COMBINATORIAL METHOD OF POLYNOMIAL EXPANSION OF SYMMETRIC BOOLEAN FUNCTIONS

Danila A. Gorodecky

The United Institute of Informatics Problems of National Academy of Sciences of Belarus, Minsk, 220012, Belarus, danila.gorodecky@gmail.com.

Abstract

A novel polynomial expansion method of symmetric Boolean functions is described. The method is efficient for symmetric Boolean function with small set of valued numbers and has the linear complexity for elementary symmetric Boolean functions, while the complexity of the known methods for this class of functions is quadratic. The proposed method is based on the consequence of the combinatorial Lucas theorem.

Keywords: polynomial expansion, symmetric Boolean function, carrier vector, reduced Zhegalkin spectrum, complexity

1. Introduction

The polynomial expansion is among the most complex tasks of the discrete mathematics. The polynomial expansion can be used to define the fifty-fifty distribution of 0 and 1 in the Steinhaus triangle, to synthesize modular summators, to find an algebraic immunity in cryptography and to solve various theoretical problems and practical applications.

Because of high computational complexity of generation of the polynomial for an arbitrary Boolean function the universal methods of the polynomial expansion are not effective. Therefore the methods of generation of expansions for various classes of Boolean functions are more effective. One of these classes is symmetric Boolean functions (SBF). It is known many methods of the polynomial expansion of SBF. One of the most effective methods is the transeunt triangle method [1]. It has the complexity $O(n^2)$. The known methods have the redundant computations, i.e. the intermediate computations should be produced to generate the polynomial expansion.

The article represents the method of the polynomial expansion with the complexity $O(n)$ in particular cases. The method could be applied to solve the task of polynomial expansion, as well as the reverse task, i.e. representation of the function described by the polynomial.

The method is based on the consequence of the combinatorial Lucas theorem, since it is referred as the combinatorial method.

2. Main definitions

An arbitrary Boolean function $F = F(X)$ of the $n$ variables, where $X = (x_1, x_2, ..., x_n)$, with unchanged value after swapping any couple of variables $x_i$ and $x_j$, where $i \neq j$ and $i, j = 1, 2, ..., n$, is called SBF.

SBF $F$ of the $n$ variables is characterised by the set of valued numbers $A(F) = \{a_1, a_2, ..., a_r\}$. The function $F$ is equal 1 if and only if the set of variables $x_1, x_2, ..., x_n$ has exactly $a_i$ numbers of 1’s, where $0 \leq a_i \leq n$, $0 \leq i \leq r$ and $0 \leq r \leq n + 1$. These SBFs are referred as $F^a_n(x_1, x_2, ..., x_n)$. If $r = 1$, then a function $F = F^a_n(X)$ is called elementary SBF (ESBF).

There is one-to-one correspondence between SBF $F^a_n(x_1, x_2, ..., x_n)$ and $(n+1)$-bits binary code $\pi = (\pi_0, \pi_1, ..., \pi_n)$ – the carrier vector [2] (or the reduced truth vector [3]), where the $i$-th entry
\( \pi_i \) is a value of the function \( F \) with the \( i \) numbers of 1’s, where \( 0 \leq i \leq n \). In other words, \( \pi_i = 1 \) if and only if the \( i \) is the valued number of the SBF \( F \).

The following formula is true for an arbitrary SBF \( F \):

\[
F(X) = \sqrt[n]{\prod_{i=0}^{n} \pi_i F_i(X)} = \bigoplus_{i=0}^{n} \pi_i F_i(X).
\] (1)

Positive polarity Reed-Muller polynomial (all variables are uncomplemented) is called as Zhegalkin polynomial and is referred as \( P(F) \).

SBF \( F \) of the \( n \) variables is called the polynomial-unate SBF (PUSBF or homogeneous SBF [4]), if the Zhegalkin polynomial form \( P(F) \) contains \( \binom{n}{i} \) \( i \)-rank products with the \( i \) positive literals, where \( 0 \leq i \leq n \). This function referred as \( F = E^i_n \). Hence it follows

\[
E^0_n = 1,
E^1_n = x_1 \oplus x_2 \oplus \ldots \oplus x_n,
E^2_n = x_1x_2 \oplus \ldots \oplus x_1x_n \oplus \ldots \oplus x_{n-1}x_n,
\ldots
E^n_n = x_1x_2 \ldots x_n.
\]

In general case, the polynomial form \( P(F) \) of SBF \( F = F(X) \) can be represented as:

\[
P(F) = \gamma_0 \oplus \gamma_1 (x_1 \oplus x_2 \oplus \ldots \oplus x_n) \oplus
\oplus \gamma_2 (x_1x_2 \oplus \ldots \oplus x_1x_n \oplus \ldots \oplus x_{n-1}x_n) \oplus \ldots \oplus \gamma_n x_1x_2 \ldots x_n,
\]

where \( \gamma(F) = (\gamma_0, \gamma_1, \gamma_2, \ldots, \gamma_n) \) is the reduced Zhegalkin (Reed-Muller) spectrum of SBF. It follows

\[
E(X) = \bigoplus_{i=0}^{n} \gamma_i E^i_n(X).
\] (2)

From the other hand PUSBF \( F \) of the \( n \) variables is characterised by the set of polynomial numbers \( B(E) = \{ b_1, b_2, \ldots, b_q \} \). The \( j \)-th entry of the reduced Zhegalkin spectrum \( \gamma(E) = (\gamma_0, \gamma_1, \ldots, \gamma_n) \) is equal 1 if and only if \( b_j = 1 \), where \( 0 \leq j \leq q \) and \( 0 \leq q \leq n \). If \( q = 1 \), then a function \( E^b_n \) is called the elementary PUSBF (EPUSBF).

The article provides the method of the transformation of the reduced truth vector \( \pi(F) \) to the reduced spectrum \( \gamma(F) \), i.e. \( \gamma(E^{b_1, b_2, \ldots, b_q}_n) \) to \( \pi(E^{b_1, b_2, \ldots, b_q}_n) \), and backwards, i.e. \( \pi(F^{a_1, a_2, \ldots, a_r}_n) \) to \( \gamma(F^{a_1, a_2, \ldots, a_r}_n) \).

3. Combinatorial method of generation of the carrier vector

The combinatorial method of the generating of the reduced truth vector \( \pi(E^{b_1, b_2, \ldots, b_q}_n) \) and the reduced spectrum \( \gamma(F^{a_1, a_2, \ldots, a_r}_n) \) is proposed below.

3.1. Generation of the carrier vector \( \pi(E^{b_1}_n) \)
The process of the generating of the carrier vector \( \pi(E^b_n) \) of the EPUSBF \( E^b_n \) could be demonstrated on the example.

**Example 1.** Let’s assume that it is necessary to get the carrier vector \( \pi = (\pi_0, \pi_1, ..., \pi_6) \) for the PUSBF \( F(X) = E^2_6(X) \).

From the condition it follows that \( \gamma = (0, 0, 1, 0, 0, 0) \) and

\[
E^2_6 = \begin{pmatrix}
1 & 2 & 3 & 4 & 5 \\
\oplus & x_1 x_2 & \oplus & x_1 x_3 & \oplus & x_1 x_4 & \oplus & x_1 x_5 & \oplus & x_1 x_6 & \oplus \\
\oplus & x_2 x_3 & \oplus & x_2 x_4 & \oplus & x_2 x_5 & \oplus & x_2 x_6 & \oplus \\
\oplus & x_3 x_4 & \oplus & x_3 x_5 & \oplus & x_3 x_6 & \oplus \\
\oplus & x_4 x_5 & \oplus & x_4 x_6 & \oplus \\
\oplus & x_5 x_6 \\
\end{pmatrix}
\] (3)

Note, that number of the column is equal to the number of factors in the column which are included in the polynomial of the function \( E^2_6(X) \). The polynomial \( P(E^2_6) \) contains \( \begin{pmatrix} 6 \\ 2 \end{pmatrix} = 15 \) 2–rank products.

To generate the carrier vector \( \pi = (\pi_0, \pi_1, ..., \pi_6) \) the \( i \)–th entry \( \pi_i \) should be defined with the following arguments, where \( i = 0, 6 \):

- assuming \( \pi_0 = 1 \) then \( E^2_6 \) contains \( F^0_6 \). But it is impossible, because the polynomial (3) of \( E^2_6(X) \) doesn’t contain term 1. In this case \( E^2_6 = 0 \) and therefore \( \pi_0 = 0 \);
- assuming \( \pi_1 = 1 \) then \( E^2_6 \) contains \( F^1_6 \). According to the definition of the ESBF \( F^1_6 = 1 \iff \) polynomial (3) is equal 1 for \( x_1 = 1 \) and \( x_2 = x_3 = ... = x_6 = 0 \). But it is impossible. In this case \( E^2_6 = 0 \) and therefore \( \pi_1 = 0 \);
- assuming \( \pi_2 = 1 \) then \( E^2_6 \) contains \( F^2_6 \). According to the definition of the ESBF \( F^2_6 = 1 \iff \) polynomial (3) is equal 1 for \( x_1 = x_2 = 1 \) and \( x_3 = x_4 = x_5 = x_6 = 0 \). Thus the only factor from the first column of the polynomial (3) is equal 1. In this case \( E^2_6 = 1 \) and therefore \( \pi_2 = 1 \);
- assuming \( \pi_3 = 1 \) then \( E^2_6 \) contains \( F^2_6 \oplus F^3_6 \). According to the definition of the ESBF \( F^3_6 = 1 \iff \) polynomial (3) is equal 1 for \( x_1 = x_2 = x_3 = 1 \) and \( x_4 = x_5 = x_6 = 0 \). Thus the factors from the first and second columns of the polynomial (3) are equal 1. Since the number of the unity components is the odd number, then in this case \( E^2_6 = 1 \) and therefore \( \pi_3 = 1 \);
- assuming \( \pi_4 = 1 \) then \( E^2_6 \) contains \( F^2_6 \oplus F^3_6 \oplus F^4_6 \). According to the definition of the ESBF \( F^4_6 = 1 \iff \) polynomial (3) is equal 1 for \( x_1 = x_2 = x_3 = x_4 = 1 \) and \( x_5 = x_6 = 0 \). Since the factors from the first, second and third columns of the polynomial (3) are equal 1, then the number of the unity components is the even number. In this case \( E^2_6 = 0 \) and therefore \( \pi_4 = 0 \);
- assuming \( \pi_5 = 1 \) then \( E^2_6 \) contains \( F^2_6 \oplus F^3_6 \oplus F^4_6 \). According to the definition of the ESBF \( F^5_6 = 1 \iff \) polynomial (3) is equal 1 for \( x_1 = x_2 = x_3 = x_4 = x_5 = 1 \) and \( x_6 = 0 \). Since the factors from the first, second, third and fourth columns of the polynomial (3) are equal 1, then the number of the unity components is the even number. In this case \( E^2_6 = 0 \) and therefore \( \pi_5 = 0 \);
assuming \( \pi_6 = 1 \) then \( E^2_6 \) contains \( F^2_6 \oplus F^3_6 \oplus F^6_6 \). According to the definition of the ESBF \( F^6_6 = 1 \Leftrightarrow \) polynomial (3) is equal 1 for \( x_1 = x_2 = x_3 = x_4 = x_5 = x_6 = 1 \). Since the factors from all columns of the polynomial (3) are equal 1, the number of the unity components is the odd number. In this case \( E^2_6 = 1 \) and therefore \( \pi_6 = 1 \).

As the result the carrier vector of the EPUSBF \( E^2_n \) is \( \pi(E^2_6) = (0, 0, 1, 0, 0, 1) \) and \( P(E^2_6) = F^2_6 \oplus F^3_6 \oplus F^6_6 \).

It is worth to pay attention to the fact that the value of the polynomial depends only on the parity number of unity factors.

The reasoning used in the example 1 may be summarized with the theorem:

**Theorem 1.** The \( i \)-th entry \( \pi_i \) of the carrier vector \( \pi = (\pi_0, \pi_1, \ldots, \pi_n) \) of the PUSBF \( E^b_n = E^b_n(x_1, x_2, \ldots, x_n) \) is calculated by using the formula:

\[
\pi_i = \begin{cases} 
1, & \text{if } \left( \begin{array}{c} i \\ b \end{array} \right) = 1 \pmod{2}; \\
0 & \text{otherwise},
\end{cases}
\]

where \( i = \overline{b, n} \).

**Proof.** Let’s consider three cases of relations \( i \) and \( b \).

The first case where \( i < b \). Then the number of the unity terms \( i \) is less than the \( b \)-rank products and \( E^b_i = 0 \) (see the first and second cases of the example 1). Therefore \( \pi_i = 0 \).

The second case where \( i = b \), i.e. \( x_1 = x_2 = \ldots = x_b = 1 \) and \( x_{b+1} = x_{b+2} = \ldots = x_n = 0 \). Thus just one \( b \)-rank term of the PUSBF \( E^b_b \) is equal 1 and \( P(E^b_b) = x_1x_2\ldots x_b = 1 \) (see the third case of the example 1). In this case \( \pi_i = 1 \).

The third case where \( i > b \), i.e. \( x_1 = x_2 = \ldots = x_b = x_{i+1} = \ldots = x_n = 0 \). Thus the function \( E^b_i \) is represented by the polynomial \( P(E^b_i) = x_1x_2\ldots x_b \oplus x_1x_2\ldots x_{b-1}x_{b+1} \oplus \ldots \oplus x_1x_2\ldots x_{i-1}x_i \oplus \ldots \oplus x_1x_2\ldots x_{n-b}x_{i-b+1}x_{i-b+2}x_{i-b+3} \). Since the value of \( P(E^b_i) \) is determined by even-odd of \( i \). Thus \( E^b_i = \begin{cases} 
1, & \text{if } \left( \begin{array}{c} i \\ b \end{array} \right) = 1 \pmod{2}; \\
0 & \text{otherwise}.
\end{cases} \)

\( \pi_i = \begin{cases} 
1, & \text{if } \left( \begin{array}{c} i \\ b \end{array} \right) = 1 \pmod{2}; \\
0 & \text{otherwise}.
\end{cases} \)

The statement of the theorem is proved.

As a result of Theorem 1 the carrier vector \( \pi \) of the PUSBF \( E^b_n \) corresponds to the following form

\[
\pi = \left( 0, 0, \ldots, 0, 1, \underbrace{\pi_{b+1}, \ldots, \pi_n}_{b}, \underbrace{n-b}_{n-b} \right).
\]

\[ \text{(5)} \]
Consequence of the Lucas theorem is helpful for calculation using the formula (4). It determines the even-odd of the number \( \binom{i}{b} \) and as follows.

**Theorem 2. (Consequence of the Lucas theorem)** [5]. The number \( \binom{n}{b} = 1 \pmod{2} \) if and only if each bit of \( b \) is no more than the same bit of \( n \), where \( n \geq b \) in decimal representation.

Note, that the binary length \( n = \left( n_{\delta_1}, n_{\delta_2}, \ldots, n_1 \right) \) and \( b = \left( b_{\delta_2}, b_{\delta_1}, \ldots, b_1 \right) \) is defined as \( \delta_1 = \left\lceil \log_2 n \right\rceil + 1 \) and \( \delta_2 = \left\lceil \log_2 b \right\rceil + 1 \) respectively.

**Example 2.** Let's define even-odd of the number \( \binom{n}{b} \) using Theorem 2 and assuming \( n = 11 \) and \( b \) for two cases a) \( b = 2 \) and b) \( b = 5 \).

The length of \( n \) is \( \delta_1 = \left\lceil \log_2 11 \right\rceil + 1 = 4 \), then \( n = (n_4, n_3, n_2, n_1) = (1010) \), and

a) \( \delta_2 = \left\lceil \log_2 2 \right\rceil + 1 = 2 \), then \( b = (b_2, b_1) = (10) \);

b) \( \delta_2 = \left\lceil \log_2 5 \right\rceil + 1 = 3 \), then \( b = (b_3, b_2, b_1) = (101) \).

For the case a) the binary representations \( b \) and \( n \) are comparable and satisfy the condition of Theorem 2, as pictured in Figure 1 a).

For the case b) the binary representations \( b \) and \( n \) are not comparable and do not satisfy the condition of Theorem 2, as pictured in Figure 1 b).

\[
\begin{array}{c|c|c|c|c|c|c}
\hline
i & 4 & 3 & 2 & 1 & b & 1 \\
\hline
i & 1 & 0 & 1 & 1 & b & 1 \\
\hline
i & 1 & 0 & 1 & 1 & b & 1 \\
\hline
\end{array}
\]

- not involved in the comparison
- satisfied the condition of Theorem 2
- does not satisfy the condition of Theorem 2

Figure 1. Definition of even-odd of the numbers a) \( \binom{11}{2} \); b) \( \binom{11}{5} \)

As a result, in the case a) the number \( \binom{11}{2} \) is the odd and thus \( \binom{11}{2} = 1 \pmod{2} \); in the case b) the number \( \binom{11}{5} \) is the even and thus \( \binom{11}{5} = 0 \pmod{2} \).

Let's generate the carrier vector for the function in above example 1 using Theorem 1 and Theorem 2.

**Example 3.** Let’s assume that it is necessary to generate \( \pi\left(E_6^2\right) \).

From the condition it follows \( \gamma\left(E_6^2\right) = (0, 0, 1, 0, 0, 0, 0) \). According to the formula (5) \( \pi_0 = \pi_1 = 0 \), \( \pi_2 = 1 \) and \( \pi = (0, 0, 1, \pi_3, \pi_4, \pi_5, \pi_6) \).

Therefore, in order to find \( \pi_3, \pi_4, \pi_5, \pi_6 \) the even-odd order of the Binomial coefficients \( \binom{3}{2}, \binom{4}{2}, \binom{5}{2}, \binom{6}{2} \) respectively should be defined. From the Theorem 2 they could be defined as shown on figure 2.
The figure 2 is analogous to representation as follows: \( \binom{3}{2} = \binom{11}{10} \Rightarrow \pi_3 = 1; \)
\[
\binom{4}{2} = \binom{100}{10} \Rightarrow \pi_4 = 0; \quad \binom{5}{2} = \binom{101}{10} \Rightarrow \pi_5 = 0; \quad \binom{6}{2} = \binom{110}{10} \Rightarrow \pi_6 = 1.
\]
As the result the carrier vector is \( \pi(E_6^2) = (0, 0, 1, 1, 0, 0, 1) \) and \( P(E_6^2) = F_6^2 \oplus F_6^3 \oplus F_6^6 \).

The procedure of calculating of the entries of the carrier vector using consequence of the Lucas theorem is called combinatorial methods.

![Figure 2. Definition of even-odd of the numbers](image)

- not involved in the comparison
- does not satisfy the condition of Theorem 2
- satisfy the condition of Theorem 2

**3.2. Generation of the carrier vector** \( \pi(E_n^{b_1, b_2, ..., b_q}) \)

The combinatorial method of generating of the carrier vector for the EPUSBF \( E_n^b = E_n^b(x_1, x_2, ..., x_n) \) can be generalized for an arbitrary PUSBF \( E_n^{b_1, b_2, ..., b_q} = E_n^{b_1, b_2, ..., b_q}(x_1, x_2, ..., x_n) \) with the following theorem.

**Theorem 3.** The \( i \)-th entry \( \pi_i \) of the carrier vector \( \pi(E_n^{b_1, b_2, ..., b_q}) = (\pi_0, \pi_1, ..., \pi_n) \) of the PUSBF \( E_n^{b_1, b_2, ..., b_q} \) is calculated with the following formula:

\[
\pi_i = \begin{cases} 
1, & \mbox{if } \left\lfloor \frac{i}{b_1} \right\rfloor + \left\lfloor \frac{i}{b_2} \right\rfloor + ... + \left\lfloor \frac{i}{b_q} \right\rfloor = 1 \pmod{2}; \\
0, & \mbox{otherwise},
\end{cases}
\]

where \( i = b_1 + 1, n \). Note, that \( \left\lfloor \frac{i}{b_j} \right\rfloor \) for \( i < b_j \) is meaningless, where \( j = 1, q \), therefore let’s assume \( \left\lfloor \frac{i}{b_j} \right\rfloor = 0 \) for \( i < b_j \).

The proof of Theorem 3 follows from Theorem 1.

**Example 4.** Let’s assume that it is necessary to generate \( \pi(E_{10}^{5,7,8}) \).

From the condition it follows \( \pi(E_{10}^{5,7,8}) = (0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0) \). According to the formula (5) it follows \( \pi_0 = \pi_1 = \pi_2 = \pi_3 = \pi_4 = 0 \) and \( \pi_5 = 1 \). Thus \( \pi(E_{10}^{5,7,8}) = (0, 0, 0, 0, 1, \pi_6, \pi_7, \pi_8, \pi_9, \pi_{10}) \).
According to the formula (6) and Theorem 2 it is easy to define $\pi_6, \pi_7, \pi_8, \pi_9$ and $\pi_{10}$ as shown on the figure 3.

Figure 3. Calculating of the $\pi_6, \pi_7, \pi_8, \pi_9, \pi_{10}$ for $\pi(E_{10}^{5.7.8})$

The figure 3 is analogous to representation as follows:

\[
\binom{6}{5} = \binom{110}{101} = 0 \pmod{2} \Rightarrow \pi_6 = 0;
\]

\[
\binom{7}{5} + \binom{7}{7} = \binom{111}{101} + \binom{111}{111} = 1 \pmod{2} + 1 \pmod{2} = 0 \pmod{2} \Rightarrow \pi_7 = 0;
\]
\[
\begin{align*}
\left( \begin{array}{ccc}
8 \\
5 \\
\end{array} \right) + \left( \begin{array}{ccc}
8 \\
7 \\
\end{array} \right) + \left( \begin{array}{ccc}
8 \\
8 \\
\end{array} \right) &= \left( \begin{array}{ccc}
1000 \\
101 \\
\end{array} \right) + \left( \begin{array}{ccc}
1000 \\
111 \\
\end{array} \right) + \left( \begin{array}{ccc}
1000 \\
1000 \\
\end{array} \right) = 0 \mod 2 + 0 \mod 2 + 1 \mod 2 = 1 \mod 2 \\
\Rightarrow \pi_8 &= 1; \\
\left( \begin{array}{ccc}
9 \\
5 \\
\end{array} \right) + \left( \begin{array}{ccc}
9 \\
7 \\
\end{array} \right) + \left( \begin{array}{ccc}
9 \\
8 \\
\end{array} \right) &= \left( \begin{array}{ccc}
1001 \\
101 \\
\end{array} \right) + \left( \begin{array}{ccc}
1001 \\
111 \\
\end{array} \right) + \left( \begin{array}{ccc}
1001 \\
1000 \\
\end{array} \right) = 0 \mod 2 + 0 \mod 2 + 1 \mod 2 = 1 \mod 2 \\
\Rightarrow \pi_9 &= 1; \\
\left( \begin{array}{ccc}
10 \\
5 \\
\end{array} \right) + \left( \begin{array}{ccc}
10 \\
7 \\
\end{array} \right) + \left( \begin{array}{ccc}
10 \\
8 \\
\end{array} \right) &= \left( \begin{array}{ccc}
1010 \\
101 \\
\end{array} \right) + \left( \begin{array}{ccc}
1010 \\
111 \\
\end{array} \right) + \left( \begin{array}{ccc}
1010 \\
1000 \\
\end{array} \right) = 0 \mod 2 + 0 \mod 2 + 1 \mod 2 = 1 \mod 2 \\
\Rightarrow \pi_{10} &= 1.
\end{align*}
\]

Thus \( \pi_6 = \pi_7 = 0 \) and \( \pi_8 = \pi_9 = \pi_{10} = 1 \). As the result the carrier vector of the PUSBF \( E_{10}^{5,7,8} \) is \( \pi(E_{10}^{5,7,8}) = (0,0,0,0,0,1,0,0,1,1,1) \) and \( P(E_{10}^{5,7,8}) = F_{10}^{5} \oplus F_{10}^{7} \oplus F_{10}^{8} \).

4. Generation of the reduced spectrum \( \gamma(F_{n}^{a_1,a_2,\ldots,a_r}) \)

The combinatorial method of the generation of the carrier vector \( \pi(F_{n}^{a_1,a_2,\ldots,a_r}) \) can be applied to the generation of the reduced spectrum \( \gamma(F_{n}^{a_1,a_2,\ldots,a_r}) \), where \( F_{n}^{a_1,a_2,\ldots,a_r} \) is the SBF.

To solve the task of the generating of the reduced spectrum \( \gamma(F_{n}^{a_1,a_2,\ldots,a_r}) \) Theorem 1 and Theorem 3 can be adapted to the two following forms.

**Theorem 4.** The \( i \)-th entry \( \gamma_i \) of the reduced spectrum \( \gamma = (\gamma_0,\gamma_1,\ldots,\gamma_n) \) of the PUSBF \( F_{n}^{a} = F_{n}^{a}(x_1,x_2,\ldots,x_n) \) is calculated with the following formula:

\[
\gamma_i = \begin{cases} 
1, & \text{if } i \mod 2 = 1; \\
0, & \text{otherwise},
\end{cases}
\]

where \( i = a\cdot n \).

**Theorem 5.** The \( i \)-th entry \( \gamma_i \) of the reduced spectrum \( \gamma(F_{n}^{a_1,a_2,\ldots,a_r}) = (\gamma_0,\gamma_1,\ldots,\gamma_n) \) of the SBF \( F_{n}^{a_1,a_2,\ldots,a_r} \) is calculated with the following formula:

\[
\gamma_i = \begin{cases} 
1, & \text{if } i \mod 2 = 1; \\
0, & \text{otherwise},
\end{cases}
\]

where \( i = a\cdot1 + a\cdot n \). Note, that \( \left( \begin{array}{c} i \\
\alpha_j \\
\end{array} \right) \) for \( i < a_j \) is meaningless, where \( j = 1, r \), therefore let’s assume \( \left( \begin{array}{c} i \\
\alpha_j \\
\end{array} \right) = 0 \) and \( i < a_j \).

According to Theorem 4 and Theorem 5 the reduced spectrum \( \gamma = (\gamma_0,\gamma_1,\ldots,\gamma_n) \) of the SBF \( F_{n}^{a_1,a_2,\ldots,a_r} \) corresponds to the following form
\[ \gamma = \left\{ \frac{0,0,\ldots,0,1,\gamma_{a+1},\ldots,\gamma_n}{a} \right\}. \] (9)

The example of the application of Theorem 5 will be considered.

**Example 5.** Let’s generate the reduced spectrum \( \gamma(F_7^{2,3}) \).

From the condition it follows the carrier vector is \( \pi(E_7^{2,3}) = (0,0,1,1,0,0,0,0) \). According to the formula (9) it follows \( \gamma_0 = \gamma_1 = 0, \gamma_2 = 1 \) and \( \gamma(F_7^{2,3}) = (0,0,1,\gamma_3,\gamma_4,\gamma_5,\gamma_6,\gamma_7) \).

According to the formula (8) and Theorem 5 it is easy to define \( \gamma_3, \gamma_4, \gamma_5, \gamma_6 \) and \( \gamma_7 \) as shown below:

\[
\begin{align*}
\begin{pmatrix} 3 \\ 2 \\ 3 \\ 3 \end{pmatrix} + \begin{pmatrix} 11 \\ 10 \end{pmatrix} &= 1 \pmod{2} + 1 \pmod{2} = 0 \pmod{2} \Rightarrow \gamma_3 = 0; \\
\begin{pmatrix} 4 \\ 2 \\ 3 \\ 3 \end{pmatrix} + \begin{pmatrix} 100 \\ 100 \end{pmatrix} &= 0 \pmod{2} + 0 \pmod{2} = 0 \pmod{2} \Rightarrow \gamma_4 = 0; \\
\begin{pmatrix} 5 \\ 2 \\ 3 \\ 3 \end{pmatrix} + \begin{pmatrix} 101 \\ 101 \end{pmatrix} &= 0 \pmod{2} + 0 \pmod{2} = 0 \pmod{2} \Rightarrow \gamma_5 = 1; \\
\begin{pmatrix} 6 \\ 2 \\ 3 \\ 3 \end{pmatrix} + \begin{pmatrix} 110 \\ 110 \end{pmatrix} &= 1 \pmod{2} + 0 \pmod{2} = 1 \pmod{2} \Rightarrow \gamma_6 = 1; \\
\begin{pmatrix} 7 \\ 2 \\ 3 \\ 3 \end{pmatrix} + \begin{pmatrix} 111 \\ 111 \end{pmatrix} &= 1 \pmod{2} + 1 \pmod{2} = 0 \pmod{2} \Rightarrow \gamma_7 = 0.
\end{align*}
\]

As the result the reduced spectrum of the function \( F_7^{2,3} \) is \( \gamma(F_7^{2,3}) = (0,0,1,0,0,1,1,0) \) and \( P(F_7^{2,3}) = E_2^2 \oplus E_3^2 \oplus E_5^6 \).

**5. The complexity of the combinatorial method**

The complexity of the proposed method can be defined as the number of the binary operations XOR (or OR) and is referred \( S_1 \) for the EPUSBF \( E_n^b \) (or ESBF \( F_n^a \)) and \( S_2 \) for the PUSBF \( E_n^{b_1,b_2,\ldots,b_q} \) (or SBF \( F_n^{a_1,a_2,\ldots,a_r} \)).

The positive relationship of two binary vectors is \( (x_1,x_{i-1},\ldots,x_1) \geq (y_1,y_{i-1},\ldots,y_1) \), if \( x_i \geq y_i \), where \( i = 1 \ldots t \). In this way to define the relationship \( x_i \geq y_i \) the following condition should be satisfied

\[ x_i \vee y_i = 1. \] (10)

Therefore according to the condition of Theorem 2 the complexity (the number of operations (10)) of the computation of the number \( \binom{i}{b} = 1 \pmod{2} \) is \( \lfloor \log_2 b \rfloor + 1 \). From Theorem 1 it follows the complexity of the computation of the carrier vector \( \pi(E_n^b) = (\pi_0,\pi_1,\ldots,\pi_n) \) is

\[ S_1 = (\lfloor \log_2 b \rfloor + 1) \cdot (n-b). \] (11)

From Theorem 3 it follows the complexity of the computation of the carrier vector \( \pi(E_n^{b_1,b_2,\ldots,b_q}) = (\pi_0,\pi_1,\ldots,\pi_n) \) is
\[ S_2 = (\lfloor \log_2 b_1 \rfloor + 1) \cdot (n - b_1) + (\lfloor \log_2 b_2 \rfloor + 1) \cdot (n - b_2) + \ldots + (\lfloor \log_2 b_q \rfloor + 1) \cdot (n - b_q) + (n - b_2 + 1) + (n - b_3 + 1) + \ldots + (n - b_q + 1) = \\
= (\lfloor \log_2 b_1 \rfloor + 1) \cdot (n - b_1) + \sum_{i=2}^{q} (\lfloor \log_2 b_i \rfloor + 1) \cdot (n - b_i) + \sum_{i=2}^{q} (n - b_i + 1) = \\
= (\lfloor \log_2 b_1 \rfloor + 1) \cdot (n - b_1) + \sum_{i=2}^{q} (\lfloor \log_2 b_i \rfloor + 1) \cdot (n - b_i + 1). \] (12)

The complexity of the calculation of \( \gamma(F_n^a) \) and \( \gamma(F_n^{a_1,a_2,\ldots,a_r}) \) can be calculated with (11) and (12) respectively.

6. Discussion

There are some effective methods to solve the task of polynomial expansion, i.e. generating of the reduced spectrum \( \gamma(F_n^{a_1,a_2,\ldots,a_r}) = (\gamma_0, \gamma_1, \ldots, \gamma_n) \) and the reverse task of generating of the carrier vector \( \pi(E_{n_1},E_{n_2},\ldots,E_{n_r}) = (\pi_0, \pi_1, \ldots, \pi_n) \). One of these methods is the transeunt triangle method. It was originally proposed by V.P. Suprun for SBF [1]. Then the method was generalized for arbitrary Boolean functions [6]. The transeunt triangle method is the most effective method for polynomial expansion of the SBF \( F = F(x_1, x_2, \ldots, x_n) \) and has the complexity \( O(n^2) \).

The transeunt triangle form is generated from the upper base to the bottom using the XOR-operation (see example 6). Thus the number of XOR-operations defines the complexity \( S_T \) of the transeunt triangle method and is

\[ S_T = \frac{n^2 + n}{2}. \] (13)

There is the example of the implementation of the transeunt triangle method for generating of the carrier vector \( \pi(E_{n_1}^b) = (\pi_0, \pi_1, \ldots, \pi_n) \).

**Example 6.** Let’s assume the reduced spectrum \( \gamma(E_6^2) = (0, 0, 1, 0, 0, 0, 0) \), i.e. \( n = 6 \) and \( b = 2 \). In order to generate \( \pi(E_6^2) \) the transeunt triangle method will be used.

From the condition the upper base of the triangle will \( \gamma(E_6^2) = (0, 0, 1, 0, 0, 0, 0) \) and it takes the form as follow:

\[
\begin{array}{cccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\
\end{array}
\]

According to the transeunt triangle method the left side of the triangle corresponds to the reduced carrier vector and it is equal to \( \pi(E_6^2) = (0, 0, 1, 1, 0, 0, 1) \). Therefore \( E_6^2 = F_6^2 \oplus F_6^3 \oplus F_6^6 \).
Using the formula (13) the complexity of the computation of the \( \pi\left( E_0^2 \right) \) with the transeunt triangle method is \( S_T = 15 \). On the other side, to complexity of performing the same task using the combinatorial method, according to the formula (11) and as shown in example 3, is \( S_1 = 8 \).

Firstly, let’s compare the complexity \( S_1 \) (formula (11)) of the combinatorial method proposed in the article and the complexity \( S_T \) (formula (13)) of the transeunt triangle method for SBF \( E_n^a \) (or \( F_n^a \)). The illustration of the comparison of \( S_1 \) and \( S_T \) is shown on the Figure 4.

As it can be seen at figure 4 the combinatorial method for EPUSBF has the linear complexity. The proposed method can provide ten times more efficiency for 80 variables in comparison with the transeunt triangle method. The complexity of the combinatorial method (formula (11)) is calculated for the worst case, i.e. for the EPUSBF \( E_n^b \), where \( b = 16 \).

The complexity of the combinatorial method in comparison with the complexity of the transeunt triangle method for PUSBF \( E_n^{b_1,b_2,\ldots,b_r} \), where \( r > 1 \), strongly depends on numbers included in the set of the polynomial numbers. As a result, the table demonstrates the threshold of the efficiency of the combinatorial method in comparison with the transeunt triangle method.

The second column contains the power set of the polynomial numbers \( B(E) \) for which the complexity \( S_2 \) and \( S_T \) is approximately equal. The third column contains the set of the polynomial numbers \( B(E) \) for which the complexities of both methods are the same. Any other set of the polynomial numbers \( B(E) \) provides a lower complexity of the combinatorial method for the number of the variables \( B(E) \) provides a lower complexity of the combinatorial method for the number of the variables specified in the first column. The fourth column shows the ratio of the set of the polynomial numbers to all variables specified in the first column. Two right columns show the comparable complexities of the combinatorial and the transeunt triangle methods.
Table

The efficiency of the combinatorial method and the transeunt triangle method

| Number of the variables \( n \) that the function \( E_n^{b_1, b_2, \ldots, b_r} \) depends on | Number \( r \) of the polynomial numbers \( b_1, b_2, \ldots, b_r \) for \( S_2 \approx S_T \) | Set of the polynomial numbers \( B(E) = b_1, b_2, \ldots, b_r \) | Percentage of the number \( r \) of the variables \( n \), \% | Complexity of the combinatorial method, \( S_2 \) | Complexity of the transeunt triangle method, \( S_T \) |
|---|---|---|---|---|---|
| 10 | 3 | \{2, 3, 4\} | 30 | 53 | 55 |
| 20 | 5 | \{4,..8\} | 25 | 235 | 210 |
| 30 | 6 | \{4,..9\} | 20 | 483 | 465 |
| 40 | 7 | \{8,..14\} | 18 | 836 | 820 |
| 50 | 8 | \{11,..18\} | 16 | 1266 | 1275 |
| 60 | 9 | \{16,..24\} | 15 | 1840 | 1830 |
| 70 | 10 | \{16,..25\} | 14 | 2520 | 2485 |
| 80 | 11 | \{16,..26\} | 14 | 3295 | 3240 |
| 90 | 12 | \{16,..27\} | 13 | 4765 | 4095 |
| 100 | 13 | \{16,..28\} | 13 | 5130 | 5050 |
| 255 | 26 | \{32,..57\} | 10 | 32988 | 32640 |
| 511 | 44 | \{64,..107\} | 9 | 131355 | 130816 |
| 1023 | 76 | \{128,..203\} | 7 | 521960 | 523776 |
| 2047 | 135 | \{256,..390\} | 7 | 2095866 | 2096128 |
| 4095 | 242 | \{512,..753\} | 6 | 8381660 | 8386560 |

7. Conclusions

The combinatorial method is a new method of generating of the carrier vector \( \pi \left( F_n^{b_1, b_2, \ldots, b_r} \right) \) and the reduced spectrum \( \gamma \left( F_n^{a_1, a_2, \ldots, a_r} \right) \) of SBF, i.e. polynomial expansion of SBF.

The proposed method is the linear complexity and the complexity of the known methods (for example, transeunt triangle method) is quadratic for EPUSBF (or ESBF). The combinatorial method provides high efficiency for small number of variables for PUSBF (or SBF).

References

[1] V.P. Suprun “Polynomial Expression of Symmetric Boolean Functions”, Soviet Journal of Computer and Systems Science, Vol. 23, № 6, Nov.–Dec. 1985, p. 88-91. (translation from Izv. Adad. SSSR, Tekhicheskaia Kibernetika, 1985, № 4, p. 123-127).
[2] J.T. Butler, G.W. Dueck, S.N. Yanushkevich, V.P. Shmerko “On the use of transeunt triangle to synthesis fixed-polarity Reed-Muller expansions”, Proceedings of the Reed-Muller Workshop 2009, May 23-24, 2009, Naha. Okinawa, Japan, p. 119-126.
[3] T. Sasao, J.T. Butler “The eigenfunction of the Reed-Muller transformation”, Proceedings Reed-Muller 2007 Workshop, May 16, 2007, Oslo, Norway.
[4] A. Braeken, B. Preneel “On the Algebraic Immunity of Symmetric Boolean Functions”, Progress in Cryptology – INDOCRYPT 2005, Lecture Notes in Computer Science, Vol. 3797, 2005, p. 35-48.
[5] A. Granville “Arithmetic Properties of Binomial Coefficients I: Binomial coefficients modulo prime powers”, Canadian Mathematical Society Conference Proceedings, 1997, p. 253-275.
[6] V.P. Suprun “Table Method for Polynomial Decomposition of Boolean Functions”, Kibernetika, № 1, 1987, p. 116-117 (in Russian).