Control and Cybernetics

vol. **33** (2004) No. 4

Efficient spectral method of identification of linear Boolean function

by

Piotr Porwik

Institute of Informatics, Silesian University
Będzińska 39, 41–200 Sosnowiec, Poland
e-mail: porwik@.us.edu.pl

Abstract: This paper discusses a problem of recognition of the Boolean function's linearity. The article describes the spectral method of analysis of incompletely specified Boolean functions using the Walsh Transform. The linearity and nonlinearity play an important role in design of digital circuits. The analysis of the spectral coefficients' distribution allows to determine the various combinatorial properties of the Boolean functions: redundancy, monotonicity, self-duality, correcting capability, etc. which seems to be more difficult to obtain by means of other methods. In particular, the distribution of spectral coefficients allows us to determine whether Boolean function is linear. The method described in the paper can be easily used in investigations of large Boolean functions (of many variables), what seems to be very attractive for modern digital technologies. Experimental results demonstrate the efficiency of the approach.

Keywords: Walsh coefficients, coefficients distribution, incompletely defined Boolean functions, linear Boolean function.

1. Introduction

Transformations between the Boolean and various spectral domains have been extensively studied by a number authors (Ahmed, Rao, 1975; Harmuth, 1977, Hurst et al., 1985; Karpovsky, 1976; Porwik, Falkowski, 1999). These studies have been carried out because some problems of digital logic may be solved more efficiently in the spectral domain than in the Boolean domain. Theoretically, techniques based on the Walsh transform provide some nice applications such as Boolean functions' classification, disjoint decomposition, multiplexer and threshold logic synthesis, state assignment, testing and evaluation of logic complexity (Hurst et al., 1985; Falkowski et al., 1992; Porwik, Falkowski, 1999). In some cases spectral methods can be effectively applied to solve mathematical and practical problems (Blahut, 1983; Harmuth, 1977). One of these problems is finding of linearity of Boolean functions by means of the Walsh-Hadamard spectral technique. The classical approaches to computation of the Walsh-Hadamard spectrum is based on the truth tables. The most effective truth table based algorithm is the Fast Walsh-Hadamard Transform (Ahmed, Rao, 1975). Unfortunately, the main disadvantage of truth table based techniques is that they cannot be used for logic functions with large numbers of variables. One way to overcome the above difficulty is to use Decision Diagrams (DDs) to compute the Walsh-Hadamard spectral coefficients (Fujita, Yang, 1995; Stanković, Falkowski, 2002). Nowadays the most widely known are the Binary Decision Diagrams (BDDs) for representation of switching functions.

BDDs are used in design automation for efficient representation of Boolean functions. Such diagrams can successfully describe properties of Boolean functions (Fujita, Yang, 1995; Stanković, Falkowski, 2002; Thornton, Drechsler, 2001; Yang, Ciesielski, 2002). Different BDDs (Wegener, 2000) have proved to be very convenient data structures for majority of discrete function representations permitting manipulations and calculation with large discrete functions efficiently in terms of space an time. Therefore they are frequently used to represent data structures in modern CAD systems. This article solves the problem of spectral analysis for such CAD systems and for systems based on BDDs.

The theoretical and practical backgrounds proposed in this paper can be used in the DD domain. DD structures, although often used, are difficult for hardware realization, unlike spectral algorithms, which are very efficient as circuit applications. Such hardware applications are used for example in cryptography and data encryption (Maitra, Sarkar, 2002; Seberry, Zhang, 1994). The method presented in this paper can be used in hardware applications as well.

The described method allows us to check whether a Boolean function (especially partially defined) is linear. This information can be obtained directly on the basis of the Walsh coefficients. The search for linearity is especially profitable for some Boolean function implementations. Such functions are important in practical designs of, for example, adders, multipliers and parity checkers. Linearity or nonlinearity measure is a very important feature of a Boolean function. Nowadays, some investigations of the linearity (nonlinearity) of functions are carried out in many areas, for instance in cryptography, data encryption, cipher, error control codes, project of the so-called s-boxes, etc. (Maitra, Sarkar, 2002; Mister, Adams, 1996; Seberry, Zhang, 1994).

2. Preliminaries

A Boolean function f of n variables $x_1, x_2, ..., x_n$ is a mapping $\{0,1\}^n \to \{0,1\}$. The variables $x_1, x_2, ..., x_n$ and their complements $\overline{x}_1, \overline{x}_2, ..., \overline{x}_n$ are called literals.

Let V_n be a vector space of n tuples of elements from the Galois field, $GF(2)$. For this space there is a natural one-to-one correspondence between vectors in

V_n and integer numbers in $[0, ..., 2^n - 1]$. It allows for the ordering of the vectors according to their corresponding integer values. If f is a Boolean function from V_n, then f can be expressed as a unique polynomial of n co-ordinates $x_1, x_2, ..., x_n$. For this reason f will be identified as a unique multi-variable polynomial $f(x)$, where $x = (x_1, x_2, ..., x_n)$.

Definition 2.1 *An n variable Boolean function $f(x_1, x_2, ..., x_n)$ can be written as $\sum\limits_{j=0}^{2^n-1} y_j x_1^{b_1} x_2^{b_2} ... x_n^{b_n}$, where $b_1, b_2, ..., b_n \in \{0, 1\}$ and $b_1 b_2 ... b_n$ is an n bit binary number represented by j, $x_i^{b_i=0} = \overline{x}_i$, $x_i^{b_i=1} = x_i$ for $i = 1, 2, ..., n$. Then $\mathbf{Y} = [y_0, y_1, ..., y_{2^n-1}]$, $y_j \in \{0, 1\}$ is the truth vector of f.*

The true (false) set of function f, denoted by T_f (F_f) is the collection of the true (false) points of f, i.e. $T_f = \{x \in \{0, 1\}^n : f(x) = 1\}$ and $F_f = \{x \in \{0, 1\}^n : f(x) = 0\}$. A term (elementary conjunction) is a conjunction of literals of the form:
$\prod\limits_{i \in P} x_i \prod\limits_{i \in N} \overline{x}_i$, where P (positive) and N (negative) are disjoint subset of $\{1, ..., n\}$.

Example 2.1 *The truth vector of the three-variable Boolean function $f(x_1, x_2, x_3) = \overline{x}_1\overline{x}_2\overline{x}_3 + \overline{x}_1\overline{x}_2 x_3 + \overline{x}_1 x_2 x_3 + x_1\overline{x}_2 x_3 + x_1 x_2 x_3$ is $[1, 1, 0, 1, 0, 1, 0, 1]$.*

Definition 2.2 *The Boolean function is balanced if $card(T_f) = card(F_f)$.*

In other words a Boolean function is balanced if it contains an equal number of zeros and ones. In the theory of Boolean functions there exist many types of balanced functions. For instant linear, majority, minority, self-dual functions are balanced.

Definition 2.3 *A partially defined Boolean function f is a function $h : T \cup F \rightarrow \{0, 1\}$ defined as*

$$h(r) = \begin{cases} 1 \text{ if } r \in T \\ 0 \text{ if } r \in F \end{cases}$$

where $T \subseteq \{0, 1\}^n$denotes a set of true vectors and $F \subseteq \{0, 1\}^n$denotes a set of false vectors. An extension of f occurs when $T \subseteq T_f$ and $F \subseteq F_{f.}$

From Definition 2.3 it follows that the number of undefined points of f can be calculated from the formula $d = 2^n - [card(T_f) + card(F_f)]$. If $d = 0$ then Boolean function is fully defined.

Any partially defined Boolean function f which is undefined at k points can be obviously extended to a fully defined form. There are 2^k such forms.

3. The spectral analysis

Spectral data are used for many applications in digital logic design. Some of them allow function classification (Hurst et al., 1985; Porwik, 2002), fault synthesis, signal processing (Karpovsky, 1976; Porwik, Falkowski, 1999; Sasao, 1993) etc. A Boolean function $f(x_1, x_2, ..., x_n)$ can be transformed from the domain $\{0, 1\}$ into the spectral domain by linear transformation $\mathbf{H} \cdot \mathbf{Y} = \mathbf{S}$, where $\mathbf{H}$ is a $2^n \times 2^n$ orthogonal transform matrix, $\mathbf{Y} = [y_0, y_2, ..., y_{2^n-1}]^T$ is the two-valued truth vector of $f(x_1, x_2, ..., x_n)$, and $\mathbf{S} = [s_0, s_1, ..., s_{2^n-1}]^T$ is the vector of spectral coefficients. In order to obtain coefficients of $\mathbf{S}$ type, values $\{0, 1\}$ of vector $\mathbf{Y}$ are, respectively, replaced by the $\{1, -1\}$ values. One of the several ways to interpret the meaning of each spectral coefficient is to view it as a measure of correlation between two functions (vectors) (Hurst et al., 1985; Porwik, 2000a, 2002). Hence, the first function f is a Boolean function represented by two-valued truth vector $\mathbf{Y}$ and the second function is one from the collection of constituent functions of the transformation matrix $\mathbf{H}$. The type of information that is obtained from spectral coefficients depends on the transformation matrix. In this paper the well-known Hadamard matrices have been used as transform matrices. It has been observed in Harmuth (1977) that for some N, where $n = \log_2 N$, the Hadamard matrices include the discrete Walsh functions.

Definition 3.1 *The Sylvester-Hadamard (the Walsh-Hadamard) matrix of order 2^n is generated by the following recursive formula:*

$$\mathbf{H}_0 = [1],\ \mathbf{H}_n = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \otimes \mathbf{H}_{n-1}, \quad n = 1, 2, ... \tag{1}$$

where $\otimes$ denotes the Kronecker product.

The square matrix (1) can be alternatively generated on the basis of formula:

$$\mathbf{H}_0 = [1], \mathbf{H}_n = \begin{bmatrix} \mathbf{H}_{n-1} & \mathbf{H}_{n-1} \\ \mathbf{H}_{n-1} & -\mathbf{H}_{n-1} \end{bmatrix}. \tag{2}$$

Additionally: $\mathbf{H}_n = \mathbf{H}_n^T$ and $\mathbf{H}_n \cdot \mathbf{H}_n^T = 2^n \cdot I_n$, where I_n is the identity matrix of order 2^n. Because $\mathbf{H}_n^{-1} = \frac{1}{2^n}\mathbf{H}_n^T$ the matrix $\mathbf{H}_n$ is orthogonal. The spectral coefficients calculated on the basis of matrix (1) are the so-called Walsh's coefficients. This transformation is known as the Walsh-Hadamard Transform (WHT).

Each row of the matrix $\mathbf{H}_n$, created in this way, includes a discrete Walsh sequence $wal(w, t)$ (in other words, a discrete Walsh function). In this notation, $w = 1, ..., 2^n$ identifies the number of the Walsh function, and $t = 1, ..., 2^n$ stands for the discrete point of the function determination interval. The relationship between Walsh's coefficients and variables of Boolean function f can be described as follows.

Definition 3.2 *Any Boolean function $f(x_1, x_2, ..., x_n)$ of n variables can be written by means of the Walsh-Hadamard coefficients as an arithmetical polynomial:*

$$\begin{aligned} f(x_1, x_2, ..., x_n) &= \frac{1}{2^{n+1}}[2^n - s_0 - s_1 \cdot (-1)^{x_n} - s_2 \cdot (-1)^{x_{n-1}} - \\ & \quad s_3 \cdot (-1)^{x_n \oplus x_{n-1}} ... - s_{2^n-1} \cdot (-1)^{x_n \oplus x_{n-1} ... \oplus x_1}] \end{aligned}$$

where: $\oplus$ stands for the modulo-2 addition, and $s_0, s_1, ..., s_{2^n-1} \in \mathbf{S}$ are spectral coefficients.

Each spectral coefficient $s_i \in \mathbf{S}$ is described by its order. The order is equal to the number of variables describing the linear function, which corresponds to row in the matrix $\mathbf{H}_n$ for a given spectral coefficient. The s_i elements of the vector $\mathbf{S}$ are ordered according to a straight binary code of literals describing the minterms of the original truth vector $\mathbf{Y}$:

s_0	$C_0^n = 1$	the zero order coefficient,
s_i	$C_1^n = n$	the first order coefficients, $i = 1, ..., n$,
s_{ij}	C_2^n	the second order coefficients, $ij = 12, 13, 1n, ..., (n-1)n$,
s_{ijk}	C_3^n	the third order coefficients, $ijk = 123, 124, ..., (n-2)(n-1)n$,
...	...	
$s_{12...,n}$	$C_n^n = 1$	the coefficient of order n.

Generally, the number of spectral coefficients of p^{th} order is equal to $C_n^p = \binom{n}{p}$ for $p = 0, 1, ..., n$.

In this notation s_{1234} is a spectral coefficient, which has been calculated for a given Boolean function for input variables $x_1 = x_2 = x_3 = x_4 = 1$. The s_0 coefficient is directly related to the number of minterms for which Boolean function f has the value 1. If the number of minterms which have value of 1 is denoted by a then $s_0 = 2^n - 2a$. The properties of $\mathbf{H}_n$ matrices and specific distribution of spectral coefficients s_i can be easily applied in practice in many investigation areas. For example in information theory, especially in the Reed-Muller coder to form the generating matrices (Fujita, Yang, 1995).

4. Spectral description of linear Boolean function

Definition 4.1 *The Boolean function $f_k(x_1, x_2, ..., x_n)$ of n variables is called affine if it takes the form of the polynomial: $f_k(x) = a_1x_1 \oplus a_2x_2 ... \oplus a_nx_n \oplus c$, where $a_j, c \in GF(2)$ and $k = c + \sum_{i=1}^{n} a_i 2^i$.*

In particular, if $c = 0$ then f is called a linear function.

In the affine Boolean functions each coefficient a_i corresponds to a unique ordering x_i. Hence, the ordering set of all a_i corresponds to a unique ordering of a Boolean function.

Corollary 4.1 *(Porwik, 200b) By definition of the Walsh functions for any affine Boolean function f_k we have:*

for $c = 0$

$$\mathbf{Y}_k = f_k(x) = \frac{1}{2}(\mathbf{1} - wal(w,t)),$$

for $c = 1$

$$\mathbf{Y}_k = f_k(x) = \frac{1}{2}(\mathbf{1} - ((-1) \cdot wal(w,t))).$$

Corollary 4.1 implies that any affine Boolean function can be generated immediately from Hadamard matrices (Porwik, 2000a, b):

$$\begin{array}{lll} \text{for} \quad c = 0 & \text{from} & \mathbf{H}_n, \\ \text{for} \quad c = 1 & \text{from} & \overline{\mathbf{H}}_n = -1 \cdot \mathbf{H}_n. \end{array} \tag{3}$$

The V_n space generates 2^{2^n} different Boolean functions. This space includes 2^{n+1} of affine functions (Porwik, 2000b). By means of the simple Walsh-Hadamard Transform only 2^n linear functions can be found. Theorem 4.1 allows for finding of all the affine Boolean functions in V_n space.

Theorem 4.1 *(Falkowski, Porwik, 1999) Any affine Boolean function f_k is characterized by the unique Walsh-Hadamard spectrum distribution:*

$$s_x = \begin{cases} +2^n & \textit{for} \quad x = k/2 \quad \Leftrightarrow c = 0 \\ -2^n & \textit{for} \quad x = (k-1)/2 \quad \Leftrightarrow c = 1 \\ 0 & \textit{otherwise} \end{cases} \tag{4}$$

where: k and c have the same meaning as in Definition 4.1 and $x = 0, 1, ..., 2^n - 1$.

Proof. Directly from the definition of the Walsh functions it is known that these functions form the complete orthogonal system. From mutual orthogonality of the rows of the Hadamard matrix we have:

$$\sum_{t=0}^{2^n-1} wal(i,t) \cdot wal(j,t) = \begin{cases} 2^n & \text{for} \quad i = j \\ 0 & \text{for} \quad i \neq j. \end{cases} \tag{5}$$

For any Walsh function we have (Hurst et al., 1985):

$$\sum_{t=0}^{2^n-1} wal(i,t) = \begin{cases} 2^n & \text{for} \quad i = 0 \\ 0 & \text{for} \quad i \neq 0 \end{cases} \tag{6}$$

Using equations (5), (6) and Corollary 4.1 we obtain formula (4). ■

Hence in the proposed method, the affine Boolean function can be defined by means of the Walsh functions (Corollary 4.1) or by means of spectral coefficients $s_x \in \mathbf{S}$ (Theorem 4.1). Thus, in order to decide whether a Boolean function is linear it is only necessary to calculate its spectrum. If the spectrum contains only one nonzero value, then function is affine and it has the polynomial form (see: Definition 4.1).

PROPERTY 4.1 *The Boolean function of n variables is affine, if and only if coefficient $s_0 = 0$ and value of p^{th} order of spectral coefficient is $\pm 2^n$.*

EXAMPLE 4.1 *Table 1 includes the description of the given Boolean functions $f_{k=15}$ and $f_{k=14}$. It is necessary to check whether these functions are linear. From the analysis of spectral coefficients it follows that the spectrum includes only one nonzero coefficient s_7.*

Table 1. Examples of Boolean functions and their spectra

$x_1x_2x_3$	$x = \sum_{i=1}^{n} x_i 2^{n-i}$	$f_{k=15}(x)$	s_x^{15}	$f_{k=14}(x)$	s_x^{14}	order of coeff.
000	0	1	0	0	0	s_0
001	1	0	0	1	0	s_3
010	2	0	0	1	0	s_2
011	3	1	0	0	0	s_{23}
100	4	0	0	1	0	s_1
101	5	1	0	0	0	s_{13}
110	6	1	0	0	0	s_{12}
111	7	0	-8	1	8	s_{123}

Hence, according to Theorem 4.1, both functions f_{15} and f_{14} are affine. Additionally, function f_{14} is linear.

From Table 1 it follows that $f_{15}(x) = \overline{f}_{14}(x)$, and these functions can be described by the Boolean formulas: $f_{15}(x_1, x_2, x_3) = 1 \oplus x_1 \oplus x_2 \oplus x_3$ and $f_{14}(x_1, x_2, x_3) = x_1 \oplus x_2 \oplus x_3$. The obtained results are consistent with Theorem 4.1 and Property 4.1.

In many cases Boolean functions are given as incompletely defined, and then vector $\mathbf{Y}$ contains values $\{0, 1, -\}$, where symbol "$-$" denotes the "don't care minterms". In order to obtain coefficients of $\mathbf{S}$ type, values $\{0, 1, -\}$ of $\mathbf{Y}$ are coded by $\{1, -1, 0\}$, respectively.

THEOREM 4.2 *An n variable Boolean function f, undefined at one point only, can be affine if f is characterized by the Walsh-Hadamard spectrum distribution:*

$$s_x = \begin{cases} +1 \textit{ for } x = 0 & \rightarrow \textit{undefined point should have value } 1 \\ -1 \textit{ for } x = 0 & \rightarrow \textit{undefined point should have value } 0 \\ +(2^n - 1) \textit{ for } x = k/2 & \rightarrow c = 0 \\ -(2^n - 1) \textit{ for } x = (k-1)/2 & \rightarrow c = 1 \\ \pm 1 \textit{ otherwise} & \end{cases} \tag{7}$$

where: k has the same meaning as in Definition 4.1 and $x = 0, 1, ..., 2^n - 1$.

Proof. The proof results immediately from Corollary 4.1 and Theorem 4.1. ∎

It is obvious that for this case we have $2^n - [card(T_f) + card(F_f)] = 1$.

If any Boolean function f is undefined at d points, then $s_0 = 2^n - 2a - d$, where $a = card(T_f)$ or in other words a is a number of places where

$$f(x_1, x_2, ..., x_n) = 1$$

and

$$2^n - [card(T_f) + card(F_f)] = d.$$

Additionally, incompletely defined Boolean function can be completed as affine if:

$$s_0 = \begin{cases} +d & \text{then all } d \text{ points should have value } 1 \rightarrow \{\underbrace{-,-,...,-}_{d \text{ times}}\} \rightarrow \{1\} \\ -d & \text{then all } d \text{ points should have value } 0 \rightarrow \{\underbrace{-,-,...,-}_{d \text{ times}}\} \rightarrow \{0\} \\ \neq d & \text{then } d \text{ points have different values} \rightarrow \{\underbrace{-,-,...,-}_{d \text{ times}}\} \rightarrow \{0,1\} \end{cases}$$

and $\max\{|s_1|, |s_2|, ..., |s_{2^n-1}|\} = |2^n - d|$.

If this condition holds, then f has affine extensions. Furthermore, note that $\frac{1}{2}(2^n - s_i)$ is equal to the Hamming distance between f and the i^{th} affine function if $s_i \geq 0$ or is equal to the distance between f and the negation of the i^{th} affine function if $s_i < 0$. Hence, an affine extension of f is the i^{th} row of the $\mathbf{H}_n$ matrix if $s_i = 2^n - d$ or its negation if $s_i = d - 2^n$.

Affine Boolean functions are balanced, which directly results from the properties of the Hadamard matrices $\mathbf{H}_n$ and $\overline{\mathbf{H}}_n$.

EXAMPLE 4.2 *Table 2 includes descriptions of Boolean functions. The function from column 2 is fully defined. This function is linear. The rest of the functions are incompletely defined and each of them have different undefined numbers of points ("don't care" minterms).*

Table 2. The illustrative Boolean functions and their spectra

$x_1x_2x_3$	$f_{\substack{d=0\\k=10}}(x)$	s_x	$f_{\substack{d=1\\k=10}}(x)$	s_x	$f_{\substack{d=2\\k=10}}(x)$	s_x
000	0	0	0	-1	0	-2
001	1	0	1	-1	1	0
010	0	0	$-$	1	$-$	0
011	1	0	1	1	1	2
100	1	0	1	-1	1	0
101	0	8	0	7	$-$	6
110	1	0	1	1	1	2
111	0	0	0	1	0	0

$f_{\substack{d=2\\k=10}}(x)$	s_x	$f_{\substack{d=2\\k=10}}(x)$	s_x	$f_{\substack{d=3\\k=10}}(x)$	s_x
0	2	0	0	0	-1
–	-2	–	-2	–	-1
0	0	–	2	–	1
–	0	1	0	1	1
1	2	1	0	1	1
0	6	0	6	–	5
1	0	1	2	1	3
0	0	0	0	0	-1

From Theorem 4.2 it follows that the function $f_{\substack{d=1\\k=10}}$ can be made complete as linear if the undefined point has the value of 1. From the obtained spectral coefficients for the functions of type $f_{\substack{d=2\\k=10}}$ it results that functions can be linear if at all the undefined points the values 1 or 0 occur respectively. From Theorem 4.2 it follows that the Boolean function $f_{\substack{d=3\\k=10}}$ can be also brought to linearity, as shown in Table 2.

In this way any undefined Boolean function can be checked whether it can be made complete as an affine function.

Unfortunately, the described above matrix-based method is impractical for large n, but, as it has been shown, Boolean functions have particular properties allowing for the modification of the methods of calculations.

5. The method of spectra calculation

The Walsh-Hadamard spectral coefficients can be calculated by means of different methods, like FFT-algorithms or BDDs applications (Ahmed, Rao, 1975; Fujita, Yang, 1995; Stanković, Falkowski, 2002; Thornton, Drechsler, 2001). Recently, a method to compute Walsh-Hadamard spectrum directly from the sum-of-products (SOP) representation has been proposed. However, many practical logic functions cannot be represented in the SOP form, because the numbers of such products can be too large (Fujita, Yang, 1995).

Furthermore, computations, even with the fast transforms, can be additionally difficult, because the truth-table of Boolean functions grows expotentially with n. For that reason the methods based on DDs are preferred. Additionally, in the DD approach, no explicit matrix product is formed, but the Walsh-Hadamard matrix definition is explicitly embedded. In this paper fast spectrum computation using spectral DDs (SDDs) has been used. Readers are referred to Fujita, Yang (1995) and Thornton, Drechsler (2001) for details of SDDs properties. The CUDD tool (Somenzi, 2004) was used for the creation of the spectral decision diagrams.

It can be observed that besides the BDDs technique, fast transforms based on butterfly charts are still applied. Butterfly algorithms are well known, and therefore will not be presented here. From Fujita, Yang (1995) and Stanković, Falkowski (2002) it follows that FFT algorithms have the same computational complexity as the BDD techniques but only for the worst cases. In significant number of cases BDD techniques have huge advantage. Additionally, the main disadvantage of the FFT algorithms is that they cannot be used for logic functions with large numbers of variables, because the main limiting factor of spectral methods in processing of switching functions is their calculation complexity. For example, the time and space complexities of FFT-like algorithms are $O(n2^n)$ and $O(2^n)$, respectively, for Boolean functions of n variables.

One way to overcome the above difficulties is to use Spectral Decision Diagrams to compute Walsh-Hadamard spectral coefficients (Fujita, Yang, 1995; Stanković, Falkowski, 2002).

For some classes of Boolean functions in order to speed up the spectra calculations the Haar functions are used. For example, Karpovsky (1976) shows that the spectral complexity of conjunction and disjunction increases with the number of variables exponentially for the Walsh functions and only linearly for the Haar functions. Hence, new methods of spectra calculations for various classes of functions are still being sought.

Example 5.1 *Suppose that a Boolean function is described as $f = x_1x_2 \oplus x_3$. The truth vector of such function has the form $\mathbf{Y}_f = [0,1,0,1,0,1,1,0]$. The SDD and reduced SDD are shown in Fig. 1.*

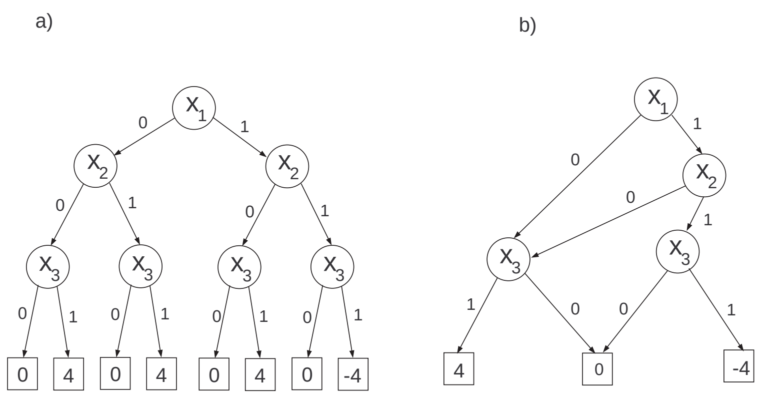

Figure 1. a) Spectral Decision Diagram b) reduced SDD

According to Theorem 4.1, the analyzed function is non linear.

Let $\mathbf{Y} = [y_0, y_1, ..., y_{2^n-1}]$ be the two-valued truth vector of the function $f(x_1, x_2, ..., x_n)$. From the properties of the Hadamard matrices it follows that all of the Walsh-Hadamard spectral coefficients of a Boolean function can be calculated recursively from the equation:

$$\mathbf{H}_n \times [y_0, y_1, ..., y_{2^n-1}] = \mathbf{H}_n[y_0, y_1, ..., y_{2^n-1}]^T = \begin{bmatrix} A+B \\ A-B \end{bmatrix} \tag{8}$$

where: $A = \mathbf{H}_{n-1}\ [y_0, y_1, ..., y_{2^{n-1}-1}]^T$ and $B = \mathbf{H}_{n-1}\ [y_{2^{n-1}}, y_{2^n}, ..., y_{2^n-1}]^T$.

Formula (8) can be used to calculate the Walsh-Hadamard spectrum, because instead of inconvenient large matrices $\mathbf{H}_n$ some small matrices can be used. The described formula can be obviously easily implemented in parallel computations. The parallel algorithms significantly accelerate computations. In these cases the matrices $\mathbf{H}_i$ can be first calculated by table lookup. Additionally, by means of formula (8) it is easy to check whether specified Boolean function is linear. In these instances each part of the spectrum calculated by means of equation (8) must satisfy the conditions of Theorem 4.1. Additionally, for that function we have $s_i^A = |s_i^B|$, where s^A, s^B denote spectral coefficients of parts A and B, respectively. The dependence mentioned follows from the observation that any n-variable affine Boolean function has truth vector $\mathbf{Y} = [a, b]$ which can be divided into two subvectors of the length 2^{n-1} such that $a = b$ or $a = \overline{b}$ or $\overline{a} = b$ or $\overline{a} = \overline{b}$.

Calculations described by equation (8) can be realized by means of flow chart (Fig. 2).

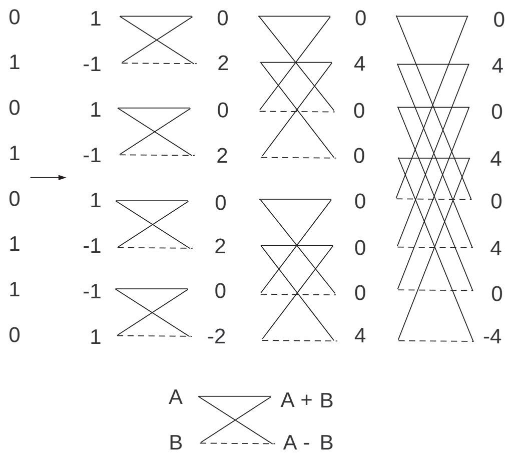

Figure 2. Butterfly diagram for spectrum computation

From (8) it follows that the complexity of this method is the same as in the FFT algorithm. If it is necessary to check whether Boolean function is linear (affine), then the complexity of such calculations is only $O(2^n)$, because only the first step of the algorithm is needed. This is because after the first step the parts of spectra are known and each of them must describe a spectrum of a linear function.

6. Experimental results

The computational advantages of the Walsh transform cause that it is of a considerable interest to VLSI designers as well, therefore the time complexity of calculating spectra is the most important factor. Computation of the spectrum of an n variable Boolean function is a complex operation requiring, in the general case, $n2^n$ operations of addition and subtraction and 2^n memory locations. Unfortunately, for large Boolean functions, storing and manipulating of spectral coefficients is difficult and in some cases impossible. Many types of DDs use very compact data structures for the representation and manipulation of large Boolean functions.

The described above method of affine functions identification can be implemented by means of SDDs. For small Boolean functions, classical Fast Walsh Transform can be used as well. It is suitable for hardware realizations, where matrix multiplication is simple.

The Fast Walsh-Hadamard Transform (FWHT) (Ahmed, Rao, 1975; Karpovsky, 1976; Porwik, 2002) and SDDs have been used and compared. Additionally, FWHT has been realized in two versions: as the classical approach (Ahmed, Rao, 1975) and as "butterfly diagram" (Stanković, Falkowski, 2002). All methods have been compared by means of time measurements which were necessary to find the Boolean function spectra and evaluate whether function can be affine. The method based on SDDs gives good results because, according to Theorem 4.1, information about linearity of a Boolean function can be obtained immediately from the spectra. For a large function its SDD representation gives reduced spectra only, but as has been shown, it is sufficient for the linearity estimation.

All experiments were performed by means of a PC computer (with Linux operating system). The computer was equipped with AMD Duron (Morgan) 1.2GHz processor and 512 Mbyte main memory.

In experiment, spectra calculation computations times have been determined. Results of investigations are shown in Fig. 3.

From Fig. 3 it follows that the time of spectra computation by means of the butterfly algorithm gives slightly better results when compared to the well known FWHT method. The method presented was tested by means of a set of standard benchmarks functions. Each benchmark was treated as a set of separate single-output functions and tested individually. Additionally, each benchmark function was specially prepared to include 25% undefined ("don't care") places.

The symbol #inóut indicates the number of inputs and outputs in the benchmark function. Column 3 in Tables 5 and 6 includes information on how many nodes in the reduced SDD have been determined for the appropriate benchmark. The columns 4, 5 and 6 indicate the total time of spectra computation and Boolean function identification. All values smaller than 0.01 are indicated in tables as 0.00.

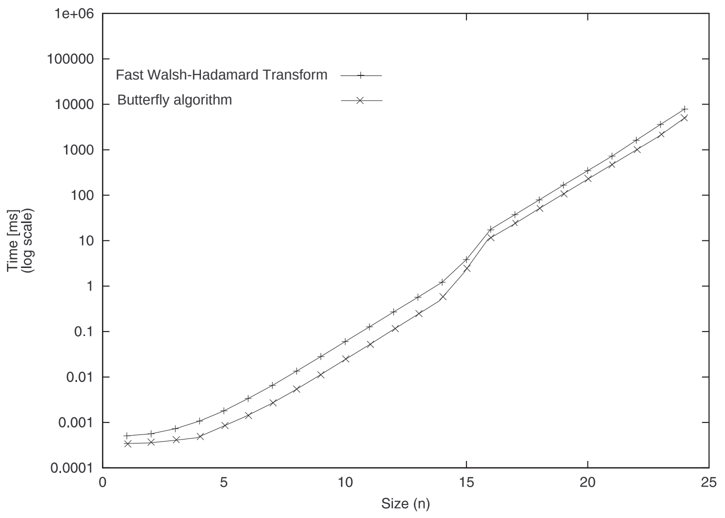

Figure 3. Time of affine Boolean function search based on WHT

Table 5. Experimental results for single output functions ($n < 20$)

Circuit	#in/out	nodes SDD	t_1[ms] SDD	t_2[ms] FWHT	t_3[ms] Butterfly
xor5	5/1	7	0.00	0.0012	0.0008
sym10	10/1	60	0.00	0.0413	0.0232
intb (1st)	15/7	336	0.00	2.3375	1.4260
spla (4th)	16/46	155	0.08	0.8452	0.4939
t481	16/1	204	0.00	5.1387	3.7300
table5 (12th)	17/15	12526	0.02	18.6053	15.1150
in2 (2nd)	19/10	151	0.00	81.9875	62.2200

Table 6. Experimental results for large variable single output functions ($n > 20$)

Benchmark	#in/out	nodes SDD	t_1[s] SDD	t_2[s] FWHT	t_3[s] Butterfly
t1 (23rd)	21/23	41	0.04	0.38740	0.31970
ts10 (1st)	22/16	100	0.01	0.80475	0.65730
in5 (5th)	24/14	743	0.00	3.28300	2.71567
chkn (1st)	29/7	152	0.01	106.79702	88.08214
b3 (2nd)	32/20	462	0.00	427.59010	339.18663
seq (29th)	41/35	121	0.04	5565.22418	4717.43246
x1 (5th)	51/35	148	0.05	memout	memout
x4 (10th)	94/71	40	0.09	memout	memout
apex5 (88th)	117/88	233	0.02	memout	memout
apex6 (1st)	135/99	4	0.02	memout	memout
frg2 (2nd)	143/139	22	0.08	memout	memout

From the results contained in Tables 5 and 6 it follows that investigation of the Boolean function's linearity can be carried out with the aid of different methods, but the SDDs approach is the most profitable.

7. Conclusions

The proposed spectral method of investigation allows us to obtain fast information about linearity of the analyzed function. The proposed method of spectral coefficients analysis and spectra calculation can be used for any incompletely specified Boolean functions.

In the paper, the relationship between the Walsh-Hadamard spectrum and the incompletely defined Boolean functions has been discussed. The simple methods for determining the linearity of incompletely defined Boolean functions directly from their Walsh-Hadamard spectrum has been shown.

The theorems, propositions and equations presented show a new approach which allows for determining efficiently the linear Boolean functions, especially the incompletely defined ones. The method presented can be implemented using matrix operations and SDDs. As it has been shown, the reduced spectrum obtained from SDD is sufficient for the Boolean function's linearity determination, because only one spectral coefficient should be analyzed. Hence, when it is necessary to check whether a Boolean function is linear, the test of linearity can be carried out on the basis of the pruned part of spectra. For this reason, linearity of very large Boolean functions can be checked.

The method described in this paper can be used in linearization of Boolean functions, where one part of a function can be realized as linear and the second part as nonlinear. 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. The linear block consists of XOR circuits only. For an n-variable function, complexity (number of equivalent two-input gates) of the linear blocks increases asymptotically not faster than $n^2/\log_2 n$ (for $n \to \infty$), whereas the complexity of the nonlinear block is almost always an expotentially increasing function of n (Karpovsky et al., 2003). In the latter reference, unlike in the presented method, only completely defined functions can be used. Our approach overcomes such difficulties. Therefore, the complexity of the linear blocks may be ignored in the linearization problems.

Acknowledgements We are grateful to the anonymous referees for useful comments and suggestions.

References

Ahmed, N. and Rao, K.R. (1975) *Orthogonal Transforms for Digital Signal Processing.* Springer-Verlag. Berlin.

Blahut, R.E. (1983) *Theory and Practice in Error Control Codes.* Addison-Wesley Publishing Company, London.

Falkowski, B.J. and Porwik, P. (1999) Evaluation of Nonlinearity in Boolean Functions by Extended Walsh-Hadamard Transform. *2nd Int. Conf. on Information Communications and Signal Processing ICISC'99, Singapore paper 2B2.2*, 1-4.

Falkowski, B.J., Schafer, I. and Perkowski, M.A. (1992) Effective Computer Methods for the Calculation of Rademacher-Walsh Spectrum for Completely and Incompletely Specified Boolean Functions. *IEE Tran. on Computer-Aided Design* **11** (10), 1207-1226.

Fujita, M. and Yang, J. (1995) Fast Spectrum Computation for Logic Functions using Binary Decision Diagrams. *Int'l Symp. Circ. and Systems*, 275-278.

Harmuth, H.F. (1977) *Sequency Theory. Foundations and Applications.* Academic Press, New York

Hurst, S.L., Miller, D.M. and Muzio, J.C. (1985) *Spectral Techniques in Digital Logic.* Academic Press, London.

Karpovsky, M.G. (1976) *Finite Orthogonal Series in the Design of Design of Digital Devices.* John Wiley, New York.

Karpovsky, M.G., Stanković, R.S. and Astola, J.T. (2003) Reduction of Size Decision Diagrams by Autocorrelation Functions. *IEEE Trans. on Comp.* **52** (5), 592-606.

Maitra, S. and Sarkar, P. (2002) Cryptographically significant Bolean functions with five valued spectra. *Theoretical Computer Science*, **276**, 133-146.

Mister, S. and Adams, C. (1996) Practical S-Box Design. *Workshop on selected areas in cryptography (SAC'96).* Queen's University Kingston, Ontario, Canada, 61-76.

Porwik, P. and Falkowski, B.J. (1999) Informatics Properties of the Walsh Transform. *2nd Int. Conf. on Information Communications and Signal Processing ICISC'99, Singapore, paper 2B2.4*, 1-5.

Porwik, P. (2000) Towards Calculation of Boolean Functions Nonlinearity Using Walsh Transform. *Arch. Theoret. Appl. Comp. Sci. Polish Acad. Sci.* **12** (1), 51-64.

Porwik, P. (2002) Efficient calculation of the Reed-Muller forms by means of the Walsh spectrum. *Int. Journal of Applied Mathematics and Computer Science* **12** (4), 571-579.

Porwik, P. (2000) Spectral modelling of digital systems with specified features. Prace Naukowe Uniwersytetu Śląskiego **1898**, Katowice (in Polish).

Porwik, P. (2003) The Spectral Test of the Boolean Function Linearity. *Journal of Applied Mathematics and Computer Science* **13** (4), 567-575.

Sasao, T. (1993) *Logic Synthesis and Optimalization.* Kluwer Academic Publishers, Dordrecht, Holland.

Seberry, J. and Zhang, X.M. (1994) Construction of Bent Function from Two Known Bent Functions. *Australasian Journal of Combinatorics* **9**, 21-34.

Somenzi, F. (2004) BDD Package. CUDD v.2.3.0. http://vlsi.colorado.edu/˜fabio/CUDD/cuddIntro.html

Stanković, R. and Falkowski, B. (2002) Spectral Transform Calculation through Decision Diagrams. *VLSI Design* **14** (1), 5-12.

Thornton, M. and Drechsler, R. (2001) Spectral Decision Diagrams Using Graph Transformations. *Int. Conf. Design, Automation and Test in Europe*, Munich, Germany, 713-719.

Wegener, I. (2000) Branching Programs and Dinary Decision Diagrams. *SIAM Monographs on Discrete Mathematics and Applications*, Philadelphia.

Yang, C. and Ciesielski, M. (2002) BDS: A BDD-based logic optimization system. *IEEE Trans. CAD* **21** (7), 866-876.