers in the first and second ways, as well as the complexity of performing operations in these ways.

For the first method, it is known that the algorithm of converting an integer *n* from one number system to another has the complexity $O((\log n)^2)$ regardless of the radix [12]. However, the question arises: how to approximate the result under conditions of limited digit capacity? The same algorithm is used to this end. Indeed, let us convert the obtained result of multiplication from the binary to decimal representation, keep the first three decimal digits, and delete the remaining digits. Then again convert the obtained number to binary one. Thus, the complexity will double; however, the adequacy of the result is guaranteed.

To implement the project, taking into account time delays, let us calculate the sum of two FS by the second method.

Example 4. Let the FS A and B be the same as in Example 3, and the same FS be calculated:

$$A = \{\mu_A^1 | x_1 + \mu_A^2 | x_2 + \mu_A^3 | x_3 + \mu_A^4 | x_4 + \mu_A^5 | x_5\},$$

$$B = \{\mu_B^1 | x_1 + \mu_B^2 | x_2 + \mu_B^3 | x_3 + \mu_B^4 | x_4 + \mu_B^5 | x_5\}.$$

The obtained set C has the form:

$$C = A + B = \{\mu_C^1 | x_1 + \mu_C^2 | x_2 + \mu_C^3 | x_3 + \mu_C^4 | x_4 + \mu_C^5 | x_5\}.$$

To calculate the value of the operation of algebraic sum $\mu_{A+B}^i(x_i) = \mu_A^i(x_i) + \mu_B^i(x_i) - \mu_A^i(x_i) \cdot \mu_B^i(x_i)$, we will use the following algorithm.

Algorithm (*x*, *y*) (16-bit components).

Input: vector x of values of the membership function of FS A, vector y of values of the membership function of FS B.

Output: vector z of values of the membership function of FS $A + B$.

Method:

1. $x[1.5] := \mu_A^1 = 0.2;\ \mu_A^2 = 0.5;\ \mu_A^3 = 0.4;\ \mu_A^4 = 0.7;\ \mu_A^5 = 0.7;$

   $y[1.5] := \mu_B^1 = 0.4;\ \mu_B^2 = 0.6;\ \mu_B^3 = 0.1;\ \mu_B^4 = 0.3;\ \mu_B^5 = 0.8;$

2. For $i = 1$–5, do

2.1. Calculate $\mu_{C=A+B}^i(x_i) = \mu_A^i(x_i) + \mu_B^i(x_i) - \mu_A^i(x_i) \cdot \mu_B^i(x_i)$.

2.2. Subtract half of the lower-order digits from the multiplication result.

2.3. Assign the obtained value $z[i]\ (* = \mu_{A+B}(x_i) *)$.

3. Obtain vector z of values of the membership function of FS $A + B$.

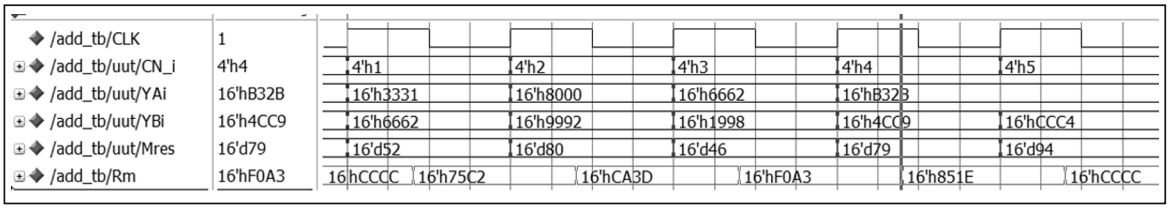

Fig. 9. Timing diagram of the algorithm of implementing FS addition operations.

Let us implement and model the project (taking into account time delays) of the algorithm of calculating the operations of the algebraic sum of fuzzy sets (elements of the FS are presented in direct codes) based on FPGA microcircuits (series XA6SLX9-2FTGI 256) using CAD ISE 14.07 (Integrated Synthesis Environment) Foundation by Xilinx and modeling system ModelSimSE 10.4c. The simulation results (the time diagram is presented in Fig. 9) confirm the correct operation of the structure for implementation of the arithmetic operation of addition of FS and have the following notation:

for $i = 1 = CN\_1 = 4'h1$

$$\mu_A^1 = 0.2 = YA1 = 16'h3331,\ \mu_B^1 = 0.4 = YB1 = 16'h6662,$$

$$\mu_C^1 = 0.52 = Mres = 16'd52,\ Rm = 16'75C2;$$

for $i = 2 = CN\_2 = 4'h2$

$$\mu_A^2 = 0.5 = YA2 = 16'h8000,\ \mu_B^2 = 0.6 = YB2 = 16'h9992,$$

$$\mu_C^2 = 0.52 = Mres = 16'd80,\ Rm = 16'C3AD;$$

for $i = 3 = CN\_3 = 4'h3$

$$\mu_A^3 = 0.4 = YA3 = 16'h6662,\ \mu_B^3 = 0.1 = YB3 = 16'h1998,$$

$$\mu_C^3 = 0.46 = Mres = 16'd46,\ Rm = 16'hF0A3;$$

for $i = 4 = CN\_4 = 4'h4$

$$\mu_A^4 = 0.7 = YA4 = 16'hB32B,\ \mu_B^4 = 0.3 = YB4 = 16'h4CC9,$$

$$\mu_C^4 = 0.79 = Mres = 16'd79,\ Rm = 16'h851E;$$

for $i = 5 = CN\_5 = 4'h5$

$$\mu_A^5 = 0.7 = YA5 = 16'hB32B,\ \mu_B^5 = 0.8 = YB5 = 16'hCCC4,$$

$$\mu_C^5 = 0.94 = Mres = 16'd94,\ Rm = 16'hCCCC.$$

In the considered Example 4, the algebraic operation of the sum of FS is implemented with the clock frequency of 375 MHz (synchronization period of 2.66 ns).

CONCLUSIONS

We have considered an automaton interpretation of algebraic operations on fuzzy sets and give an example of hardware implementation of algebraic operations with the help of the corresponding networks of automata. If an analytical expression of the FS is given, then an adaptive logic network, which calculates its value, is synthesized

based on the structure of this expression. The use of the identities of the algebra of fuzzy sets makes it possible to optimize the expression, and this optimizes the hardware by reconfiguring the logic network. To implement the algebraic operations, we used automata and networks of automata of the types $A_{\min}$ ($A_{\cap}$), $A_{\max}$ ($A_{\cup}$), A_{-}, A_{+}, A_{01}, and A_{2}.

REFERENCES

1. S. L. Kryvyi, V. N. Opanasenko, and S. B. Zavyalov, "Logical operations over fuzzy sets and relations in automaton interpretation," Cybern. Syst. Analysis, Vol. 56, No. 6, 1012–1020 (2020). https://doi.org/10.1007/s10559-020-00321-x.
2. V. N. Opanasenko and S. L. Kryvyi, "Synthesis of neural-like networks on the basis of conversion of cyclic Hamming codes," Cybern. Syst. Analysis, Vol. 53, No. 4, 627–635 (2017). https://doi.org/10.1007/s10559-017-9965-z.
3. S. L. Kryvyi, V. M. Opanasenko, and S. B. Zavyalov, "Partitioning a set of vectors with integer coordinates by means of logical hardware," Cybern. Syst. Analysis, Vol. 55, No. 3, 462–473 (2019). https://doi.org/10.1007/s10559-019-00154-3.
4. S. L. Kryvyi and V. M. Opanasenko, "Partitioning a set of vectors with nonnegative integer coordinates using logical hardware," Cybern. Syst. Analysis, Vol. 54, No. 2, 310–319 (2018). https://doi.org/10.1007/s10559-018-0033-0.
5. Y. P. Kondratenko and Ie. V. Sidenko, "Decision-making based on fuzzy estimation of quality level for cargo delivery," in: Recent Developments and New Directions in Soft Computing. Studies in Fuzziness and Soft Computing, Vol. 317, Springer, Cham (2014), pp. 331–344. https://doi.org/10.1007/978-3-319-06323-2_21.
6. A. V. Palagin, V. N. Opanasenko, and S. L. Kryvyi, "Resource and energy optimization oriented development of FPGA-based adaptive logical networks for classification problem," in: V. Kharchenko, Y. Kondratenko, and J. Kacprzyk (eds.), Green IT Engineering: Components, Networks and Systems Implementation, Vol. 105 (2017), pp. 195–218. https://doi.org/10.1007978 -3-319-55595-9_10.
7. A. N. Borisov, A. V. Alekseev, and G. V. Merkur'eva, Processing of Fuzzy Information in Decision-Making Systems [in Russian], Radio i Svyaz', Moscow (1989).
8. A. Palagin and V. Opanasenko, "The implementation of extended arithmetics on FPGA-based structures," in: Proc. IEEE 9th Intern. Conf. on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS 2017), Vol. 2, Bucharest, Romania (2017), pp. 1014–1019. https://doi.org/10.1109/IDAACS.2017.8095239.
9. Y. P. Kondratenko and N. Y. Kondratenko, "Soft computing analytic models for multiplication of asymmetrical fuzzy numbers," in: Recent Developments and the New Direction in Soft-Computing Foundations and Applications. Studies in Fuzziness and Soft Computing, Vol. 393, Springer, Cham (2021), pp. 201–214. https://doi.org/10.1007/978-3-030-47124-8_17.
10. J. Drozd, O. Drozd, S. Antoshchuk, A. Kucshnerov, and V. Nikul, "Effectiveness of matrix and pipeline FPGA-based arithmetic components of safety-related systems," in: Proc. 8th IEEE Intern. Conf. on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS 2015) (Warsaw, Poland, 24–26 Sept, 2015), Vol. 2 (2015), pp. 785–789. https://doi.org/10.1109/IDAACS.2015.7341410.
11. R. Bellman and L. Zadeh, Decision-Making under Fuzzy Conditions. Analysis and Decision-Making Procedures [Russian translation], Mir, Moscow (1976), pp. 172–215.
12. S. L. Kryvyi, An Introduction to the Methods of Creating Software Products [in Ukrainian], NaUKMA, Kyiv (2018).