Minimizing AND-EXOR Expressions for Multiple-Valued Two-Input Logic Functions (Extended Abstract)

Takaaki Mizuki[1], Hitoshi Tsubata[2], and Takao Nishizeki[2]

[1] Cyberscience Center, Tohoku University, Aramaki-Aza-Aoba 6–3, Aoba-ku, Sendai 980–8578, Japan
tm-paper@rd.isc.tohoku.ac.jp
[2] Graduate School of Information Sciences, Tohoku University, Aramaki-Aza-Aoba 6–6–05, Aoba-ku, Sendai, 980–8579, Japan

Abstract. A minimum ESOP (Exclusive-OR Sum-of-Products) form of a logic function f is an AND-EXOR 2-level expression of f having the minimum number of product terms. In the paper we deal with multiple-valued 2-input logic functions f, and give an algorithm to find a minimum ESOP form of a given function f in polynomial time.

1 Introduction

An ESOP (Exclusive-OR Sum-of-Products) form of a logic function f is an AND-EXOR 2-level expression of f, i.e., a logical expression that combines products of literals by Exclusive-ORs. For example,

$$f(x_1, x_2, x_3) = (x_1 \wedge \bar{x}_2 \wedge x_3) \oplus (\bar{x}_1 \wedge \bar{x}_2 \wedge \bar{x}_3) \oplus (x_2 \wedge \bar{x}_3) \tag{1}$$

is an ESOP form defining a (2-valued) 3-input logic function f. The logic function f can be expressed by another ESOP form, say

$$f(x_1, x_2, x_3) = x_1\bar{x}_2 \oplus \bar{x}_3. \tag{2}$$

(Hereafter, as in this expression, we omit the conjunction symbol $\wedge$.) The ESOP form in Eq. (1) has exactly three product terms, while the ESOP form in Eq. (2) has only two product terms. Thus, there exists a minimization problem regarding ESOP forms. This paper deals with such a minimization problem; more specifically, we give an efficient algorithm to minimize ESOP forms for multiple-valued 2-input logic functions.

1.1 ESOP Forms

First of all, we formally define "multiple-valued input logic functions" and "literals." Throughout the paper, for a positive integer m, we define $\mathbb{Z}_m$ as follows:

$$\mathbb{Z}_m \overset{\text{def}}{=} \{0, 1, \ldots, m-1\}.$$

Let $n \geq 1$ be the number of logical variables, and let $m_1, m_2, \ldots, m_n \geq 2$ be n positive integers. Then, a function

$$f(x_1, x_2, \ldots, x_n)$$

such that

$$f : \mathbb{Z}_{m_1} \times \mathbb{Z}_{m_2} \times \cdots \times \mathbb{Z}_{m_n} \to \{0, 1\}$$

is called a *multiple-valued n-input logic function*; in particular, when $m_1 = m_2 = \cdots = m_n = m$, we call it an *$m$-valued n-input logic function.* (Needless to say, when $m = 2$, it is a 2-valued n-input logic function and hence is a so-called Boolean function.) Furthermore, for an i-th variable x_i, $1 \leq i \leq n$, and a subset $S \subseteq \mathbb{Z}_{m_i}$, we define a function $x_i^S : \mathbb{Z}_{m_i} \to \{0, 1\}$, called a *literal*, as follows:

$$x_i^S(x_i) = \begin{cases} 1 \text{ if } \ x_i \in S; \\ 0 \text{ otherwise.} \end{cases}$$

We often denote $x_i^S(x_i)$ simply by x_i^S. When $S = \mathbb{Z}_{m_i}$, the literal x_i^S is just the constant 1; also, when $S = \emptyset$, the literal x_i^S $(= x_i^\emptyset)$ is just the constant 0. For instance, when $m_i = 2$, i.e., $\mathbb{Z}_{m_i} = \{0, 1\}$, there are exactly four literals $x_i^{\{0,1\}}$, $x_i^{\{0\}}$, $x_i^{\{1\}}$ and $x_i^\emptyset$, which are often denoted by 1, $\overline{x}_i$, x_i and 0, respectively.

A product $x_1^{S_1} x_2^{S_2} \cdots x_n^{S_n}$ of literals is called a *product term.* If a logic function $f(x_1, x_2, \ldots, x_n)$ has a logical expression

$$F = \bigoplus_{(S_1, S_2, \cdots, S_n)} x_1^{S_1} x_2^{S_2} \cdots x_n^{S_n} \tag{3}$$

which combines product terms by Exclusive-ORs, then the expression F is called an *ESOP form.* If an ESOP form F has a product term containing a literal $x_i^\emptyset$ $(= 0)$, then the resulting ESOP form F' obtained by removing such a product term represents the same logic function as the original ESOP form F. For an ESOP form F, we denote by $\tau(F)$ the number of product terms in F. A *minimum ESOP form* of a logic function f is an ESOP form having the minimum number of product terms among all possible ESOP forms representing f.

1.2 Known Results

For many decades, the problem of minimization or simplification of ESOP forms has attracted much attention of the researchers in the logic design community. (A comprehensive survey appears in a book [6].) Although no efficient algorithm to minimize ESOP forms has been known, many good heuristic algorithms to simplify ESOP forms have been proposed (e.g. [1,5,7,8,11]). On the other hand, there also exist efficient exact minimization algorithms which efficiently work only for a limited (small) number of variables or product terms (e.g. [3,9,10]).

Historically, the binary case of $m_1 = m_2 = \cdots = m_n = 2$, namely ESOP forms for 2-valued input logic functions have been much investigated, of course; the most famous ESOP form is probably a Reed-Muller expression. For such a 2-valued input case, there are many good heuristic (or exact) algorithms to simplify (or minimize) ESOP forms (e.g. [1,3,9,11]). In particular, the best known upper bound on the number $\tau(F)$ of product terms in a minimum ESOP form F for any 2-valued n-input logic function is $\tau(F) \leq 29 \cdot 2^{n-7}$ (provided that $n \geq 7$) [2].

On the other hand, there are relatively a small number of papers dealing with multiple-valued input logic functions, but there are a few works on the case where integers m_i are larger than 2. In particular, the case of $m_1 = m_2 = \cdots = m_n = 4$, namely ESOP forms for 4-valued input logic functions have been greatly studied, e.g. [5,7]; it is motivated by improving input decoders in PLA (Programmable Logic Array) structures. Furthermore, the case where $m_1 = m_2 = \cdots = m_{n-1} = 2$ and $m_n \geq 3$ has been studied in [10].

1.3 Our Results

As mentioned in the previous subsection, no efficient ESOP minimization algorithm for general logic functions has been known; in particular, for the multiple-valued input case, every existing efficient minimization algorithm, to our knowledge, has a limitation in the input sizes m_i. In this paper, instead of restricting the input sizes m_i in multiple-valued input logic functions, we fix the number n of variables to 2. We thus deal with m-valued 2-input logic functions, and give an algorithm to find a minimum ESOP form of any given function in polynomial time in m, say time $O(m^3)$, where m is any integer larger than 1.

It is known that the minimization of ESOP forms of m-valued 2-input logic functions, which this paper addresses, can be applied to improving a cryptographic protocol [4], as follows. The cost (communication complexity) of the cryptographic protocol developed in [4] to securely compute a function $f(a, b)$ is proportional to the number $\tau(F)$ of product terms in an ESOP form F of f. Thus, if one can find a minimum ESOP form of f, then one can achieve the most efficient secure computation by the protocol. Therefore, applying the results in this paper to the protocol proposed in [4], one can execute the protocol most efficiently.

The remainder of the paper is organized as follows. In Section 2, we present some preliminaries necessary to explain our algorithm. In Section 3, we introduce a method to express an ESOP form of an m-valued 2-input logic function in a matrix form. This matrix-based expression helps us to easily and intuitively understand the minimization of ESOP forms. In Section 4, we present our efficient algorithm to find a minimum ESOP form of any given m-valued 2-input logic function by using elementary row operations for matrices. This paper concludes in Section 5 with some discussions.

2 Preliminaries

In this section, we define some terms and present some of the known results.

2.1 Multiple-Valued Shannon Expansion

Let $f(a, b)$ be an m-valued 2-input logic function with variables a and b, that is, let

$$f : \mathbb{Z}_m \times \mathbb{Z}_m \to \{0, 1\}.$$

Then, throughout this paper, we call the ESOP form

$$f(a,b) = a^{\{0\}}b^{T_1} \oplus a^{\{1\}}b^{T_2} \oplus \cdots \oplus a^{\{m-1\}}b^{T_m}$$

the *multiple-valued Shannon expansion* of f. It should be noted that $T_i \subseteq \mathbb{Z}_m$, $i \in \{1, 2, \ldots, m\}$, is uniquely determined as follows:

$$T_i = \{b \in \mathbb{Z}_m \mid f(i-1, b) = 1\}.$$

Consider for example a 5-valued 2-input logic function $h(a,b)$, whose truth table is given in Table 1. The multiple-valued Shannon expansion of h is

$$h(a,b) = a^{\{0\}}b^{\{0,3,4\}} \oplus a^{\{1\}}b^{\{1,4\}} \oplus a^{\{2\}}b^{\{1,3\}} \oplus a^{\{3\}}b^{\{0,3,4\}} \oplus a^{\{4\}}b^{\{0,1,3\}}. \quad (4)$$

Table 1. A truth table for the 5-valued 2-input logic function $h(a,b)$

		b				
		0	1	2	3	4
	0	1	0	0	1	1
	1	0	1	0	0	1
a	2	0	1	0	1	0
	3	1	0	0	1	1
	4	1	1	0	1	0

Let F be the multiple-valued Shannon expansion of an m-valued 2-input logic function f, then the number $\tau(F)$ of the product terms in F satisfies $\tau(F) = m$ (before removing a product term containing a constant literal $b^{\emptyset}$).

2.2 Transformation Rules for ESOP Forms

One of the most famous currently known algorithms to simplify ESOP forms is EXMIN2, which was developed by Sasao [5]. The transformation rule for ESOP forms described in the following Theorem 1 is one of the rules utilized by the algorithm EXMIN2. Hereafter, the binary operator $\oplus$ for two sets denotes the symmetric difference of the two sets, that is,

$$S \oplus T = (\overline{S} \cap T) \cup (S \cap \overline{T}).$$

Theorem 1 ([5]). *For any four literals $a^{S_p}, b^{T_p}, a^{S_q}, b^{T_q}$ of two variables a and b,*

$$a^{S_p}b^{T_p} \oplus a^{S_q}b^{T_q} = a^{S_p \oplus S_q}b^{T_p} \oplus a^{S_q}b^{T_p \oplus T_q}$$

holds, where $S_p, T_p, S_q, T_q \subseteq \mathbb{Z}_m$.

Note that, according to Theorem 1, if $T_p = T_q$, then

$$a^{S_p}b^{T_p} \oplus a^{S_q}b^{T_p} = a^{S_p \oplus S_q}b^{T_p} \oplus a^{S_q}b^{\emptyset} = a^{S_p \oplus S_q}b^{T_p},$$

and hence the number of product terms decreases by exactly 1. For example, applying the transformation rule in Theorem 1 to the first and fourth product terms in the ESOP form in Eq. (4) results in

$$\begin{aligned} &h(a,b) \\ &= a^{\{0\}\oplus\{3\}}b^{\{0,3,4\}} \oplus a^{\{1\}}b^{\{1,4\}} \oplus a^{\{2\}}b^{\{1,3\}} \oplus a^{\{3\}}b^{\{0,3,4\}\oplus\{0,3,4\}} \oplus a^{\{4\}}b^{\{0,1,3\}} \\ &= a^{\{0,3\}}b^{\{0,3,4\}} \oplus a^{\{1\}}b^{\{1,4\}} \oplus a^{\{2\}}b^{\{1,3\}} \oplus a^{\{4\}}b^{\{0,1,3\}}. \qquad (5) \end{aligned}$$

In this paper, as seen later in Sections 3 and 4, applying the transformation rule given in Theorem 1, we propose an efficient algorithm to find a minimum ESOP form of any given m-valued 2-input logic function.

3 ESOP Matrices

In this section, we propose a method for expressing an ESOP form of an m-valued 2-input logic function in a Boolean matrix. The method makes it easier for us to intuitively understand the transformations of ESOP forms.

We begin with an example; consider the following ESOP form of the 5-valued 2-input logic function h (already seen in Eq. (5)):

$$a^{\{0,3\}}b^{\{0,3,4\}} \oplus a^{\{1\}}b^{\{1,4\}} \oplus a^{\{2\}}b^{\{1,3\}} \oplus a^{\{4\}}b^{\{0,1,3\}}.$$

Given such a 5-valued ESOP form having 4 product terms, we construct a Boolean 4×10 matrix as follows:

$$\left(\begin{array}{c|c} 1\,0\,0\,1\,0 & 1\,0\,0\,1\,1 \\ 0\,1\,0\,0\,0 & 0\,1\,0\,0\,1 \\ 0\,0\,1\,0\,0 & 0\,1\,0\,1\,0 \\ 0\,0\,0\,0\,1 & 1\,1\,0\,1\,0 \end{array}\right) \begin{array}{l} \leftarrow a^{\{0,3\}}b^{\{0,3,4\}} \\ \leftarrow a^{\{1\}}b^{\{1,4\}} \\ \leftarrow a^{\{2\}}b^{\{1,3\}} \\ \leftarrow a^{\{4\}}b^{\{0,1,3\}} \end{array}$$

which represents the ESOP form above; the first row of the Boolean matrix corresponds to the first term $a^{\{0,3\}}b^{\{0,3,4\}}$ in the ESOP form, i.e., the five elements in the left half of the first row correspond to the literal $a^{\{0,3\}}$, and the five elements in the right half correspond to the literal $b^{\{0,3,4\}}$, and so on. (Notice that each literal is described by a bit pattern of length 5.)

Generally, let $F = \bigoplus_{i=1}^{t} a^{S_i}b^{T_i}$ be an m-valued ESOP form, then we call the following $t \times 2m$ Boolean matrix H the *ESOP matrix* of F:

$$H = \left(\begin{array}{cccc|cccc} \ell_{11} & \ell_{12} & \cdots & \ell_{1m} & r_{11} & r_{12} & \cdots & r_{1m} \\ \ell_{21} & \ell_{22} & \cdots & \ell_{2m} & r_{21} & r_{22} & \cdots & r_{2m} \\ \vdots & \vdots & \cdots & \vdots & \vdots & \vdots & \cdots & \vdots \\ \ell_{t1} & \ell_{t2} & \cdots & \ell_{tm} & r_{t1} & r_{t2} & \cdots & r_{tm} \end{array}\right)$$

where

$$\ell_{ij} = a^{S_i}(j-1) = \begin{cases} 1 \text{ if } j-1 \in S_i; \\ 0 \text{ if } j-1 \notin S_i \end{cases}$$

and

$$r_{ij} = b^{T_i}(j-1) = \begin{cases} 1 \text{ if } j-1 \in T_i; \\ 0 \text{ if } j-1 \notin T_i \end{cases}$$

for every $i \in \{1, 2, \ldots, t\}$ and $j \in \{1, 2, \ldots, m\}$.

Given an ESOP form of t product terms, its corresponding $t \times 2m$ ESOP matrix is uniquely determined. Conversely, given a $t \times 2m$ Boolean matrix, the corresponding ESOP form of t product terms is also uniquely determined.

Hereafter, for a $t \times 2m$ ESOP matrix H, partitioning H into the left block H^{L} and the right block H^{R}, we often write

$$H = \left(H^{\mathrm{L}} \middle| H^{\mathrm{R}} \right),$$

where H^{L} and H^{R} are $t \times m$ matrices. For example, let H be the ESOP matrix of the multiple-valued Shannon expansion

$$f(a,b) = a^{\{0\}} b^{T_1} \oplus a^{\{1\}} b^{T_2} \oplus \cdots \oplus a^{\{m-1\}} b^{T_m}$$

of an m-valued 2-input logic function f, then H^{L} must be an identity matrix (unit matrix) of size m, i.e., H must be like

$$H = \left(I \middle| H^{\mathrm{R}} \right),$$

where I denotes the identity matrix (also in the sequel).

Furthermore, for each of the left block H^{L} and the right block H^{R} of a $t \times 2m$ ESOP matrix H, using row vectors $\boldsymbol{\ell}_i$ and $\boldsymbol{r}_i$ of length m, we often write

$$H = \left(H^{\mathrm{L}} \middle| H^{\mathrm{R}} \right) = \left(\begin{array}{c|c} \boldsymbol{\ell}_1 & \boldsymbol{r}_1 \\ \boldsymbol{\ell}_2 & \boldsymbol{r}_2 \\ \vdots & \vdots \\ \boldsymbol{\ell}_t & \boldsymbol{r}_t \end{array} \right).$$

As will be seen in Section 4, our algorithm uses the following two transformation rules (named R1 and R2) for ESOP forms:

(R1) $a^{S_p} b^{T_p} \oplus a^{S_q} b^{T_q} = a^{S_p \oplus S_q} b^{T_p} \oplus a^{S_q} b^{T_p \oplus T_q}$ (Theorem 1);
(R2) $a^{S_p} b^{T_p} \oplus a^{S_q} b^{T_q} = a^{S_q} b^{T_q} \oplus a^{S_p} b^{T_p}$ (commutativity of Exclusive-OR).

Considering the two rules above applied to ESOP matrices, we naturally get the following two definitions.

Definition 1. *Given an ESOP matrix of t rows,* applying rule R1 *to the p-th and q-th rows means the following row operation:*

$$\left(\begin{array}{c|c} \boldsymbol{\ell}_1 & \boldsymbol{r}_1 \\ \vdots & \vdots \\ \boldsymbol{\ell}_p & \boldsymbol{r}_p \\ \vdots & \vdots \\ \boldsymbol{\ell}_q & \boldsymbol{r}_q \\ \vdots & \vdots \\ \boldsymbol{\ell}_t & \boldsymbol{r}_t \end{array} \right) \xrightarrow{\mathrm{R1}_{(p,q)}} \left(\begin{array}{c|c} \boldsymbol{\ell}_1 & \boldsymbol{r}_1 \\ \vdots & \vdots \\ \boldsymbol{\ell}_p \oplus \boldsymbol{\ell}_q & \boldsymbol{r}_p \\ \vdots & \vdots \\ \boldsymbol{\ell}_q & \boldsymbol{r}_p \oplus \boldsymbol{r}_q \\ \vdots & \vdots \\ \boldsymbol{\ell}_t & \boldsymbol{r}_t \end{array} \right),$$

where the operator $\oplus$ represents bitwise Exclusive-OR of two row vectors.

Definition 2. *Given an ESOP matrix of t rows,* applying rule R2 *to the p-th and q-th rows means the following row operation:*

$$\left(\begin{array}{c|c} \boldsymbol{\ell}_1 & \boldsymbol{r}_1 \\ \vdots & \vdots \\ \boldsymbol{\ell}_p & \boldsymbol{r}_p \\ \vdots & \vdots \\ \boldsymbol{\ell}_q & \boldsymbol{r}_q \\ \vdots & \vdots \\ \boldsymbol{\ell}_t & \boldsymbol{r}_t \end{array}\right) \xrightarrow{\mathrm{R2}_{(p,q)}} \left(\begin{array}{c|c} \boldsymbol{\ell}_1 & \boldsymbol{r}_1 \\ \vdots & \vdots \\ \boldsymbol{\ell}_q & \boldsymbol{r}_q \\ \vdots & \vdots \\ \boldsymbol{\ell}_p & \boldsymbol{r}_p \\ \vdots & \vdots \\ \boldsymbol{\ell}_t & \boldsymbol{r}_t \end{array}\right).$$

Based on these two definitions, the following lemma immediately holds.

Lemma 1. *Let F be an arbitrary ESOP form, and let H be the ESOP matrix of F. Assume that applying rule* R1 *(*R2*) to the p-th and q-th product terms in F results in an ESOP form F', and that applying rule* R1 *(*R2*) to the p-th and q-th rows of H results in a matrix H'. Then, H' is the ESOP matrix of F'.*

Note that rules R1 and R2 for an ESOP matrix H can be regarded just as the elementary row operations (on a Boolean matrix) for each of the left block H^{L} and the right block H^{R} of H.

4 Our Algorithm

We are now ready to present our algorithm.

Given a truth table of an m-valued 2-input logic function f as an input, the following algorithm outputs a minimum ESOP form of f.

[Our algorithm]

1. Find the multiple-valued Shannon expansion $\bigoplus_{i=1}^{m} a^{\{i-1\}} b^{T_i}$ of f from the truth table of f, and let
$$\left(I \middle| H^{\mathrm{R}}\right)$$
be its $m \times 2m$ ESOP matrix. (Recall that the left block of the ESOP matrix of a multiple-valued Shannon expansion is always an identity matrix I.)
2. Apply a series of rules R1 and R2 to the ESOP matrix so that the right block H^{R} is transformed into a Boolean matrix in row echelon form. (Using a known algorithm, e.g. Gaussian elimination algorithm, one can obtain such a row echelon form within an $O(m^2)$ number of transformations. Each transformation can be done in $O(m)$ bit operations.) Note that, since these transformations are elementary row operations for each of the left and right blocks, the rank of the right block never changes, and hence its rank remains rank(H^{R}) (after the series of rules R1 and R2), where rank(M) denotes the rank of a matrix M.

3. Note that the current ESOP matrix in row echelon form has an all-zero submatrix O in the lower part of its right block as follows:

$$\left(\begin{array}{c|c} * & * \\ * & O \end{array}\right)\begin{array}{l} \}\mathrm{rank}(H^{\mathrm{R}}) \\ \end{array}.$$

   Construct the ESOP form corresponding to this ESOP matrix, and remove all the $m - \mathrm{rank}(H^{\mathrm{R}})$ terms containing constant 0. The resulting ESOP form is the output of our algorithm.

We now demonstrate the execution of our algorithm with the 5-valued 2-input logic function h which was given in Table 1. In step 1 of our algorithm, the multiple-valued Shannon expansion of h is given in Eq. (4), and hence we have a 5×10 Boolean matrix

$$\left(\begin{array}{ccccc|ccccc} 1&0&0&0&0&1&0&0&1&1 \\ 0&1&0&0&0&0&1&0&0&1 \\ 0&0&1&0&0&0&1&0&1&0 \\ 0&0&0&1&0&1&0&0&1&1 \\ 0&0&0&0&1&1&1&0&1&0 \end{array}\right)$$

as its ESOP matrix. In step 2, applying a series of four transformations $\mathrm{R1}_{(1,4)}$, $\mathrm{R1}_{(1,5)}$, $\mathrm{R1}_{(2,3)}$ and $\mathrm{R1}_{(2,5)}$ to the matrix, we make the right block of the ESOP matrix be a Boolean matrix in row echelon form

$$\left(\begin{array}{ccccc|ccccc} 1&0&0&1&1&1&0&0&1&1 \\ 0&1&1&0&1&0&1&0&0&1 \\ 0&0&1&0&0&0&0&0&1&1 \\ 0&0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0&0 \end{array}\right).$$

In step 3, from the ESOP matrix above, our algorithm outputs the following ESOP form:

$$a^{\{0,3,4\}}b^{\{0,3,4\}} \oplus a^{\{1,2,4\}}b^{\{1,4\}} \oplus a^{\{2\}}b^{\{3,4\}},$$

which is a minimum ESOP form of h as will be guaranteed in Theorem 2.

Since applying rules R1 and R2 in step 2 of our algorithm can be regarded exactly as elementary row operations for each of the left and right blocks as mentioned above, the rank of the right block never changes, and hence its rank remains $\mathrm{rank}(H^{\mathrm{R}})$. Therefore, the following Lemma 2 holds.

Lemma 2. *Let f be an m-valued 2-input logic function, and let*

$$\left(I \middle| H^{\mathrm{R}}\right)$$

be the ESOP matrix of its multiple-valued Shannon expansion. Then, our algorithm outputs an ESOP form F such that $\tau(F) = \mathrm{rank}(H^{\mathrm{R}})$.

Using Lemma 2, one can verify the correctness of our algorithm and obtains the following Theorem 2.

Theorem 2. *For every m-valued 2-input logic function f, our algorithm outputs a minimum ESOP form.*

Proof. Omitted due to the page limitation. □

5 Conclusions

In this paper, we first introduced a method for expressing an ESOP form as a matrix, and then, utilizing the method, we proposed an algorithm to find a minimum ESOP form of any given m-valued 2-input logic function in $O(m^3)$ bit operations.

Lemma 2 and Theorem 2 also imply that, given an m-valued 2-input logic function f, the minimum number of product terms among all the ESOP forms of f is equal to $\mathrm{rank}(H^{\mathrm{R}})$, where

$$\left(I \middle| H^{\mathrm{R}}\right)$$

is the ESOP matrix of the multiple-valued Shannon expansion of f. Furthermore, even if a logic function f is given in an ESOP form F which is not necessarily that of the multiple-valued Shannon expansion of f, one can efficiently find a minimum ESOP form of f having $\min\{\mathrm{rank}(H^{\mathrm{L}}), \mathrm{rank}(H^{\mathrm{R}})\}$ product terms by extending the results in Section 4, where

$$\left(H^{\mathrm{L}} \middle| H^{\mathrm{R}}\right)$$

is the ESOP matrix of F.

We have so far considered the minimization of ESOP forms of m-valued 2-input logic functions, namely only for the case of $m_1 = m_2 = m$. However, even for the case of $m_1 \neq m_2$, i.e., for multiple-valued 2-input logic functions $f : \mathbb{Z}_{m_1} \times \mathbb{Z}_{m_2} \to \{0, 1\}$ with $m_1 \neq m_2$, one can easily construct an efficient minimization algorithm by redefining ESOP matrices as $t \times (m_1 + m_2)$ matrices.

Acknowledgments

We thank Dr. Xiao Zhou and Dr. Takehiro Ito for their valuable discussions and suggestions. This work was supported by KAKENHI (19700002).

References

1. Fleisher, H., Tavel, M., Yeager, J.: A computer algorithm for minimizing Reed-Muller canonical forms. IEEE Transactions on Computers 36(2), 247–250 (1987)
2. Gaidukov, A.: Algorithm to derive minimum ESOP for 6-variable function. In: Proceedings of the Fifth International Workshop on Boolean Problems, Freiberg (2002)
3. Hirayama, T., Nishitani, Y., Sato, T.: A faster algorithm of minimizing AND-EXOR expressions. IEICE Trans. Fundamentals E85-A(12), 2708–2714 (2002)
4. Mizuki, T., Otagiri, T., Sone, H.: An application of ESOP expressions to secure computations. Journal of Circuits, Systems, and Computers 16(2), 191–198 (2007)
5. Sasao, T.: EXMIN2: a simplification algorithm for exclusive-or sum-of-products expressions for multiple-valued-input two-valued-output functions. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 12(5), 621–632 (1993)

6. Sasao, T.: Switching Theory for Logic Synthesis. Kluwer Academic Publishers, Boston (1999)
7. Sasao, T., Besslich, P.: On the complexity of mod-2 sum PLA's. IEEE Transactions on Computers 39(2), 262–266 (1990)
8. Song, N., Perkowski, M.A.: Minimization of exclusive sum-of-products expressions for multiple-valued input, incompletely specified functions. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 15(4), 385–395 (1996)
9. Stergiou, S., Papakonstantinou, G.: Exact minimization of ESOP expressions with less than eight product terms. Journal of Circuits, Systems and Computers 13(1), 1–15 (2004)
10. Stergiou, S., Voudouris, D., Papakonstantinou, G.: Multiple-value exclusive-or sum-of-products minimization algorithms. IEICE Trans. Fundamentals E87-A(5), 1226–1234 (2004)
11. Ye, Y., Roy, K.: An XOR-based decomposition diagram and its application in synthesis of AND/XOR networks. IEICE Trans. Fundamentals E80-A(10), 1742–1748 (1997)