Reduction of Sizes of Decision Diagrams by Autocorrelation Functions

Mark G. Karpovsky
Dept. of Electrical and Computer Engineering
8 Saint Marry's Street
Boston University
Boston, Ma 02215
USA

Radomir S. Stanković
Dept. of Computer Science
Faculty of Electronics
Beogradska 14
18 000 Niš
Serbia

Jaakko T. Astola
Tampere Int. Center for Signal Processing
Tampere University of Technology
Tampere
Finland

This work was partially supported by the BSF Grant 1998154.
This work was supported by the Nokia Oyj Foundation Visiting Fellowship, and the Academy of Finland, Finnish Center of Excellence Programme, Grant No. 44876, and the EXSITE Project No. 51520.

Abstract

This paper discusses optimization of decisions diagrams (DDs) by total autocorrelation functions. We present an efficient algorithm for construction of Linearly Transformed Binary Decision Diagrams (LT-BDDs) and Linearly transformed multi-terminal BDDs (LT-MTBDDs)for systems of Boolean functions, based on linearization of these functions by the corresponding autocorrelation functions. Then, we present a method for reduction of sizes of DDs by a level by level reduction of the width of DDs using the total autocorrelation functions. The approach provides for a simple procedure for minimization of LT-BDDs and LT-MTBDDs and upper bounds on their sizes. Experimental results for benchmarks illustrate that the proposed method on average is very efficient.

key words: *Logic synthesis, spectral techniques, decision diagrams, linear transforms, autocorrelation functions.*

I. Introduction

An n variable k-output discrete function $f = (f^{(0)}, \ldots, f^{(k-1)})$ is defined as a mapping $f : \times_{i=1}^{n} C_i \rightarrow P^k$, where $\times$ denotes the direct product, C_i, $i = 1, \ldots n$ and P are finite sets of cardinalities c_i and p, respectively, $n, k \in N$, N is the set of natural numbers.

To get a mathematically tractable model for discrete functions, we assume that C_i endorse the structure of a group, and P is a field [36]. Switching functions are a particular case when $C_i = \{0, 1\}, \oplus$ for each i, where $\oplus$ denotes the addition modulo 2 (EXOR), and $P = GF(2)$ is the Galois field of order 2 [1]. Thus, a multi-output switching function is defined as $f : C_2^n \rightarrow GF(2)^k$.

Decision diagrams (DDs) are a data structure permitting efficient representation of discrete functions defined on groups of large orders [27]. DDs are defined for representation of different classes of discrete functions by using decomposition rules to assign a given f to a DD, [27], [31], [34]. In this paper, the considerations are restricted to basic DDs for functions on C_2^n. Binary DDs (BDDs) are the basic concept used to represent single output switching functions [4]. Multiple-output switching functions are represented by Shared BDDs [21]. Multi-terminal binary DDs (MTBDDs) [5] are used to represent functions on C_2^n. They can represent systems of Boolean functions described by the corresponding integer equivalent functions $f(x)$.

Extensions and generalizations of the considerations presented to DDs for functions on arbi[3]trary not-necessarily Abelian groups are straightforward [32], [34]. The multiple-valued logic functions are included as an example of functions on p-adic groups C_p into $GF(p)$, $p \in N$ [13].

DDs are derived by the reduction of decision trees (DTs). The reduction is performed by sharing the isomorphic subtrees and deleting the redundant information from the DT. The reduction procedure is formalized through the reduction rules [27] adapted to the range of functions represented and the used decomposition rules.

For many applications, the efficiency of DD representations is determined by the size of the DD defined as the number of nodes in the DD for a given f. The width of the DD is defined as the maximal number of nodes at a level, where a level consist of nodes to which the same variable is assigned. The size and the width determine the area of the DD, which is also an important parameter in applications and comparisons of different DDs [32].

There are at least two approaches to the reduction of the size of DDs for a given function or a given class of functions

1. Usage of different DDs defined by using different decomposition rules.
2. Transformations and manipulations with functions represented, such as functional decomposition, variable ordering, and outputs pairing.

The following example illustrates optimization of DDs by ordering and transformation of variables.

Example 1: Consider the function $f(x) = f(x_1, x_2)$ given by the vector of function values $\mathbf{F} = [f(0), f(1), f(2), f(3)]^T = [1, 2, 2, 1]^T$. Permutation of variables converts f into a function $f_\sigma = f(y_1, y_2)$, where $y_1 = x_2$, $y_2 = x_1$, given by the vector $\mathbf{F}_\sigma = [f(0), f(2), f(1), f(3)]^T = [1, 2, 2, 1]^T$. This vector is generated from $\mathbf{F}$ by a permutation σ of function values which in the matrix notation $\sigma = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$. This permutation of variables does not reduce the

size of MTBDD. However, if we select the permutation matrix $\sigma_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$[4], which converts f into a function $f_{\sigma_1} = f(z_1, z_2)$, where $z_1 = x_1 \oplus x_2$, and $z_2 = x_2$, then the vector of function values is $\mathbf{F}_{\sigma_1} = [f(0), f(3), f(2), f(1)]^T = [1, 1, 2, 2]^T$. Fig. 1 shows MTBDDs of f, f_σ, and f_{σ_1}. Therefore, by extending the transformations over variables besides the permutation, we can achieve reduction of DDs.

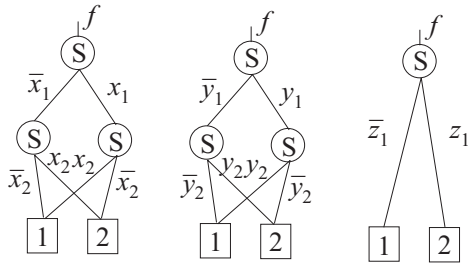

Fig. 1. MTBDD(f), MTBDD(f_σ), and MTBDD(f_{σ_1}).

This example shows that the problem of reduction of sizes of DDs by transformation of variables can be reduced to determination of an optimal permutation in the vector of function values which results in a DD of a smaller size.

In spectral interpretation of DDs, the use of different decomposition rules reads as the use of different spectral transforms to define a DD [35] and various DDs are uniformly considered as particular examples of Spectral Transform DDs (STDDs) defined as graphic representations of some Fourier series-like expressions [34].

The main drawback of these approaches is the lack of exact algorithms to determine the suitable decomposition rules for a given f, which in spectral interpretation of DDs requires the choice of the most suited spectral transform [35], or to determine the best ordering of variables or outputs pairing.

The dynamic reordering of variables for optimization of DDs has been introduced for BDDs in [8] and further improved in [25]. The same problem was considered and elaborated by many authors in a number of publications, see for example, [6], [19]. The ordering of variables

is an NP-hard problem [3]. Moreover, it is NP-hard to compute an almost optimal variable ordering when a given function is already represented by a BDD [29], [28], [37]. Therefore, several heuristic algorithms are proposed for reduction of DDs by ordering of variables with heuristic based on local transformations. For example, this ordering can be implemented as dynamic sifting [25] and further generalized into group sifting [22], [23].

Linearly transformed BDDs (LT-BDDs) are generalizations of BDDs derived by using the Shannon expansion with respect to a linear combination of a subset of variables. It is obvious that the same generalization applies to MTBDDs, since in BDDs and MTBDDs the same underlying group of binary vectors is assumed as the domain for the functions represented. Moreover, extensions to STDDs are straightforward.

Construction of LT-BDDs is an interesting and important task, since in many practical cases LT-BDDs permit exponential reduction of the size compared to the BDDs.

We note that regarding applications in circuit design from DDs, a price for the reduced size of LT-BDDs is

1. A hardware required to implement a linear transformation of variables.
2. Difficulty to determine an optimal transformation of variables.

A linear transformation over the variables can be represented by an $(n \times n)$ matrix over $GF(2)$, and the required space is small compared to the space required to store a BDD or a LT-BDD. However, the other requirement can be considered as a bottleneck for applications of LT-BDDs, although there are heuristic algorithms to determine a suitable linear combination of variables. The algorithm proposed in [18] splits the set of variables into subsets of adjacent variables and combines variables within a subset. A similar algorithm implemented as a windowing procedure is proposed in [10]. The algorithm for construction of LT-BDDs presented in [12] is an application of evolutionary computation techniques to this problem. In [19], the algorithm presented in [18] is combined with sifting method used in variable ordering in DDs [25] with special attention paid to the integration of the method into the existing CAD systems. Algorithms for efficient manipulations with LT-BDDs, prepared as an extension of CUDD package [30] further support the applications of LT-BDDs [11].

In this paper, we discuss applications of total autocorrelation functions to reduction of sizes of SBDDs. We show that a method for linearization of switching functions introduced in [14],

and further elaborated, and extended to multiple-valued functions in [13] provides for a deterministic algorithm to construct an optimal linear transformation of variables in LT-BDDs by the total autocorrelation functions and the inertia groups of the original systems of Boolean functions [13]. Then, we show that the maximum values of the total autocorrelation function can be used as a measure to determine the number of pairs of isomorphic subtrees rooted at the nodes at the same level in the MTBDD. Using this property, we developed a method for minimization of the width of MTBDDs level by level, providing at each level the maximum number of isomorphic subtrees. Under this assumption, the method presented can be considered as a deterministic method for reduction of sizes of MTBDDs. The differences with the existing methods, advantages and limitations of the proposed method can be summarized as follows.

1. Instead of using heuristics (like sifting) to minimize the size L of a MTBDD for a given function f (this problem is NP-hard [3]) we represent L as $L = L_{n-1}+L_{n-2}+\cdots+L_1$, where L_i is a number of nodes at the level i and since in most cases $L_{n-1} \geq L_{n-2} \geq \cdots \geq L_1$, we first construct a set of linearly transformed MTBDDs (LT-MTBDDs), minimizing L_{n-1} for the given system of functions, then within this set we construct a subset, minimizing L_{n-2}, etc.
2. The complexity of the proposed approach is determined by the complexity of a procedure for computing for a given function f it's total autocorrelation function B_f. If f is a system of k Boolean functions of n variables, then B_f can be computed by the Fast Walsh Transform [13] and Wiener-Khincin theorem [13] by the procedure which will require not more than $min(O(n2^{n+k}), O(2^{2n}))$ steps. In many cases, if f is defined by a Boolean expression, then B_f (and the corresponding optimized MTBDD) can be computed analytically from f, (see Example 9 below).
3. The proposed approach provides for upper bounds on sizes of MTBDDs for a given f.
4. Experimental results for benchmarks and randomly generated multiple-output switching functions show that the proposed approach in many cases results in smaller LT-MTBDDs as compared with the existing approaches such as sifting. In addition to this, unlike the optimization by ordering of variables, the proposed approach can be applied to symmetric functions.

Autocorrelation is very useful in spectral methods for analysis and synthesis of networks realizing logic functions.

For a given n-variable switching function f, the autocorrelation function B_f is defined as

$$B_f(\tau) = \sum_{x=0}^{2^n-1} f(x)f(x \oplus \tau), \quad \tau \in \{0, \ldots, 2^n - 1\},$$

The Winer-Khinchin theorem [13] states a relationship between the autocorrelation function and Walsh (Fourier) coefficients $B_f = 2^n W^{-1}(Wf)^2$.

We note that the autocorrelation function is invariant to the shift operator $\oplus$ in terms of which B_f is defined. Due to that, it performs some compression of data in the sense that several functions may have the same autocorrelation function B_f. Fig. 2 illustrates this property of B_f. In this figure, $\varphi_\tau(x) = f(x \oplus \tau)$ is a shifted function for f, WK denotes the Wiener-Khinchin theorem, and C_{B_f} denotes the set of functions having the same autocorrelation function B_f. However, although we are sacrificing a part of data, this compression makes description of problems where the shift is not important very efficient. For example, the autocorrelation is very useful in the applications where we are interested in the equality of some function values, and not in their magnitude.

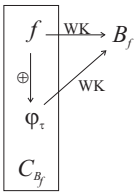

Fig. 2. Autocorrelation functions.

A. Total autocorrelation function

For a system of k switching functions $f^{(i)}(x_1, \ldots, x_n)$, $i = 0, \ldots, k-1$, the total autocorrelation function is defined as the sum of autocorrelation functions of each function in the system. Thus, $B_f(\tau) = \sum_{i=0}^{k-1} B_{f^{(i)}}(\tau)$.

Note that for any $\tau \neq 0$, $B_f(\tau) \leq B_f(0)$. Set $G_I(f)$ of all values for τ such that $B_f(\tau) = B_f(0) = \sum_{i=0}^{k-1} \sum_{x=0}^{2^m-1} f^{(i)}(x)$ is a group with respect to the EXOR as the group operation

which is denoted as the inertia group of the system f.

We note that the complexity of computing the total autocorrelation function $B_f(\tau)$ by the Wiener-Khinchin theorem [13] and by the fast Walsh transform [1], [13], expressed in the number of arithmetic operations does not exceed $O(n2^{n+k})$ and this approach is efficient only for small k. The straightforward application of the definition of B_f requires at most $O(2^{2n})$ computations for any k. It should be noted that the Walsh transform, can be performed over BDDs [5], [9], which reduces the limitations to the number of variables in calculations related to the implementation of Wiener-Khinchin theorem.

Four approaches for calculation of autocorrelation functions by DDs are presented in [33] providing for possibility of compromising between the space and time restrictions, as well as between the requirements to calculate all the autocorrelation coefficients, subsets of coefficients or a single coefficient. In particular, for the considerations in this paper, the most interesting is the method that for a given function f reduces the calculation of autocorrelation coefficients to the manipulation with labels in LT-BDD(f) derived by exploiting the recursive structure of the autocorrelation matrix for f. The space complexity of this method is proportional to the size of the BDD(f). An experimental verification of some of these methods is given in [20]. For example, for randomly generated Boolean functions of 10 variables with 5,20,35,50,65,80, and 95 % of elements equal to 1 in the truth-vector, the autocorrelation function is calculated by the Wiener-Khinchin theorem performed over vector representations, in 0.22, 0.55, 0.50, 0.77, 0.72, 0.50, and 0.27 seconds, respectively. If the calculation is performed over BDDs, then, it is required 1.82, 3.29, 4.45, 4.78, 4.89, 4.17, and 2.42 seconds.

A generalization of autocorrelation to systems of p-valued m-variable functions or functions defined over finite Abelian groups is straightforward and can be found e.g. in [13].

III. Linearization of Boolean Functions

Linearization of Boolean functions assumes representing a given system of Boolean functions as the superposition of a system of linear Boolean functions and a residual nonlinear part of minimal complexity. Fig. 2 shows the realization of a given function f based on the linearization. The network produced consists of a serial connection of a linear and a nonlinear blocks. The linear block consists of EXOR circuits only. For an n-variable function, complexity

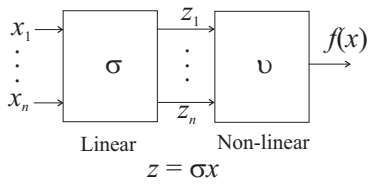

Fig. 3. Realization by linearization of f.

(number of equivalent two-input gates) of the linear block increases asymptotically no faster than $n^2/\log_2 n$ as $n \to \infty$ [13], whereas the complexity of the nonlinear block is almost always an exponentially increasing function of n. Therefore, the complexity of the linear block may be ignored in linearization problems.

The linearization of a system of Boolean functions is performed to meet a selected criterion for the complexity of the realization of $f(x)$. In this paper, as in [13], we consider the complexity of $f(x)$ as the minimum number of two input gates to implement $f(x)$.

A. Linearization problem

Consider a complexity criterion as $\mu(\nu) = |\{(x,y)|x,y \in C_2^n, d(x,y) = 1, \nu(x) = \nu(y)\}|$, where $d(x,y)$ denotes the Hamming distance of binary vectors x and y, and ν is a Boolean function of n variables. Then, the linearization problem may be formulated as follows.

For a given $f : C_2^n \to C_2$, find a nonsingular $(n \times n)$ matrix σ over $GF(2)$ such that f is mapped into another function f_σ defined by the requirement $f(x) = f_\sigma(\sigma \odot x)$, and $\mu(f_\sigma)$ takes the maximum value over the set of all nonsingular over $GF(2)$ matrices σ,

where $\odot$ denotes the multiplication in $GF(2)$ of the matrix σ with the vector x.

B. Solution of the linearization problem

The following procedure, which we denote as the Linearization of Switching Functions (LSF) procedure, provides for a solution of the linearization problem.

LSF-procedure

1. Construct (for example by the Wiener-Khinchin theorem and Fast Walsh Hadamard Transform (FWHT)) the autocorrelation function $B_f(\tau) = \sum_x f(x)f(x \oplus \tau)$,
2. Find τ_0 such that $B(\tau_0) = max_{\tau \neq 0} B(\tau)$.

3. Find τ_i, $i = 1, \ldots, n-1$, such that $B(\tau_i) = max_{\tau \notin T_i} B(\tau)$, where $T_i = \{c_0\tau_0 \oplus c_1\tau_1 \oplus \cdots \oplus c_{i-1}\tau_{i-1}\}$, $c_i \in \{0, 1\}$.
4. Construct $\mathbf{T} = \left[\tau_0, \tau_1, \cdots, \tau_{n-1} \right]^T$, and determine $\sigma = \mathbf{T}^{-1}$, where all the calculations are in $GF(2)$.

Complexity of solving the linearization problem for a given f in terms of a number of the required arithmetic operations does not exceed $O(n2^n)$ and may be much smaller than this if we have a compact description of f [13]. For randomly generated Boolean functions, the linearization results in about 20% reduction in the gate counts [17], [16]. Generalization to multi-valued p-ary logic (p-prime) is straightforward.

We note that a set of τ_i, such that $B(\tau_i) = B(0)$ form a group (inertia group [13]) and by selecting vectors which form a basis for this group as columns of $\mathbf{T}$, the above procedure results in the minimum number of essential variables for the non-linear part. Example 2 illustrates this feature, which will be used to determine a quasioptimal linear transformation of variables in LT-BDDs. We note also that the above linearization procedure maximizes the number of neighboring minterms, as specified by the minimization criterion, $\mu(\nu)$. This feature will be used in reduction of the size of MTBDDs.

Example 2: Table 1 shows a system of two four-variable Boolean functions $f^{(0)}$ and $f^{(1)}$, and the total autocorrelation function B of this system determined as in the Step 1 of the LSF-procedure. Fig. 4 shows a direct AND-EXOR realization of this system with two-input circuits. This realization is chosen for a comparison to the realization by linearization of f, since in the linearization method the linear part is realized by EXOR circuits, and the criterion for minimization is expressed in terms of two-input gates. The maximum value of $B(\tau) = B(0) = 16$ for the inputs $5 = (0101)$, $10 = (1010)$, and $15 = (1111)$. This performs the Step 2 in the LSF-procedure. Thus, the inertia group for this system is $G_I = \{(0000), (0101), (1010), (1111)\}$. As a basis for G_I we take (0101) and (1010), and determine $\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$. Then, $\sigma = \mathbf{T}^{-1} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$, which completes the Step 3 and Step 4 in the LSF-procedure.

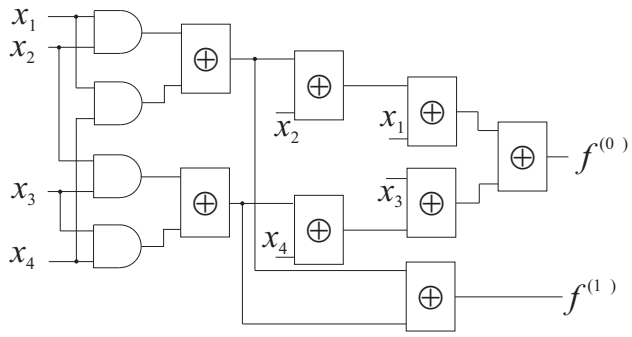

Fig. 4. Realization of $f^{(0)}$ and $f^{(1)}$ in Example 2.

Matrix $\mathbf{T}$ determines a reordering of the truth-vector $\mathbf{F}$ of a four-variable function f as

$$\mathbf{F}_\sigma = [f(0), f(10), f(5), f(15), f(4), f(14), f(1), f(11), f(8), f(2), f(13), f(7), f(12), f(6), f(9), f(3)]^T.$$

Table 1 shows functions $f_\sigma^{(0)}$ and $f_\sigma^{(1)}$ produced by this reordering from $f^{(0)}$ and $f^{(1)}$. Thus, these functions satisfy the relation

$$f_\sigma(x) = f(\sigma^{-1} \odot x).$$

Therefore, we have for z_1, z_2, z_3, z_4 in Fig. 3 $z_1 = x_1 \oplus x_3$, $z_2 = x_2 \oplus x_4$, $z_3 = x_4$, $z_4 = x_3$. From there,

$$\begin{aligned} f^{(0)} &= z_1 \vee z_2 = (x_1 \oplus x_3) \vee (x_2 \oplus x_4) \\ f^{(1)} &= z_1 z_2 = (x_1 \oplus x_3)(x_2 \oplus x_4), \end{aligned}$$

where $\vee$ denotes logical OR. It should be noted that both $f^{(0)}$ and $f^{(1)}$ do not essentially depend on z_3 and z_4. Fig. 5 shows the corresponding AND-EXOR realization of this system of Boolean functions by two-input circuits. Thus, for a comparison with the direct realization in Fig. 4, the logical OR is realized by using AND and EXOR circuits.

IV. Linearization of Boolean functions and LT-BDDs

The linearization method for Boolean functions presented in Section 3 can be used for construction of linear transformations for BDDs and Shared BDDs (SBDDs) for systems of Boolean functions. This statement will be explained and illustrated by the following example.

TABLE I

System of Boolean functions.

	$x_1x_2x_3x_4$	$f^{(0)}$	$f^{(1)}$	B	$f_\sigma^{(0)}$	$f_\sigma^{(1)}$
0	0000	0	0	16	0	0
1	0001	1	0	8	0	0
2	0010	1	0	3	0	0
3	0011	1	1	8	0	0
4	0100	1	0	8	1	0
5	0101	0	0	16	1	0
6	0110	1	1	8	1	0
7	0111	1	0	8	1	0
8	1000	1	0	8	1	0
9	1001	1	1	8	1	0
10	1010	0	0	16	1	0
11	1011	1	0	8	1	0
12	1100	1	1	8	1	1
13	1101	1	0	8	1	1
14	1110	1	0	8	1	1
15	1111	0	0	16	1	1

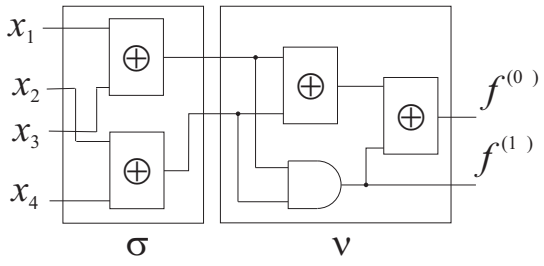

Fig. 5. Realization by linearization of $f^{(0)}$ and $f^{(1)}$ in Example 2.

Example 3: Fig. 6 shows SBDD for the system of Boolean functions $f^{(0)}(x)$ and $f^{(1)}(x)$ defined in Table 1. This SBDD represents the given system in the form of expressions

$$f^{(0)} = \overline{x}_1\overline{x}_2\overline{x}_3x_4 \oplus \overline{x}_1\overline{x}_2x_3 \oplus \overline{x}_1x_2x_3 \oplus \overline{x}_1x_2\overline{x}_3\overline{x}_4 \oplus x_1\overline{x}_2\overline{x}_3x_4 \oplus x_1x_2x_3\overline{x}_4 \oplus x_1\overline{x}_2\overline{x}_3 \oplus x_1x_2\overline{x}_3,$$
$$f^{(1)} = \overline{x}_1\overline{x}_0x_3x_4 \oplus \overline{x}_5x_2x_3\overline{x}_4 \oplus x_1\overline{x}_2\overline{x}_3x_4 \oplus x_1x_2\overline{x}_3\overline{x}_4.$$

As it is shown in Example 2 where we performed first five steps in the procedure for construction of LT-BDDs defined in what follows, after linearization, this system can be converted into the system $f_\sigma^{(0)}(z)$ and $f_\sigma^{(1)}(z)$, in terms of new variables z_i, $i = 1, 2, 3, 4$ expressed as the linear combination of original variables x_i, $i = 1, 2, 3, 4$. Then the given system can be represented by a SBDD derived by decomposition in terms of this linear combination of variables as is specified in the Step 6 of the for construction of LT-BDDs. Fig. 7 shows SBDD for the system of Boolean functions from Example 3 derived by the linearization method, where in the Step 7 of the procedure for determination of LT-BDDs (see below), the labels at the edges are determined. This SBDD represents the given system in the following form

$$f^{(0)} = (x_1 \oplus x_3) \oplus (\overline{x_1 \oplus x_3})(x_2 \oplus x_4),$$
$$f^{(1)} = (x_1 \oplus x_3)(x_2 \oplus x_4).$$

Thus, we can formulate the following procedure for determination of a linear transformation of variables in LT-BDDs.

Procedure 1: Procedure for generation of LT-BDD

1. Given an n-variable k-output switching function $f = (f^{(0)}, \ldots, f^{(k-1)})$.
2. Represent f by the integer-valued equivalent function $f(x) = \sum_{i=0}^{k-1} f^{(i)}(x)2^i$.
3. Construct characteristic functions $f_r(x)$ for $f(x)$, where $f_r(x) = 1$ if $f(x) = r$ and $f_r(x) = 0$ for $f(x) \neq r$.
4. Construct a total autocorrelation function for system $\{f_r(x)\}$.
5. Perform the LSF- procedure described in the Section 3, subsection B, and assign to $f(x)$ a function $f_\sigma(z)$, where $z = \sigma \odot x$.
6. Determine SBDD for $f_\sigma(z)$.
7. Relabel edges in SBDD($f_\sigma(z)$) by replacing each z_i with the corresponding linear combination of initial variables x_i.

Compared to the present methods for linear transformation of DDs, an advantage is that the linearization method based on total autocorrelation functions provides for a deterministic algorithm, in the sense that all steps are uniquely determined. At the same time, the method can be used for systems of Boolean functions.

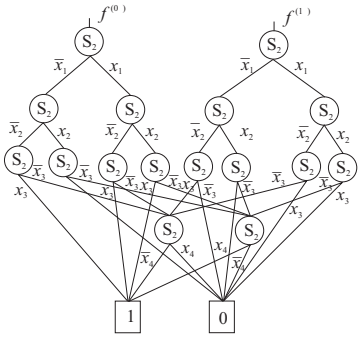

Fig. 6. SBDD for the system of functions.

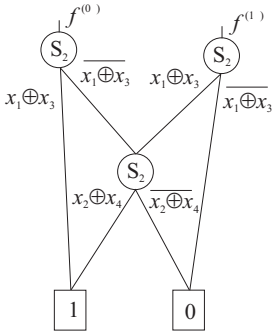

Fig. 7. Shared LT-BDD for the system of functions derived by the linearization method.

V. Reduction of Sizes of DD

In this section, we present a procedure for minimization of MTBDDs for systems of Boolean functions by their total autocorrelation functions. It is assumed that a given system is represented by the integer equivalent function $f(x)$. We note that the reduction of $size(MTBDD(f_x))$ is an NP-complete problem [3]. The proposed procedure provides for the nearly minimal solutions in the following way: it performs minimization of the MTBDD for a given $f(x_1, \ldots x_n)$ level by level by starting from the bottom level corresponding to x_n. It guarantees the maxi-

mal number of pairs of equal values of f for input vectors which differ in the value of x_n. Thus, for each pair of equal values of f we can reduce a node at the lowest level in the MTBDD. Then, we perform reordering of pairs of equal values of f, and repeat the procedure at the new MTBDD for $n-1$ variables. Under the assumption that we already minimized the width at the previous level, we get a minimum width at the present level. The width is determined by the maximum value of the total autocorrelation function for $f(x)$. This maximum value may be achieved for many different n-tuples of variables $x = (x_1, \ldots, x_n)$. Therefore, the procedure depends on the choice of x in the sense that for different choices of x different reduction possibilities at the upper levels may be achieved. However, this is a usual feature of nearly optimal solutions of NP-hard problems.

Unlike the method described in [24], the method presented in this paper can be used for both single output and multiple-output networks, and extends the class of permutation matrices which are used in optimization of DDs by ordering of variables. Therefore, the proposed method always produce MTBDDs with smaller or at most equal sizes compared to the methods using the ordering of variables.

Procedure 2: K-procedure

1. Assign to a given multi-output function $f^{(0)}, \ldots, f^{(k-1)}$, an integer equivalent function $Q_n = f(x) = \sum_{i=0}^{k-1-i} 2^i f^{(i)}(x)$.
2. Denote by R the range of $f(z)$ assigned to f. For every $i \in R$, construct characteristic functions $f_i(x) = \begin{cases} 1, & \text{if } f(x) = h, \\ 0, & otherwise, \end{cases}$.
3. Calculate the autocorrelation functions B_{f_i} for each $f_i(x)$, and the total autocorrelation function $B_f = \sum_i B_{f^{(i)}}$.
4. Determine the n-tuple of input variables $\tau = (x_1, \ldots, x_n)$, where B_f takes the maximum value, excepting the value $B_f(0)$. If there are several choices, select anyone of them.
5. Determine a matrix $\sigma_n = \sigma$ from the requirement $\sigma \odot \tau = (0, \ldots, 0, 1)^T$, where $\odot$ denotes the multiplication over $GF(2)$.
6. Determine a function f_σ such that $f_\sigma(\sigma \odot x) = f(x)$. That means, reorder values in a vector $\mathbf{F}$ representing values of f by the mapping $x = (x_1, \ldots, x_n) \rightarrow x_\sigma$, where $x_\sigma = \sigma^{-1} \odot x$.
7. In a vector $\mathbf{F}_\sigma$ representing the values of f_σ, perform an encoding of pairs of adjacent

values by assigning the same symbol to the identical pairs. Denote the resulting function of $(n-1)$ variables by $\mathbf{Q}_{n-1}$.

8. Replace $f = Q_n = Q_{n-1}$ and repeat the procedure.
9. Repeat the previous procedure for $i = i - 1$ to some k until there are identical pairs in $\mathbf{Q}_k$.
10. Determine MTBDD for f_{σ_k}.

Remark 1: The K-procedure produces the maximal number of identical pairs of values or subtrees in a MTBDD at the positions pointed by the outgoing edges $\overline{x}_i$ and x_i for all $i = n, n-1, \ldots, 1$.

Remark 2: (Upper bound on a number of deleted nodes in MTBDDs)
The number of nodes in the resulting MTBDD(f_σ) is upperbounded by $L \leq 2^n - 1 - \frac{1}{8}\sum_{i=1}^{n} B_{max}^{(n-i-1)} 2^{i-7}$, where $B_{max}^{(k)}$ is the maximum value of the total autocorrelation function at the level k.

Remark 3: For each pair of equal values of f at adjacent positions which is produced by the reordering of function values determined by the K-procedure, a node in the MTBDD($f_{\sigma_i^{-1}}$) may be deleted form the BDD. It follows, that K-procedure produces the minimal number of different nodes at each level in the MTBDD($f_{\sigma_i^{-1}}$). However, since the pairing of nodes at the i-th level is performed by the total autocorrelation function for Q_i, this is not necessarily the exact minimum of nodes in the MTBDD(f), which may be achieved by an ordering of elements of $\mathbf{F}$ optimal in the sense that produces the MTBDD(f) of the minimum size.

Remark 4: A reordering of elements in $\mathbf{F}$ can be represented by the corresponding permutation matrix. We denote by $\mathbf{P}_{dv}$, $\mathbf{P}_{FreeBDD}$, and $\mathbf{P}_K$, the set of permutation matrices used in optimization of MTBDDs by ordering of variables, in FreeBDDs [6], and in MTBDDs for f_σ determined by K-procedure. Then, $P_{dv} \subset P_{FreeBDD} \subset P_K$.

We illustrate the K-procedure by the following example.

Example 4: Table 2 shows two functions f_0, and f_1 of four variables. These functions are represented by the integer equivalent function $f = 2f_0 + f_1$. The maximum value for the total autocorrelation function B_f is 8 which corresponds to the n-tuple $\tau_{max} = (1111)$.

Fig. 8 shows Multi-terminal Binary Decision Tree (MTBDT(f)) for f and Fig. 9 shows the corresponding MTBDD(f). We determine the matrix σ_4 from the requirement $\sigma_4 \odot \tau_{max} =$

$$\sigma_4 \odot \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

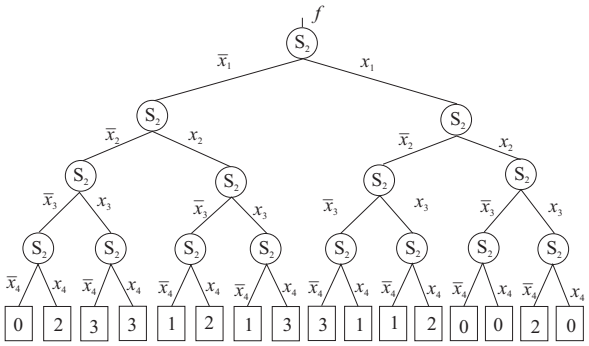

Fig. 8. MTBDT for f.

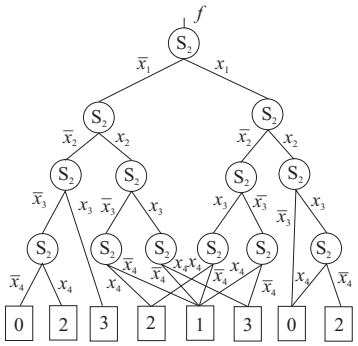

Fig. 9. MTBDD for f.

Therefore, we can choose $\sigma_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$. We determine the inverse matrix for σ_4 over $GF(2)$ as $\sigma_4^{-1} = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix}$. Table 3 shows the mapping of vectors of variables in f by

using σ_4^{-1}. Then, we determine

$$\begin{aligned}\mathbf{F}_\sigma &= [f(0), f(15), f(12), f(3), f(6), f(9), f(10), f(5), f(11), f(4), f(7), f(8), f(13), f(2), f(1), f(14)]^T, \\ &= [0, 6, 0, 3, 5, 1, 3, 2, 2, 1, 3, 3, 5, 3, 2, 2]^T,\end{aligned}$$

for $\sigma = \sigma_4$. We perform the encoding $\mathbf{F}_\sigma \rightarrow \mathbf{Q}_3$ of pairs of function values in $\mathbf{F}_\sigma$ as follows $\mathbf{Q}_3 = [0, 4, 1, 5, 6, 3, 4, 2]^T$, where $(0,0) = 0$, $(0,3) = 4$, $(1,1) = 1$, $(1,2) = 5$, $(2,1) = 6$, $(3,3) = 3$, $(2,2) = 2$. Fig. 10 shows Multi-terminal binary decision tree MTBDT($f_{\sigma_4^{-1}}$) and Fig. 11 shows MTBDT($f_{\sigma_4^{-1}}$) with encoded pairs of equal values for constant nodes. We denote the characteristic functions for 0,1,2,3,4,5,6 in $\mathbf{Q}_3$ as f_i. There is a single non-trivial characteristic function f_4. It is given by $f_4 = [0, 1, 0, 0, 0, 0, 1, 0]^T$, and its autocorrelation function is $B_{f_4} = [2, 0, 0, 0, 0, 0, 0, 2]^T$.

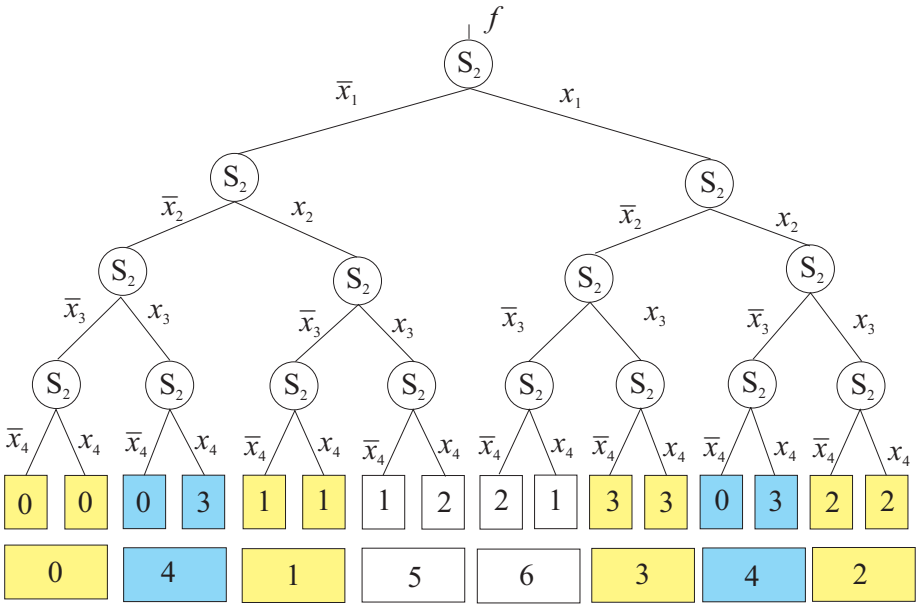

Fig. 10. MTBDT for $f_{\sigma_4^{-1}}$.

Since $max_{\tau \neq 0} B_{f_4}(\tau) = B_{f_4}(111) = 2$, we have for σ_3 that $\sigma_3 \odot \tau_{max} = \sigma_3 \odot \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$.

Therefore, $\sigma_3 = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$, and $\sigma_3^{-1} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$. Table 4 shows the mapping of vectors

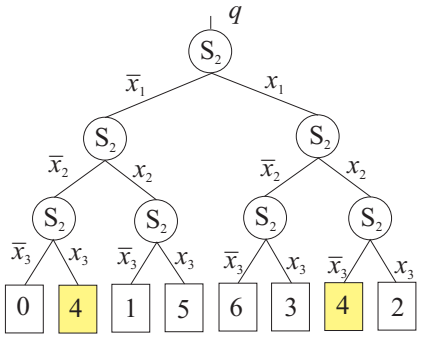

Fig. 11. MTBDT for $f_{\sigma_4^{-1}}$ with encoded pair of function values.

of variables in $\mathbf{Q}_3$ by using σ_3^{-1}. For f in Table 2, and $\sigma = \sigma_3$, we have

$\mathbf{Q}_\sigma = [0, 2, 4, 4, 3, 1, 6, 5]^T$. Fig. 12 shows the corresponding MTBDD($\mathbf{Q}_\sigma$). Therefore, $\mathbf{F}_\sigma = [f(0), f(15), f(1), f(14), f(12), f(3), f(13), f(2), f(7), f(8), f(6), f(9), f(11), f(4), f(10), f(5)]^T$, $= [0, 0, 2, 2, 0, 3, 0, 3, 3, 3, 1, 1, 2, 1, 1, 2]^T$. Fig. 13 shows the corresponding final MTBDD(f_σ).

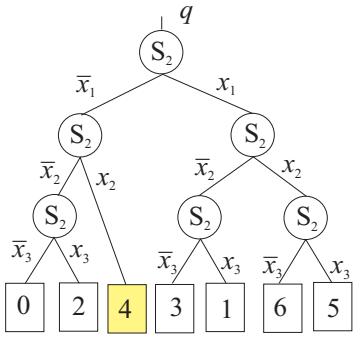

Fig. 12. MTBDD($\mathbf{Q}_\sigma$.)

Note that the recursive application of σ_4^{-1} and σ_3^{-1} to f is identical to the application of a composite mapping $\sigma_{4,3}^{-1} = \sigma_4^{-1} \odot \sigma_{3,1}^{-1}$, where $\sigma_{3,1} = \begin{bmatrix} \sigma_3^{-1} & \mathbf{0} \\ \mathbf{0} & 1 \end{bmatrix}$, where $\mathbf{0}$ is (3×1) zero matrix. Therefore, $\sigma_{4,3}^{-1} = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix} \odot \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}$. Table 5 shows the mapping of vectors of variables in f by using $\sigma_{4,3}^{-1}$. It produces the same permutation of values in $\mathbf{F}$ as a recursive application of σ_4^{-1}, and σ_3^{-1} to f, respectively. In this example, the

TABLE II

Function f and f_σ.

	Function		Characteristic functions				Autocorrelation functions				
x, w	f_0, f_1	$f(x)$	$f_0(z)$	$f_1(z)$	$f_2(z)$	$f_3(z)$	$B_0(z)$	$B_1(z)$	$B_2(z)$	$B_3(z)$	$B(z)$
0	00	0	1	0	0	0	4	4	4	4	16
1	10	2	0	0	1	0	2	0	0	2	4
2	11	3	0	0	0	1	2	2	0	0	4
3	11	3	0	0	0	1	2	2	0	0	4
4	01	1	0	1	0	0	0	0	2	2	4
5	10	2	0	0	1	0	0	0	2	2	4
6	01	1	0	1	0	0	0	0	0	0	0
7	11	3	0	0	0	1	0	0	0	0	0
8	11	3	0	0	0	1	0	0	0	0	0
9	01	1	0	1	0	0	0	0	0	0	0
10	01	1	0	1	0	0	0	0	2	2	4
11	10	2	0	0	1	0	0	0	2	2	4
12	00	0	1	0	0	0	2	2	0	0	4
13	00	0	1	0	0	0	2	2	0	0	4
14	10	2	0	0	1	0	0	2	2	2	4
15	00	0	1	0	0	0	2	2	2	2	8

size of the MTBDD(f) was reduced from 13 to 9 non-terminal nodes by using the proposed method.

A. Advantages and limitations of the proposed method

Remark 5: The K-procedure performs the decomposition of f with respect to the expansion rule $f = (x_i \oplus \cdots \oplus x_n) f_0 \oplus (\overline{x_i \oplus \cdots \oplus x_n}) f_1$, where f_0 and f_1 are co-factors of f for $x_i \oplus \cdots \oplus x_n = 0$, and 1, respectively.

The following example illustrates dependency of the solutions on the choice of permutation matrices σ and vectors τ where the total autocorrelation functions take the maximum values,

TABLE III

Mapping of function values by σ_4^{-1}.

x_0	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}
0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	1	0	0	1	1	0	1	0	0	1	1	0	0	1
0	1	1	0	1	0	0	1	0	1	1	0	1	0	0	1
0	1	0	1	1	0	1	0	1	0	1	0	0	1	0	1
0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0
0	15	12	3	6	9	10	5	11	4	7	8	13	2	1	14

TABLE IV

Mapping of function values by σ_3^{-1}.

x_0	x_1	x_2	x_3	x_4	x_5	x_6	x_7
0	0	0	0	1	1	1	1
0	0	1	1	0	0	1	1
0	1	0	1	0	1	0	1
0	1	2	3	4	5	6	7
0	1	0	1	1	0	1	0
0	1	0	1	0	1	0	1
0	1	1	0	1	0	0	1
0	7	1	6	5	2	4	3

TABLE V

Mapping of function values by $\sigma_4^{-1} \odot \sigma_3^{-1}$.

x_0	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}	x_{14}	x_{15}
0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0
0	1	0	1	1	0	1	0	1	0	1	0	0	1	0	1
0	1	0	1	0	1	0	0	1	0	1	0	1	0	1	0
0	1	1	0	0	1	1	0	1	0	0	1	1	0	0	1
0	15	1	14	12	3	13	2	7	8	6	9	11	4	10	5

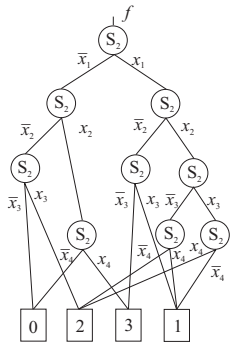

Fig. 13. MTBDD(f_σ).

in the cases when there are several maxima in the total autocorrelation function for a given function f.

Example 5: (Dependency on τ)

Consider a Boolean function f given by the truth-vector $\mathbf{F} = [0,0,1,0,1,0,0,0,1,0,0,1,0,1,0,0]^T$. For this function $size(MTBDD(f)) = 9$.

The maximum value of the autocorrelation function $B_f(\tau) = 14$ for the inputs $\tau = 6$,

9, and 15. For $\tau = 15$ and $\sigma_4(\tau = 15) = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$, we determine $\sigma_4^{-1}(\tau = 15) = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix}$. The elements of the truth-vector for f are reordered as $\mathbf{F}_\sigma = [0,0,0,0,0,0,0,0,1,1,0,1,1,1,0,0]^T$. We perform encoding of pairs of adjacent values as $\mathbf{Q}_\sigma = [0,0,0,0,1,2,1,0]^T$, where (0,0)=0, (1,1)=1, and (0,1)=2. For this function, the maximum value of the total autocorrelation function $max_\tau B_{\mathbf{Q}(\tau)} = 6$ for the input $2 = (010)$. For $\sigma_3(\tau = 2) = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$, it follows $\sigma_3^{-1}(\tau = 2) = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$, and the corresponding reordering is $\mathbf{Q}_\sigma = [0,0,0,0,1,1,2,0]^T$, from where $\mathbf{F}_\sigma = [0,0,0,0,0,0,0,0,1,1,1,1,0,1,0,0]^T$. For the resulting f_σ, $size(MTBDD(f_\sigma)) = 4$.

If for the maximum value of the autocorrelation function $B_f(\tau)$, we choose the input $\tau = 6$, then for $\sigma_4(\tau = 6) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$, we determine $\sigma_4^{-1}(\tau = 6) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$. Thus, we reorder the elements of $\mathbf{F}$ for the given f as $\mathbf{F}_\sigma = [0,0,0,0,1,1,0,0,0,0,1,0,1,1,0,0]^T$.

For encoding $\mathbf{Q}_\sigma = [0,0,1,0,0,2,1,0]^T$, where $(0,0)) = 0$, $(1,1) = 1$, and $(1,0) = 2$, the maximum values of the total autocorrelation function of $\mathbf{Q}_\sigma$ is 6 for the input $4 = (100)$.

For $\sigma_3(\tau = 4) = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$, we determine $\sigma_3^{-1}(\tau = 4) = \sigma_3(\tau = 4)$. Therefore, the corresponding reordering is $\mathbf{Q}_\sigma = [0,0,1,1,0,2,0,0]^T$, which produces $\mathbf{F}_\sigma = [0,0,0,0,1,1,1,1,0,0,1,0,0,0,0,0]^T$. For the resulting f_σ, we determine $size(MTBDD(f_\sigma)) = 5$.

Example 6: (Dependency on σ)

For f in the previous example, if we chose for the maximum value of $B_f(\tau)$ the input $\tau = 15$ and the matrix $\sigma_{4,\tau}(\tau = 15) = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$, which requires $\sigma^{-1}_{4,\tau}(\tau = 15) = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$, then we determine the reordering $\mathbf{F}_\sigma = [0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0]^T$. For the encoding $\sigma(\mathbf{Q}) = [0,0,0,1,1,2,0,0]^T$, the maximum value of the total autocorrelation function is $max_\tau B_{\mathbf{Q}_\sigma}(\tau) = 6$ for the input $\tau = 7 = (111)$. For $\sigma_3(\tau = 7) = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$, we get $\sigma_3^{-1}(\tau = 7) = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$, which induces a reordering $\mathbf{Q}_\sigma = [0,0,0,0,2,0,1,1]^T$. From there, $\mathbf{F}_\sigma = [0,0,0,0,0,0,0,0,1,0,1,1]^T$. For the resulting f_σ, $size(MTBDD(f_\sigma)) = 5$.

However, if we chose $\sigma_{3,\tau}(\tau = 7) = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix}$, and the corresponding $\sigma^{-1}_{3,\tau}(\tau = 7) = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}$, we get the reordering $\mathbf{Q}_\sigma = [0,0,1,1,0,0,2,0]^T$, which produces the vector $\mathbf{F}_\sigma = [0,0,0,0,1,1,1,1,0,0,0,0,1,0,0,0]^T$. For the resulting f_σ, $size(MTBDD(f_\sigma)) = 5$.

The reason for the increased size is that $\sigma_{3,\tau}(\tau = 7)$, unlike $\sigma_3(\tau = 7)$, did not paired together sequences of four 0. This pairing in MTBDD(f_σ) means assignment of identical subvectors of length four to the same logic value for x_1. In this case that is the negative literal $\overline{x}_i$. Thanks to that, the subtree rooted in the node pointed by $\overline{x}_1$in the MTBDD is reduced to a single constant node.

The proposed method provides for reduction of a number of subtrees consisting of a nonterminal node and two constant nodes, since it produces pairs of equal function values. The larger subtrees, which correspond to the equal subvectors of orders 2^k, $k > 1$ are not taken into account at this step. The method fails in the case when we chose a permutation matrix which does not provide a grouping of isomorphic smallest subtrees into a larger subtree. Example 7

illustrates that feature of the method.

Example 7: Consider a function $f = \overline{x}_2\overline{x}_4 + x_2\overline{x}_3x_4 + x_1x_2x_4 + \overline{x}_1x_3\overline{x}_4$. The truth-vector for this function is given by $\mathbf{F} = [1,0,1,0,0,1,1,0,1,0,1,0,0,1,0,1]^T$. $size(MTBDD(f)) = 6$ for this truth vector.

The maximum value of $B_f(\tau)$ is equal to 12, which means that we may generate six pairs of equal values for f at the level for x_4. Then, the method proposed in this paper produces MTBDDs with the size equal to 7. However, the ordering of variables approach results in the MTBDDs of sizes 5, 6, and 7 [24].

However, if we first perform encoding $\mathbf{Q} = [2,2,3,2,2,2,3,3]^T$, where $(1,0) = 2$, and $(0,1) = 3$, and then apply the proposed method, we get a MTBDD of size 5, by always taking the smallest value for τ. This follows from the property that in $\mathbf{Q}$, we have five pairs denoted by 2 and three pairs denoted by 3, which permits an immediate reduction of subtrees consisting of three non-terminal nodes.

We note that the method proposed in the paper is based on an extended set of allowed permutation matrices for the inputs, compared to the one used in DD optimization by ordering of variables. The price for such extension is minor, since the values for f can be easily determined from f_σ assigned to f. Therefore, the proposed method permits to derive efficient solutions which can not be achieved by the ordering of variables. In this respect, the proposed method relates to the considerations in [2] and [7]. In [2] , the same approach to BDDs minimization by using an extended set of permutation matrices was proposed starting from cube representations of functions and performing transformations of cubes. However, no algorithm or heuristic for determination of a transformation for cubes has been proposed. Instead, for each given function f, a particular transformation is determined by the inspection of the characteristics of f. In [7], the method in [2] was extended into the method of truth table permutation, and further elaborated by proposing two heuristic algorithms for determination of a suitable permutation of the function values for f permitting reduction of the size of the BDD for f.

To conclude this section we note that the method proposed in this paper results in BDDs which are not larger and in most cases smaller than BDDs produced by methods based on ordering of variables.

Example 8: Consider a function $f = x_1x_2x_3 + x_2x_3x_4$. The truth-vector for this function is $\mathbf{F} = [0,0,0,0,0,0,0,1,0,0,0,0,0,0,1,1]^T$. The optimization by ordering of variables produces a MTBDD of size 5. However, the method proposed in this paper, produces a MTBDD of size 4 in the following way. The maximum value for $B_f(\tau) = 14$ for the inputs $\tau = 1, 8, 9$. For simplicity, we choose $\tau = 1$, which implies $\sigma_4(\tau = 1)$ is the identity matrix of order 4, and perform the encoding as $\mathbf{Q} = [0,0,0,2,0,0,0,1]^T$. The maximum of the total autocorrelation function for $\mathbf{Q}$ is 6 and it is achieved for $\tau = 4 = (100)$. For a matrix $\sigma_3(\tau = 4) = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$, which is self-inverse over $GF(2)$, we get the reordering $\mathbf{Q}_\sigma = [0,0,0,0,0,0,2,1]^T$, which produces $\mathbf{F}_\sigma = [0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,1]^T$. For thus determined f_σ, $size(MTBDD(f_\sigma)) = 4$.

B. Modified K-procedure

The approach for design of optimized LT-BDDs described in the previous section was developed for the case when $L_{n-1} \geq L_{n-2} \geq \ldots \geq L_1$, where L_i is the number of nodes at the level corresponding to x_{i+1} in the BDD corresponding to the original ordering of variables. Since these inequalities are not always satisfied the following simple modification of the proposed procedure may be very efficient.

Procedure 3: (*Modified K-procedure*)

1. Compute $max_{\tau \neq 0} B_{f_n}(\tau) = B_n$, where $f_n = f$, and n is the number of variables in f_n.
2. Compress f_n into f_{n-q} by encoding q-tuples of successive function values for f_n, for $q = 1, \ldots, n-1$.
3. Compute $max_i B_{f_i} = B_{f_t}$. 4. Apply K-procedure to f_t.

This modification of the original procedure may increase the amount of computation by the factor of 2 at most.

C. Analytical computation of autocorrelation functions

We note that total autocorrelation functions and optimal LT-BDDs in many cases may be computed analytically. To illustrate this point we consider a device implementing an error-correcting procedure based on a linear code V of length n with k-information bits [13].

A code V correct errors from set $E \subset Z_2^n$ iff $v_1 \oplus e_1 \neq v_2 \oplus e_2$ for any $v_1, v_2 \in V$ and

$e_1, e_2 \in E$. If V correct l errors, then E contains all $\sum_{i=0}^{l} \begin{pmatrix} n \\ i \end{pmatrix}$ vectors e with $||e|| \neq l$ and some errors e with $||e|| \geq l+1$, where $||e||$ is the Hamming weight of e.

A code V is not extendable for a given E iff for any $x \in Z_2^n$ there exists a unique $v \in V$ and a unique $e \in E$ such that $x = v \oplus e$, $|E| = 2^{n-k}$.

A device implementing an error-correcting procedure based on a given V has n inputs, n outputs, and for input x produces output $e = f(x)$ such that there exist $v \in V$ and $x = v \oplus e$. (Since V corrects set of errors E, this e is unique.) For this device, we have for the total aucorrelation function [13] $B_f(\tau) = \begin{cases} 2^n, & \tau \in V, \\ 0, & \tau \notin V. \end{cases}$

If V is an (n,k) code, the K-procedure will require k steps. For the resulting optimal linear transform σ, the rows of $(h_1, \ldots, h_{n-k})$ of σ form a basis in the null-space for V. Thus, $\sigma = \begin{bmatrix} h_1 & \\ \vdots & \\ h_{n-k} & \\ \mathbf{O} & \mathbf{I}_k \end{bmatrix}$, where $\mathbf{I}_k$ is the $(k \times k)$ identity matrix, and the corresponding LT-BDD will have $2^{(n-k)} - 1$ nodes.

Example 9: Consider a $(5,2)$ code with the generating matrix [13] $\mathbf{G} = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 \end{bmatrix}$. It is easy to check that this code can correct all single errors and two double errors 00011, and 10001. In this case, $B_f(\tau) = \begin{cases} 32, & \text{if } \tau = 00000, 10110, 01101, 11011 \\ 0, & \text{otherwise,} \end{cases}$, and $\sigma^{-1} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$. Therefore, $z_1 = x_1 \oplus x_2 \oplus x_3$, $z_2 = x_1 \oplus x_4$, $z_3 = x_2 \oplus x_5$. Fig. 14 shows the resulting optimal BDD.

VI. Experimental Results

We performed experiments comparing BDDs and LT-BDDs for benchmark functions used in logic design and for randomly generated multiple-output switching functions.

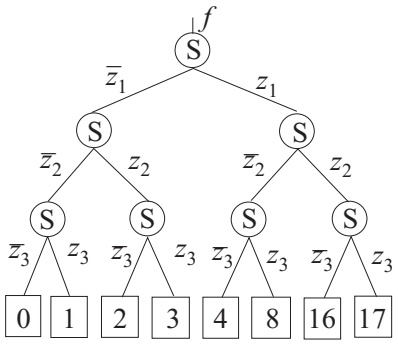

Fig. 14. LT-BDD for the decoder for the $(5, 2)$ shortened Hamming code.

We used a program for calculation of autocorrelation functions and performing the Procedure for linearization of Boolean functions working with vector representations of Boolean functions. For this reason, the experiments are restricted to functions of a small number of variables. However, if the calculation are performed over BDDs by using the DD-methods for computation of Walsh transform, and the Wiener-Khinchin theorem, then the proposed approach can be applied for functions which can be processed by other DD-methods, since the main complexity of the proposed method relates to the calculation of the total autocorrelation functions.

In Table VI, we present numbers of inputs (In), outputs (Out), constant nodes (cn), and in columns denoted by MTBDD(f) and MTBDD(f_σ), we compared numbers of non-terminal nodes (ntn), whose sum with the numbers of constant nodes produces the size (s), and the width (w) of the MTBDDs for the initial ordering of variables and LT-MTBDDs derived by the autocorrelation functions. The column MTBDD(f_v) shows the number of non-terminal nodes and widths of MTBDDs for the optimal orderings of variables determined by the brute force method based on comparing all possible orderings. The existing methods for optimization of DDs by variables ordering are heuristics and mostly produce the nearly optimal solutions. Therefore, the provided comparison is the strongest challenge for the proposed method. This table presents the results for the method based on autocorrelation functions for the smallest values for τ where the autoorrelation functions take the maximum values. The other choices for τ and σ, may produce smaller LT-MTBDDs.

In Table VII we compare the number of non-terminal nodes of MTBDDs for the initial

ordering of variables (MTBDD(f)), the optimal ordering (MTBDD(f_v)), the initial ordering with negated edges (MTBDD(f_w)), the ordering determined by lover-bound sifting method with negated edges (MTBDD(f_r), and by the autocorrelation functions (MTBDD(f_σ)) for binary-valued single output randomly generated functions. It should be noted that in the used package for sifting, the nodes with negated edges are used, while for other methods this optimization is not performed. This is the reason that for some functions, the numbers of nodes and the width for DDs produced by lower-bound sifting are smaller than the number of nodes for optimal ordering of variables without using the negated edges. It is interesting to note that in some cases, as for example f_1, f_7, n_6, n_7, the method based on autocorrelation functions although without negated edges produces the results close to the sifting with negated edges.

From these experiments, the following conclusions can be made.

1. The proposed method is very effective when the integer equivalent function $f(x)$ defined by a given multiple-output function (see [13]) takes a large number of different values, which however, do not repeat periodically as sequences of order 2^k, $k = 1, \ldots, n-1$. This is the case, for example, of n-bit adders. It follows from Table VI that for adders transition from BDDs to LT-BDDs results in almost 50% reduction of the sizes of the corresponding decision diagrams.

   For adders, the method produces the LT-MTBDDs with minimal width, however, but the sizes of LT-MTBDDs are larger than in the case of optimal ordering of variables. It should be noted that reordering of variables does not reduce the widths of MTBDDs for adders. The self-inverse matrix σ describing the optimal ordering of variables for adders can be derived from the values τ_i corresponding to the maximum values of total autocorrelation functions by writing each 1-bit of τ_i in the separate row of σ starting from the largest τ_i. For example, for the 2-bit adder, the total autocorrelation function B_f takes the maximum value for $\tau_1 = 5 = (0101)$ and $\tau_2 = 10 = (1010)$. Therefore, $\sigma = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$.

2. The method is less efficient when the equal values of the original function f repeat themselves as sequences of order 2^k. In the corresponding MTBDDs, these sequences

result in isomorphic subtrees, which permits reduction of numbers of nodes at the upper levels in the MTBDDs. In these cases, pairing function values at the Hamming distance 1 by the total autocorrelation function may destroy the equal sequences of order 2^k, where $k > 1$, which results in larger MTBDDs.

As it is discussed in [7], that feature is characteristic for methods using permutations of function values for multiple-output functions [2], [7]. Since any reordering of variables is a permutation of function values, the same remark applies to optimization of DDs by reordering of variables.

The functions con1 and ex1010 in Table 6, are examples where the proposed method reduces both the sizes and the widths of the MTBDD. For the function misex1 the method provides reduced width and the same size. The function clip is an example where the proposed method increased the width but not the size. For the function t481, the method cannot reduce neither the size and the width of the MTBDD. The explanation is that in the function t481 sequence 1101 repeats itself many times, which permits reduction of a number of nodes at upper levels resulting in a MTBDD of a smaller size.

The method is inefficient for multiple-output functions whose integer valued equivalent functions contain few equal values. In these cases, we can not produce large numbers of pairs of equal values resulting in a reduction of the number of nodes in MTBDDs. The examples are multipliers and the benchmark function bw.

3. The method is efficient for randomly generated multiple-output functions. It should be noted that in this case the initial MTBDDs are usually large and the ordering of variables does not provide for reduction of their sizes. Savings in the number of non-terminal nodes of MTBDDs for randomly generated functions (see Table 7) range from 1.4% for f_3 to 45.06% for n_3. Savings in the width of MTBDDs range from 1.06% for f_9 to 17.86% for f_7 and 20% for f_2, and 51.62% for n_3. For f_6, the method produced the larger MTBDD, since in this case the random numbers generator produced equal sequences. In some cases, as for example for n_6, the method produced the smaller LT-MTBDsD than for the original ordering, however, larger than MTBDDs for the optimal orderings.
4. An important feature of the proposed method is that, unlike sifting, it can be applied to the reduction of MTBDDs of symmetric functions, where the permutation of variables

does not permit reduction of nodes. Symmetry implies equal sequences of order 2^k for some large k in the function values.

First, we perform encoding of such sequences, and after this we apply the method to the function g of 2^{n-k} variables derived in this way. Then we determine g_σ for this function, and after the decoding we get f_σ for the initial function f. Table VI in rows 20 to 24 illustrates the method and compares MTBDD(f) and MTBDD(f_σ) for some symmetric benchmark functions. For these benchmarks, we first perform encoding of sequences of four successive input vectors, and then use the proposed method, and perform re-coding before we determine the size of the MTBDD. For rd84/8, and 9sym/16, we performed encoding of the sequences of 8 and 16 successive input vectors, respectively.

VII. Closing Remarks

Two important problems in optimization of DD representations, ordering of variables, and determination of an optimal linear transformation of variables, can be related, expressed, and solved using total autocorrelation functions. The computational complexity of this approach does not exceed $min(O(n2^{n+k}, O(2^n))$, where n is the number of variables and k is the number of functions. The approach permits uniform consideration of single and multiple-output functions.

We show that the method for linearization of Boolean functions provides for a simple and efficient algorithm for determination of a nearly optimal linear transformation of variables. Then, we show that the total autocorrelation function may be used to determine the order of function values, which recursively determine and minimize the width of each level in the related MTBDDs. Since the proposed algorithm minimizes a number of nodes at each level, it results in a reduction of the complexity of MTBDDs. The proposed algorithms for both ordering of variables and constructing quasioptimal linear transform of variables have simple software implementations. The algorithms are deterministic in the sense that there are no heuristic involved at any step of the algorithms. Experiments with benchmarks and with randomly generated functions illustrate that the proposed method is on average very efficient.

The proposed method performs a larger class of transformations over variables compared to dynamic reordering and related methods where just the reordering of variables is used. The

TABLE VI

MTBDD(f), MTBDD(f_v), and MBTDD(f_σ) for benchmark and randomly generated functions.

				MTBDD(f)		MTBDD(f_v)		MTBDD(f_σ)	
f	In	Out	cn	ntn	w	ntn	w	ntn	w
add2	4	3	7	13	6	12	6	8	3
add3	6	4	15	51	20	33	14	24	7
add4	8	5	31	113	30	78	30	64	15
add5	10	6	63	289	62	171	62	160	31
add6	12	7	127	705	126	360	126	384	63
add7	14	8	255	1665	254	741	254	896	127
ex1010	10	10	177	894	383	-	-	871	367
misex1	8		11	17	6	17	6	17	5
clip	9	5	33	339	120	141	35	159	32
t481	16	1	2	32	4	-	-	103	46
rd53	5	3	6	15	5	15	5	14	5
rd73	7	3	8	28	7	28	7	17	6
rd84	8	4	9	36	8	36	8	23	7
rd84/8	8	4	9	36	8	36	8	18	6
9sym	9	1	2	33	6	33	6	24	5
9sym/16	9	1	2	33	6	33	6	9	3
Randomly generated functions									
f_2	8	1	2	75	30	68	25	66	24
f_3	8	1	2	73	28	67	23	72	27
f_6	8	1	2	58	18	58	18	69	25
f_7	8	1	2	72	28	70	26	67	23
f_8	8	3	8	174	64	167	60	168	59
f_9	8	4	16	222	95	216	90	219	94
f_{10}	8	3	5	139	56	135	54	136	54
n_1	8	2	3	84	30	77	25	80	26
n_2	8	2	3	89	29	80	26	87	27
n_3	8	2	3	91	31	85	27	50	15
n_4	8	2	3	90	31	83	24	89	28
n_5	8	2	3	82	27	76	22	77	24
n_6	8	2	2	68	25	59	18	62	19
n_7	8	2	2	72	28	64	21	63	20
n_8	8	2	2	72	29	64	22	72	28
n_9	8	2	2	73	28	69	25	68	25
n_{10}	8	2	2	118	41	111	36	115	42

TABLE VII

Non-terminal nodes in MTBDDs for initial ordering of variables, optimal ordering, initial ordering with negated edges, lower-bound sifting with negated edges, and autocorrelation functions.

f	MTBDD(f)	MTBDD(f_v)	MTBDD(f_w)	MTBDD(f_r)	MTBDD(f_σ)
f_1	75	68	64	62	66
f_2	73	67	67	60	72
f_6	58	58	52	51	69
f_7	72	70	64	62	67
n_1	84	77	64	62	80
n_6	68	59	63	55	62
n_7	72	64	65	60	63
n_8	72	64	65	58	72
n_9	73	69	64	62	68

method can be used to improve the results derived as the output of the dynamic reordering.

Acknowledgment

The authors are grateful to Dr. Dragan Janković from Faculty of Electronics, Niš, Yugoslavia, and Dr. Ari Trachtenberg from Boston University, Boston, USA, for the help with programming and performing experiments reported in this paper.

The authors thank the Referees whose valuable comments improved the presentation in this paper.

References

[1] S. Agaian, J. Astola, K. Egiazarian, *Binary Polynomial Transforms and Nonlinear Digital Filters*, Marcel Dekker, 1995.

[2] J. Bern, C. Meinel, A. Slobodova, "Efficient OBDD-based manipulation in CAD beyond current limits", *32nd Design Automation Conference*, 1995, 408-413.

[3] B. Bollig, I. Wegener, "Improving the variable ordering of OBDDs is NP-complete", *IEEE Trans. Comput.*, Vol. C-45, No. 9, 1996, 993-1002.

[4] R.E. Bryant, "Graph-based algorithms for Boolean functions manipulation", *IEEE Trans. Comput.*, Vol.C-35, No.8, 1986, 667-691.

[5] E.M. Clarke, K.L. Mc Millan, X. Zhao, M. Fujita, "Spectral transforms for extremely large Boolean functions", in Kebschull, U., Schubert, E., Rosenstiel, W., Eds., *Proc. IFIP WG 10.5 Workshop on Applications of the Reed-Muller Expression in Circuit Design*, Hamburg, Germany, September 16-17, 1993, 86-90. Workshop Reed-Muller'93, 86-90.

[6] R. Drechsler, B. Becker, *Binary Decision Diagrams, Theory and Impementation*, Kluwer Academic Publishers, 1998.

[7] M. Fujita, Y. Kukimoto, R.K. Brayton, "BDD minimization by truth table permutation", *Proc. Int. Szmp. on Circuits and Systems, ISCAS'96*, May 12-15, 1996, Vol. 4, 596-599.

[8] M. Fujita, Y. Matsunaga, T. Kukuda, "On variable ordering of binary decision diagrams for the optimization in multi-level synthesis", *European Conf. on Design Automation*, 1991, 50-54.

[9] M. Fujita, J. C.-H. Yang, E.M. Clarke, X. Zhao, M.C. Geer, "Fast spectrum computation for logic functions usign binary decision diagrams", *ISCAS-94*, 1994, 275-278.

[10] W. Günther, R. Drechsler, "BDD minimization by linear transforms", in *Advanced Computer Systems*, 1998, 525-532.

[11] W. Günther, R. Drechsler, "Efficient manipulation algorithms for linearly transformed BDDs", *Proc. 4th Int. Workshop on Applications of Reed-Muller Expansion in Circuit Design*, Victoria, Canada, May 20-21, 1999, 225-232.

[12] W. Günther, R. Drechsler, "Minimization of BDDs using linear transformations based on evolutionary techniques", *Proc. Int. Symp. Circuit and Systems*, 1999.

[13] M.G. Karpovsky, *Finite Orthogonal Series in the Design of Digital Devices*, John Wiley, 1976.

[14] M.G. Karpovsky, E.S. Moskalev, "Utilization of autocorrelation characteristics for the realization of systems of logical functions", *Avtomatika i Telmekhanika*, No. 2, 1970, 83-90, English translation *Automatic and Remote Control*, Vol. 31, 1970, 342-350.

[15] M.G. Karpovsky, (ed.), *Spectral Techniques and Fault Detection*, Academic Press, 1985, 35-90.

[16] M.G. Karpovsky, R.S. Stanković, J.T. Astola, "Spectral techniques for design and testing of computer hardware", *Proc. Int. Worksop on Spectral Techniques in Logic Design, SPECLOG-2000*, Tampere, Finland, June 2-3, 2000, 1-34.

[17] R.J. Lechner, A. Moezzi, "Synthesis of encoded PLAs", in [15].

[18] C. Meinel, F. Somenzi, T. Theobald, "Linear sifting of decision diagrams", *Proc. Design Automation Conference*, 1997, 202-207.

[19] Ch. Meinel, F. Somenzi, T. Tehobald, "Linear sifting of decision diagrams and its application in synthesis", *IEEE Trans. CAD*, Vol. 19, No. 5, 2000, 521-533.

[20] D. Milošević, R.S. Stanković, C. Moraga, "Calculation of dyadic autocorrelation throguh decision diagrams", *Proc. Int. Workshop on Computational Intelligence and Information Technologies*, Niš, Serbia, June 20-21, 2001, 129-134.

[21] S. Minato, "Graph-based representations of discrete functions", in [27], 1-28.

[22] S. Panda, F. Somenzi, "Who are the variables in your neigborhood", *Proc. IEEE Int. Conf. Computer-Aided Design*, San Joseé, CA, 1995, 74-77.

[23] S. Panda, F. Somenzi, B.F. Pleisser, "Symmetry detection and dynamic variable ordering of decision diagrams", *Proc. IEEE Int. Conf. Computer-Aided Design*, 1994, 628-631.

[24] J. Rice, M. Serra, J.C. Muzio, "The use of autocorrelation coefficients for variable ordering for ROBDDs",

Proc. 4th Int. Workshop on Applications of Reed-Muller Expansion in Circuit Design, Victoria, Canada, August 20-21, 1999, 185-196.

[25] R. Rudell, "Dynamic variable ordering for ordered binary decision diagrams", *Proc. IEEE Conf. Computer Aided Design*, Santa Clara, CA, 1993, 42-47.

[26] T. Sasao, *Switching Theory for Logic Synthesis*, Kluwer Academic Publishers, 1999.

[27] T. Sasao, M. Fujita, (eds.), *Representations of Discrete Functions*, Kluwer, 1996.

[28] M. Sauerhoff, I. Wegener, R. Werchner, "Optimal ordered binary decision diagrams for read-once formulas", *Discrete Applied Mathematics*, Vol. 103, 2000, 237-258.

[29] D. Sieling, "On the existence of polynomial time approximation schemes for OBDD minimization", *Proc. STACS'98*, Lecture Notes in Computer Sci., Vol. 1373, Springer, Berlin, 1998, 205-215.

[30] F. Somenzi, *CUDD - Colorado University Decision Diagram Package*, 1996.

[31] R.S. Stanković, *Spectral Transform Decision Diagrams in Simple Questions and Simple Answers*, Nauka, Belgrade, 1998.

[32] R.S. Stanković, "Some remarks on basic characteristics of decision diagrams", *Proc. 4th Int. Workshop on Applications of Reed-Muller Expansion in Circuit Design*, August 20 - 21, 1999, 139-146.

[33] R.S. Stanković, M. Bhattacharaya, J.T. Astola, "Calulation of dyadic autocorrelation through decision daigrams", *Proc. European Conference on Circuit Theory and Design, ECCTD'01*, Espoo, Finland, August 28-31, 2001, II-337 - II-340.

[34] R.S. Stanković, T. Sasao, "Decision diagrams for representation of discrete functions: uniform interpretation and classification", *Proc. ASP-DAC'98*, Yokohama, Japan, February 13-17, 1998.

[35] R.S. Stanković, T. Sasao, C. Moraga, "Spectral transform decision diagrams", in: [27], 55-92.

[36] E.A. Trachtenberg, "SVD of Frobenius matrices for approximate and multiobjective signal processing tasks", in E.F. Derettere, Ed., *SVD and Signal Processing*, Elsevier, North-Holland, Amsterdam, New York, 1988, 331-345.

[37] I. Wegener, "Worst case examples for operations over OBDDs", *Information Processing Letters*, 74, 2000, 91-94.