DOI 10.1007/s10559-018-0033-0
Cybernetics and Systems Analysis, Vol. 54, No. 2, March, 2018

NEW MEANS OF CYBERNETICS, INFORMATICS, COMPUTER ENGINEERING, AND SYSTEMS ANALYSIS

PARTITIONING A SET OF VECTORS WITH NONNEGATIVE INTEGER COORDINATES USING LOGICAL HARDWARE

S. L. Kryvyi[1] **and V. M. Opanasenko**[2]

UDC 51.681.3

Abstract. *This article considers problems of partition of a set of vectors with nonnegative integer coordinates into two classes with respect to a threshold value and a threshold relation using adaptable logical networks. The correctness of the corresponding algorithms for implementing such a partition is proved for an arbitrary threshold value and a dimension of vectors.*

Keywords: *Boolean function, classification, integer vector, threshold relation.*

INTRODUCTION

The adaptation of hardware to the solution of the problem of partitioning a full set of values of Boolean functions based on a threshold value and a threshold relation was considered in [1]. This approach is well-known as "the technology of reconfigurable computing" [2, 3], and its embodiment in real projects became possible owing to the appearance of programmable logic integrated circuits (PLIC). In [4-8], the problem of adaptation of hardware with a formalized substantiation of algorithms of adapting structures of adaptive logic networks (ALNs) to the implementation of classification algorithms was considered.

This article considers the problem of classification of a given set of n-dimensional vectors with nonnegative integer coordinates on the basis of the well-known method of partitioning a given set of vectors into subsets [7–10] with an implementation based on structures such as ALNs.

STATEMENT OF THE PROBLEM

Let $V = \{v_1 = (u_{11}, \dots, u_{1n}), v_2 = (u_{21}, \dots, u_{2n}), \dots, v_m = (u_{m1}, \dots, u_{mn})\}$ be a set n-dimensional vectors, where $u_{ij} \in N$ and N is the set of natural numbers. We consider the following classification problems.

Problem 1. Find the partition of the set V into two subsets V_1 and V_2 under the following condition: if, for given j, $1 \le j \le n$, and a fixed $k \in N$, a coordinate u_{ij} is smaller than or equal to k, then a vector $v_i \in V_1$ and, otherwise, $v_i \in V_2$ for all i, where $1 \le i \le m$.

Problem 2. Find the partition of the set V into two subsets V_1 and V_2 under the following condition: if, for all $1 \le i \le m$, $1 \le j \le n$, and a fixed $k \in N$, the value of u_{ij} is smaller than or equal to k (for all coordinates of vectors from V), then $v_i \in V_1$ and, otherwise, $v_i \in V_2$.

[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, *opanasenkovm@nas.gov.ua*. Translated from Kibernetika i Sistemnyi Analiz, No. 2, March–April, 2018, pp. 151–161. Original article submitted May 29, 2017.

Problem 3. Find the partition of the set V into two subsets V_1 and V_2 under the following condition: for a given vector $a=(a_1,\dots,a_n)$, which belongs to the set V or does not to belong to it, and a fixed $k\in N$, assign $v_i\in V$ to the set V_1 if the vector $v=v_i-a=(u_{i1}-a_1,\dots,u_{in}-a_n)$ has all coordinates smaller or equal to k and, otherwise, assign v_i to V_2.

Note that Problems 1 and 2 are particular cases of Problem 3. In fact, Problems 1 and 2 turn out to be Problem 3 if $a=(0,\dots,0)$ is a zero vector.

The above statement of the classification problem makes it possible to reduce the solution of Problems 1–3 to the problem of classification of Boolean functions based on a threshold value and a threshold relation [1].

REDUCING PROBLEMS TO BOOLEAN FUNCTIONS

We represent coordinates of vectors from V in binary notation, i.e., in the form of words over the alphabet $X=\{0,1\}$. Let $t=\max\limits_{1\le i\le m,\,1\le j\le n}(u_{ij}+a_i)$ be the greatest value of coordinates of vectors from V. Then the length of each vector does not exceed the value of $m\cdot\log t$, and the lengths of coordinates do not exceed $\log t$. As a result, vectors from V assume the form $v_i=(p_{i1},p_{i2},\dots,p_{in})$, where p_{ij} is a word of length $\le\log t$ in the binary alphabet X. We refine the partition problem for Boolean vectors and describe the computational structure with the help of which the partition is performed.

Let $X'=\{x_1,x_2,\dots,x_n\}$ be an alphabet of Boolean variables assuming their values in the set $X=\{0,1\}$. Let also $v_i=(p_{i1},p_{i2},\dots,p_{in})$ be a Boolean vector, and let A be the set of all vectors that correspond to vectors from V. We call the set A the full set corresponding to the set of vectors V. We fix some vector a that has the same dimension as vectors from V (the vector a does not necessarily belong to V) and a binary relation R on the set A. We call the vector a a threshold vector and the relation R a threshold relation. This article considers the case of the following four binary threshold relations specified on the set $A: R_1(<), R_2(>), R_3(\le)$, and $R_4(\ge)$.

The problem of partitioning the set A into two subsets with respect to the threshold value a and threshold relation R is to divide the set A into two subsets A_1 and A_2 such that $A_1=\{x\in A:(x,a)\in R\}$ and $A_2=\{x\in A:(x,a)\notin R\}$. For example, if $R=\le$, then

$$A_1=\{x\in A:x\le a\},\ A_2=\{x\in A:a>x\}.$$

In practice, there is no need to explicitly construct a partition of the set A, and only the value of the following function corresponding to the relation R should be computed:

$$\varphi_R(x,a)=\begin{cases}1 & \text{if } (x,a)\in R;\\ 0 & \text{if } (x,a)\notin R.\end{cases}$$

It follows from the definition of this function that it assumes the value 1 on the set A_1 and the value 0 on the set A_2.

(In this article, some calculations from [1] are used).

DIRECT ALGORITHM FOR SOLVING THE PROBLEM

Since the set of values consists of Boolean words, i.e., words over the alphabet $X=\{0,1\}$, the following bit-by-bit Boolean operations over these words are naturally introduced: + (conjunction), & (disjunction), – (negation), and $\oplus$ (modulo 2 addition). The following logical functions are constructed from these operations:

$$a+b,\ \bar a+b,\ a+\bar b,\ \bar a+\bar b,\ a\&b,\ \bar a\&b,\ a\&\bar b,\ \bar a\&\bar b,\ a\oplus b,\ a\oplus\bar b,\ \bar a,\ \bar b.$$

The result of computing the value of the function $\varphi_R(x,a)$ is obvious since the threshold relation R can be extended to digits. Then, comparing the threshold value of the vector a with the values of the current vector x beginning with high-order digits, we obtain $(x,a)\in R$ whose kth digits are different, all previous digits are identical, and $(x_k,a_k)\in R$, where x_k and a_k are the kth digits (kth symbols) of the words x and a, respectively. If all digits of the words a and x are identical, then $(x,a)\in R\Leftrightarrow R$ is one of the relations $\{\le,\ge\}$.

Based on this description, we obtain an obvious algorithm for solving the partition problem. Let $a = a_n a_{n-1} \ldots a_2 a_1$ be the value of the threshold vector, and let $x = x_n x_{n-1} \ldots x_2 x_1$ be the current vector whose coordinate values are applied to the input of the algorithm, where a_k is the kth digit of the threshold vector a. When using this notation, the algorithm assumes the following form.

PARTITION $(\boldsymbol{a}, \boldsymbol{R}, \boldsymbol{x})$

Input: a is a threshold vector, R is a threshold relation, and x is an input vector.

Output: the value of the function $\varphi_R(x, a)$.

Method:

```
begin
  $i = n$;
  while $(x_i = a_i \wedge i \neq 0)$ do $i = i - 1$ od
  if $i = 0$ then if $R \in \{\leq, \geq\}$ then return (1) else return (0)
  else if $(x_i, a_i) \in R$ then return (1) else return (0)
end
```

The correctness of this algorithm is obvious and does not require any substantiation. We call it a direct algorithm.

Example 1. Assume that $a = 11001$, $x = 11010$, $y = 01110$, $z = 11001$, and $R_1 = <$. Then, for R_1 and x, y, and z, we obtain the following values generated by the algorithm:

— for x, the first three digits are identical and the other digits are different, i.e., $a_2 \neq x_2$ and $a_2 < x_2$ and, therefore, $a < x$ and $(x, a) \notin R_1$; hence, the final value of the predicate $\varphi_R(x, a)$ equals 0;

— for y, we have $a_5 \neq y_5$ and $a_5 > y_5$ and, therefore, $y < a$ and the final value of the predicate $\varphi_R(x, a)$ equals 1;

— for z, all digits are equal to the corresponding digits of a, and since $R_1 \notin \{\leq, \geq\}$, the final value of the predicate $\varphi_R(x, a)$ equals 0.

ALGORITHM FOR SOLVING THE PROBLEM ON AN ALN

The situation with substantiating the algorithm changes if there is a fixed ALN consisting of universal logical elements (implementing an arbitrary logical function) that should be used to implement the function $\varphi_R(x, a)$ computed by the algorithm PARTITION. A feature of the structure of ALN elements is that they can be tuned to implementing any logical function $+$, $\&$, $\oplus$, or $-$ [11]. It should be noted that the digits of the current word x are applied to the input of the structure in such a way that the high-order digit is applied to each lower level node and to all subsequent levels as one of arguments.

The solution of the partition problem on the computational structure shown in Fig. 1 is informally described as follows. The formation of the environment of computations based on this structure is realized by tuning each of its levels to a given logical function depending on the value of the threshold of the vector a and the threshold relation R. The type of the logical function for the ith level $(i \in [n-1, 2])$ is determined according to the rule

$$F_i^R = a + b \text{ if } a_i = 0;$$

$$F_i^R = a \& b \text{ if } a_i = 1. \tag{1}$$

For the threshold relation and some threshold vector a, the computation of the value of the function $\varphi_R(x, a)$ differs from the accepted rule. Such vectors are called singular points of the threshold relation.

Before determining the values of threshold relations at singular points, we will consider the computational structure presented in Fig. 1 as an illustrative example.

Example 2. Put $n = 5$, $R = <$, $a = 10101$, $x = 10111$, and $y = 01010$. Then, at each level of the structure, for a and x, we have the following values: $0111 \to 111 \to 11$, which determines the value 0 for the threshold relation R and, for a and y, the values $0000 \to 000 \to 00$, which determines the value 1 for the threshold relation R. This means that $\varphi_R(x, a) = 0$ and $\varphi_R(y, a) = 1$ since $a < x$ and $y < a$.

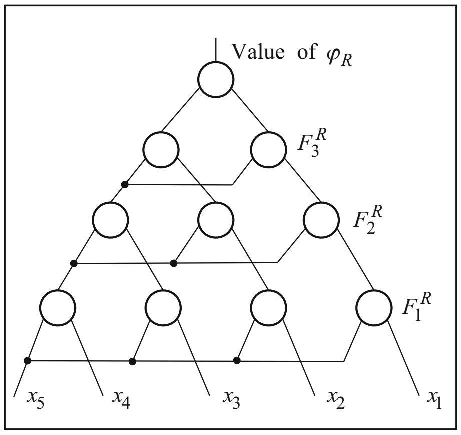

Fig. 1. Structure of a computing environment ($n = 5$).

It follows from this example that the final value of the function $\varphi_R(x, a)$ for a given threshold and a given specified relation is determined by the values of the last two bits obtained at the level $i = 1$. Let $a_2\ a_1$ be the last two bits obtained as a result of computation. Then the values of the threshold relation at this level of the computational structure are specified by the following functions:

$$a_2 = 0,\ a_1 = 0 \text{ implies } F_1^{R_2} := a + b,\ F_1^{R_3} := \bar{a} \& \bar{b};$$

$$a_2 = 0,\ a_1 = 1 \text{ implies } F_1^{R_1} := \bar{a} \& \bar{b},\ \ F_1^{R_2} := a,\ F_1^{R_3} := \bar{a},\ \ F_1^{R_4} := a + b; \tag{2}$$

$$a_2 = 1,\ a_1 = 0 \text{ implies } F_1^{R_1} := \bar{a},\ F_1^{R_2} := a \& b,\ F_1^{R_3} := \bar{a} + \bar{b},\ \ F_1^{R_4} := a;$$

$$a_2 = 1,\ a_1 = 1 \text{ implies } F_1^{R_1} := \bar{a} + \bar{b},\ F_1^{R_4} := a \& b.$$

Using these functions in Example 2, we obtain the required values.

We now consider threshold relations and their singular points. Tuning to the function $\varphi_R(x, a)$ of computing the value of the threshold relation is performed by the analysis of the first bit of the threshold vector and the threshold relation of one of the following types.

Threshold relation R_1 =<. The singular point of this relation is the value of the threshold $a = 100\ldots0$. Then the value of R_1 is computed with the help of the functions

$$F_i^{R_1} = \bar{a},\ F_{i-1}^{R_1} = a \& b.$$

For the level $i = 1$, the value of the relation is equal to zero if zero is obtained at the lower level and is equal to one if one is obtained at the lower level. In particular, assume that $a = 10000$, $x = 11001$, and $y = 01101$. Then we have the following values at levels for a and x:

$$0000 \to 000 \to 00 \to 0 \text{ since } a < x;$$

for a and y, we obtain

$$1111 \to 111 \to 11 \to 1 \text{ since } y < a.$$

Threshold relation R_2 =>. The singular point of this relation is the value of the threshold $a = 011\ldots1$. Then the value of R_1 is computed with the help of the functions

$$F_i^{R_2} = a,\ F_{i-1}^{R_2} = a \& b.$$

For the level $i = 1$, the value of the relation is equal to zero if zero is obtained at the lower level and is equal to one if one is obtained at the lower level. For example, assume that $a = 01111$, $x = 11001$, and $y = 01101$. Then we have the following values at levels for a and x:

$$1111 \to 111 \to 11 \to 1 \text{ since } a < x;$$

for a and y, we obtain the value

$$0000 \to 000 \to 00 \to 0 \text{ since } y < a.$$

Threshold relation R_3 $= \leq$. The singular point of this relation is the value of the threshold $a = 011\ldots1$. Then the value of R_3 is computed with the help of the functions

$$F_i^{R_3} = \bar{a}, \; F_{i-1}^{R_3} = a \& b.$$

For the level $i = 1$, the value of the relation is equal to zero if zero is obtained at the lower level and is equal to one if one is obtained at the lower level. For example, assume that $a = 01111$, $x = 11001$, and $y = 01101$. Then we have the following values at levels for a and x:

$$0000 \to 000 \to 00 \to 0 \text{ since } a < x;$$

for a and y, we obtain

$$1111 \to 111 \to 11 \to 1 \text{ since } y < a.$$

Threshold relation R_4 $= \geq$. The singular point of this relation is the value of the threshold $a = 100\ldots0$. Then the value of R_4 is determined with the help of the functions

$$F_i^{R_4} = a, \; F_{i-1}^{R_4} = a \& b.$$

For the level $i = 1$, the value of the relation is equal to zero if zero is obtained at the lower level and is equal to one if one is obtained at the lower level. For example, assume that $a = 01111$, $x = 11001$, and $y = 01101$. Then we have the following values at levels for a and x:

$$1111 \to 111 \to 11 \to 1 \text{ since } a < x;$$

for a and y, we obtain

$$0000 \to 000 \to 00 \to 0 \text{ since } y < a.$$

SUBSTANTIATION OF THE ALGORITHM

The algorithm implemented on the computational structure presented in Fig. 1 is called PART-ALN and is considered to be correct if the value found by this algorithm for any a, x, and R coincides with the value of the function $\varphi_R(x, a)$ computed by the algorithm PARTITION.

The complete substantiation of the correctness of operation of this algorithm is presented in [1] where the following statement is proved.

THEOREM 1. The algorithm PART-ALN is correct.

The proof of the theorem is based on several lemmas of the same type for each type of the threshold relation. To demonstrate the method of proof, we restrict ourselves to the proof of only Lemma 1, and the formulations of other lemmas are presented in [1].

LEMMA 1. The algorithm PART-ALN correctly computes the value of $\varphi_{R_1}(x, a)$ for the threshold relation $R_1 = <$.

Proof. Let $a = a_n a_{n-1} \ldots a_2 a_1$ be a threshold value, and let $x = x_n x_{n-1} \ldots \ldots x_2 x_1$ be an input vector. Let also $a_j = x_j$ for all $j = n, \ldots, i+1$ and $a_i \neq x_i$ and, at the same time, let $a = a_j a_{j-1} \ldots a_2 a_1$ do not contain the singular point of the relation R_1 for $j = n, \ldots, i+1$. Let us consider possible cases.

1. Let $a_i = 1$, and let $x_i = 0$.

Then, at all subsequent levels, according to the PART-ALN algorithm, before the appearance of a singular point, we obtain vectors consisting of zero strings.

If we obtain $a_2 a_1 = 00$ at the end of the threshold value, then this means that, at the kth level ($i \le k < 3$), there is a singular point of the relation R_1. Then, at this level, the computational structure is adjusted to the functions $F_i^{R_1} = \bar{a}$ and $F_{i-1}^{R_1} = a \,\&\, b$ and, beginning with this level, all input values become equal to one. Therefore, at the output of the computational structure, we obtain $F_1^{R_1} = a \,\&\, b = 1$. This value coincides with the value computed by the algorithm PARTITION for the function $\varphi_{R_1}(x, a) = 1$.

If we obtain $a_2 a_1 = 01$ at the end of threshold value, then $F_1^{R_1} = \bar{a} \,\&\, \bar{b}$ and, with the values $x_2 x_1 = 00$ at the output of the computational structure, we obtain $F_1^{R_1} = \bar{a} \,\&\, \bar{b} = 1$ and, with the other values of the vector $x_2 x_1$, the output values will be equal to zero. The obtained value coincides with the value computed by the algorithm PARTITION for the function $\varphi_{R_1}(x, a)$.

If we obtain $a_2 a_1 = 10$ at the end of the threshold value, then $F_1^{R_1} = \bar{a}$ and, again, with the values of $x_2 x_1 = 00$ and $x_2 x_1 = 01$, we obtain $F_1^{R_1} = \bar{a} = 1$ at the output and, for the other values of the vector $x_2 x_1$, the output values are equal to zero. The obtained values coincide with the values computed by the algorithm PARTITION for the function $\varphi_{R_1}(x, a)$.

If we obtain $a_2 a_1 = 11$ at the end of the threshold value, then $F_1^{R_1} = \bar{a} \vee \bar{b}$ and, with all values of $x_2 x_1$ except for the value of $x_2 x_1 = 11$, we obtain $F_1^{R_1} = \bar{a} \vee \bar{b} = 1$ at the output. These values coincide with the values of the function $\varphi_{R_1}(x, a)$ computed by the algorithm PARTITION.

2. Let $a_i = 0$ and $x_i = 1$. Then, according to the PART-ALN algorithm, we obtain vectors consisting of ones at all subsequent levels before the appearance of a singular point (if any).

If we obtain $a_2 a_1 = 00$ at the end of the threshold value, then this means that, at the kth level ($i \le k < 3$), there is a singular point of the relation R_1. Then the computational structure at this level is adjusted to the functions $F_i^{R_1} = \bar{a}$ and $F_{i-1}^{R_1} = a \,\&\, b$ and, beginning with this level, all input values become zero strings. Therefore, at the output of the computational structure, we obtain $F_1^{R_1} = a \,\&\, b = 0$. This value coincides with value computed by the algorithm PARTITION for the function $\varphi_{R_1}(x, a) = 0$.

If we obtain $a_2 a_1 = 01$ at the end of the threshold value, then $F_1^{R_1} = \bar{a} \,\&\, \bar{b}$ and, again, with the values of $x_2 x_1 = 00$, we obtain $F_1^{R_1} = \bar{a} \,\&\, \bar{b} = 1$ at the output of the computational structure, and, with the other values of the vector $x_2 x_1$, the output values will equal zero. The obtained value coincides with the value computed by the algorithm PARTITION for the function $\varphi_{R_1}(x, a)$.

If we obtain $a_2 a_1 = 10$ at the end of the threshold value, then $F_1^{R_1} = \bar{a}$ and, with all values of $x_2 x_1 = 00$ and $x_2 x_1 = 01$, we obtain $F_1^{R_1} = \bar{a} = 1$ at the output and, with the other values of the vector $x_2 x_1$, the output values will be equal to zero. The obtained values coincide with the values computed by the algorithm PARTITION for the function $\varphi_{R_1}(x, a)$.

If we obtain $a_2 a_1 = 11$ at the end of the threshold value, then $F_1^{R_1} = \bar{a} \vee \bar{b}$ and, with all values of $x_2 x_1$ except for the value of $x_2 x_1 = 11$, we obtain $F_1^{R_1} = \bar{a} \vee \bar{b} = 1$ at the output. The obtained values coincide with the values computed by the algorithm PARTITION for the function $\varphi_{R_1}(x, a)$.

Lemma 1 is proved.

LEMMA 2. The algorithm PART-ALN correctly computes the value of $\varphi_{R_2}(x, a)$ for the threshold relation $R_2 = >$.

LEMMA 3. The algorithm PART-ALN correctly computes the value of the function $\varphi_R(x, a)$ at the singular points of the threshold relations.

TABLE 1

Boolean Vectors of the Set A	Value of the Vector	Obtained Types of ALN Functions of			
		the 1st level ($f_1 = a + b$)	the 2nd level ($f_2 = a\&b$)	the 3rd level ($f_3 = a\&b$)	the 4th level ($f_4 := \bar{a}$)
p_1	(00111)	(0111)	(000)	(00)	1
	(10000)	(1111)	(111)	(11)	0
	(00101)	(0101)	(000)	(00)	1
p_2	(01011)	(1011)	(011)	(00)	1
	(01010)	(1010)	(010)	(00)	1
	(00011)	(0011)	(000)	(00)	1
p_3	(10001)	(1111)	(111)	(11)	0
	(10110)	(1111)	(111)	(11)	0
	(01000)	(1000)	(000)	(00)	1

SOLUTION OF PROBLEMS 1–3

It is obvious that the above ALN structure (see Fig. 1) is independent of the width of the input word and is adjusted to any threshold vector. This feature allows one to solve Problems 1–3 formulated above using the following structure.

Solution of Problem 1. The value of the parameter k used in the problem statement is represented in the form a binary word that is the threshold vector a in the problem of partitioning the set A. Specifying the threshold relation R, we obtain all necessary values for adjusting ALN.

Example 3. Let $k = 14$, and let the set of vectors V include the vectors

$$v_1 = (7, 16, 5),\ v_2 = (11, 10, 3),\ v_3 = (17, 22, 8).$$

The maximum values of coordinates of the vectors do not exceed 22, and it suffices five bits (since $\log 22 < 5$) to represent these coordinates by binary words. In this representation, to vectors of the set V correspond Boolean vectors that belong to the set A and are as follows:

$$p_1 = (00111, 10000, 00101),\ p_2 = (01011, 01010, 00011),\ p_3 = (10001, 10110, 01000).$$

If the partition is carried out with respect to the second coordinate, then the set A will be divided into the subsets $A_1 = \{v_2\}$ and $A_2 = \{v_1, v_3\}$. This is easily verified directly. To the value of the parameter k will correspond the threshold vector $p = (01110)$ (i.e., the vector a), and if the threshold relation R is given (for example, $R = <$), then all necessary values for adjusting ALN are available (the size of input words of ALN equals 5 bits, i.e., four levels). According to the algorithm PART-ALN considered above, we will determine the types of functions for all four levels of ALN for the ALN structure (see Fig. 1) on the basis of the analysis of bits (starting with the most significant bit) of the threshold vector $p = (01110)$ for $R = <$ with allowance for rules (1) and (2) as follows:

(1) the most significant bit is equal to zero and, therefore, the type of the logical function for the first level is defined as $f_1 = a + b$;

(2) the next bit is equal to one and, therefore, the type of the logical function for the second level is defined as $f_2 = a \& b$;

(3) the next bit equals one and, therefore, the type of the logical function for the third level is defined as $f_3 = a \& b$;

(4) based on the analysis of two least significant bits (01), for the threshold relation $R = <$, the type of the logical function for the last (fourth) level is defined as follows: $f_4 = \bar{a}$.

The results of transformation that are presented in Table 1 confirm that the partition is carried out with respect to the second coordinate, and the set A is divided into the subsets $A_1 = \{v_2\}$ and $A_2 = \{v_1, v_3\}$.

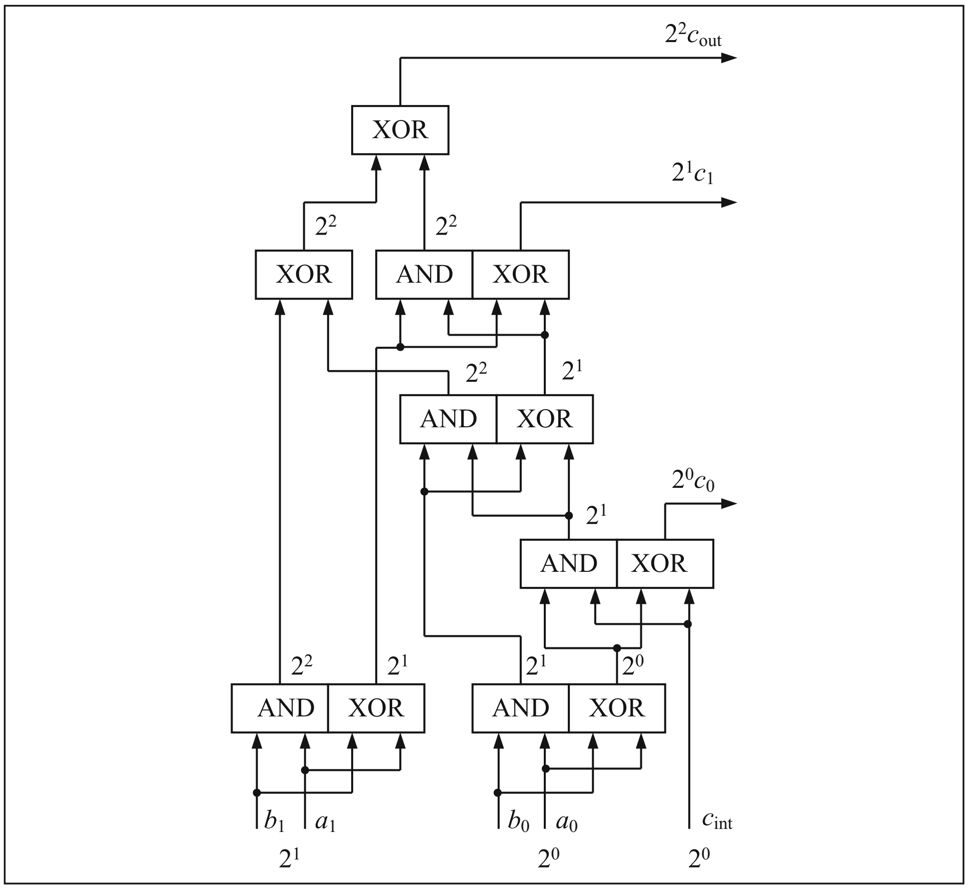

Fig. 2. Block diagram of the adder based on the basic logical operations XOR and AND.

Solution of Problem 2. The solution follows from the method of solving Problem 1. In fact, representing the coordinates of the threshold vector $a = (k, k, \ldots, k)$ in the form of binary words, we execute n iterations for each coordinate. It should be noted that this partition can be performed concurrently by adjusting each of n ALNs to the same threshold value and the same relation. In this case, further acceleration of the classification is possible as follows: if, for one coordinate of some vector, the value of the function $\varphi_R(x, a) = 0$, then further computations with this vector should not be executed since it will be assigned to the second subset V_2.

If necessary, ALN can be adjusted to different values of coordinates of the threshold vector and different relations for each coordinate separately.

Solution of Problem 3. To solve the problem, the vector a should be summed with a vector $K = (k, k, \ldots, k)$ and the obtained vector $a + K$ should be considered to be the threshold vector. Then verification is performed by the way of solution of Problem 2. If a vector v_i satisfies the threshold relation R, then it belongs to the set A_1 and, otherwise, it belongs to the set A_2.

The considered ways of solving problems imply the need for executing the addition operation. To implement this operation, we will consider the implementation of an adder based on ALN and, in this case, in contrast to the structure of the threshold device from [12], an ALN structure of the trapezoidal type is used [3], i.e., a structure with $2n+1$ inputs and α outputs ($\alpha = \log_2 n + 1$). The adder implements the following operation [13]:

$$C = A + B + c_{\text{int}} = \sum_{\lambda=0}^{n-1} 2^{\lambda} a_{\lambda} + \sum_{\lambda=0}^{n-1} 2^{\lambda} b_{\lambda} + c_{\text{int}},$$

where a_λ and $b_\lambda$are components containing the value of the λth bit of the binary representation of the binary vectors A and B, respectively; 2^λ is the weight of the λth bit of binary vectors; c_{int} are values of the carry input. The result of the adder is represented by $(n+1)$ bits (taking into account the carry output c_{out}).

The implementation of the adder is based on a bit-level (h is the number of levels of processing, and $s = 1 \div h$) transformation scheme that uses modulo 2 addition and multiplication (the logical operations XOR and AND) as basic bit operations.

Transformations are executed by pairwise processing of input vectors. Processing groups of weighted components of vectors at each level involves the formation of information bits with their current weights (the operation of modulo 2 addition) and carry bits whose weight is larger by one (the operation of modulo 2 multiplication). Thus, at the level $s = 1$, groups with the weights 2^1 and 2^0 are formed. The group with the weight 2^1 represents the carry bit created with the help of the operation AND. The group with the weight 2^0 is the information bit created with the help of the operation XOR, etc. The group with a weight 2^γ is processed until the cardinality of the weighted group becomes equal to 1, i.e., will be represented by one information bit. The information bit with the greatest weight 2^γ $(\gamma = n)$ will be created last.

Let us consider an example of synthesizing the adder for binary vectors A and B $(n = 1)$, which is based on the basic logical operations XOR and AND and whose block diagram is presented in Fig. 2.

Example 4. Assume that the vectors from the set V are such as in Example 2, that the parameter $k = 8$, that the threshold relation $R = \leq$, and that the threshold vector $a = (7, 6, 3)$. Then $a + K = (7, 6, 3) + (8, 8, 8) = (15, 14, 11)$ will be the threshold vector, and vectors v_i are divided by the ALN structure (see Fig. 1) into two subsets $V_1 = \{v_2\}$ and $V_2 = \{v_1, v_3\}$.

CONCLUSIONS

This article considers the solution of the problem of classifying a set of vectors with nonnegative integer coordinates with the help of adaptable logical networks. The presented solution is independent of the digit capacity of coordinates of vectors and is applicable to arbitrary digit capacities of coordinates of vectors and threshold values. This solution requires the execution of the operation of addition and, to this end, an adder is constructed that uses only the operations AND and XOR that underlie the construction of ALN for solving the classification problems being considered. The proposed solution is also applicable to the problem of classifying a set of vectors with negative integer coordinates. To this end, it is necessary to perform a displacement as a result of which negative quantities are transformed into nonnegative ones and, at the end of transformations, the reverse displacement is performed for obtaining true values.

REFERENCES

1. V. N. Opanasenko and S. L. Kryvyi, "Partitioning the full range of Boolean functions based on the threshold and threshold relation," Cybernetics and Systems Analysis, Vol. 48, No. 3, 459–468 (2012).
2. A. V. Palagin and V. N. Opanasenko, "Reconfigurable computing technology," Cybernetics and Systems Analysis, Vol. 43, No. 5, 675–686 (2007).
3. A. V. Palagin and V. N. Opanasenko, "Design and application of the PLD-based reconfigurable devices," in: M. Adamski, A. Barkalov, and M. Wegrzyn (eds.), Design of Digital Systems and Devices; Lecture Notes in Electrical Engineering, Springer, Berlin–Heidelberg, Vol. 79, 59–91 (2011).
4. V. N. Opanasenko and S. L. Kryvyi, "Synthesis of adaptive logical networks on the basis of Zhegalkin polynomials," Cybernetics and Systems Analysis, Vol. 51, No. 6, 969–977 (2015).
5. A. Palagin, V. Opanasenko, and S. Kryvyi, "The structure of FPGA-based cyclic-code converters," Optical Memory & Neural Networks (Information Optics), Vol. 22, No. 4, 207–216 (2013).
6. V. Opanasenko and S. Kryvyi, "Synthesis of multilevel structures with multiple outputs," in: Proc. 10th Intern. Conf. of Programming (UkrPROG'2016), Vol. 1631, Code 122904, Kyiv, Ukraine (2016), pp. 32–37.

7. Y. P. Kondratenko and E. Gordienko, "Implementation of the neural networks for adaptive control system on FPGA," in: B. Katalinic (ed.), Annals of DAAAM for 2012 & Proceeding of 23rd DAAAM International Symposium on Intelligent Manufacturing and Automation, DAAAM International, Vienna, Austria, 23 (1) (2012), pp. 0389–0392.
8. Y. P. Kondratenko and Ie. V. Sidenko, "Decision-making based on fuzzy estimation of quality level for cargo delivery," in: L. A. Zadeh et al. (eds.), Recent Developments and New Directions in Soft Computing, Studies in Fuzziness and Soft Computing, Springer, Switzerland (2014), pp. 331–344.
9. Yu. G. Krivonos, Yu. V. Krak, and M. F. Kirichenko, Modeling, Analysis, and Synthesis of Manipulation Systems [in Ukrainian], Naukova Dumka, Kyiv (2006).
10. A. I. Stasiuk and L. L. Goncharova, "Mathematical models of computer intellectualization of technologies for synchronous phasor measurements of parameters of electric networks," Cybernetics and Systems Analysis, Vol. 52, No. 5, 186–192 (2016).
11. R. K. Brayton, G. D. Hechtel, and A. L. Sangiovanni-Vincentelli, "Synthesis of multilevel combinational logic circuits," TIIER, Vol. 78, No. 2, 38–83 (1990).
12. A. V. Palagin, V. N. Opanasenko, and L. G. Chigirik, "Synthesizing a Hamming adder of arbitrary word width," Journal of Automation and Information Sciences, Vol. 27, No. 2, 39–42 (1995).
13. J. Drozd, A. Drozd, S. Antoshchuk, A. Kushnerov, 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: Technology and Applications, Warsaw, Poland (2015), pp. 785–789.