Transformations amongst the Walsh, Haar, Arithmetic and Reed-Muller Spectral Domains

M. A. Thornton
Electrical and Computer Engineering
Mississippi State University
Starkville, MS
USA
mitch@ece.msu.edu

D. M. Miller
Computer Science
University of Victoria
Victoria, BC
CANADA
mmiller@csr.uvic.ca

R. Drechsler
Siemens AG
Munich
Germany
address
drechsle@informatik.uni-freiburg.de

Abstract

Direct transformation amongst the Walsh, Haar, Arithmetic and Reed-Muller spectral domains is considered. Matrix based techniques are developed and it is shown that these can be implemented as fast *in-place transforms. It is also shown that these transforms can be implemented directly on decision diagram representations.*

1 Introduction

Transformation between the Boolean and various spectral domains has been extensively studied [1, 3, 13, 14, 15]. In this paper, we review the fundamental structures and properties of spectral transforms and the resulting spectra of Boolean functions. Matrix based computation of the spectra is considered as are *fast* transform techniques derived from the matrix structures. The contribution of this paper is to show that fast transform techniques can be developed for direct transformation amongst certain spectral domains, *i.e.* transforms from one spectral domain to another that do not pass through the Boolean domain. These fast transform techniques can be directly implemented on decision diagram representations.

The paper is organized as follows. Section 2 provides the necessary mathematical background for this paper. Section 3 introduces the spectral domains considered. Fast transform techniques are discussed in Section 4. Direct transforms amongst the spectral domains are discussed in Section 5. Section 6 considers the use of decision diagram techniques to implement the fast transforms. The paper concludes with some closing remarks and suggestions for further research.

2 Background

An n-input completely-specified Boolean function f can be represented by $\mathbf{Y} = \{m_0, m_1, m_2 \ldots m_{2^n-1}\}^t$ a column vector with 2^n entries each giving the functional value for the corresponding minterm. $\mathbf{Y}$ is often called the truth vector for f.

f represented by $\mathbf{Y}$ can be transformed from the Boolean to a spectral domain as follows:

$$\mathbf{R} = \mathbf{T}^n \mathbf{Y} \tag{1}$$

where $\mathbf{T}^n$ is a 2^n by 2^n transform matrix the precise specification of which defines the spectral domain in question. In many cases, the matrix has a simple recursive structure which can be used to significant computational advantage as will be shown.

We restrict our interest to invertible transforms, hence:

$$\mathbf{Y} = (\mathbf{T}^n)^{-1}\mathbf{R}$$

The consequence is that the transforms between the Boolean and spectral domains fully preserve information, but, as is well known, the spectral domains make certain properties easier to consider than in the Boolean domain, and different spectral domains illuminate different functional properties.

Often the transform matrix can be expressed as a sequence of Kronecker products of a single base matrix. We here provide a brief review of the Kronecker product. More detail can be found in [10].

Given a matrix $\mathbf{A}$ of order $(m \times n)$ with the element in the i^{th} row and j^{th} column denoted a_{ij} and a matrix $\mathbf{B}$ of order $(r \times s)$, the Kronecker product $\mathbf{A} \otimes \mathbf{B}$ is given by

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} & \cdots & a_{1n}\mathbf{B} \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} & \cdots & a_{2n}\mathbf{B} \\ \vdots & \vdots & & \vdots \\ a_{m1}\mathbf{B} & a_{m2}\mathbf{B} & \cdots & a_{mn}\mathbf{B} \end{bmatrix}$$

The product matrix has order $(mr \times ns)$. Note that unlike the normal matrix product, the Kronecker product is defined for any matrix orders.

For matrices **A**, **B**, **C** and **D** and a scalar α, the following properties hold

$$\begin{aligned}
(\alpha\mathbf{A}) \otimes \mathbf{B} &= \alpha(\mathbf{A} \otimes \mathbf{B}) \\
\mathbf{A} \otimes (\alpha\mathbf{B}) &= \alpha(\mathbf{A} \otimes \mathbf{B}) \\
(\mathbf{A}+\mathbf{B}) \otimes \mathbf{C} &= \mathbf{A} \otimes \mathbf{C} + \mathbf{B} \otimes \mathbf{C} \\
\mathbf{A} \otimes (\mathbf{B}+\mathbf{C}) &= \mathbf{A} \otimes \mathbf{B} + \mathbf{A} \otimes \mathbf{C} \\
\mathbf{A} \otimes (\mathbf{B} \otimes \mathbf{C}) &= (\mathbf{A} \otimes \mathbf{B}) \otimes \mathbf{C} \\
(\mathbf{A} \otimes \mathbf{B})^t &= \mathbf{A}^t \otimes \mathbf{B}^t \\
(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) &= \mathbf{AC} \otimes \mathbf{BD} \qquad (2) \\
(\mathbf{A} \otimes \mathbf{B})^{-1} &= \mathbf{A}^{-1} \otimes \mathbf{B}^{-1}
\end{aligned}$$

Equation 2 is termed the *mixed product rule* and is only valid when the matrices are of appropriate dimension for the normal matrix products.

Finally, two observations from the above properties are particularly relevant to our work. The Kronecker product of two symmetric matrices is itself a symmetric matrix, and since the Kronecker product is an associative operation, the order of application of a sequence of Kronecker products does not matter.

Although not stated in [10], it is clear from the development there that the Kronecker product properties we make use of in this Chapter hold over $GF(2)$, a fact that will be useful below in the consideration of the Reed-Muller transform.

The following theorem concerning inverses of matrices expressed as Kronecker products of a base matrix is very useful.

Theorem 2.1 *Given a square invertible matrix* **A**,

$$\left[\bigotimes_{i=1}^{n} \mathbf{A}\right]^{-1} = \bigotimes_{i=1}^{n} \mathbf{A}^{-1}$$

Proof: This follows immediately by the iterative application of the identity $(\mathbf{A} \otimes \mathbf{B})^{-1} = \mathbf{A}^{-1} \otimes \mathbf{B}^{-1}$ and the associativity of the Kronecker product. $\square$

3 Spectral Transforms

In this section we present four particular spectral transforms that have been extensively studied in the literature: the Walsh, the Reed-Muller, the Arithmetic, and the Haar transforms.

3.1 Walsh Transform

Perhaps the most well known and most widely studied spectral transforms are based on a set of orthogonal functions defined by J. L. Walsh in 1923 [24] which are an extension of a set of functions defined by H. Rademacher [19] a year earlier. The transform itself is a form of Hadamard matrix [23].

The Walsh transform matrix $\mathbf{W}^n$ in Hadamard order can be defined as

$$\mathbf{W}^0 = [\ 1\ ] \quad \mathbf{W}^n = \begin{bmatrix} \mathbf{W}^{n-1} & \mathbf{W}^{n-1} \\ \mathbf{W}^{n-1} & -\mathbf{W}^{n-1} \end{bmatrix}$$

An equivalent definition using the Kronecker product is particularly useful here

$$\mathbf{W}^1 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

and

$$\mathbf{W}^n = \mathbf{W}^1 \otimes \mathbf{W}^{n-1}$$

Since the Kronecker product is associative, this may be written as

$$\mathbf{W}^n = \bigotimes_{i=1}^{n} \mathbf{W}^1$$

The rows of $\mathbf{W}^n$ are the set of 2^n n-variable Walsh functions of which the n-variable Rademacher functions are a subset. In addition to the Hadamard (Walsh-Hadamard), the Walsh (Walsh-Kaczmarz), the Paley-Walsh, and the Rademacher-Walsh orderings have been studied [4][14] . The Hadamard ordering has seen most use since the simple recursive structure of the transform matrix allows for 'fast transform' methods [9] [21]. The Hadamard, Walsh and Walsh-Paley orderings share the very useful property that the transform matrix is its own inverse up to a scaling factor of $\frac{1}{2^n}$ as will be shown below for the Hadamard case. The practical importance of this is that the same computational procedure can be used for transforming between the function and spectral domains with the simple adjustment of scaling.

The Walsh spectrum **R** of f is given by

$$\mathbf{R} = \mathbf{W}^n\mathbf{Y}$$

where the matrix multiplication is carried out over the integers, *i.e.* logic 0(1) is treated as the integer 0(1). We term this *R-encoding*.

An alternate formulation represents the function by the vector **Z** in which logic 0 is coded as $+1$ and logic 1 is coded as -1, which we term *S-encoding*. In this case the spectrum is given by

$$\mathbf{S} = \mathbf{W}^n\mathbf{Z}$$

Since $\mathbf{Z} = \mathbf{1} - 2\mathbf{Y}$, it can be shown that

$$s_0 = 2^n - 2r_0; \quad s_\alpha = -2r_\alpha, \forall\, \alpha \subset \{1, 2 \ldots n\}$$

so the information content of the **R** and the **S** spectral coefficients is the same.

Theorem 3.1 $(\mathbf{W}^n)^{-1} = \frac{1}{2^n}\mathbf{W}^n$.

Proof: The proof follows from Theorem 2.1, the fact $(\mathbf{W}^1)^{-1} = \frac{1}{2}\mathbf{W}^1$ and the fact that scalar multipliers can be factored out of a Kronecker product. □

3.2 Reed-Muller Transform

The Reed-Muller transfrom is motivated by the seminal work in 1954 of I.S. Reed [20] and R.E. Muller [18] which led to considerable interest in the Reed-Muller (AND-XOR) expansion of Boolean functions. The transform matrix $\mathbf{M}^n$ is defined by

$$\mathbf{M}^0 = [\ 1\ ] \quad \mathbf{M}^n = \begin{bmatrix} \mathbf{M}^{n-1} & 0 \\ \mathbf{M}^{n-1} & \mathbf{M}^{n-1} \end{bmatrix} \qquad (3)$$

and the spectrum **R** is given by

$$\mathbf{R} = \mathbf{M}^n\mathbf{Y} \qquad (4)$$

In this case, the matrix multiplication is over $GF(2)$ *i.e.* integer addition is replaced with summation modulo-2. $\mathbf{M}^n$ can be expressed using the Kronecker product as

$$\mathbf{M}^1 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$
$$\mathbf{M}^n = \bigotimes_{i=1}^{n} \mathbf{M}^1 \qquad (5)$$

Theorem 3.2 $(\mathbf{M}^n)^{-1} = \mathbf{M}^n$ *over* $GF(2)$.

Proof: The proof follows from Theorem 2.1 and the fact $(\mathbf{M}^1)^{-1} = \mathbf{M}^1$ over $GF(2)$. □

From this theorem we have:

$$\mathbf{Y} = \mathbf{M}^n\mathbf{R} \qquad (6)$$

The above shows that **Y** is a linear combination (over $GF(2)$) of the columns of $\mathbf{M}^n$ for which the relevant coefficient in **R** is 1. Each column of $\mathbf{M}^n$ represents a function which is the logical AND of a subset of $x_1, x_2, \ldots, x_n$. The leftmost column is the constant function 1 which corresponds to the AND of no variables. Hence the Reed-Muller spectrum identifies a representation of a Boolean function as a sum over $GF(2)$ of a collection of products of variables. To be precise,

$$\mathbf{Y} = \sum_{i=0}^{2^n-1} \mathbf{r}_i \mathbf{M}_i^n \qquad (7)$$

where $\mathbf{M}_i^n$ is the i^{th} column of $\mathbf{M}^n$.

3.3 Arithmetic Transform

The arithmetic transform [12], which is also known as the *probability transform* [22] and the *inverse integer Reed-Muller transform* [8] was initially introduced by S.K. Kumar and M.A. Breuer in 1981 [16] in work on probabilistic aspects of Boolean functions. The transform matrix has a recursive structure analogous to that of the Walsh and Reed-Muller transforms and is given by

$$\mathbf{A}^0 = [\ 1\ ] \quad \mathbf{A}^n = \begin{bmatrix} \mathbf{A}^{n-1} & 0 \\ -\mathbf{A}^{n-1} & \mathbf{A}^{n-1} \end{bmatrix} \qquad (8)$$

or alternatively

$$\mathbf{A}^1 = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}$$
$$\mathbf{A}^n = \bigotimes_{i=1}^{n} \mathbf{A}^1 \qquad (9)$$

As before, we define the spectrum as

$$\mathbf{R} = \mathbf{A}^n\mathbf{Y} \qquad (10)$$

Theorem 3.3

$$(\mathbf{A}^n)^{-1} = \bigotimes_{i=1}^{n} \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$

Proof: The proof follows from Theorem 2.1 and the fact $(\mathbf{A}^1)^{-1} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$. □

Note that while $(\mathbf{A}^1)^{-1} = \mathbf{M}^1$, their use is quite different since the arithmetic spectrum is computed over the integers whereas the Reed-Muller spectrum is computed over $GF(2)$. It is for this reason the arithmetic transform was termed the inverse integer Reed-Muller transform in [8].

3.4 Haar Transform

The orthogonal Haar functions presented by A. Haar in 1910 [11] form a set of 2^n continuous orthogonal functions over the interval [0,1]. They can be defined as follows where *k* is over the continuous interval 0 to 1:

$$\begin{aligned} H_0^0(k) &= +1.0 \\ H_i^q(k) &= (\sqrt{2})^{i-1}(+1.0), \ \text{for } \frac{q}{2^{i-1}} \le k < \frac{q+\frac{1}{2}}{2^{i-1}} \\ &= (\sqrt{2})^{i-1}(-1.0), \ \text{for } \frac{q+\frac{1}{2}}{2^{i-1}} \le k < \frac{q+1}{2^{i-1}} \\ &= 0, \ \text{at all other points} \end{aligned} \qquad (11)$$

where $i = 1, 2, \ldots, n$ and $q = 0, 1, \ldots, 2^{i-1} - 1$.

Discrete sampling of the set of Haar functions gives a $2^n \times 2^n$ orthogonal matrix $\mathbf{T}^n$. For $n = 3$,

$$\mathbf{T}^3 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sqrt{2} & \sqrt{2} & -\sqrt{2} & -\sqrt{2} \\ 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 \end{bmatrix}$$

$\mathbf{T}^n$ is a complete, orthogonal matrix with $\Sigma_{k=0}^{2^n-1} t_{ik} t_{jk} = 2^n$ if $i \neq j$ and 0 otherwise. The following result follows directly.

Theorem 3.4 $[\mathbf{T}^n]^{-1} = \frac{1}{2^n}[\mathbf{T}^n]^t$.

Note that $\mathbf{T}^n$ is not symmetric so the transpose is needed for the inverse.

A computationally more practical *modified* Haar transform $\mathbf{K}^n$ is derived from $\mathbf{T}^n$ by normalizing the nonzero entries of $\mathbf{T}^n$ to take the values +1 and -1 yielding for $n = 3$ for example:

$$\mathbf{K}^3 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix}$$

Theorem 3.5 *The modifiednormalized Haar transform can be expressed as*

$$\mathbf{K}^0 = [\ 1\ ] \quad \mathbf{K}^n = \begin{bmatrix} \mathbf{K}^{n-1} \otimes [\ 1 \quad 1\ ] \\ \mathbf{I}^{n-1} \otimes [\ 1 \quad -1\ ] \end{bmatrix} \tag{12}$$

Proof: For the modified normalized Haar transform, Equation 11 becomes:

$$\begin{aligned} H_0^0(k) &= +1 \\ H_i^q(k) &= +1, \text{ for } \frac{q}{2^{i-1}} \leq k < \frac{q+\frac{1}{2}}{2^{i-1}} \\ &= -1, \text{ for } \frac{q+\frac{1}{2}}{2^{i-1}} \leq k < \frac{q+1}{2^{i-1}} \\ &= 0, \text{ at all other points} \end{aligned} \tag{13}$$

where $i = 1, 2, \ldots, n$ and $q = 0, 1, \ldots, 2^{i-1} - 1$.

For $i = n$, 2^{n-1} Haar functions are defined, each sampled at 2^n points which are q and $q + \frac{1}{2}$ for $q = 0, 1, \ldots, 2^{n-1} - 1$. The first of these functions, $H_n^0(k)$ is a 1, followed by a -1, followed by $2^n - 2$ 0's. The second, $H_n^1(k)$, is two 0's, followed by a 1, followed by -1 followed by $2^n - 4$ 0's. The ongoing pattern should be apparent and is illustrated above for the case of $n = 3$. These functions in order are the bottom 2^{n-1} rows of $\mathbf{K}^n$. They can be expressed in matrix form as $\mathbf{I}^{n-1} \otimes [1 \quad -1]$.

For $i = 1, 2, \ldots, n-1$, the Haar functions defined preceded by $H_0^0(k)$ are precisely those that compose $\mathbf{K}^{n-1}$ and it is these functions that comprise the upper half of $\mathbf{K}^n$. The difference is that to correspond to the lower half of $\mathbf{K}^n$, these functions must be sampled twice as often. This corresponds to duplicating the values across the function which can be expressed in matrix form as $\mathbf{K}^{n-1} \otimes [1 \quad 1]$.

Concatenating the two matrix expressions yields Equation 12. □

Theorem 3.4 does not hold for the normalized Haar transform matrix since while the rows do maintain pairwise orthogonality the resultant values are not the same. The inverse of $\mathbf{K}^n$ is given by the following theorem.

Theorem 3.6 $(\mathbf{K}^0)^{-1} = [1]$

$$(\mathbf{K}^n)^{-1} = \frac{1}{2^n}\left[(\mathbf{K}^{n-1})^{-1} \otimes \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \mathbf{I}^{n-1} \otimes \begin{bmatrix} 2^{n-1} \\ -2^{n-1} \end{bmatrix}\right]$$

Proof: Let

$$\mathbf{B}^n = \left[(\mathbf{B}^{n-1}) \otimes \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \mathbf{I}^{n-1} \otimes \begin{bmatrix} 2^{n-1} \\ -2^{n-1} \end{bmatrix}\right]$$

and consider $\mathbf{K}^n\mathbf{B}^n$. This yields

$$\mathbf{K}^n\mathbf{B}^n = \begin{bmatrix} \mathbf{Q}_{00}, \mathbf{Q}_{01} \\ \mathbf{Q}_{10}, \mathbf{Q}_{11} \end{bmatrix}$$

where

$$\begin{aligned} \mathbf{Q}_{00} &= (\mathbf{K}^{n-1} \otimes [1 \quad 1]) \left(\mathbf{B}^{n-1} \otimes \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right) \\ \mathbf{Q}_{01} &= (\mathbf{K}^{n-1} \otimes [1 \quad 1]) \left(\mathbf{I}^{n-1} \otimes \begin{bmatrix} 2^{n-1} \\ -2^{n-1} \end{bmatrix}\right) \\ \mathbf{Q}_{10} &= (\mathbf{I}^{n-1} \otimes [1 \quad -1]) \left(\mathbf{B}^{n-1} \otimes \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right) \\ \mathbf{Q}_{11} &= (\mathbf{I}^{n-1} \otimes [1 \quad -1]) \left(\mathbf{I}^{n-1} \otimes \begin{bmatrix} 2^{n-1} \\ -2^{n-1} \end{bmatrix}\right) \end{aligned}$$

Applying the mixed product rule $(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = \mathbf{AC} \otimes \mathbf{BD}$ and then reducing, the above becomes

$$\mathbf{K}^n\mathbf{B}^n = \begin{bmatrix} 2\mathbf{K}^{n-1}\mathbf{B}^{n-1} & 0 \\ 0 & 2^n\mathbf{I}^{n-1} \end{bmatrix} \tag{14}$$

We hypothesize that $(\mathbf{K}^n)^{-1} = \frac{1}{2^n}\mathbf{B}^n$. From Equation 14 this is clearly true when $n = 1$. Induction on n assumes $\mathbf{K}^{n-1}\mathbf{B}^{n-1} = [2^{n-1}\mathbf{I}^{n-1}]$ substituted into Equation 14 yields $\mathbf{K}^n\mathbf{B}^n = [2^n\mathbf{I}^n]$. Hence $(\mathbf{K}^n)^{-1} = \frac{1}{2^n}\mathbf{B}^n$ and the theorem is proven. □

For $n = 3$ for example, the inverse is

$$[\mathbf{K}^3]^{-1} = \frac{1}{2^3}\begin{bmatrix} 1 & 1 & 2 & 0 & 4 & 0 & 0 & 0 \\ 1 & 1 & 2 & 0 & -4 & 0 & 0 & 0 \\ 1 & 1 & -2 & 0 & 0 & 4 & 0 & 0 \\ 1 & 1 & -2 & 0 & 0 & -4 & 0 & 0 \\ 1 & -1 & 0 & 2 & 0 & 0 & 4 & 0 \\ 1 & -1 & 0 & 2 & 0 & 0 & -4 & 0 \\ 1 & -1 & 0 & -2 & 0 & 0 & 0 & 4 \\ 1 & -1 & 0 & -2 & 0 & 0 & 0 & -4 \end{bmatrix}$$

As is apparent from the above example, $(\mathbf{K}^n)^{-1}$ is the transpose of $\mathbf{K}^n$ with scaling factors applied to certain columns. From the recursive structure of Equation 14, one can verify that the appropriate scaling factor is 2^{n-k} where k is the $\log_2(p)$ and p is the number of non-zero entries in the column. It is clear from the definition of $\mathbf{K}^n$ that p is always a power of 2 so k is always a positive integer.

4 Transform Procedures

The above spectra can be directly computed by appropriate matrix multiplication, however the computational cost of this approach is generally prohibitive for functions of significant size. Fortunately, more efficient alternative techniques exist. In this section, we present so-called *fast* transform techniques.

4.1 Fast Walsh-Hadamard Transform

The recursive definition of the Hadamard-ordered Walsh transform is the basis for a fast Hadamard transform (FHT) method analogous to a fast Fourier transform (FFT) over discrete data. Observe that

$$\mathbf{R} = \begin{bmatrix} \mathbf{W}^{n-1} & \mathbf{W}^{n-1} \\ \mathbf{W}^{n-1} & -\mathbf{W}^{n-1} \end{bmatrix} \begin{bmatrix} \mathbf{Y}^0 \\ \mathbf{Y}^1 \end{bmatrix}$$

where $\mathbf{Y}^0$ and $\mathbf{Y}^1$ represents a partitioning of $\mathbf{Y}$ into two equal sized subvectors. It follows that

$$\mathbf{R} = \begin{bmatrix} \mathbf{W}^{n-1}\mathbf{Y}^0 + \mathbf{W}^{n-1}\mathbf{Y}^1 \\ \mathbf{W}^{n-1}\mathbf{Y}^0 - \mathbf{W}^{n-1}\mathbf{Y}^1 \end{bmatrix} \tag{15}$$

$$= \begin{bmatrix} \mathbf{W}^{n-1}(\mathbf{Y}^0 + \mathbf{Y}^1) \\ \mathbf{W}^{n-1}(\mathbf{Y}^0 - \mathbf{Y}^1) \end{bmatrix} \tag{16}$$

The above shows that the computation of the n^{th} order transform involves the application of $(n-1)^{th}$ order transforms to two subvectors of $\mathbf{Y}$ followed by the addition and subtraction of the results. Alternatively, the transform can be computed as the addition and subtraction of two subvectors of $\mathbf{Y}$ followed by the application of two $(n-1)^{th}$ order transforms to the resultant subvectors. A similar reduction can be applied to the computation of the $(n-1)^{th}$ order transforms. Indeed the reduction can be iteratively applied down to the trivial case of applying $\mathbf{W}^0$ transforms. The result is that the objective of computing $\mathbf{W}^n\mathbf{Y}$ is reduced to a sequence of vector additions and subtractions.

At each iteration, there are twice as many additions and subtractions as for the previous iteration involving subvectors of half the size. The computational work at each iteration is thus the same. In total, n iterations are involved with each involving 2^n elements yielding time complexity $O(n2^n)$. The operations can be applied in place on a single vector so the space complexity is $O(2^n)$. This is in contrast to the matrix multiplication approach where the time and space complexities are both $O(2^{2n})$.

The computational sequence arising from the above is illustrated in Figure 1 for the case of $n = 3$. For clarity, we show the computation as creating new vectors but note again that the computation can in fact be done in place. The interpretation of the *butterfly* signal flowgraphs in Figure 1 is as shown in Figure 2.

The FHT method represents a substantial improvement over computing the spectrum by matrix multiplication but it is still prohibitive for large functions due to its exponential complexity. A major importance of this approach is that it forms the basis for very efficient decision diagram approaches.

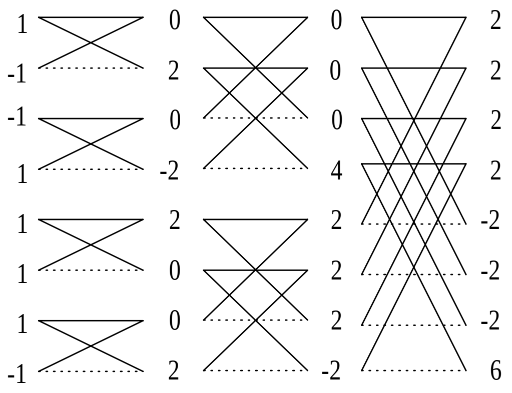


Figure 1. Example of Fast Transform Computation of Walsh Spectrum

4.2 Fast Reed-Muller Transform

A similar approach is possible for developing a fast Reed-Muller transform since $\mathbf{M}^n$ has a similar recursive structure to that of $\mathbf{W}^n$. The situation for $n = 3$ is illustrated in Figure 3 with the interpretation of signal flow subgraph is as shown in Figure 4. The computations for a fast Reed-Muller transform are of course over $GF(2)$.

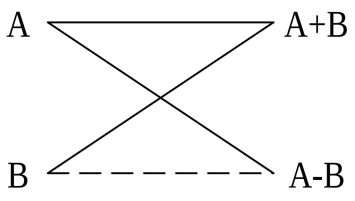


Figure 2. Interpretation of a "Butterfly" Signal Flowgraph for the Walsh Transform

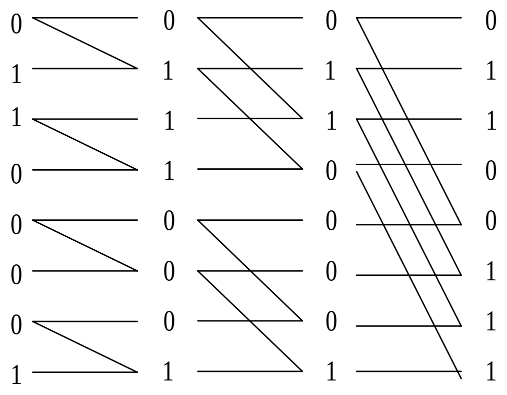


Figure 3. Example of Fast Transform Computation of Reed-Muller Spectrum

4.3 Fast Arithmetic Transform

The situation for the arithmetic transform is analogous to the Walsh and Reed-Muller cases and thus not explicitly shown here.

4.4 Fast Haar Transform

The signal flowgraph for a fast normalized Haar transform can be identified directly from the recursive definition of $\mathbf{K}^n$ given in Theorem 12. The case for $n = 3$ is depicted in Figure 5. The "butterfly" structures are as defined in the Walsh case, Figure 2.

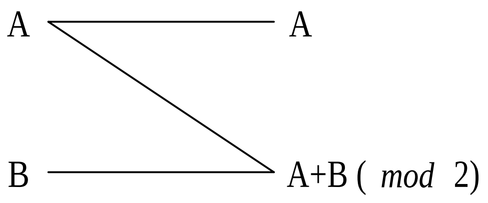


Figure 4. Interpretation of a Signal Flow Subgraph for the Reed-Muller Transform

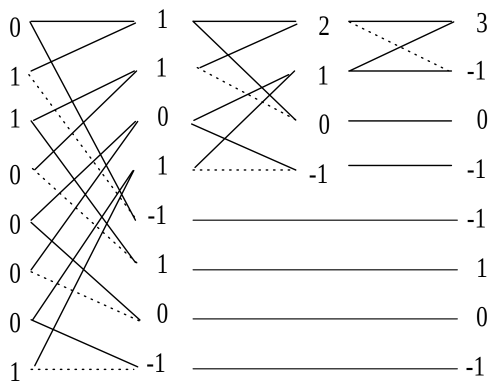


Figure 5. Example of Fast Transform Computation of Haar Spectrum

Figure 5 depicts the normalized Haar transform. For the unnormalized transform defined by Equation 11 the structure is the same but appropriate multipliers must be applied in the computations.

The inverse transform has the reverse structure and once again appropriate multipliers must be applied, this time in both the normalized and unnormallzed cases. Figure 6 depicts the situation for the inverse normalized transform using the same example as Figure 5. A value passing through a phase without going through a "butterfly" is multiplied by 2. The result is scaled by 2^3.

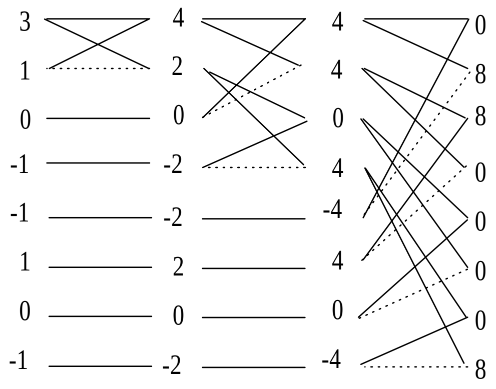


Figure 6. Example of Fast Transform Computation of Inverse Haar Transform

The Haar transform considered thus far and particularly the fast transform illustrated in Figure 5 is in sequency order. A drawback is that it can not be done in place since as is apparent from the flow diagram, pairs of elements are combined and, except for the first and last element in each transform phase, the results go to other positions. An alternative is to rearrange the computations into natural (Hadamard) order which does allow for in-place computation. It is this ordering that we employ in computing Haar spectra using decision diagrams.

The natural (Hadamard) order Haar transform can be defined as follows (we use $\mathbf{H}^n$ to distinguish this transform from the sequency ordered Haar transform $\mathbf{K}^n$):

$$\begin{aligned}
\mathbf{H}^n &= \mathbf{B}^n + \mathbf{D}^n \\
\mathbf{D}^n &= \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix} \otimes \mathbf{D}^{n-1} \\
\mathbf{B}^n &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \otimes \mathbf{B}^{n-1} + \begin{bmatrix} 0 & 0 \\ 1 & -1 \end{bmatrix} \otimes \mathbf{D}^{n-1} \\
\mathbf{D}^0 &= [1], \mathbf{B}^0 = [0]
\end{aligned} \tag{17}$$

For example, for $n = 3$ the above yields:

$$\mathbf{H}^3 = \begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\
0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & -1
\end{bmatrix}$$

Numbering the rows of $\mathbf{K}^3$ from 0 to 7, the rows of $\mathbf{H}^3$ adhere to the permutation [0,4,2,5,1,6,3,7]. Hence, the spectral coefficients determined using $\mathbf{H}^n$ in place of $\mathbf{K}^n$ will be similarly permuted.

We first show that the formulation given generates the Haar functions and then consider the related inverse transform..

Theorem 4.1 *Equation 17 generates the complete set of Haar functions in natural order.*

Proof: Two initial observations for all n: $\mathbf{D}^n$ is of order $(2^n \times 2^n)$ and consists of a top row of all 1's with 0's everywhere else; $\mathbf{B}^n$ is of order $(2^n \times 2^n)$ and has a top row of all 0's.

It is apparent from the definition of $\mathbf{H}^n$ that it can be written

$$\mathbf{H}^n = \begin{bmatrix} \mathbf{B}^{n-1} + \mathbf{D}^{n-1} & \mathbf{D}^{n-1} \\ \mathbf{D}^{n-1} & \mathbf{B}^{n-1} - \mathbf{D}^{n-1} \end{bmatrix} \tag{18}$$

It is useful to let $\mathbf{C}^n$ be a $(2^n - 1 \times 2^n)$ matrix which is $\mathbf{B}^n$ with its top row removed. $\mathbf{H}^n$ can then be written

$$\mathbf{H}^n = \begin{bmatrix} 1\,1\cdots 1 & 1\,1\cdots 1 \\ \mathbf{C}^{n-1} & \mathbf{0} \\ & \\ 1\,1\cdots 1 & -1\;-1\cdots -1 \\ \mathbf{0} & \mathbf{C}^{n-1} \end{bmatrix}$$

where $\mathbf{0}$ denotes a $(2^{n-1} - 1 \times 2^{n-1})$ matrix of 0's.

The top row of $\mathbf{H}^n$ consists of 2^n 1's and is H_0^0. The row at the top of $\mathbf{H}^n$ is 2^{n-1} 1's followed by 2^{n-1} -1's, which is H_1^0. It is important to note from Equation 13 that these are the only two Haar functions that are non-zero in both halves of the definition space. We must next show that the remaining Haar functions are also generated which we do by induction.

Clearly

$$\mathbf{H}^1 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

Assume $\mathbf{H}^{n-1}$ includes all the $(n-1)^{th}$ order Haar functions which means $\mathbf{C}^{n-1}$ includes them all except H_0^0. This is precisely what is required since a review of Equation 13 show the second and higher order Haar functions generated for n are two occurrences of the second and higher order Haar functions generated for the case of $n-1$ one in the lower half of the definition space and the second in the higher half. It follows that $\mathbf{H}^n$ includes all Haar functions.

It is also clear from the construction that $\mathbf{H}^n$ orders the Haar function in natural order, that is earliest zero-crossing first.

A fast transform technique for the naturally ordered Haar transform is easily developed from the recursive structure in Equation 17. Figure 7 illustrates the situation for $n = 3$. It is interesting to observe that the structure is essentially the structure from the Walsh case with certain "butterflies" removed. The number of computations is the same as for the sequency ordered Haar transform, namely $2^n - 2$ but the significant advantage is the computations can be done in place since each butterfly combines two elements and places the results in the same locations.

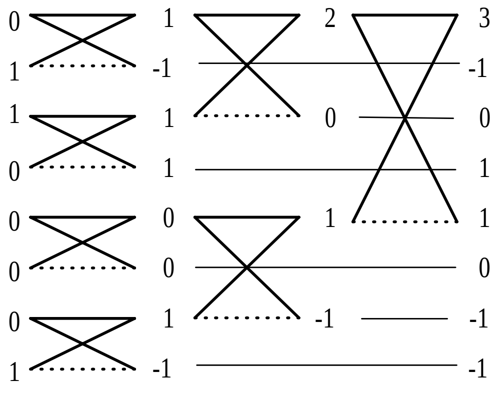


Figure 7. Example of Fast Transform Computation of Haar Spectrum in Natural Order

We next consider the inverse of $\mathbf{H}^n$.

Theorem 4.2 $\mathbf{H}^{n^{-1}} = \frac{1}{2^n}\mathbf{G}^n$ *where*

$$\mathbf{G}^n = \mathbf{C}^n + \mathbf{E}^n$$

$$\begin{aligned}\mathbf{E}^n &= \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix} \otimes \mathbf{E}^{n-1} \\ \mathbf{C}^n &= \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \otimes \mathbf{C}^{n-1} + \begin{bmatrix} 0 & 1 \\ 0 & -1 \end{bmatrix} \otimes \mathbf{E}^{n-1} \\ \mathbf{E}^0 &= [1], \mathbf{C}^0 = [0] \end{aligned} \tag{19}$$

Proof: From Equations 17 and 19 we have

$$\begin{aligned}\mathbf{H}^n\mathbf{G}^n &= (\mathbf{B}^n + \mathbf{D}^n)(\mathbf{C}^n + \mathbf{E}^n) \\ &= \mathbf{B}^n\mathbf{C}^n + \mathbf{B}^n\mathbf{E}^n + \mathbf{D}^n\mathbf{C}^n + \mathbf{D}^n\mathbf{E}^n\end{aligned}$$

From the previous theorem we know the rows of $\mathbf{B}^n$ are Haar functions (with the exception of H_0^0) each with an equal number of +1's and -1's, so the sum across each row of $\mathbf{B}^n$ is 0. Hence, $\mathbf{B}^n\mathbf{E}^n = \mathbf{0}$ where $\mathbf{0}$ denotes the matrix of all 0's.

$\mathbf{C}^n$ is constructed as the transpose of $\mathbf{B}^n$ with multipliers applied to certain columns. Hence each column of $\mathbf{C}^n$ sums to 0, so $\mathbf{D}^n\mathbf{C}^n = \mathbf{0}$ and

$$\mathbf{H}^n\mathbf{G}^n = \mathbf{B}^n\mathbf{C}^n + \mathbf{D}^n\mathbf{E}^n$$

$\mathbf{D}^n\mathbf{E}^n$ yields a matrix with 2^n in the top left corner and 0's everywhere else.

Now

$$\begin{aligned}\mathbf{B}^n\mathbf{C}^n = &\left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \otimes \mathbf{B}^{n-1} + \begin{bmatrix} 0 & 0 \\ 1 & -1 \end{bmatrix} \otimes \mathbf{D}^{n-1}\right) \\ &\left(\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \otimes \mathbf{C}^{n-1} + \begin{bmatrix} 0 & 1 \\ 0 & -1 \end{bmatrix} \otimes \mathbf{E}^{n-1}\right)\end{aligned}$$

Multiplying this through, applying the Kronecker mixed product rule and multiplying the constant matrices we have

$$\begin{aligned}\mathbf{B}^n\mathbf{C}^n = &\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \otimes \mathbf{B}^{n-1}\mathbf{C}^{n-1} + \\ &\begin{bmatrix} 0 & 1 \\ 0 & -1 \end{bmatrix} \otimes \mathbf{B}^{n-1}\mathbf{E}^{n-1} + \\ &\begin{bmatrix} 0 & 0 \\ 2 & -2 \end{bmatrix} \otimes \mathbf{D}^{n-1}\mathbf{C}^{n-1} + \\ &\begin{bmatrix} 0 & 0 \\ 0 & 2 \end{bmatrix} \otimes \mathbf{D}^{n-1}\mathbf{E}^{n-1}\end{aligned}$$

As above, $\mathbf{B}^{n-1}\mathbf{E}^{n-1} = \mathbf{D}^{n-1}\mathbf{C}^{n-1} = \mathbf{0}$ so

$$\mathbf{B}^n\mathbf{C}^n = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \otimes \mathbf{B}^{n-1}\mathbf{C}^{n-1} + \begin{bmatrix} 0 & 0 \\ 0 & 2 \end{bmatrix} \otimes \mathbf{D}^{n-1}\mathbf{E}^{n-1}$$

We hypothesize that $\mathbf{B}^n\mathbf{C}^n$ is a diagonal matrix with a 0 in the top left entry and 2^n for every other diagonal entry. It is readily verified that this is the case for $n = 1$. Assuming, it is true for $n - 1$ and substituting we find it is true for n since $\mathbf{D}^{n-1}\mathbf{E}^{n-1}$ is a matrix with 2^{n-1} in the top left corner and 0's elsewhere. Substituting this result back we find $\mathbf{H}^n\mathbf{G}^n = 2^n\mathbf{I}^n$ and the theorem is proven. □

Figure 8 illustrates the fast reverse transform procedure for $n = 3$. As in the sequency case, a value which passes through a phase without going through a "butterfly" must be multiplied by 2.

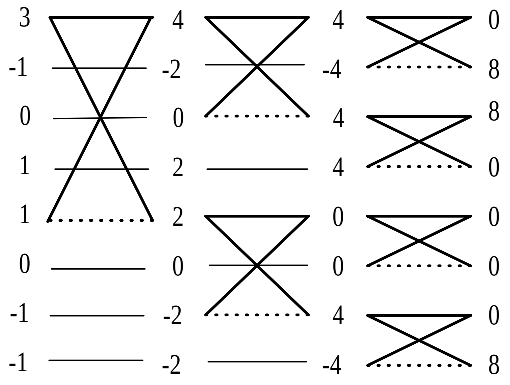


Figure 8. Example of Fast Transform Computation of Inverse Haar Transform in Natural Order

5 Relationships Amongst the Transforms

Since each of the transforms discussed has an inverse it is clearly possible to create a transform from one spectral domain to another in the worst case by simply passing through the Boolean functional domain. The issue is of course can such a transformation be done more efficiently.

For example, as identified in [16], if $\mathbf{S}$ is the arithmetic spectrum of a function, its Walsh spectrum $\mathbf{R}$ in R-encoding is given by $\mathbf{R} = \mathbf{W}^n(\mathbf{A}^n)^{-1}\mathbf{S}$. $(\mathbf{A}^n)^{-1}\mathbf{S}$ transforms the arithmetic spectrum to the functional domain after which the multiplication by $\mathbf{W}^n$ yields the Walsh spectrum. It is more efficient of course to treat $\mathbf{W}^n(\mathbf{A}^n)^{-1}$ as a single matrix which we can write as

$$\left(\bigotimes_{i=1}^{n} \mathbf{W}^1\right)\left(\bigotimes_{i=1}^{n} (\mathbf{A}^1)^{-1}\right)$$

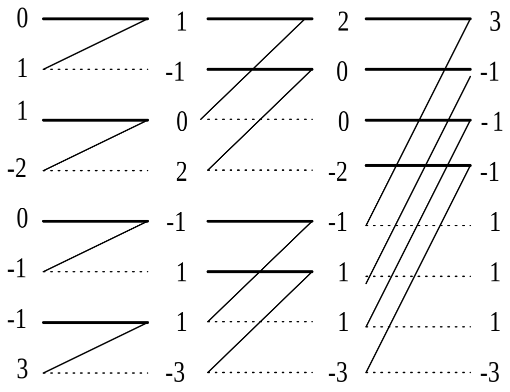
By the properties of the Kronecker product this can be written as

$$\bigotimes_{i=1}^{n} \mathbf{W}^1\,(\mathbf{A}^1)^{-1}$$

So the transform from the arithmetic to the Walsh domain can be accomplished using the transform matrix

$$\mathbf{T}^n = \bigotimes_{i=1}^{n} \begin{bmatrix} 2 & 1 \\ 0 & -1 \end{bmatrix}$$

which can be used as the basis for a fast transform approach. This is illustrated for $n = 3$ in Figure 9. Following a similar



The dark lines indicate multiplication by 2.

Figure 9. Example of Direct Fast Transform from the Arithmetic to Walsh Spectrum

approach, we can show that

$$\mathbf{T}^n = \frac{1}{2^n}\left(\bigotimes_{i=1}^{n}\left[\begin{array}{cc} 1 & 1 \\ 0 & -2 \end{array}\right]\right)$$

is a direct transform from the Walsh to the arithmetic spectral domain.

Transforming to and from the Haar domain is also possible. We here consider Walsh to Haar and Haar to Walsh transforms. arithmetic to Haar and Haar to arithmetic transforms can be developed in a similar fashion.

Theorem 5.1 *The Walsh-Hadamard spectrum of a function can be transformed to the natural order Haar spectrum using the transform*

$$\begin{aligned}
\mathbf{T}^n &= \frac{1}{2^n}(\mathbf{P}^n + \mathbf{Q}^n) \\
\mathbf{P}^n &= \left[\begin{array}{cc} 1 & 1 \\ 1 & -1 \end{array}\right] \otimes \mathbf{P}^{n-1} + \left[\begin{array}{cc} 0 & 0 \\ 0 & 2 \end{array}\right] \otimes \mathbf{Q}^{n-1} \\
\mathbf{Q}^n &= \left[\begin{array}{cc} 2 & 0 \\ 0 & 0 \end{array}\right] \otimes \mathbf{Q}^{n-1} \\
\mathbf{P}^0 &= [0] \;\; \mathbf{Q}^0 = [1]
\end{aligned}$$

Proof: We need to transform from the Walsh to the function domain and then to the Haar domain. The transform is thus given by

$$\mathbf{T}^n = \mathbf{H}^n\left(\frac{1}{2^n}\mathbf{W}^n\right)$$

Employing Equation 17 we have

$$\mathbf{H}^n\left(\frac{1}{2^n}\mathbf{W}^n\right) = \frac{1}{2^n}(\mathbf{B}^n\mathbf{W}^n + \mathbf{D}^n\mathbf{W}^n)$$

By substitution and applying the Kronecker mixed product rule we have

$$\begin{aligned}
\mathbf{B}^n\mathbf{W}^n &= \left[\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array}\right]\left[\begin{array}{cc} 1 & 1 \\ 1 & -1 \end{array}\right] \otimes \mathbf{B}^{n-1}\mathbf{W}^{n-1} \\
&+ \left[\begin{array}{cc} 0 & 0 \\ 1 & -1 \end{array}\right]\left[\begin{array}{cc} 1 & 1 \\ 1 & -1 \end{array}\right] \otimes \mathbf{D}^{n-1}\mathbf{W}^{n-1}
\end{aligned}$$

and

$$\mathbf{D}^n\mathbf{W}^n = \left[\begin{array}{cc} 1 & 1 \\ 0 & 0 \end{array}\right]\left[\begin{array}{cc} 1 & 1 \\ 1 & -1 \end{array}\right] \otimes \mathbf{D}^{n-1}\mathbf{W}^{n-1}$$

Defining $\mathbf{P}^n = \mathbf{B}^n\mathbf{W}^n$ and $\mathbf{Q}^n = \mathbf{D}^n\mathbf{W}^n$ we have

$$\mathbf{P}^n = \left[\begin{array}{cc} 1 & 1 \\ 1 & -1 \end{array}\right] \otimes \mathbf{P}^{n-1} + \left[\begin{array}{cc} 0 & 0 \\ 0 & 2 \end{array}\right] \otimes \mathbf{Q}^{n-1}$$

and

$$\mathbf{Q}^n = \left[\begin{array}{cc} 2 & 0 \\ 0 & 0 \end{array}\right] \otimes \mathbf{Q}^{n-1}$$

Substitution shows $\mathbf{P}^0 = [0]$ and $\mathbf{Q}^0 = [1]$ and the theorem is proven. □

Theorem 5.2 *The natural order Haar spectrum of a function can be transformed to the Walsh-Hadamard spectrum using the transform*

$$\begin{aligned}
\mathbf{T}^n &= \frac{1}{2^n}(\mathbf{P}^n + \mathbf{Q}^n) \\
\mathbf{P}^n &= \left[\begin{array}{cc} 2 & 2 \\ 2 & -2 \end{array}\right] \otimes \mathbf{P}^{n-1} + \left[\begin{array}{cc} 0 & 0 \\ 0 & 2 \end{array}\right] \otimes \mathbf{Q}^{n-1} \\
\mathbf{Q}^n &= \left[\begin{array}{cc} 2 & 0 \\ 0 & 0 \end{array}\right] \otimes \mathbf{Q}^{n-1} \\
\mathbf{P}^0 &= [0] \;\; \mathbf{Q}^0 = [1]
\end{aligned}$$

Proof: The proof is analogous to the proof of Theorem 5.1. □

These two theorems are the basis for fast transform procedures as illustrated in Figures 10 and 11. Note that while the structure of the transform is the same in each case, the butterflies in the Haar to Walsh direction are all scaled by a factor of 2. The structure is similar to the Walsh butterfly diagram presented earlier except the first butterfly in each group is replaced by the straight through passage of the two data values scaled by 2.

The above approach of combining transforms to go from one spectral domain to another can not be used when the Reed-Muller is involved because it is carried out over $GF(2)$ while the others are carried out over the integers. However, it was shown in [16] that the Reed-Muller spectral coefficients can be found by taking the modulo-2 of the absolute values of the arithmetic coefficients, a result that is not unexpected given the similar nature of the two transform

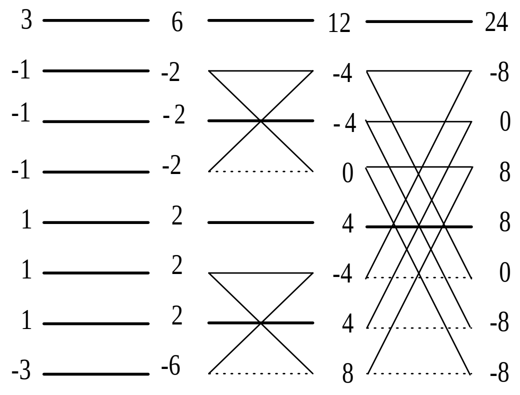


Figure 10. Example of Fast Transform from Walsh to Haar Speatrum

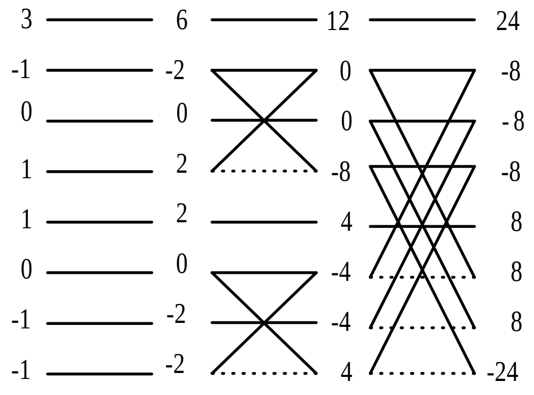


Figure 11. Example of Fast Transform from Haar to Walsh Spectrum

matrices. Hence, it is possible to express the transform from a domain to the Reed-Muller domain as a matrix multiplication followed by the taking of the modulo-2 of the absolute values of the result. For the domains considered here, the matrix multiplication can be implemented as a fast transform.

6 Decision Diagram Implementation

Reduced ordered binary decision diagrams (ROBDD) [6] are now a widely used data structure in many applications including VLSI CAD. Many extensions to the basic idea have been introduced including multi-terminal binary decision diagrams (MTBDD)[7] and edge-valued binary decision diagrams (EVBDD) [17]. (MTBDDs are also called *Algebraic DDs* (ADDs) [2].) We assume the reader is familiar with decision diagrams and refer anyone who is not to the extensive literature on the subject. In terms of notation, each nonterminal vertex v is labeled with a variable from *X* denoted $index(v)$ and has two outgoing edges denoted as $low(v)$ and $high(v)$. Each terminal vertex v is labeled by a $value(v)$ from *T* and has no outgoing edges, hence no successors. For an ROBDD, *T* contains 0 and 1 while for an MTBDD, *T* contains a set of integer values. An EVBDD has factors on certain edges and reduces the complexity of an MTBDD. Details can be found in [17].

It is insightful to present the *fast* DD transformation technique by first considering the case of transformations over trees. Although this is impractical for implementation, it does allow for the basis of the method to be easily explained. Unlike the BDD representation, a tree representation is complete and consists of $2^n - 1$ non-terminal vertices and 2^n terminal vertices.

In terms of graph operations, the one-variable Walsh transform replaces the subtree $low(v)$ with a subtree representing the sum of the original subtrees $low(v)$ and $high(v)$ and correspondingly replaces the subtree $high(v)$ with a subtree representing the difference of the original subtree $low(v)$ minus the subtree $high(v)$. This is the butterfly operation for the Walsh transform.

In terms of implementation, computing the sum (or difference) of two MTBDDs is typically performed as a recursive procedure similar to the classic `ite` operation used in most BDD package implementations [5].

The order of transformation of the tree is important. Initially, transforming the terminal vertices to the integers ± 1 allows for the non-terminal nodes at the bottom of the tree to be transformed. By successively applying the transformations in a bottom-up manner, the tree representing the Boolean function is transformed to a tree representing the Walsh spectrum in Hadamard order from left to right. Figure 16 illustrates this procedure. The vertices drawn with dashed lines indicate portions of the graph that have undergone the transformation.

This technique can be stated in a succinct form as a depth-first algorithm as given in Figure 12 where *Value()* is a function which returns the value of a terminal node, *Label()* is a function which returns the label of a nonterminal node, and *New_Terminal()* and *New_Nonterminal* are procedures which produce new nodes of the specified types.

The tree-based algorithm offers no computational advantage over the direct computation of the spectrum using matrix algebra since the size of the tree is exponential in the number of dependent function variables. In order to take advantage of shared topological isomorphic subgraphs as are found in reduced DD structures, the tree-based algorithm must be modified to account for the case when non-terminal variables are present along a path without subsequent valued level indices. This case never occurs in a tree but often does occur in a reduced DD. As an example, consider the case where the function $f = \overline{x}_1\overline{x}_3 + x_2\overline{x}_3 + x_1\overline{x}_2x_3$ is to be transformed to the Walsh domain. Figure 13 contains a di-

```
Walsh_Tree_Transform (f)
  if(f is a terminal) return
  Walsh_Tree_Transform(Low(f))
  Walsh_Tree_Transform(High(f))
  low_temp = Tree_Add(Low(f),High(f))
  High(f) = Tree_Sub(Low(f),High(f))
  Low(f) = low_temp

Tree_Add(g,h)
  if(g is a terminal)
    return(New_Terminal(Value(g)+Value(h))
  return(New_Nonterminal(Label(g),
               Tree_Add(Low(g),Low(h)),
               Tree_Add(High(g),High(h))))

Tree_Sub(g,h)
  if(g is a terminal)
    return(New_Terminal(Value(g)-Value(h))
  return(New_Nonterminal(Label(g),
                Tree_Sub(Low(g),Low(h)),
                Tree_Sub(High(g),High(h))))
```

Figure 12. Pseudo-code for Tree-based Walsh Transformation

agram representing the reduced BDD of this function with variable order $x_1 \prec x_2 \prec x_3$. As is easily seen, the path specified by $x_1 = 0$ and $x_3 = 0$ skips the intermediate variable x_2. However, in transforming the non-terminal vertex x_1, the absence of a vertex representing variable x_2 cannot be ignored and must be inherently considered.

An *absent* vertex is in effect a vertex v with $low(v) = high(v)$. Applying the Walsh butterfly to such a vertex replaces the subtree $low(v)$ by that subtree multiplied by 2 and the subtree $high(v)$ by the terminal value 0. Based on this observation the algorithm for transforming a BDD is given in Figure 14 where for ease of explanation we assume the labels of the nonterminals are ordered increasingly from the root of the BDD towards the terminals and *Label()* applied to a terminal yields a maximum value. *Twice()* doubles the terminal values of the argument BDD an operation which is easily accomplished if EVBDD are used.

The approach illustrated in Figure 14 for the Walsh case can be applied to the other transforms described above. The major complication is to identify when butterflys should or should not be applied. For example, the case for the tree-based computation of Haar spectra is addressed in Figure 15.

7 Concluding Remarks

Direct transformation amongst the Walsh, Haar, Arithmetic and Reed-Muller spectral domains has been considered. It has been shown that fast transform techniques are possible with the exception of transformation from the Reed-Muller domain. Implementation using decision diagram methods has been outlined.

Current work involves developing efficient generic *universal* program code for transforming from one domain to another. We are also considering how the transforms presented can be used to map spectral conditions, *e.g.* symmetry conditions, from one domain to another.

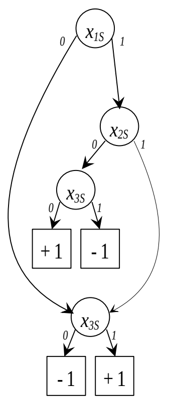


Figure 13. Reduced BDD for the Example Function

References

[1] N. Ahmed and K. R. Rao. *Orthogonal Transforms for Digital Signal Processing*. Springer-Verlag, New York, New York, 1975.

[2] R.I. Bahar, E.A. Frohm, C.M. Gaona, G.D. Hachtel, E. Macii, A. Pardo, and F. Somenzi. Algebraic decision diagrams and their application. In *Int'l Conf. on CAD*, pages 188–191, 1993.

[3] K. G. Beauchamp. *Applications of Walsh and Related Functions*. Academic Press, 1984.

[4] P. W. Besslich. *Spectral Techniques and Fault Detection, (M. G. Karpovsky, editor)*. Academic Press Publishers, Boston, Massachusetts, 1985.

[5] K.S. Brace, R.L. Rudell, and R.E. Bryant. Efficient implementation of a BDD package. In *Design Automation Conf.*, pages 40–45, 1990.

[6] R.E. Bryant. Graph-based algorithms for Boolean function manipulation. *IEEE Trans. on Comp.*, 35(8):677–691, 1986.

[7] E. Clarke, M. Fujita, P. McGeer, K.L. McMillan, J. Yang, and X. Zhao. Multi terminal binary decision diagrams: An efficient data structure for matrix representation. In *Int'l Workshop on Logic Synth.*, pages P6a:1–15, 1993.

[8] E.M. Clarke, M. Fujita, and X. Zhao. Hybrid decision diagrams - overcoming the limitations of MTBDDs and BMDs. In *Int'l Conf. on CAD*, pages 159–163, 1995.

[9] J. W. Cooley and J. W. Tukey. An algorithm for the machine calculation of complex Fourier series. *Math. Computation*, 19:297–301, 1965.

[10] A. Graham. *Kronecker Products and Matrix Calculus: with Applications*. Ellis Horwood Limited and John Wiley & Sons, New York, 1981.

[11] A. Haar. Zur Theorie der orthogonalen Funktionensysteme. *Math. Ann.*, 69:331–371, 1910.

[12] K. D. Heidtmann. Arithmetic spectrum applied to fault detection for combinational networks. *IEEE Trans. on Comp.*, 40(3), 1991.

[13] S. L. Hurst. *The Logical Processing of Digital Signals*. Crane-Russack, New York, 1978.

[14] S.L. Hurst, D.M.Miller, and J.C.Muzio. *Spectral Techniques in Digital Logic*. Academic Press Publishers, 1985.

[15] M. Karpovsky. *Finite Orthogonal Series in the Design of Digital Devices*. Wiley and JUP, 1976.

[16] S. K. Kumar and M. A. Breuer. Probabilistic aspects of Boolean switching functions via a new transform. *Journal of the ACM*, 28(3):502–520, 1981.

[17] Y.-T. Lai and S. Sastry. Edge-valued binary decision diagrams for multi-level hierarchical verification. In *Design Automation Conf.*, pages 608–613, 1992.

[18] D. E. Muller. Application of Boolean algebra to switching circuit design and error detection. *IRE Transactions*, 1:6–12, 1954.

[19] H. Rademacher. Einige Sätze über Reihen von allgemeinen orthogonal Funktionen. *Math. Ann.*, 87:112–138, 1922.

[20] I.S. Reed. A class of multiple-error-correcting codes and their decoding scheme. *IRE Trans. on Inf. Theory*, 3:6–12, 1954.

[21] J. L. Shanks. Computation of the fast Walsh-Fourier transform. *IEEE Trans. on Comp.*, 18:457–459, 1969.

[22] S.B.K. Vrudhula, M. Pedram, and Y.-T. Lai. Edge valued binary decision diagrams. In T. Sasao and M. Fujita, editors, *Representation of Discrete Functions*, pages 109–132. Kluwer Academic Publisher, 1996.

[23] J. S. Wallis. *Hadamard Matrices*. (Lecture Notes No. 292), Springer-Verlag, 1972.

[24] J. L. Walsh. A closed set of normal orthogonal functions. *American Journal of Mathematics*, 55:5–24, 1923.

```
Walsh_BDD_Transform (f)
  if(f is a terminal) return
  if(f has already been transformed)
    return
  Walsh_BDD_Transform(Low(f))
  Walsh_BDD_Transform(High(f))
  low_temp = BDD_Add(Low(f),High(f))
  High(f) = BDD_Sub(Low(f),High(f))
  Low(f) = low_temp

BDD_Add(g,h)
  if(g and h are terminals)
    return(New_Terminal(Value(g)+Value(h))
  if(Label(g)=Label(h))
    return(New_Nonterminal(Label(g),
           BDD_Add(Low(g),Low(h)),
           BDD_Add(High(g),High(h))))
  else if(Label(g)<Label(h))
    return(New_Nonterminal(Label(g),
           BDD_Add(Low(g),Twice(h)),
                          High(g))
  else return(New_Nonterminal(Label(h),
           BDD_Add(Low(h),twice(g)),
                          High(h))

BDD_Sub(g,h)
  if(g and h are terminals)
    return(New_Terminal(Value(g)-Value(h))
  if(Label(g)=Label(h))
    return(New_Nonterminal(Label(g),
           BDD_Sub(Low(g),Low(H)),
           BDD_Sub(High(g),High(h))))
  else if(Label(g)<Label(h))
    return(New_Nonterminal(Label(g),
           BDD_Sub(Low(g),Twice(h)),
                          High(g))
  else return(New_Nonterminal(Label(h),
           BDD_Sub(Low(h),Twice(g)),
                          High(h))
```

Figure 14. Pseudo-code for BDD-based Walsh Transformation

```
Haar_Tree_Transform (f)
   if(f is a terminal) return
   Haar_Tree_Transform(Low(f))
   Haar_Tree_Transform(High(f))
   low_branch = Low(f)
   while(low_branch not a terminal)
     low_branch = Low(low_branch)
   high_branch = High(f)
   while(high_branch not a terminal)
     high_branch = Low(high_branch)
   temp_value = Value(low_branch)+Value(high_branch)
   Value(high_branch) = Value(low_branch)-Value(high_branch)
   Value(low_branch) = temp_value
```

Figure 15. Pseudo-code for the Tree-based Haar Transformation

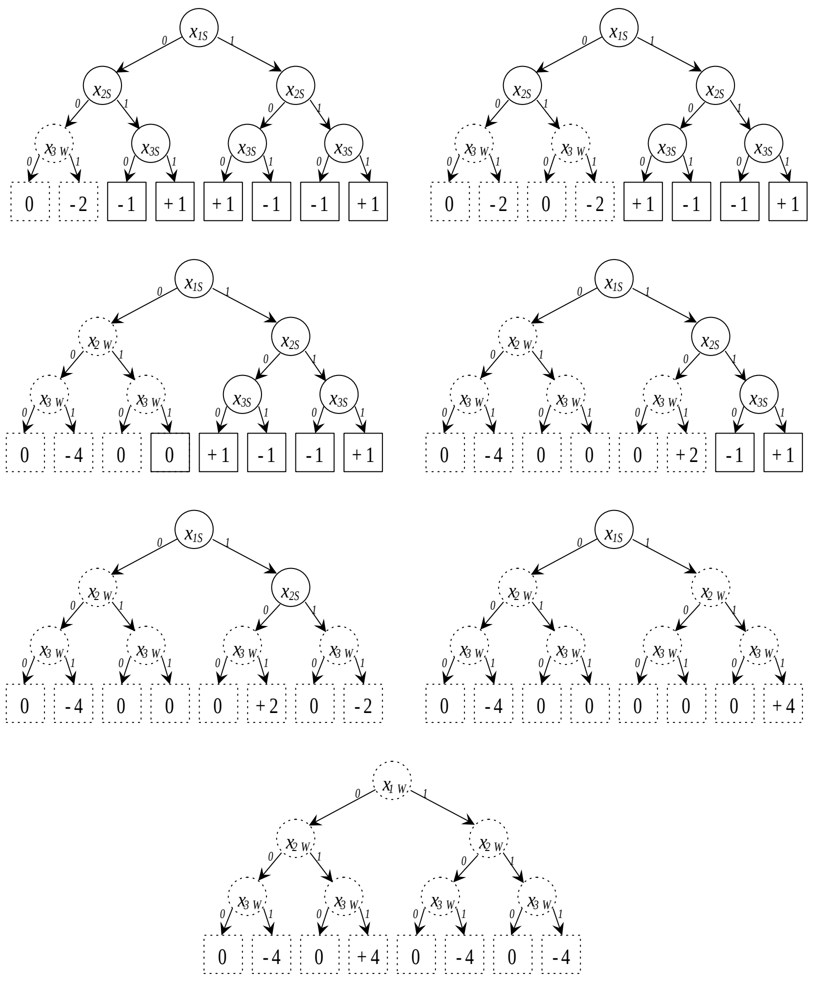


Figure 16. Example of a Tree-based Walsh Transformation