GENERATING BOOLEAN FUNCTIONS VIA ARITHMETIC POLYNOMIALS

V. L. Artyukhov, V. N. Kondrat'ev, and A. A. Shalyto

UDC 519.714

We will expand the concept of an arithmetic polynomial by introducing the "absolute value" operation. We propose methods for constructing a polynomial for Boolean functions. The conditions of polynomial linearity are defined.

1. Introduction

The possibility of using nontraditional methods to generate Boolean functions arises when microprocessors and microcomputers are used in logic control systems. These methods are based on the extensive capabilities of the indicated hardware to do arithmetic. Among these methods are, e.g., spectral methods, methods for computing threshold functions, and computing by means of arithmetic polynomials (AP).

The latter group of methods was examined in [1, 2]. This work will further investigate the issues in generating Boolean functions (BF) and systems of Boolean functions (SBF) via arithmetic polynomials.

2. Expanding the Concept of An "Arithmetic Polynomial"

It is a well-known fact that a Boolean equation based on AND, OR, and NOT operations can be generated via an AP by replacing the logical operations with arithmetic operations according to the rules:

$$x_1\&x_2=x_1x_2;\quad x_1\vee x_2=x_1+x_2-x_1x_2;\quad \bar{x}=1-x. \tag{1}$$

Notice too, that if the Boolean functions f_1 and f_2 are orthogonal ($f_1 \cdot f_2 = 0$), then

$$f_1\vee f_2=f_1+f_2=f_1\oplus f_2.$$

We will prove a theorem that frequently makes it possible to simplify an AP obtained by means of the indicated rules.

Theorem 1. Let x_1, $x_2\in\{0, 1\}$, and let N and k be random integers. Then

$$x_i=(x_i)^N, \tag{2}$$

$$x_i+(2^N-2)x_1x_2+x_2=(x_1+x_2)^N, \tag{3}$$

$$(x_1-x_2)^{2k}=|x_1-x_2|, \tag{4}$$

$$(x_1-x_2)^{2k+1}=x_1-x_2. \tag{5}$$

are valid.

Proof. The first relation is obvious. We will prove the others. To do this, we consider the binomial theorem:

$$(x_1+x_2)^N=x_1^N+Nx_1^{N-1}x_2+\frac{N(N-1)}{1\cdot 2}x_1^{N-2}x_2^2+\ldots+x_2^N. \tag{6}$$

Starting from the fact that when raising the sum of the variables x_1 and x_2 to a power the sum of the coefficients on the binomial is equal to 2^N and that Eq. (2) is valid, we can rewrite Eq. (6) as:

$$(x_1+x_2)^N=x_1+(2^N-2)x_1x_2+x_2.$$

When raising the difference between the variables x_1 and x_2 to the Nth power the sum of the coefficients on the binomial is zero, and therefore

Leningrad. Translated from Avtomatika i Telemekhanika, No. 4, pp. 138-147, April, 1988. Original article submitted December 24, 1986.

Automation and Remote Control. Vol. 49. No 4, Part 2, April, 1988.

 0005-1179/88/4904-0508$12.50

$$(x_1-x_2)^{2k+1}=x_1-x_2 \text{ when } N=2k+1;$$

$$(x_1-x_2)^{2k}=x_1-2x_1x_2+x_2=x_1^2-2x_1x_2+x_2^2=(x_1-x_2)^2=|x_1-x_2| \text{ when } N=2k.$$

are valid.

The theorem has been proved.

An AP in which the "absolute value" operation is used is called a closed arithmetic polynomial (CAP).

Example 1. We will generate the formula $f = x_1 \bullet x_2$ for a CAP. Using the relations given earlier we obtain:

$$f=x_1\oplus x_2=\bar{x}_1x_2\vee x_1\bar{x}_2=(1-x_1)x_2+x_1(1-x_2)=x_1-2x_1x_2+x_2=x_1^2-2x_1x_2+x_2^2=(x_1-x_2)^2=|x_1-x_2|.$$

It follows from this example that if f_1 and f_2 are conjunctions of rank $r \geq 1$,

$$f_1\oplus f_2=|f_1-f_2|, \tag{7}$$

and if f_1 and f_2 are not conjunctions,

$$f_1\oplus f_2=|f_1'-f_2'|, \tag{8}$$

where f_1' and f_2' are APs of the functions f_1 and f_2, respectively.

These relations make it possible to obtain a CAP for formulas of the indicated type directly.

Example 2. We generate the formula $f = (x_1 \vee x_2) \bullet x_3$ for a CAP.

It follows from Eqs. (1) and (8) that $f = |x_1 + x_2 - x_1x_2 - x_3|$; at the same time, Eq. (1) makes it possible for us to obtain the AP:

$$f=x_1+x_2+x_3-x_1x_2-2x_1x_3-2x_2x_3+2x_1x_2x_3.$$

3. Generating Boolean Functions Via Zhegalkin Polynomials

A Zhegalkin polynomial (ZhP) can be obtained from a given formula by using the relations

$$a\vee b=a\oplus b\oplus ab; \quad \bar{a}=1\oplus a.$$

We will examine the other, more usual methods of constructing ZhPs. The choice of method depends on what form (table or formula) the function being generated is in. If given as a formula, the choice of method depends on the specifics of this formula (does the formula allow simple disjunctive decomposition of Eq. (9) or not).

1. If a Boolean function has been given and it can be written [3] as

$$f(X)=f(\varphi(X_1), X_2), \tag{9}$$

where $X = X_1 \cup X_2$, $X_1 \cap X_2 = 0$, then a ZhP can be found from the relations

$$f=[\bar{\varphi}f(\varphi=0)\vee\varphi\overline{f(\varphi=1)}]\oplus\varphi =[\bar{\varphi}f(\varphi=0)\oplus\varphi\overline{f(\varphi=1)}]\oplus\varphi, \tag{10}$$

$$f=[\bar{\varphi}\overline{f(\varphi=0)}\oplus\varphi f(\varphi=1)]\oplus\bar{\varphi}, =[\bar{\varphi}\overline{f(\varphi=0)}\oplus\varphi f(\varphi=1)]\oplus\bar{\varphi}, \tag{11}$$

which are extensions of the formulas in [4] for the random functions

$$f=[\bar{x}_if(x_i=0)\vee x_i\overline{f(x_i=1)}]\oplus x_i =[x_if(x_i=0)\oplus x_i\overline{f(x_i=1)}]\oplus x_i, \tag{12}$$

$$f=[\bar{x}_i\overline{f(x_i=0)}\vee x_if(x_i=1)]\oplus\bar{x}_i =[\bar{x}_i\overline{f(x_i=0)}\oplus x_if(x_i=1)]\oplus\bar{x}_i. \tag{13}$$

Example 3. We will define a ZhP for the formula $f = x_1x_2 \vee x_3x_4$. By making repeated use of Eq. (10) we obtain

$$f=x_1x_2\vee x_3x_4=(\overline{x_3x_4}x_1x_2\oplus x_3x_4\cdot 0)\oplus x_3x_4=x_1x_2\overline{x_3x_4}\oplus x_3x_4=$$
$$=(\overline{x_1x_2}\cdot 0\oplus x_1x_2\overline{x_3x_4})\oplus x_1x_2\oplus x_3x_4=x_1x_2x_3x_4\oplus x_1x_2\oplus x_3x_4.$$

Note that the repeated use of Eq. (10) in this case can be avoided if we consider that $\overline{x_3x_4} = 1 \bullet x_3x_4$.

2. If a Boolean formula that can be written as Eq. (9) has been given, a ZhP can also be found by using

$$f=f(\varphi=0)\oplus[f(\varphi=0)\oplus f(\varphi=1)]\cdot\varphi, \tag{14}$$

which is an extension of the Reed's expansion [3] proposed for random formulas:

$$f=f(x_i=0)\oplus[f(x_i=0)\oplus f(x_i=1)]\oplus x_i. \tag{15}$$

Example 4. We will find a ZhP for the formula $f = x_1x_2 \vee x_3x_4$. Using Eq. (14), we find

$$f=x_1x_2\oplus(x_1x_2\oplus 1)x_3x_4=x_1x_2\oplus x_1x_2x_3x_4\oplus x_3x_4.$$

3. A third approach is the following: the orthogonal normal disjunctive form (ONDF) for the general case is found by repeated use of the rule

$$f=f_1\vee f_2=f_1\vee\bar{f}_1f_2.$$

The $\vee$ symbol in the ONDF is replaced by $\bullet$ and $\bar{x}_i$ is replaced by $1\bullet x_i$. Then the formulas are multiplied out and similar terms, if they exist, are reduced.

Notice that the amount of effort needed to construct the ONDF depends on the order in which the formulas are written. If, when this is done, φ_1 has h_1 letters and φ_2 has h_2, $h_1 < h_2$, then $\varphi=\varphi_1\vee\varphi_2$. If $h_1 > h_2$, then $\psi=\varphi_2\vee\varphi_1$.

Example 5. We will define a ZhP for the formula $f = (x_1\vee x_2)x_3\vee x_1x_2$. It follows from what has been said previously that $f = (x_1\vee x_2)x_3\vee x_1x_2 = x_1x_2\vee(x_1\vee x_2)x_3 = x_1x_2\vee\overline{x_1x_2}(x_1\vee x_2)x_3 = x_1x_2\vee(\bar{x}_1x_2\vee x_1\bar{x}_2x_3) = x_1x_2\bullet(x_1\bullet x_2)x_3 = x_1x_2\bullet x_1x_3\bullet x_2x_3$.

4. The matrix method of [5] for constructing a ZhP is based on specifying the function in absolute normal disjunctive form (ANDF) or in a truth table. The method is based on using a binary Pascal's triangle [6] found in a square matrix of size $2^n \times 2^n$ every element of which that lies above the main diagonal is equal to zero.

For this matrix the relations

$$|Q_n|=\begin{vmatrix} Q_{n-1} & 0 \\ Q_{n-1} & Q_{n-1}\end{vmatrix},\quad |Q_1|=\begin{vmatrix}1 & 0\\ 1 & 1\end{vmatrix}. \tag{16}$$

are satisfied.

The coefficients of a ZhP

$$f=g_0\oplus g_1x_n\oplus g_2x_{n-1}\oplus g_3x_{n-1}x_n\oplus\ldots\oplus g_{2^n-1}x_1x_2\ldots x_n \tag{17}$$

are associated with the coefficients of the ANDF

$$\begin{aligned} f=&y_0\bar{x}_1\bar{x}_2\ldots\bar{x}_n\vee y_1\bar{x}_1\bar{x}_2\ldots\bar{x}_{n-1}x_n\vee y_2\bar{x}_1\ldots\\ &\ldots x_{n-1}\bar{x}_n\vee\ldots\vee y_{2^n-1}x_1x_2\ldots x_n\end{aligned} \tag{18}$$

by the matrix relation

$$|G|=|Q_n|\otimes|Y|, \tag{19}$$

where $|G|$, and $|Y|$ are column matrixes of the coefficients for a ZhP and the ANDF, respectively, $\bullet$ is the symbol for the operation of multiplying matrixes in which the arithmetic sum is replaced by the sum in modulo 2.

As an example, we will reduce Eq. (19) when n = 2:

$$\begin{vmatrix}g_0\\ g_1\\ g_2\\ g_3\end{vmatrix}=\begin{vmatrix}1&0&0&0\\1&1&0&0\\1&0&1&0\\1&1&1&1\end{vmatrix}\otimes\begin{vmatrix}y_0\\ y_1\\ y_2\\ y_3\end{vmatrix}=\begin{vmatrix}y_0\\ y_0\ominus y_1\\ y_0\ominus y_2\\ y_0\ominus y_1\ominus y_2\oplus y_3\end{vmatrix}.$$

Making the reverse transition from a ZhP to an ANDF is done through

$$|Y|=|Q_n|\otimes|G|. \tag{20}$$

4. Constructing a Closed Arithmetic Polynomial for a Random Boolean Function

The CAP for a random Boolean formula (function) can be found as follows: a ZhP is defined for the given formula (function) which in the general case is then converted to a CAP by repeatedly using Eq. (7).

Example 6. We will construct a CAP for $f = (x_1\vee x_2)\bullet x_3$ by using the fourth method for constructing a ZhP.

Having found the ANDF for this function we obtain a column matrix $|Y|$ and then we compute the coefficients g_i by means of Eq. (19):

$$\begin{vmatrix} g_0 \\ g_1 \\ g_2 \\ g_3 \\ g_4 \\ g_5 \\ g_6 \\ g_7 \end{vmatrix} = \begin{vmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{vmatrix} \begin{vmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{vmatrix} = \begin{vmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{vmatrix}.$$

Consequently, $f = g_0 \oplus g_1 x_3 \oplus g_2 x_2 \oplus g_3 x_2 x_3 \oplus g_4 x_1 \oplus g_5 x_1 x_3 \oplus g_6 x_1 x_2 \oplus g_7 x_1 x_2 x_3 = x_3 \oplus x_2 \oplus x_1 \oplus x_1 x_2 = |||x_3 - x_2| - x_1| - x_1 x_2|$.

It follows from comparing the CAPs found in Example 2 and 6 that, in contrast to the situation for APs, the representation of a CAP is not unique. Therefore, to verify the construction of the CAP the transition from the CAP to an AP must be made. In doing this, to reduce the embedment depth of the "absolute value" operators, we suggest that, beginning with the most deeply embedded operators a Shannon expansion $f = (1 - x_i)f(x_i = 0) + x_i f(x_i = 1)$ and the relations $|x_i - x_j| = x_i - 2x_i x_j + x_j$, $|1 - x_i| = 1 - x_i$, $|x_i - 1| = 1 - x_i$ be used.

By way of example we will prove the validity of

$$f = |||x_1 - x_2| - x_1 x_2| - x_3| = x_1 + x_2 + x_3 - x_1 x_2 - 2x_1 x_3 - 2x_2 x_3 + 2x_1 x_2 x_3.$$

Using the first of the relations given, we find that $f = ||x_1 + x_2 - 3x_1 x_2| - x_3|$. We expand the internal operator in terms of the variable x_1:

$$\varphi_1 = |x_1 + x_2 - 3x_1 x_2| = (1 - x_1)|x_2| + x_1|1 - 2x_2|.$$

We expand the operator $\varphi_2 = |1 - 2x_2|$ in terms of the variable x_2: $\varphi_2 = (1 - x_2)|1| + x_2|-1| = 1$. In doing this, $\varphi_1 = (1 - x_1)x_2 + x_1 = x_1 + x_2 - x_1 x_2$. Thus, $f = |x_1 + x_2 - x_1 x_2 - x_3|$. Expanding this relation in terms of x_1, we obtain: $f = (1 - x_1)|x_2 - x_3| + x_1|1 - x_3| = (1 - x_1)(x_2 - 2x_2 x_3 + x_3) + x_1(1 - x_3) = x_1 + x_2 + x_3 - x_1 x_2 - 2x_1 x_3 - 2x_2 x_3 + 2x_1 x_2 x_3$.

In spite of the fact that the computational complexity of a CAP is no less than that of a ZhP when using the method shown, it can be used when logical operations are absent from the programming language as happens, e.g., in some versions of BASIC. The proposed method frequently makes it possible to obtain a record that is more compact in comparison with the "classical" APs.

The CAPs we have examined generate one Boolean function. By analogy with APs closed polynomials (CP) that generate systems of Boolean functions (SBF) can be included in our examination. For example, the SBF $f_1 = x_1 \oplus x_2$, $f_2 = x_1 \oplus x_3$ can be generated by the CP: $Y = 2|x_1 - x_2| + |x_1 - x_3|$, without introducing additional fucntions. The AP for this SBF is $Y = 3x_1 + 2x_2 + x_3 - 2x_1 x_3 - 4x_1 x_3$.

5. Generating a Boolean Formula Via an Arithmetic Polynomial

Generating a Boolean formula via an AP can be done by the following method, which differs from the direct use of Eq. (1): an ONDF is defined: the V symbol in the ONDF is replaced by a + and $\bar{x}_i$ is replaced by $1 - x_i$; the formulas are then multiplied out and similar terms, if they exist, are reduced.

Example 7. We will generate the formula $f = (x_1 \vee x_2)x_3 \vee x_1 x_2$ via an arithmetic polynomial. In doing this, $f = (x_1 \vee x_2)x_3 \vee x_1 x_2 = x_1 x_2 \vee (\bar{x}_1 \vee x_2)x_3 = x_1 x_2 \vee \overline{x_1 x_2}(x_1 \vee x_2)x_3 = x_1 x_2 \vee (\bar{x}_1 x_2 \vee x_1 \bar{x}_2)x_3 = x_1 x_2 + [(1 - x_1)x_2 + x_1(1 - x_2)]x_3 = x_1 x_2 + x_1 x_3 + x_2 x_3 - 2x_1 x_2 x_3$.

We will present, without proof, two statements.

1. If the given formula is nonrepetitive in AND, OR, and NOT the coefficients of the AP are −1, 0, and 1. The converse is not true.

2. If the coefficients are −1, 0, and 1 a corresponding ZhP can be obtained from the AP by replacing the + and − signs by •.

6. Constructing an Arithmetic Polynomial for a System of Boolean Functions

We introduce the concept of an extended absolute normal disjunctive form (EANDF) for SBFs.

In doing this, we will understand an EANDF to be an expression of the type

$$Y=y_0\bar{x}_1\bar{x}_2\ldots\bar{x}_{n-1}\bar{x}_n\vee y_1\bar{x}_1\bar{x}_2\ldots\bar{x}_{n-1}x_n\vee\ldots \\ \ldots\vee y_{2^n-1}x_1x_2\ldots x_{n-1}x_n, \tag{21}$$

where y_i is the decimal equivalent of the SBF values in a set i. The validity of Eq. (21) follows from the orthogonality of the conjunction of the EANDF.

We will seek an AP for the SBF Y as

$$Y=a_0+a_1x_n+a_2x_{n-1}+a_3x_{n-1}x_n+\ldots+a_{2^n-1}x_1x_2\ldots x_n. \tag{22}$$

Equation (22) was obtained from Eq. (21) by replacing the y_i with a_i and eliminating complemented variables.

By substituting the 2^n values of the x_1, x_2, ..., x_n and setting Eq. (21) equal to Eq. (22) we obtain 2^n equations from which expressions for the AP coefficients can be found via the EANDF coefficients by substitution.

These equations can be given in a matrix form similar to Eq. (19) and which is also constructed as a binary Pascal's triangle:

$$|A|=|P_n|\cdot|Y|, \tag{23}$$

where |A| and |Y| are respectively column matrixes for the AP and EANDF coefficients.

For a matrix $|P_n|$ the relations:

$$|P_n|=\begin{vmatrix} P_{n-1} & 0 \\ -P_{n-1} & P_{n-1}\end{vmatrix},\quad |P_1|=\begin{vmatrix} 1 & 0 \\ -1 & 1\end{vmatrix}, \tag{24}$$

are satisfied, where P_{n-1} is a matrix obtained by changing the sign of the unit elements in the P_{n-1} matrix.

By way of example we will reduce Eq. (23) when n = 2:

$$\begin{vmatrix} a_0\\ a_1\\ a_2\\ a_3\end{vmatrix}=\begin{vmatrix} 1 & 0 & 0 & 0\\ -1 & 1 & 0 & 0\\ -1 & 0 & 1 & 0\\ 1 & -1 & -1 & 1\end{vmatrix}\cdot\begin{vmatrix} y_0\\ y_1\\ y_2\\ y_3\end{vmatrix}=\begin{vmatrix} y_0\\ -y_0+y_1\\ -y_0+y_2\\ y_0-y_1-y_2+y_3\end{vmatrix}. \tag{25}$$

Example 8. WE will define an AP for the formula $f_1 = x_1x_2$, $f_2 = x_1 \bullet x_2$. Having set up a truth table (EANDF) for these functions we find that $y_0 = 0$, $y_1 = y_2 = 1$, and $y_3 = 2$. When this occurs it follows from Eq. (25) that $a_0 = 0$, $a_1 = a_2 = 1$, and $a_3 = 0$. Therefore, $Y = a_0 + a_1x_2 + a_2x_1 + a_3x_1x_2 = x_2 + x_1$.

THEOREM 2. If an SBF has been defined in $\beta(\beta \le 2^n)$ sets of input data x_1, x_2, ..., x_n, it is always possible to construct an AP that has no more than β nonzero coefficients.

Proof. It follows from Eqs. (23) and (25) that an a_i is computed via a y_i and therefore, if the y_i has not been defined its value can always be chosen so that a_i will be zero.

Example 9. We will define an AP for the SBF $Y(X) = \{f_1(x_1, x_2), f_2(x_1, x_2)\}$ specified in the three sets numbered X = {0, 1, 2} and using the values Y = {1, 2, 3} in these sets.

It follows from Eq. (25) that, having chosen $y_3 = 4$, we obtain $a_3 = 0$. In doing this

$$Y=1+x_2+2x_1.$$

The reverse transition from an AP to an EANDF is accomplished via the relation

$$|Y|=|Q_n|\cdot|A|. \tag{26}$$

7. The Condition for a System of Boolean Functions to be Representable via an Arithmetic Polynomial

The condition for an SBF to be representable via a linear AP is determined from Eq. (23) by setting the coefficients of the nonlinear terms in Eq. (22) to zero and representing them via the base values y_i, where i = 0, 1, 2, 4, ..., 2^{n-1}.

In doing this the linearity condition AP for a SBF is defined by the relations:

$$3 \leqslant l \leqslant 2^n - 1, \quad l \neq 2^2, 2^3, \ldots, 2^{n-1},$$

$$y_l = \dot{\sum_{m=[\log_2 l]}} x_{m+1} y_{2^m} \dot{+} \left(1 - \dot{\sum_{m=[\log_2 l]}} x_{m+1}\right) y_0, \tag{27}$$

$$\text{bin } l = x_{[\log_2 l]+1} x_{[\log_2 l]} \ldots x_2 x_1,$$

where y_0, y_l, and y_{2^m} are the decimal equivalents of the SBF values in the sets having the numbers 0, l, and 2^m, respectively; bin l is the binary equivalent of l.

A more obvious form of this condition is acquired in matrix form:

$$|Y_l| = |DC| \cdot |Y_b|, \tag{28}$$

where $|Y_l|$ is a column matrix of the y_l elements; $|Y_b|$ is a column matrix of the base y_0, y_1, y_2, y_4, ..., $y_{2^{n-1}}$ elements; D is a submatrix, each row of which is bin l; and C is a column vector, each value of which is equal to one minus the sum of ones in bin l.

As an example, we will reduce Eq. (28) for n = 3:

$$\begin{vmatrix} y_3 \\ y_5 \\ y_6 \\ y_7 \end{vmatrix} = \begin{vmatrix} 0 & 1 & 1 & -1 \\ 1 & 0 & 1 & -1 \\ 1 & 1 & 0 & -1 \\ 1 & 1 & 1 & -2 \end{vmatrix} \cdot \begin{vmatrix} y_4 \\ y_2 \\ y_1 \\ y_0 \end{vmatrix} = \begin{vmatrix} y_2 + y_1 - y_0 \\ y_4 + y_1 - y_0 \\ y_4 + y_2 - y_0 \\ y_4 + y_2 + y_1 - 2y_0 \end{vmatrix}. \tag{29}$$

We will point out two special cases of AP linearity.

1. If $Y = c_1 + c_2 d$ (d is the decimal equivalent of the input set), the AP will have the form

$$Y = c_1 + c_2 \sum_{i=1}^{n} 2^{i-1} x_i.$$

2. If $Y = c_1 + c_2 \sum_{i=1}^{n} x_i$, the AP will have the same form.

Example 10. We will determine whether or not the SBF $f_1 = x_1 x_2$, $f_2 = x_1 \vee x_2$, $f_3 = x_1 \oplus x_2$ can be represented by a linear AP and generate this SBF.

We write a truth table for this SBF. We include the column Y: $y_0 = 0$, $y_1 = y_2 = 3$, $y_3 = 6$. The linearity condition is satisfied:

$$|y_3| = |1 \quad 1 \quad -1| \cdot \begin{vmatrix} y_2 \\ y_1 \\ y_0 \end{vmatrix} = |y_2 + y_1 - y_0|.$$

The values of the AP coefficients are found from Eq. (25): $a_0 = 0$, $a_1 = 3$, $a_2 = 3$, $a_3 = 0$. When this is done, $Y = a_0 + a_1 x_2 + a_2 x_1 + a_3 x_1 x_2 = 3x_1 + 3x_2$. A similar result is obtained from the second special case: $n = 2$, $c_1 = 0$, $c_2 = 3$.

The linearity condition for a ZhP has the form:

$$|Y_l| = |DB| \otimes |Y_b|, \tag{30}$$

where B is a column vector, each value of which is equal to the sum in modulo 2 of the ones and the ones in bin l.

As an example we will present the condition for n = 3:

$$\begin{vmatrix} y_3 \\ y_5 \\ y_6 \\ y_7 \end{vmatrix} = \begin{vmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{vmatrix} \otimes \begin{vmatrix} y_4 \\ y_2 \\ y_1 \\ y_0 \end{vmatrix} = \begin{vmatrix} y_2 \oplus y_1 \in y_0 \\ y_4 \in y_1 \ominus y_0 \\ y_4 \ominus y_2 \ominus y_0 \\ y_4 \oplus y_2 \in y_1 \end{vmatrix}.$$

8. Minimizing an Arithmetic Polynomial

The linearity conditions obtained for an AP can be used for minimization by changing the order of writing the functions in the SBF and introducing auxiliary functions [1, 2].

Using the linearity conditions makes it easier to find a linear form, if it exists, because rather than obtain an AP at each step of the minimization, only a test for linearity need be made.

Example 11. We will generate the SBF $f_1 = 1 \oplus x_1 \oplus x_2 \oplus x_3$, $f_2 = x_1 \vee x_2 \vee x_3$ via an arithmetic polynomial. The linearity condition is not satisfied for this SBF. Now, $Y = 2 - (x_1 + x_2 + x_3) + 3(x_1x_2 + x_1x_3) - 7x_1x_2x_3$. Substituting the functions f_2 and f_1 simplifies the polynomial, but does not lead to linearity:

$$Y=1+x_1+x_2+x_3-2x_1x_2x_3.$$

To ensure linearity we introduct the auxiliary function $f_3 = \bar{x}_1\bar{x}_2\bar{x}_3 \vee x_1x_2x_3$ at the second position: f_2, f_3, f_1. Now, $Y = 3 + x_1 + x_2 + x_3$.

The question of intentionally using the linearity conditions for the purpose of minimizing an AP for completely defined SBFs remains open.

For incompletely defined SBFs relations of the type given by Eq. (25) and (29) make it easier to find minimal APs due to choosing those values for the undetermined coefficients y_i that ensure that zeroes will be obtained for the values of the coefficients or that the linearity conditions will be satisfied. A similar conclusion may be drawn for ZhPs [7].

Computational complexity can also be reduced by converting the polynomials to parentheti arithmetical form (PAF).

Example 12. We will define the PAF for the SBF $f_1 = x_1x_2x_3$, $f_2 = x_1(x_2 \oplus x_3)$, $f_3 = \bar{x}_1x_2x_3 \vee x_1\bar{x}_2x_3$. For this SBF (without introducing auxiliary functions) the AP has the form:

$$Y=4f_1'+2f_2'+f_3'=x_1x_2+x_1x_3+x_2x_3+x_1.$$

Converting to the PAF, we obtain

$$Y=x_1(1+x_2+x_3)+x_2x_3.$$

Taking Eq. (2) into account, an AP can be converted to PAF as follows:

$$Y=x_1x_2+x_1x_3+x_2x_3+x_1^2=x_1(x_1+x_2)+(x_1+x_2)x_3=(x_1+x_2)(x_1+x_3).$$

9. Conclusion

Thesis reports of the 10th All-Union Conference for Problems in Control which contains the works [8, 9] were published after this article was submitted. Some of the results in those publications coincide with those obtained here. Also, a number of definitions that are of particular interest in classifying polynomials are found in [8]. From what has been said, we can make a few comments.

1. In keeping with [8] we will use the acronym AP to denote a polynomial for one Boolean function and the acronym PM for the polynomial of an SBF. A PM can be constructed with, or without using auxiliary functions. In the first case the bits corresponding to auxiliary functions must be masked in order to obtain a result after the polynomial is calculated, and in the second case the desired result is formed immediately upon completion of computing the polynomial.

From what has been said and because the masking operation is extremely difficult, e.g., for several high-level languages the PMs can be divided into two classes: with masking and without.

When this is done it must also be noted that the formulas examined in Example 2 can be replaced by an SBF and generated by a linear PM having a mask

$$Y=3x_1+3x_2+2x_3,$$

as the APs that generate this formula directly are nonlinear.

2. Equations (16) and (24) for Q_n and P_n are the Kronecker n-th power of the matrixes Q_1 and P_1, respectively [9].

3. Equations (19) and (23) are called, in keeping with [8], a direct conjunction transform (CT) with matrix operations $\{\bullet,\cdot\}$ and $\{+,\cdot\}$.

4. Equations (20) and (26) are called, in keeping with [8], an inverse CT with matrix operations $\{\bullet,\cdot\}$ and $\{+,\cdot\}$.

5. Fast direct and inverse CTs are examined in [8].

6. By analogy with [8] Eqs. (19) and (26) define the conversion from an AP to a ZhP:

$$|G|=|Q_n|^{\otimes}|Y|=|Q_n|^{\otimes}|Q_n|\cdot|A|,$$

and Eqs. (22) and (20) define the conversion from a ZhP to an AP:

$$|A|=|P_n|\cdot|Y|=|P_n|\cdot|Q_n|^{\otimes}|G|.$$

7. The use of a PM and a CP makes it possible to compute an SBF in parallel [9].

LITERATURE CITED

1. V. D. Malyugin, "Creating Boolean functions via arithmetic polynomials," Avtomat. Telemekh., No. 4, 84-93 (1982).
2. V. D. Malyugin, "Creating corteges of Boolean functions via linear arithmetic polynomials," Avtomat. Telemekh., No. 2, 114-122 (1984).
3. D. A. Pospelov, Logic Methods for Circuit Analysis and Synthesis [in Russian] Énergiya, Moscow (1974).
4. V. L. Artyukhov and A. A. Shalyto, Logical Control Systems for Ships [in Russian] Inst. for Improving the Skills of Lead Workers and Specialists in the Shipbuilding Industry, Leningrad (1983).
5. G. S. Avsarkisyan and G. S. Brailovskii, "Writing logical functions as Zhegalkin polynomials," Avtomat. Vychisl. Tekh., No. 6, 6-10 (1975).
6. D. Bokhman and H. Postkhof, Binary Dynamic Systems [in Russian], Énergoizdat, Moscow (1986).
7. G. S. Avsarkisyan, "Polynomial forms of special Boolean functions and some of their applications," Izv. AN SSSR. Tekh. Kibern., No. 5, 50-58 (1983).
8. V. P. Shmerko and G. A. Kukharev, "Methods and algorithms for operationally solving systems of Boolean equations given in polynomial form" in: 10 Vsesoyuz. Soveshch. po Probleman Upravleniya, Tez. Dokl., Kn. I, Inst. Probl. Upravleniya, Moscow (1986), pp. 367-368.
9. V. D. Malyugin, "Applying the algebra of corteges for logic functions" in: 10 Vsesoyuz. Soveshch. po Probleman Upravleniya, Tez. Dokl., Kn. I, Inst. Probl. Upravleniya, Moscow (1986), p. 369.