PAPER *Special Section on Multiple-Valued Logic and VLSI Computing*

Construction of Ternary Bent Functions by FFT-Like Permutation Algorithms*

Radomir S. STANKOVIĆ[†a)], ***Member***, **Milena STANKOVIĆ**[††], **Claudio MORAGA**[†††,††††], ***and*** **Jaakko T. ASTOLA**[†††††], ***Nonmembers***

SUMMARY Binary bent functions have a strictly specified number of non-zero values. In the same way, ternary bent functions satisfy certain requirements on the elements of their value vectors. These requirements can be used to specify six classes of ternary bent functions. Classes are mutually related by encoding of function values. Given a basic ternary bent function, other functions in the same class can be constructed by permutation matrices having a block structure similar to that of the factor matrices appearing in the Good-Thomas decomposition of Cooley-Tukey Fast Fourier transform and related algorithms.

key words: *ternary functions, bent functions, Vilenkin-Chrestenson transform, fast Fourier transform, permutation matrices*

Various approaches to the generalization of the concept of bentness from binary to p-valued functions are a subject of research by many authors, see, for example, [1], [4], [6], [7], [13], [14], [16], [17]. The reason for study generalized bent functions is twofold. First, they are interesting mathematical objects offering many challenging problems some of them analogous to these in the case of binary bent functions, as generation, characterization, classification, and counting bent functions. Although initial definitions of generalized bent functions are rather straightforward generalizations of the concepts from the theory of binary bent functions, a further study of them immediately leads to drastically different properties, which raises new challenges, as will be pointed out in the following discussions of ternary bent functions. Then, there are some practical applications of ternary bent functions [6], [18]. Further, there are interesting relationships to certain important concepts in mathematics and engineering [2], [3], [14], [19].

Research reported in this paper is based upon the two following properties of bent functions and related observations. First, in the binary case, there is a precisely determined number of non-zero values a bent function can take [1], [17]. It is similar for ternary bent functions, but a particular distribution of function values in the value vector is required.

Second, in the binary case, given a bent function f, the function f_1 obtained by adding any affine function to f is also bent. Recall that the affine functions are defined as linear functions and their complements. Thus, adding a linear combination of variables and the constant 1 to a binary bent function preserves the bentness [17]. It is similar for ternary functions, however, in this case, certain linear combinations of variables as well as their squares and products of variables can be added to a bent function while preserving the bentness [6]. Moreover, these linear combinations have coefficients in the set $\{0, 1, 2\}$. Thus, the multiplication by 2 of variables and their squares as well as products of variables is allowed under certain restrictions pointed out below and further, any of the constants 1 and 2 can be added. This includes multiplication of the given bent function by 2.

We use the property that functions in a set of functions with the same distribution of function values are mutually related by permutations of elements of their value vectors to construct new bent functions from a given bent function viewed as a representative of the considered set of bent functions. We show that permutation matrices relating these functions can be derived by an analogy to the matrices determining steps in the Cooley-Tukey Fast Fourier transform (FFT) and related algorithms. In this way, given a bent function f with a specified distribution of its values, new bent functions with the same distribution can be constructed by the multiplication of the value vector $\mathbf{F}$ of f with these FFT-like permutation matrices.

1. Ternary Bent Functions

Ternary bent functions are a subset of ternary functions $f : \{0, 1, 2\}^n \rightarrow \{0, 1, 2\}$, where n is the number of variables, having flat Vilenkin-Chrestenson spectra [6]. In this context, flat spectrum means the spectrum whose elements have equal absolute values $3^{n/2}$ [8]. This can be viewed as the minimal maximum value a Vilenkin-Chrestenson coefficient can take. The largest absolute value of a Vilenkin-Chrestenson coefficient is 3^n and it is for the linear ternary functions, since they are isomorphic with the Vilenkin-Chrestenson functions, and then this property follows from the orthogonality. In this case, all other coefficients are 0. The minimal maximum value is obtained when it is equally distributed over all the coefficients and the spectrum is flat.

Manuscript received September 7, 2020.
Manuscript publicized April 1, 2021.

†The author is with the Mathematical Institute of SASA, Belgrade, Serbia.
††The author is with the Department of Computer Science, Faculty of Electronic Engineering, Niš, Serbia.
†††The author is with the Faculty of Computer Science, Technical University of Dortmund, Dortmund, Germany.
††††The author is with the Department of Informatics, Technical University "Federico Santa María," Valparaíso, Chile.
†††††The author is with the Department of Signal Processing, Tampere University of Technology, Tampere, Finland.

*This paper was presented at ISMVL 2020.

a) E-mail: Radomir.Stankovic@gmail.com

DOI: 10.1587/transinf.2020LOP0006

To achieve the flatness, some restrictions on the values a function can take are necessary imposed. This leads to the concepts of composition and distribution of function values in the function vectors of bent functions.

In the case of binary bent functions, the number of non-zero values is either $2^{n-1} - 2^{n/2-1}$ or $2^{n-1} + 2^{n/2-1}$, where n is the number of variables. Due to this, all bent functions for a given n can be split into two sets σ_1 and σ_2 of equal cardinalities consisting of bent functions with equal number of non-zero values. Given a set of bent functions, either σ_1 or σ_2, the other set consists of bent functions that are their complements. Thus, if we generate a set, the other is obtained by complementing its elements, which can be viewed as different encoding of function values $(0, 1) \rightarrow (1, 0)$. These considerations can be extended to ternary bent functions.

In the value vector of a ternary bent function, except for $n = 1$, all three values, conveniently denoted as $\{0, 1, 2\}$, should be present. In this case, certain requirements on the number of different values a bent function can take from the set $\{0, 1, 2\}$ must be satisfied.

For a ternary bent function, the composition is defined as a triple $C = (c_0, c_1, c_2)$ where c_i, $i = \{0, 1, 2\}$, represents how many times the value i appears in its value vector. Thus, two bent functions with the same composition differ mutually up to the permutation of elements in their value vectors. The distribution of values in the value vector of a bent function $D = (d_1, d_2, d_3)$ is defined as composition with permutation of its elements allowed. Therefore, two bent functions with the same distribution differ up to the encoding of their values.

Compositions specify how many time each value (of the ordered value set) appear in a value vector. Distributions specify how many times different values (of the value set) appear in a value vector, but do not specify which values are repeated. This is why distributions, in a way, represent all possible permutations of compositions. They represent the "structure" of the value vectors in a rather "abstract" general way.

For instance, a single variable ternary bent function must have two identical values, while the third value is different. This is expressed as the distribution $D = (0, 1, 2)$ for $n = 1$. This means, a value appears twice, another a single time, and the third value is absent. Unlike the composition, the distribution does not specify how many times a particular value appears. Therefore, functions with different compositions can have the same distribution.

For the case $n = 2$, a possible distribution of values is $(5, 2, 2)$ meaning that all three values must be present in the value vector, two of them repeating two times, while the third value repeats 5 times. Adding a constant to a ternary bent function, preserves its bentness, but, changes the composition it has. For instance, given a function with composition $(5, 2, 2)$, adding the constant 1 changes its composition into $(2, 5, 2)$, while adding the constant 2 results into a function with the composition $(2, 2, 5)$. These three functions with different compositions have the same distribution of function values. Another possible distribution for ternary bent functions in two variables is $(1, 4, 4)$.

For $n = 3$, bent functions have the distribution $(12, 9, 6)$. By experiments we could not find a ternary bent function in three variables with other distribution and, therefore, we conjecture that for odd n, there is a single distribution of function values in the value vectors of bent functions. The functions $f(x_1, x_2, x_3) = x_1x_2 \oplus x_3^2$ and $f(x_1, x_2, x_3) = x_1^2 \oplus x_2^2 \oplus x_3^2$ have this distribution of function values.

For $n = 4$, there are again two distributions $(33, 24, 24)$ and $(21, 30, 30)$. Examples of functions with these distributions are $f(x_1, x_2, x_3, x_4) = x_1x_2 \oplus x_3x_4$ and $f(x_1, x_2, x_3, x_4) = x_1^2 \oplus x_2^2 \oplus x_3^2 \oplus x_4^2$, respectively.

For $n = 5$, the distribution is $(72, 81, 90)$ for both basic bent functions, the sum of disjoint products of variables with the square of a variable, and the sum of squares of variables. Thus, representative functions are $f(x_1, x_2, x_3, x_4, x_5) = x_1x_2 \oplus x_3x_4 \oplus x_5^2$ and $f(x_1, x_2, x_3, x_4, x_5) = x_1^2 \oplus x_2^2 \oplus x_3^3 \oplus x_4^2 \oplus x_5^2$.

For $n = 6$, there are two distributions $(225, 252, 252)$, and $(261, 234, 234)$. The example for the first distribution is the function $f(x_1, x_2, x_3, x_4, x_5, x_6) = x_1x_2 \oplus x_3x_4 \oplus x_5x_6$. For the other distribution an example is $f(x_1, x_2, x_3, x_4, x_5, x_6) = x_1^2 \oplus x_2^2 \oplus x_3^2 \oplus x_4^2 \oplus x_5^2 \oplus x_6^2$.

For $n = 7$, there is a single distribution $(702, 729, 756)$. This is the same distribution for both basic bent functions, the sum of disjoint products of variables with the square of a variable, and the sum of squares of variables.

We conjecture that for ternary bent functions there are a single and two distributions for n odd and even, respectively. The reason for the existence of two distributions for n even is to have 6 possible encodings in the cases when two values appear an equal number of times.

In [12], it is shown that for n even and p prime, the value distribution of a bent function $f : Z_p^n \rightarrow Z_p$ is $(b_0, b_1, \ldots, b_{p-1})$, where

$$\begin{aligned} b_0 &= p^{n-1} \pm (p-1)p^{\frac{n}{2}-1}, \\ b_k &= p^{n-1} \mp p^{\frac{n}{2}-1}, \end{aligned}$$

for $k = 1, 2, \ldots, p-1$, or its cyclic shift. Here the $\pm$ signs are taken correspondingly. The above examples satisfy this formula for $p = 3$. In [12], the case for n odd is not considered.

For n odd, from the above examples follows that the general formula for distribution of function values in the function vectors of ternary bent functions, i.e., for $p = 3$, is

$$\begin{aligned} b_0 &= 3^{n-1}, \\ b_k &= 3^{n-1} \pm 3^{\frac{(n-1)}{2}}. \end{aligned}$$

It can be observed that for a given n ternary bent functions with the same distribution of ternary values mutually differ in the position of particular values in the value vector. Therefore, relationships between functions with equal distributions can be expressed by permutation matrices. Ternary bent functions for any n can be split into six classes with respect to the encoding. Functions in different classes differ up to the encoding of their values. If we construct ternary bent functions in a class, the corresponding functions in the other five classes can be derived by encoding. For each distribution a representative function can be selected. For a distribution, and n even, the function with the quadratic form as a generalized Reed-Muller expression is selected as the representative. Thus, this functional expression is the sum of disjoint products of pairs of variables. For n odd, we select this function with the square of a variable added, and the function that is represented by the sum of squares of all variables for two possible distributions.

2. Fast Fourier Transform and Bent Functions

As noticed above, bent functions are alternatively defined as functions with flat spectra. For functions defined on finite domains, the spectra can be computed by using the fast computing algorithms. For completeness of the presentation and also for clarifying the analogy between the permutation matrices relating bent functions possessing equal distributions with the Fast Fourier transform (FFT), in this section we present the essential ideas behind these algorithms.

The spectral transforms for p-valued functions in n-variables are defined by $(p^n \times p^n)$ matrices. If for a transform, the transform matrix can be factorized into the product of n sparse $(p^n \times p^n)$ matrices, then a fast algorithm can be formulated. In particular, if the transform matrix has the Kronecker product structure, then we speak of the Good-Thomas factorization and the corresponding algorithm is called the Cooley-Tukey FFT [5], [11], [15]. The algorithm consists of n steps, in each of them performing the partial spectral transform with respect to the corresponding variable [5]. Thus, in each step computations to be performed are determined by the basic $(p \times p)$ transform matrix. In this factorization, the matrix defining the i-th step of the algorithm is the Kronecker product of the basic transform matrix at the i-th position and the $(p \times p)$ identity matrices in all other positions.

The Vilenkin-Chrestenson transform that is used to check bentness of ternary functions in n variables is defined by a $(3^n \times 3^n)$ transform matrix with the Kronecker product structure

$$\mathbf{V}(n) = \bigotimes_{i=1}^{n} (\mathbf{V}(1)), \quad \mathbf{V}(1) = \begin{bmatrix} 1 & 1 & 1 \\ 1 & e_2 & e_1 \\ 1 & e_1 & e_2 \end{bmatrix},$$

where $e_1 = -\frac{1}{2}(1 - i\sqrt{3})$, $e_2 = -\frac{1}{2}(1 + i\sqrt{3})$, and $\otimes$ denotes the Kronecker product.

In Good-Thomas factorization,

$$\mathbf{V}(n) = \prod_{i=1}^{n} \mathbf{C}_i, \quad \mathbf{C}_i = \bigotimes_{j=1}^{n} \mathbf{A}_j, \quad \mathbf{A}_j = \begin{cases} \mathbf{V}(1), & j = i, \\ \mathbf{I}(1), & j \neq i. \end{cases}$$

and $\mathbf{I}(1)$ is the (3×3) identity matrix.

Example 1: In Good-Thomas factorization, the Vilenkin-Chrestenson transform for functions in two variables can be factorized as $\mathbf{V}(2) = \mathbf{C}_1\mathbf{C}_2$ where

$$\mathbf{C}_1 = \mathbf{V}(1) \otimes \mathbf{I}(1), \quad \mathbf{C}_2 = \mathbf{I}(1) \otimes \mathbf{V}(1).$$

The flow-graph of the corresponding fast computing algorithm is shown by black lines in the figures below.

3. Classes of Ternary Bent Functions

For $n = 1$ there are 18 bent functions. Their value vectors have two identical values, while the third value is different. Thus, the distribution of values is $(0, 1, 2)$. The compositions of their value vectors are different, but the distribution is the same for all these functions. We arrange these functions into 6 classes as in Table 1. A class consists of functions with the same composition, and classes are related by encoding as in Table 2.

It follows, that there is a single function $f(x) = x^2$ where the square is modulo 3, which produces $F = [0, 1, 1]^T$, as the first function in the class c_1. The other two functions are obtained by the cyclic shift for 1-place. The other 5 classes are obtained from the class c_1 as explained in Table 1.

Classes c_2, c_3, c_4, c_5, and c_6 can be derived from c_1 by encoding of function values as shown in Table 2.

Functions within a class are mutually related by permutations since they have the same compositions. The possible (3×3) permutation matrices converting these functions to each other are

$$\mathbf{Q}_1 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{Q}_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix},$$

$$\mathbf{X}_1 = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \quad \mathbf{N}_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}.$$

If elements of the class c_1 are denoted as f_1, f_2, and f_3, then the conversion among them is as specified in Table 3. Transposed matrices give the transition in the opposite directions.

Table 1 Classes of ternary bent functions for $n = 1$.

Class	Functions	Composition	Relationship
c_1	[0,1,1], [1,0,1], [1,1,0]	(1,2,0)	$c_1 = x^2$, cyclic shift
c_2	[0,2,2], [2,0,2], [2,2,0]	(1,0,2)	$c_2 = 2c_1$
c_3	[1,0,0], [0,1,0], [0,0,1]	(2,1,0)	$c_3 = 2c_1 \oplus 1$
c_4	[2,0,0], [0,2,0], [0,0,2]	(2,0,1)	$c_4 = c_1 \oplus 2$
c_5	[2,1,1], [1,2,1], [1,1,2]	(0,2,1)	$c_5 = 2c_1 \oplus 2$
c_6	[1,2,2], [2,1,2], [2,2,1]	(0,1,2)	$c_6 = c_1 \oplus 1$

Table 2 Encoding of function values for ternary bent functions.

Classes	Encoding
c_1 and c_2	$1 \leftrightarrow 2$
c_1 and c_3	$0 \leftrightarrow 1$
c_1 and c_4	$0 \rightarrow 2$ and $1 \rightarrow 0$ and $2 \rightarrow 1$
c_1 and c_5	$0 \leftrightarrow 2$
c_1 and c_6	$0 \rightarrow 1$ and $1 \rightarrow 2$ and $2 \rightarrow 0$

Table 3 Conversion between the elements of a class.

Matrix	Functions
$\mathbf{Q}_1$	$f_1 \rightarrow f_2$
$\mathbf{Q}_2$	$f_2 \rightarrow f_3$
$\mathbf{X}_1$	$f_1 \rightarrow f_2, f_2 \rightarrow f_3$
$\mathbf{X}_1^T$	$f_1 \rightarrow f_3$
$\mathbf{N}_1$	$f_1 \rightarrow f_3$

Functional expressions for these functions are

$$f_1 = x^2, \quad f_2 = 1 \oplus x \oplus x^2, \quad f_3 = 1 \oplus 2x \oplus x^2.$$

Comparing the functional expressions for the initial functions and functions that are produced by the multiplication with the considered basic permutation matrices it follows that they perform the following substitutions.

The matrix $\mathbf{Q}_1$ performs the substitution $x \rightarrow 2x \oplus 1$. When applied to the function with the function vector $[0, 1, 1]$ of f_1, it produces the function vector of f_2 which is $[1, 0, 1]$. Comparing the functional expressions for $f_1 = x^2$ and $f_2 = 1 \oplus x \oplus x^2$, we see the substitution that is performed by $\mathbf{Q}_1$. The matrix $\mathbf{Q}_2$ performs the substitution $x \rightarrow 2x$. The matrix $\mathbf{X}_1$ performs the addition of the constant 2 modulo 3, i.e., the substitution $x \rightarrow x \oplus 2$. The transpose of this matrix $\mathbf{X}_1^T$ performs addition of the constant 1 modulo 3, i.e., the substitution $x \rightarrow x \oplus 1$. The complement for p-valued variables is defined as $(p - 1 - x_i) \bmod 3$, which for ternary variables is $(2 - x_i) \bmod 3$. In matrix notation, the complement is performed by the matrix $\mathbf{N}_1$. Notice that $\mathbf{N}_1\mathbf{Q}_1 = \mathbf{Q}_2\mathbf{N}_1 = \mathbf{X}_1$. This is the substitution $x \rightarrow 2x \oplus 2$.

It should be noticed that these (3×3) permutation matrices perform spectral invariant operations for $n = 1$ and, therefore, preserve bentness. In other words, for a given bent function f, 5 other bent functions can be derived, $f \oplus 1$, $2f$, $f \oplus 2$, $2f \oplus 1$, and $2f \oplus 2$. These functions are related by encoding.

4. Ternary Bent Functions in Two Variables

There are 486 ternary bent function in two variables as determined by a computer search. Two thirds of these functions, i.e., 324 functions, have composition of values either $(5, 2, 2)$, $(2, 5, 2)$, or $(2, 2, 5)$, while the remaining 162 functions have the compositions $(1, 4, 4)$, $(4, 1, 4)$, or $(4, 4, 1)$. Therefore, ternary bent functions in two variables can be split into two sets with respect to their distributions, since compositions with permuted values belong to the same distribution D. These are sets of functions with distributions $(5, 2, 2)$ and $(1, 4, 4)$, respectively.

Example 2: The functions $f_1 = x_1x_2$ and $f_2 = x_1^2 \oplus x_2^2$ have compositions $(5, 2, 2)$ and $(1, 4, 4)$, respectively, since their value vectors are $\mathbf{F}_{x_1x_2} = [0, 0, 0, 0, 1, 2, 0, 2, 1]^T$, and $\mathbf{F}_{x_1^2 \oplus x_2^2} = [0, 1, 1, 1, 2, 2, 1, 2, 2]^T$.

We use functions f_1 and f_2 in Example 2 as basic bent functions representing sets of bent functions with compositions $(5, 2, 2)$ and $(1, 4, 4)$ and related distributions. As in the case of ternary bent functions for $n = 1$, we split the set of all 486 ternary bent functions in two variables into 6 classes. Each class consists of 81 functions such that $2/3$ of functions, i.e., 54 functions, share the distribution $(5, 2, 2)$, while $1/3$ of functions, 27 of them, have the distribution $(1, 4, 4)$. We further deal with the class of 81 functions derived from the representatives of both distributions $f_1 = x_1x_2$ and $f_2 = x_1 \oplus x_2$. The other 5 classes, each with 81 functions, are obtained by encoding shown in Table 2. In this way we can construct all $81 + 5 \cdot 81 = 486$ ternary bent functions in two variables. In order to construct these functions from f_1 and f_2, we recall the following. In the binary case, adding affine functions to a bent function produces functions that are also bent. In the ternary case, it is possible to add variables, squares of variables, and their linear combinations. It is important to notice that we should stay within the sets of functions with the same distribution, and due to this, adding $x_1^2 \oplus x_2^2$ and $2x_1^2 \oplus 2x_2^2$ to f_1 is not allowed, unlike adding $2x_1^2 \oplus x_2^2$ and $x_1^2 \oplus 2x_2^2$. The reason is that these terms viewed as particular ternary functions belong to the distribution corresponding to f_2. Similarly, adding the terms that will convert f_2 into a function with the distribution for f_1 is not allowed.

Table 4 and Table 5 show the 54 functions derived from the basic function $f_1 = x_1x_2$, and 27 functions derived from $f_2 = x_1^2 \oplus x_2^2$, respectively. The meaning of matrices in the rightmost column is explained below.

5. Permutation Matrices

For a given bent function with a particular distribution of values, the addition of terms as specified in Tables 4 and 5, produces another bent function with the same distribution of function values. Therefore, these two functions differ in permutation of elements of their value vectors. In other words, the addition of terms to f_1 and f_2, respectively, which can be seen in the left part of these tables, can be expressed by permutation matrices shown in the right part.

We consider three types of permutation matrices, all of them related to the factor matrices describing steps of FFT algorithms. These matrices are naturally related to spectral invariant operations, since permute but do not change values of spectral coefficients. At the same time, as in the case of application of spectral invariant operations to bent functions, multiplication of function vectors by these permutation matrices results in adding particular terms to functional expressions of processed functions. These are terms which are allowed to be added to a bent function and preserve its bentness.

5.1 Kronecker Product Representable Matrices

The addition of x_1 to the function in two variables $f_1 = x_1x_2$ can be expressed as multiplication of the value vector F_1 by the permutation matrix

$$\mathbf{P}_1 = \mathbf{I}(1) \otimes \mathbf{X}_1^T,$$

where $\mathbf{I}(1)$ is the (3×3) identity matrix and $\mathbf{X}_1^T$ is as defined

Table 4 Bent functions with distribution $(5, 2, 2)$.

$F = x_1x_2$	$(5, 2, 2)$
$F = x_1x_2 \oplus x_1$	$\mathbf{P}_1$
$F = x_1x_2 \oplus x_2$	$\mathbf{P}_2$
$F = x_1x_2 \oplus 2x_1$	$\mathbf{P}_1\mathbf{P}_1$
$F = x_1x_2 \oplus 2x_2$	$\mathbf{P}_2\mathbf{P}_2$
$F = x_1x_2 \oplus x_1 \oplus x_2 \oplus 1$	$\mathbf{P}_1\mathbf{P}_2$
$F = x_1x_2 \oplus 2x_1 \oplus x_2 \oplus 2$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_2$
$F = x_1x_2 \oplus x_1 \oplus 2x_2 \oplus 2$	$\mathbf{P}_1\mathbf{P}_2\mathbf{P}_2$
$F = x_1x_2 \oplus 2x_1 \oplus 2x_2 \oplus 1$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_2\mathbf{P}_2$
$F = x_1x_2 \oplus (x_1)^2$	$\mathbf{P}_{1,2}$
$F = x_1x_2 \oplus x_1 \oplus (x_1)^2$	$\mathbf{P}_1\mathbf{P}_{1,2}$
$F = x_1x_2 \oplus x_2 \oplus (x_1)^2 \oplus 2$	$\mathbf{P}_1\mathbf{P}_2\mathbf{P}_{1,2}$
$F = x_1x_2 \oplus 2x_1 \oplus (x_1)^2$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_{1,2}$
$F = x_1x_2 \oplus 2x_2 \oplus (x_1)^2 \oplus 2$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{1,2}$
$F = x_1x_2 \oplus x_1 \oplus x_2 \oplus (x_1)^2$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{1,2}$
$F = x_1x_2 \oplus 2x_1 \oplus x_2 \oplus (x_1)^2 \oplus 1$	$\mathbf{P}_2\mathbf{P}_{1,2}$
$F = x_1x_2 \oplus x_1 \oplus 2x_2 \oplus (x_1)^2 \oplus 1$	$\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{1,2}$
$F = x_1x_2 \oplus 2x_1 \oplus 2x_2 \oplus (x_1)^2$	$\mathbf{P}_1\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{1,2}$
$F = x_1x_2 \oplus (x_2)^2$	$\mathbf{P}_{2,2}$
$F = x_1x_2 \oplus x_1 \oplus (x_2)^2 \oplus 2$	$\mathbf{P}_1\mathbf{P}_2\mathbf{P}_{2,2}$
$F = x_1x_2 \oplus x_2 \oplus (x_2)^2$	$\mathbf{P}_2\mathbf{P}_{2,2}$
$F = x_1x_2 \oplus 2x_1 \oplus (x_2)^2 \oplus 2$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{2,2}$
$F = x_1x_2 \oplus 2x_2 \oplus (x_2)^2$	$\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{2,2}$
$F = x_1x_2 \oplus x_1 \oplus x_2 \oplus (x_2)^2$	$\mathbf{P}_1\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{2,2}$
$F = x_1x_2 \oplus 2x_1 \oplus x_2 \oplus (x_2)^2 \oplus 1$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_{2,2}$
$F = x_1x_2 \oplus x_1 \oplus 2x_2 \oplus (x_2)^2 \oplus 1$	$\mathbf{P}_1 \cdot \mathbf{P}_{2,2}$
$F = x_1x_2 \oplus 2x_1 \oplus 2x_2 \oplus (x_2)^2$	$\mathbf{P}_1 \cdot \mathbf{P}_1 \cdot \mathbf{P}_2 \cdot \mathbf{P}_{2,2}$
$F = x_1x_2 \oplus 2(x_1)^2$	$\mathbf{P}_{1,2}\mathbf{P}_{1,2}$
$F = x_1x_2 \oplus x_1 \oplus 2(x_1)^2$	$\mathbf{P}_1\mathbf{P}_{1,2}\mathbf{P}_{1,2}$
$F = x_1x_2 \oplus x_2 \oplus 2(x_1)^2 \oplus 1$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_2\mathbf{P}_{1,2}\mathbf{P}_{1,2}$
$F = x_1x_2 \oplus 2x_1 \oplus 2(x_1)^2$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_{1,2}\mathbf{P}_{1,2}$
$F = x_1x_2 \oplus 2x_2 \oplus 2(x_1)^2 \oplus 1$	$\mathbf{P}_1\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{1,2}\mathbf{P}_{1,2}$
$F = x_1x_2 \oplus x_1 \oplus x_2 \oplus 2(x_1)^2 \oplus 2$	$\mathbf{P}_2\mathbf{P}_{1,2}\mathbf{P}_{1,2}$
$F = x_1x_2 \oplus 2x_1 \oplus x_2 \oplus 2(x_1)^2$	$\mathbf{P}_1\mathbf{P}_2\mathbf{P}_{1,2}\mathbf{P}_{1,2}$
$F = x_1x_2 \oplus x_1 \oplus 2x_2 \oplus 2(x_1)^2$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{1,2}\mathbf{P}_{1,2}$
$F = x_1x_2 \oplus 2x_1 \oplus 2x_2 \oplus 2(x_1)^2 \oplus 2$	$\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{1,2}\mathbf{P}_{1,2}$
$F = x_1x_2 \oplus 2(x_2)^2$	$\mathbf{P}_{2,2}\mathbf{P}_{2,2}$
$F = x_1x_2 \oplus x_1 \oplus 2(x_2)^2 \oplus 1$	$\mathbf{P}_1\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{2,2}\mathbf{P}_{2,2}$
$F = x_1x_2 \oplus x_2 \oplus 2(x_2)^2$	$\mathbf{P}_2\mathbf{P}_{2,2}\mathbf{P}_{2,2}$
$F = x_1x_2 \oplus 2x_1 \oplus 2(x_2)^2 \oplus 1$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_2\mathbf{P}_{2,2}\mathbf{P}_{2,2}$
$F = x_1x_2 \oplus 2x_2 \oplus 2(x_2)^2$	$\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{2,2}\mathbf{P}_{2,2}$
$F = x_1x_2 \oplus x_1 \oplus x_2 \oplus 2(x_2)^2 \oplus 2$	$\mathbf{P}_1\mathbf{P}_{2,2}\mathbf{P}_{2,2}$
$F = x_1x_2 \oplus 2x_1 \oplus x_2 \oplus 2(x_2)^2$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{2,2}\mathbf{P}_{2,2}$
$F = x_1x_2 \oplus x_1 \oplus 2x_2 \oplus 2(x_2)^2$	$\mathbf{P}_1\mathbf{P}_2\mathbf{P}_{2,2}\mathbf{P}_{2,2}$
$F = x_1x_2 \oplus 2x_1 \oplus 2x_2 \oplus 2(x_2)^2 \oplus 2$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_{2,2}\mathbf{P}_{2,2}$
$F = 2(x_1)^2 \oplus (x_2)^2$	$\mathbf{P}_{1,2}\mathbf{P}_{2,2}$
$F = x_1 \oplus 2(x_1)^2 \oplus (x_2)^2 \oplus 2$	$\mathbf{P}_2\mathbf{P}_{1,2}\mathbf{P}_{2,2}$
$F = x_2 \oplus 2(x_1)^2 \oplus (x_2)^2 \oplus 1$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_{1,2}\mathbf{P}_{2,2}$
$F = x_1 \oplus x_2 \oplus 2(x_1)^2 \oplus (x_2)^2$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_2\mathbf{P}_{1,2}\mathbf{P}_{2,2}$
$F = 2x_1 \oplus 2(x_1)^2 \oplus (x_2)^2 \oplus 2$	$\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{1,2}\mathbf{P}_{2,2}$
$F = 2x_2 \oplus 2(x_1)^2 \oplus (x_2)^2 \oplus 1$	$\mathbf{P}_1\mathbf{P}_{1,2}\mathbf{P}_{2,2}$
$F = 2x_1 \oplus x_2 \oplus 2(x_1)^2 \oplus (x_2)^2$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{1,2}\mathbf{P}_{2,2}$
$F = x_1 \oplus 2x_2 \oplus 2(x_1)^2 \oplus (x_2)^2$	$\mathbf{P}_1\mathbf{P}_2\mathbf{P}_{1,2}\mathbf{P}_{2,2}$
$F = 2x_1 \oplus 2x_2 \oplus 2(x_1)^2 \oplus (x_2)^2$	$\mathbf{P}_1\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{1,2}\mathbf{P}_{2,2}$

Table 5 Bent functions with distribution $(1, 4, 4)$.

$(x_1)^2 \oplus (x_2)^2$	$(1, 4, 4)$
$x_1 \oplus (x_1)^2 \oplus (x_2)^2 \oplus 1$	$\mathbf{P}_2\mathbf{P}_2$
$x_2 \oplus (x_1)^2 \oplus (x_2)^2 \oplus 1$	$\mathbf{P}_1\mathbf{P}_1$
$x_1 \oplus x_2 \oplus (x_1)^2 \oplus (x_2)^2 \oplus 2$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_2\mathbf{P}_2$
$2x_1 \oplus (x_1)^2 \oplus (x_2)^2 \oplus 1$	$\mathbf{P}_2$
$2x_2 \oplus (x_1)^2 \oplus (x_2)^2 \oplus 1$	$\mathbf{P}_1$
$2x_1 \oplus x_2 \oplus (x_1)^2 \oplus (x_2)^2 \oplus 2$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_2$
$x_1 \oplus 2x_2 \oplus (x_1)^2 \oplus (x_2)^2 \oplus 2$	$\mathbf{P}_1\mathbf{P}_2\mathbf{P}_2$
$2x_1 \oplus 2x_2 \oplus (x_1)^2 \oplus (x_2)^2 \oplus 2$	$\mathbf{P}_1\mathbf{P}_2$
$2x_1x_2 \oplus (x_1)^2 \oplus 2(x_2)^2$	$\mathbf{P}_{2,2}$
$2x_1x_2 \oplus x_1 \oplus (x_1)^2 \oplus 2(x_2)^2 \oplus 2$	$\mathbf{P}_1\mathbf{P}_2\mathbf{P}_{2,2}$
$2x_1x_2 \oplus x_2 \oplus (x_1)^2 \oplus 2(x_2)^2 \oplus 1$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_2\mathbf{P}_{2,2}$
$2x_1x_2 \oplus 2x_1 \oplus (x_1)^2 \oplus 2(x_2)^2 \oplus 2$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_2\mathbf{P}_{2,2}$
$2x_1x_2 \oplus 2x_2 \oplus (x_1)^2 \oplus 2(x_2)^2 \oplus 1$	$\mathbf{P}_1\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{2,2}$
$2x_1x_2 \oplus x_1 \oplus x_2 \oplus (x_1)^2 \oplus 2(x_2)^2 \oplus 1$	$\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{2,2}$
$2x_1x_2 \oplus 2x_1 \oplus x_2 \oplus (x_1)^2 \oplus 2(x_2)^2 \oplus 2$	$\mathbf{P}_1\mathbf{P}_{2,2}$
$2x_1x_2 \oplus x_1 \oplus 2x_2 \oplus (x_1)^2 \oplus 2(x_2)^2 \oplus 2$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_{2,2}$
$2x_1x_2 \oplus 2x_1 \oplus 2x_2 \oplus (x_1)^2 \oplus 2(x_2)^2 \oplus 1$	$\mathbf{P}_2\mathbf{P}_{2,2}$
$2x_1x_2 \oplus 2(x_1)^2 \oplus (x_2)^2$	$\mathbf{P}_{1,2}$
$2x_1x_2 \oplus x_1 \oplus 2(x_1)^2 \oplus (x_2)^2 \oplus 1$	$\mathbf{P}_1\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{1,2}$
$2x_1x_2 \oplus x_2 \oplus 2(x_1)^2 \oplus (x2)^2 \oplus 2$	$\mathbf{P}_1\mathbf{P}_2\mathbf{P}_{1,2}$
$2x_1x_2 \oplus 2x_1 \oplus 2(x_1)^2 \oplus (x_2)^2 \oplus 1$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_2\mathbf{P}_{1,2}$
$2x_1x_2 \oplus 2x_2 \oplus 2(x_1)^2 \oplus (x_2)^2 \oplus 2$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{1,2}$
$2x_1x_2 \oplus x_1 \oplus x_2 \oplus 2(x_1)^2 \oplus (x_2)^2 \oplus 1$	$\mathbf{P}_1\mathbf{P}_1\mathbf{P}_{1,2}$
$2x_1x_2 \oplus 2x_1 \oplus x_2 \oplus 2(x_1)^2 \oplus (x_2)^2 \oplus 2$	$\mathbf{P}_2\mathbf{P}_2\mathbf{P}_{1,2}$
$2x_1x_2 \oplus x_1 \oplus 2x_2 \oplus 2(x_1)^2 \oplus (x_2)^2 \oplus 2$	$\mathbf{P}_2\mathbf{P}_{1,2}$
$2x_1x_2 \oplus 2x_1 \oplus 2x_2 \oplus 2(x_1)^2 \oplus (x_2)^2 \oplus 1$	$\mathbf{P}_1\mathbf{P}_{1,2}$

above for functions in $n = 1$.

In the case of function $f_2 = x_1^2 \oplus x_2^2$, multiplication with $\mathbf{P}_1$ corresponds to adding of the term $2x_2 \oplus 1$ as it can be seen in the row 6 of Table 5.

Example 3: The function $f = x_2x_2$ has the value vector $\mathbf{F}_{x_1x_2} = [0, 0, 0, 0, 1, 2, 0, 2, 1]^T$, while the variable x_1 is represented by the value vector $\mathbf{x}_1 = [0, 0, 0, 1, 1, 1, 2, 2, 2]^T$. Therefore, the function vector for $f = x_2x_2 \oplus x_1$ is $\mathbf{F} = \mathbf{F} \oplus \mathbf{x}_1 = [0, 0, 0, 1, 2, 0, 2, 1, 0]^T$. The same vector can be obtained as

$$
\begin{aligned}
\mathbf{F} &= \mathbf{P}_1\mathbf{F}_{x_1x_2} = (\mathbf{I}(1) \otimes \mathbf{X}_1^T)\mathbf{F}_{x_1x_2} \\
&= \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 2 \\ 0 \\ 2 \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} 0, 0, 0, 1, 2, 0, 2, 1, 0 \end{bmatrix}^T.
\end{aligned}
$$

Addition of x_2 to f_1 is expressed in matrix notation as

$$\mathbf{P}_2 = \mathbf{X}_1^T \otimes \mathbf{I}(1).$$

Figures 1, 2, 3, 4, and 5 show flow-graphs of algorithms for performing permutations defined by the matrices

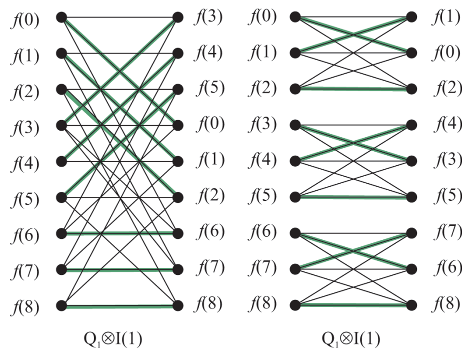

Fig. 1 Flow-graphs for $\mathbf{Q}_1 \otimes \mathbf{I}(1)$ and $\mathbf{I}(1) \otimes \mathbf{Q}_1$.

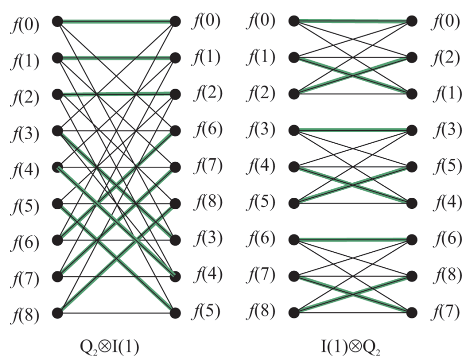

Fig. 2 Flow-graphs for $\mathbf{Q}_2 \otimes \mathbf{I}(1)$ and $\mathbf{I}(1) \otimes \mathbf{Q}_2$.

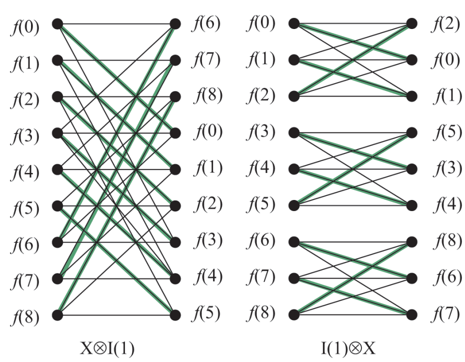

Fig. 3 Flow-graphs for $\mathbf{X}_1 \otimes \mathbf{I}(1)$ and $\mathbf{I}(1) \otimes \mathbf{X}_1$.

$\mathbf{Q}_1 \otimes \mathbf{I}(1)$, $\mathbf{I}(1) \otimes \mathbf{Q}_1$, $\mathbf{Q}_2 \otimes \mathbf{I}(1)$, $\mathbf{I}(1) \otimes \mathbf{Q}_2$, $\mathbf{X} \otimes \mathbf{I}(1)$, $\mathbf{I}(1) \otimes \mathbf{X}$, $\mathbf{X}^T \otimes \mathbf{I}(1)$, $\mathbf{I}(1) \otimes \mathbf{X}^T$, $\mathbf{N} \otimes \mathbf{I}(1)$, $\mathbf{I}(1) \otimes \mathbf{N}$. In order to justify the term FFT-like permutation matrices, the flow-graph for permutations is shown by the ticker green lines over the black lines in the flow-graph for the transform. Recall that in the flow-graph of the FFT for the Vilenkin-Chrestenson transform of ternary functions, the weights at the edges are 1, e_1, and e_2. When the basic transform matrix is replaced by the basic permutation matrices, the weights at the edges are either 0 or 1. Therefore, in the flow-graph, the edges with the weight 0 do no appear.

Example 4: The function $f = x_1x_2 \oplus x_2$ has the value vector $\mathbf{F} = [0, 1, 2, 0, 2, 1, 0, 0, 0]^T$. The same vector can be obtained from the value vector of the function $f = x_1x_2$ by

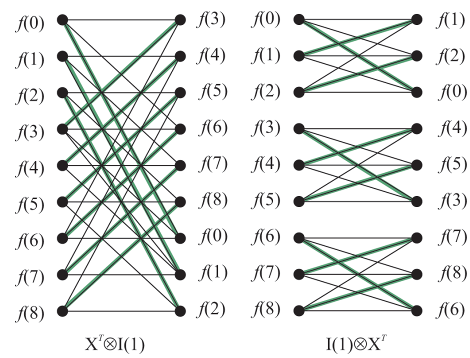

Fig. 4 Flow-graphs for $\mathbf{X}_1^T \otimes \mathbf{I}(1)$ and $\mathbf{I}(1) \otimes \mathbf{X}_1^T$.

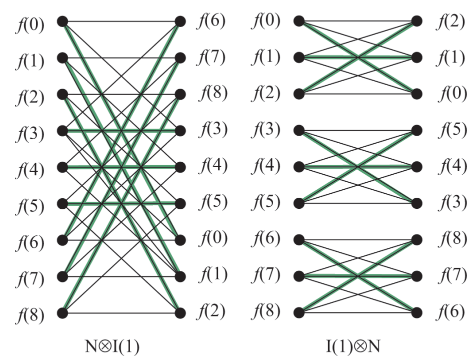

Fig. 5 Flow-graphs for $\mathbf{N}_1 \otimes \mathbf{I}(1)$ and $\mathbf{I}(1) \otimes \mathbf{N}_1$.

multiplying it with the permutation matrix

$$
\begin{aligned}
\mathbf{P}_2 &= \mathbf{X}_1^T \otimes \mathbf{I}(1) \\
&= \begin{bmatrix}
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 2 \\ 0 \\ 2 \\ 1 \end{bmatrix} \\
&= \begin{bmatrix} 0, 1, 2, 0, 2, 1, 0, 0, 0 \end{bmatrix}^T.
\end{aligned}
$$

5.2 Block Diagonal Matrices

In this section, we consider matrices of the block diagonal structure that are typical for the n-th step, in this case the second step, in the Cooley-Tukey FFT algorithms.

Addition of the term x_1^2 to f_1 can be expressed as

$$\mathbf{P}_{1,2} = diag(\mathbf{I}(1), \mathbf{X}_1^T, \mathbf{X}_1).$$

Example 5: The value vector of the term x_1^2 is $\mathbf{F}_{x_1^2} = [0, 0, 0, 1, 1, 1, 1, 1, 1]^T$, and the value vector of $f = x_1x_2 \oplus x_1^2$ is $\mathbf{F}_{x_1x_2 \oplus x_1^2} = [0, 0, 0, 1, 2, 0, 1, 0, 2]^T$. The same vector can be obtained by multiplying the value vector of $\mathbf{F} = x_1x_2$ with the permutation matrix

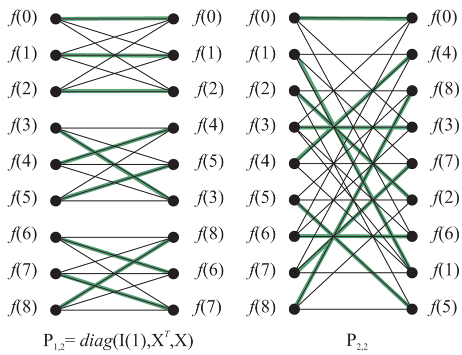

Fig. 6 Flow-graphs for the permutation matrices $\mathbf{P}_{1,2}$ and $\mathbf{P}_{2,2}$.

$$
\begin{aligned}
\mathbf{P}_{1,2} &= Diag(\mathbf{I}(1), \mathbf{X}_1^T, \mathbf{X}_1) \\
&= \begin{bmatrix} 1&0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0 \\ 0&0&0&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&1&0 \end{bmatrix} \begin{bmatrix} 0\\0\\0\\0\\1\\2\\0\\2\\1 \end{bmatrix} \\
&= \begin{bmatrix} 0,0,0,1,2,0,1,0,2 \end{bmatrix}^T.
\end{aligned}
$$

Figure 6 shows the flow-graph of the fast algorithm for permutations by the matrices $\mathbf{P}_{1,2}$, and $\mathbf{P}_{2,2}$ which is defined in Example 6.

5.3 Shift-Based Matrices

It should be noticed that the matrix $\mathbf{X}_1$ performs the cyclic shift. This observation is important since the next permutation matrix $\mathbf{P}_{2,2}$ describing the addition of x_2^2 is determined in terms of the cyclic shift.

Example 6: The function $f = x_2^2$ has the value vector $\mathbf{F}_{x_2}^2 = [0,1,1,0,1,1,0,1,1]^T$. When added to the value vector of $f = x_1x_2$, which is $\mathbf{F}_{x_1x_2} = [0,0,0,0,1,2,0,2,1]^T$, the value vector of the bent function $x_1x_2 \oplus x_2^2$ is obtained as $\mathbf{F}_{x_1x_2\oplus x_2^2} = [0,1,1,0,2,0,0,0,2]^T$. The same value vector can be obtained by the multiplication of $\mathbf{F}_{x_1x_2}$ by the permutation matrix $\mathbf{P}_{2,2}$ as

$$
\begin{aligned}
\mathbf{F}_{x_1x_2\oplus x_2^2} &= \mathbf{P}_{2,2} \cdot \mathbf{F}_{x_1x_2} \\
&= \begin{bmatrix} 1&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0&1 \\ 0&0&0&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0 \\ 0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&0&0 \\ 0&1&0&0&0&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0 \end{bmatrix} \cdot \begin{bmatrix} 0\\0\\0\\0\\1\\2\\0\\2\\1 \end{bmatrix} \\
&= \begin{bmatrix} 0,1,1,0,2,0,0,0,2 \end{bmatrix}^T.
\end{aligned}
$$

It is easy to observe that the matrix $\mathbf{P}_{2,2}$ performs the spectral invariant operation $x_1 \to x_1 \oplus x_2$.

Matrix $\mathbf{P}_{2,2}$ describing the addition of x_2^2 to x_1x_2 can be split into (3×3) submatrices. They are arranged as blocks of three submatrices in three rows. In the first row of submatrices, the first submatrix has 1 as the first elements in the first row. Then, the next submatrix has 1 as the second element in the second row. The third submatrix has 1 as the third element in the third row.

In the second row of submatrices, the same pattern repeats but shifted cyclically for a single place to the right. In the third row of submatrices, the pattern repeats but with a shift for two places. If we define the (3×3) auxiliary matrices with a single non-zero element as

$$
\mathbf{R} = \begin{bmatrix} 1&0&0 \\ 0&0&0 \\ 0&0&0 \end{bmatrix}, \mathbf{V} = \begin{bmatrix} 0&0&0 \\ 0&1&0 \\ 0&0&0 \end{bmatrix}, \mathbf{E} = \begin{bmatrix} 0&0&0 \\ 0&0&0 \\ 0&0&1 \end{bmatrix},
$$

then, the matrix $\mathbf{P}_{2,2}$ can be written as

$$
\mathbf{P}_{2,2} = \begin{bmatrix} \mathbf{R} & \mathbf{V} & \mathbf{E} \\ \mathbf{E} & \mathbf{R} & \mathbf{V} \\ \mathbf{V} & \mathbf{E} & \mathbf{R} \end{bmatrix},
$$

and if the matrix $\mathbf{P}_{2,2}$ is split into (3×3) blocks, it can be observed that the same shift based structure repeats over blocks $\mathbf{R}$, $\mathbf{V}$, and $\mathbf{E}$.

In a formal way, this matrix can be determined as follows.

Consider the sequence of symbols $S = \{a,b,c\}$, and define (3×3) matrices $\mathbf{A} = diag(a,b,c)$ and $\mathbf{B}$ as

$$
\mathbf{A} = \begin{bmatrix} a&0&0 \\ 0&b&0 \\ 0&0&c \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} a&b&c \\ c&a&b \\ b&c&a \end{bmatrix}.
$$

The Kronecker products of these symbolic matrices $(\mathbf{A}\otimes\mathbf{B})$ and $(\mathbf{B}\otimes\mathbf{A})$ followed by the replacement of the squared terms by 1 and all other product terms with mixed symbols by 0 results in the matrices $\mathbf{P}_{1,2}$ and $\mathbf{P}_{2,2}$.

Resemblance to matrices describing steps of the Cooley-Tukey FFT-like algorithms illustrated by the Example 1 is easy to notice, especially in the case of permutation matrices describing addition of variables. The difference is that instead of using the basic transform matrices for $n = 1$, here we use the (3×3) permutation matrices.

The matrices determining addition of squares of variables resemble reordering matrices appearing between steps of certain other FFT algorithms, as for example, the Winograd FFT (WFTA), Prime Factors Algorithms, and related. For more information, we refer to [5], [11], [15].

As in the case of previously considered Kronecker representable permutation matrices, these block diagonal matrices perform particular permutations of spectral coefficients without changing their values. It follows that they perform spectral invariant operations. We can conclude which operations are representable by the shift based matrices by comparing functional expressions of functions derived by

Table 6 Substitution rules corresponding to the shift-based FFT-like permutation matrices.

d	k	c	Substitution rule
a,b,c	2	0	$x_i \to 2x_j$
a,c,b	1	0	$x_i \to x_j$
b,a,c	1	2	$x_i \to x_j \oplus 2$
b,c,a	2	2	$x_i \to 2x_j \oplus 2$
c,a,b	1	1	$x_i \to x_j \oplus 1$
c,b,a	2	1	$x_i \to 2x_j \oplus 1$

permutations with these matrices and functions obtained by spectral invariant operations. Considering operations performed by Kronecker product representable permutation matrices is certainly avoided. In this way, it is easy to observe that shift based matrices under consideration perform the spectral invariant operation of substitution of a variable $x_i \to x_i \oplus k x_j \oplus c$, where $k \in \{1,2\}$, and $c \in \{0,1,2\}$, assuming that the matrices $\mathbf{A}$ and $\mathbf{B}$ are at the j-th and the i-th position in the Kronecker product, while the identity matrix $\mathbf{I}(1)$ is at all other positions. The values of k and c are determined by the permutation of elements a,b,c at the main diagonal d of the matrix $\mathbf{A}$ as summarized in Table 6.

6. Construction of Bent Functions for $n = 2$

As shown in Tables 4 and 5, all 81 bent functions in two variables can be constructed from the basic bent functions f_1 and f_2 by permutation matrices. The other ternary bent functions in two variables are obtained by encoding as specified in Table 2.

An algorithm to generate ternary bent functions in two variables by permutation matrices can be formulated as follows.

1. Select the distribution D_1 or D_2.
2. Select the basis function f_1 or f_2.
3. Apply to the value vector of the selected function the permutation matrices corresponding to terms that are allowed to be added.
4. Perform an encoding of the function values.

With respect to Step 3 of the above algorithm, notice that for all included permutation matrices $\mathbf{P}^3 = \mathbf{I}(2)$, where $\mathbf{I}(2)$ is a (9×9) identity matrix, which preserves that the same matrix can be applied no more than two times. That is the maximal number of applications of a matrix in the above tables.

Recall that in the case of binary bent functions, studied are permutations with the property $\mathbf{P}^2 = \mathbf{I}$ due to which they are called involutions [10].

In the generalization of the method to functions with more than two variables, basic bent functions to which permutation matrices are applied should be selected such that their degree is equal to the degree of functions to be produced.

Example 7: Consider the bent function $f = x_1x_2 \oplus x_1 \oplus (x_1)^2$, whose value vector is $\mathbf{F} = [0,0,0,2,0,1,0,2,1]^T$. This vector can be obtained by multiplying the value vector for $f = x_1x_2$ with the permutation matrix obtained as the product of the permutation matrices for adding x_1 and $(x_1)^2$

$$\mathbf{P} = \mathbf{P}_1 \cdot \mathbf{P}_{1,2} = Diag(\mathbf{X}_1^T, \mathbf{X}_1, \mathbf{I}(1))$$

$$= \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

In the same way, various permutation matrices can be defined.

7. Generalizations

The above method for constructing bent functions can be generalized straightforwardly to functions with a number of variables larger than $n = 2$. The corresponding permutation matrices can be derived by referring to the structure of the matrices in the steps of Cooley-Tukey FFT in the case of $\mathbf{P}_1$ and $\mathbf{P}_2$, and the symbolic computations for $\mathbf{P}_{1,2}$ and $\mathbf{P}_{2,2}$. For simplicity, the way towards the generalizations will be explained and illustrated by the following examples for $n = 3$. In this case, the basic bent functions are $q_1 = x_1x_2 \oplus (x_3)^2$ and $q_2 = x_1^2 \oplus x_2^2 \oplus x_3^2$. These both functions are of the degree 3. Therefore, an example of constructing new bent functions by permutation matrices from a bent function of the degree 4, $f = x_1x_2 \oplus (x_3)^2 \oplus (x_1)^2(x_3)^2$ is also presented.

It is important to notice that for $n = 3$, these basic bent functions, as well as all other bent functions, have the same distribution of function values $D_3 = (6,9,12)$. As in the previous cases for $n = 1$ and $n = 2$, all bent functions for $n = 3$ are split into 6 classes mutually related by encoding of function values. The first class consists of functions with the composition $(9,12,6)$ and the representant is the function $q_1 = x_1x_2 \oplus (x_3)^2$. The other classes consist of functions with permuted values of the composition elements. For example, functions with the composition $(9,6,12)$ are a different class, whose representant is $q_3 = (x_1)^2 \oplus (x_2)^2 \oplus (x_3)^2$. We define several permutation matrices by referring to matrices describing steps in Cooley-Tukey FFT

$$\begin{aligned} \mathbf{R}_1 &= \mathbf{I}(1) \otimes \mathbf{X}_1^T \otimes \mathbf{I}(1) = \mathbf{P}_1 \otimes \mathbf{I}(1), \\ \mathbf{R}_2 &= \mathbf{X}_1^T \otimes \mathbf{I}(1) \otimes \mathbf{I}(1), \\ \mathbf{R}_3 &= \mathbf{I}(1) \otimes \mathbf{I}(1) \otimes \mathbf{X}_1^T. \end{aligned}$$

Table 7 shows that these matrices add the linear terms $x_1, x_2, 2x_3 \oplus 1$, respectively, to the function $q_1 = x_1x_2 \oplus (x_3)^2$.

The function $f = x_1x_2 \oplus (x_3)^2 \oplus x_3$ is also bent, however, its value vector is $\mathbf{F} = [020020020020101212020212101]^T$. We see that this is a different composition $(12,6,9)$, therefore, it belongs to a different class and cannot be obtained by a permutation of function values in $f_1 = x_1x_2 \oplus (x_3)^2$, but by encoding.

Table 7 Functions obtained by permutation matrices $\mathbf{R}_1, \mathbf{R}_2, \mathbf{R}_3$.

Function	Matrix
$x_1x_2 \oplus (x_3)^2 \oplus x_1$	$\mathbf{R}_1$
$x_1x_2 \oplus (x_3)^2 \oplus x_2$	$\mathbf{R}_2$
$x_1x_2 \oplus (x_3)^2 \oplus 2x_3 \oplus 1$	$\mathbf{R}_3$

Table 8 Construction of ternary bent functions by permutation matrices from q_1 with the composition $(9, 12, 6)$.

Function	Permutation matrices
$x_1x_2 \oplus (x_3)^2 \oplus (x_1)^2$ $[011011011122200011122011200]^T$	$\mathbf{P}_{1,2} \otimes \mathbf{I}(1)$
$x_1x_2 \oplus (x_3)^2 \oplus (x_2)^2 \oplus 2x_2x_3$ $[011110101011221020011002212]^T$	$\mathbf{I}(1) \otimes \mathbf{P}_{1,2}$
$x_1x_2 \oplus (x_3)^2 \oplus (x_2)^2$ $[011122122011200011011011200]^T$	$\mathbf{P}_{2,2} \otimes \mathbf{I}(1)$
$x_1x_2 \oplus (x_3)^2 \oplus x_1x_3$ $[011011011020101212002221110]^T$	$\mathbf{I}(1) \otimes \mathbf{P}_{2,2}$

The matrices for adding quadratic terms are constructed as

$$\begin{aligned} \mathbf{R}_1 &= \mathbf{A} \otimes \mathbf{B} \otimes \mathbf{I}(1), & \mathbf{R}_4 &= \mathbf{B} \otimes \mathbf{A} \otimes \mathbf{I}(1), \\ \mathbf{R}_2 &= \mathbf{A} \otimes \mathbf{I}(1) \otimes \mathbf{B}, & \mathbf{R}_5 &= \mathbf{B} \otimes \mathbf{I}(1) \otimes \mathbf{A}, \\ \mathbf{R}_3 &= \mathbf{I}(1) \otimes \mathbf{A} \otimes \mathbf{B}, & \mathbf{R}_6 &= \mathbf{I}(1) \otimes \mathbf{B} \otimes \mathbf{A}, \end{aligned}$$

followed by the replacement of their elements by 1 and 0 as specified above.

Depending on the functions to which they are applied, q_1 or q_2, these matrices add terms $(x_1)^2$, $(x_2)^2$, and $(x_3)^2$ possibly combined with variables. In all the cases, coefficients in the added terms can be either 1 or 2, which increases the number of new bent functions that can be produced.

Table 8 shows bent functions derived from the function $q_1 = x_1x_2 \oplus (x_3)^2$ with the composition $(9, 12, 6)$. Table 9 shows bent functions derived from the function $q_2 = (x_1)^2 \oplus (x_2)^2 \oplus (x_3)^2$ with the composition $(9, 6, 12)$.

The matrix which expresses adding the term of order three $(x_1)^2x_3$ to the function $q_1 = x_1x_2 \oplus (x_3)^2$ is defined as

$$\mathbf{P}_3 = diag(\mathbf{I}(1) \otimes \mathbf{I}(1), \mathbf{X}_1 \otimes \mathbf{X}_1, \mathbf{X}_1^T \otimes \mathbf{X}_1).$$

The structure of this matrix is typical for the $(n-1)$-th step in Cooley-Tukey FFT algorithms.

The value vector of $f = x_1x_2 \oplus (x_3)^2 \oplus (x_1)^2x_3$ is

$$\mathbf{F}_3 = [011011011020101212020212101]^T.$$

The composition is $(9, 12, 6)$. The same value vector can be obtained as $\mathbf{F}_3 = \mathbf{P}_3\mathbf{F}_{x_1x_2 \oplus (x_3)^2}$.

Another approach to construct bent functions with the selected distribution and the degree is the following.

1. Given a bent function f with the distribution D and the degree $deg(f)$.
2. Split the function vector $\mathbf{F}$ of f into subvectors of length 3^k, $1 \leq k \leq n-1$.
3. Perform encoding of subvectors and construct a new function vector $\mathbf{F}_e$ which elements are symbols assigned to the subvectors.

Table 9 Construction of ternary bent functions by permutation matrices from q_2 with the composition $(9, 6, 12)$.

Function	Permutation matrices
$2(x_1)^2 \oplus (x_2)^2 \oplus (x_3)^2 \oplus 2x_1x_2$ $[011122122200200122200122200]^T$	$\mathbf{P}_{1,2} \otimes \mathbf{I}(1)$
$(x_1)^2 \oplus 2(x_2)^2 \oplus (x_3)^2 \oplus 2x_2x_3$ $[011221212122002020122002020]^T$	$\mathbf{I}(1) \otimes \mathbf{P}_{1,2}$
$(x_1)^2 \oplus 2(x_2)^2 \oplus (x_3)^2 \oplus 2x_1x_2$ $[011200200122200122122122200]^T$	$\mathbf{P}_{2,2} \otimes \mathbf{I}(1)$
$(x_1)^2 \oplus (x_2)^2 \oplus 2(x_3)^2 \oplus 2x_2x_3$ $[022121112100202220100202220]^T$	$\mathbf{I}(1) \otimes \mathbf{P}_{2,2}$

4. Apply a FFT-like permutation matrix $\mathbf{R}$ to $\mathbf{F}_e$ and produce a new bent function f_e.
5. Apply the FFT-like permutation matrix $\mathbf{R}$ to the subvectors and produce another bent function $f_{e,R}$.

Conversion of performing permutations over subvectors instead over the complete function vector is aimed at making the procedure convenient for implementation on parallel hardware structures. It should be noticed that the choice of the parameter k determines which step of FFT-like permutation algorithm is actually performed. This at the same time defines in terms of which variable the spectral invariant operation corresponding to the matrix $\mathbf{R}$ is performed. The choice of the matrix used to perform the permutation defines which spectral invariant operation is performed.

Example 8: The ternary function in 3 variables

$$f(x_1, x_2, x_3) = x_1x_2 \oplus x_3^2 \oplus x_2^2x_3$$

is a bent function of degree 3. Its functions vector is

$$\mathbf{F} = [011020020011101212011212101]^T.$$

The distribution of function values is $(9, 12, 6)$.

We split this function vector into three suvectors of 9 elements as

$$\begin{aligned} \mathbf{K}_0 &= [0, 1, 1, 0, 2, 0, 0, 2, 0]^T, \\ \mathbf{K}_1 &= [0, 1, 1, 1, 0, 1, 2, 1, 2]^T, \\ \mathbf{K}_2 &= [0, 1, 1, 2, 1, 2, 1, 0, 1]^T. \end{aligned}$$

With this notation, it is

$$\mathbf{F} = [\mathbf{K}_0, \mathbf{K}_1, \mathbf{K}_2]^T.$$

The matrix $\mathbf{Q}_1$ converts $\mathbf{F}$ into $\mathbf{F}_{Q_1} = [\mathbf{K}_1, \mathbf{K}_0, \mathbf{K}_2]^T$, which is

$$\mathbf{F}_{Q_1} = [011101212011020020011212101]^T.$$

The corresponding function expression is

$$f(x_1, x_2, x_3)_{Q_1} = x_3^2 \oplus x_2 \oplus x_2^2x_3 \oplus 2x_1x_2,$$

and it can obtained by the spectral invariant operation $x_1 \to 2x_1 \oplus 1$. Thus, the matrix $\mathbf{Q}_1$ applied to subvectors of length 9 performs this spectral invariant operation as the example

Table 10 Spectral invariant operations and basic permutation matrices.

Matrix	Reordering	Transformation
$\mathbf{Q}_1$	$\mathbf{F}_{Q_1} = [\mathbf{K}_1, \mathbf{K}_0, \mathbf{K}_2]^T$	$x_1 \to 2x_1 \oplus 1$
$\mathbf{Q}_2$	$\mathbf{F}_{Q_2} = [\mathbf{K}_0, \mathbf{K}_2, \mathbf{K}_1]^T$	$x_1 \to 2x_1$
$\mathbf{X}_1$	$\mathbf{F}_{X_1} = [\mathbf{K}_2, \mathbf{K}_0, \mathbf{K}_1]^T$	$x_1 \to x_1 \oplus 2$
$\mathbf{X}_1^T$	$\mathbf{F}_{X_1^T} = [\mathbf{K}_1, \mathbf{K}_2, \mathbf{K}_0]^T$	$x_1 \to x_1 \oplus 1$
$\mathbf{N}_1$	$\mathbf{F}_{N_1} = [\mathbf{K}_2, \mathbf{K}_1, \mathbf{K}_0]^T$	$x_1 \to 2x_1 \oplus 2$

Table 11 Bent functions constructed by basic permutation matrices from the function f in Example 8.

Matrix	Function
$\mathbf{Q}_1$	$f(x_1, x_2, x_3)_{Q_1} = x_3^2 \oplus x_2 \oplus x_2^2 x_3 \oplus 2x_1 x_2$
$\mathbf{Q}_2$	$f(x_1, x_2, x_3)_{Q_2} = x_3^2 \oplus x_2^2 x_3 \oplus 2x_1 x_2$
$\mathbf{X}_1$	$f(x_1, x_2, x_3)_{X_1} = x_3^2 \oplus 2x_2 \oplus x_2^2 x_3 \oplus x_1 x_2$
$\mathbf{X}_1^T$	$f(x_1, x_2, x_3)_{X_1^T} = x_3^2 \oplus x_2 \oplus x_2^2 x_3 \oplus x_1 x_2$
$\mathbf{N}_1$	$f(x_1, x_2, x_3)_{N_1} = x_3^2 \oplus 2x_2 \oplus x_2^2 x_3 \oplus 2x_1 x_2$

illustrates.

Table 10 shows the reordering of subvectors produced by the application of matrices $\mathbf{Q}_1$, $\mathbf{Q}_2$, $\mathbf{X}_1$, $\mathbf{X}_1^T$, and $\mathbf{N}_1$, the functional expressions of resulting functions and the corresponding spectral invariant operations. Table 11 shows the functions obtained by the application of these permutation matrices.

From the structure of FFT-like algorithms for the Vilenkin-Chrestenson transform of ternary functions, it is easy to see that the application of the considered (3×3) matrices to subvectors of length 9 is equivalent to multiplication of $\mathbf{F}$ with the matrix

$$\mathbf{A} \otimes \mathbf{I}(1) \otimes \mathbf{I}(1)$$

where $\mathbf{A}$ is any of the considered matrices. In other words, it is equivalent to performing the permutation corresponding to the first step of the FFT-like algorithm. Due to this, the performed spectral invariant operations are with respect to the first variable x_1. Performing the permutations corresponding to the i-th step results in the spectral invariant operations are with respect to i-th variable x_i, $i \in \{x_1, x_2, \ldots, x_n\}$.

When to the function obtained by the application of $X(1)$, we apply the same transformation to subvectors of length 9, we get the function with the function vector

$$\mathbf{F} = [101011212020011020212011101]^T,$$

which is bent and it is obtained by the successive application of the spectral invariant operations $x_1 \to x_1 \oplus 2$ and $x_2 \to x_2 \oplus 2$.

8. Closing Remarks

For ternary bent functions there are specific combinations of ternary values a function can take. This feature is expressed as the distribution of function values. As in the binary case, the distribution depends on the number of variables. For any n, the bent functions can be split into 6 classes that are related by encoding. Functions within a class having the same composition of values, are mutually related by permutation of elements in their value vectors. Therefore, by starting from a function representing the class, all other functions in the class can be derived by permutation of elements in value vectors. These permutations are not arbitrary, but strictly structured, since bentness, alternatively flatness of the Vilenkin-Chrestenson spectra should be preserved. We show that the corresponding permutation matrices have a block structure similar to that in factor matrices of Cooley-Tukey FFT.

References

[1] C. Carlet and S. Mesnager, "Four decades of research on bent functions," DES. Codes Cryptogr., vol.78, no.1, pp.5–50, Jan. 2016.

[2] R. Calderbank and W.M. Kantor, "The geometry of two-weight codes," Bull. London Math. Soc., vol.18, no.2, pp.97–122, March 1986.

[3] J.F. Dillon, Elementary Hadamard Difference Sets, Ph.D. dissertation, University of Maryland, College Park, 1974.

[4] S. Hodžić and E. Pašalić, "Generalized bent functions - some general construction methods and related necessary and sufficient conditions," Cryptogr. Commun., 2015, DOI 10.1007/s12095-015-0126-9.

[5] M.G. Karpovsky, R.S. Stanković, and J. Astola, Spectral Logic and its Applications for the Design of Digital Devices, Wiley, 2008.

[6] P.V. Kumar, R.A. Scholtz, and L.R. Welch, "Generalized bent functions and their properties," J. Combin. Theory Ser. A, vol.40, no.1, pp.90–107, Sept. 1985.

[7] O.A. Logachev, A.A. Salnikov, and V.V. Yashchenko, "Bent functions on a finite Abelian group," Discrete Math. Appl., vol.7, no.6, pp.547–564, 1997.

[8] M. Luis and C. Moraga, "Functions with flat Chrestenson spectra," Proc. 19th Int. Symp. Multiple-valued Logic, pp.406–413, Guangzhou, China, 1989.

[9] G. Luo, X. Cao, and S. Mesnager, "Several new classes of self-dual bent functions derived from involutions," Cryptography and Communications, Published online 17 May 2019, https://doi.org/10.1007/s12095-019-00371-9.

[10] S. Mesnager, "On constructions of bent functions from involutions," Proc. Int. Symp. on Inform. Theory (ISIT), Barcelona, Spain, pp.110-114, July 10-15, 2016.

[11] H.J. Nussbaumer, Fast Convolution Algorithms, Fast Fourier Transform and Convolution Algorithms, Springer-Verlag, Berlin, 1981.

[12] K. Nyberg, "Constructions of bent functions and difference sets," I.B. Damgard, (ed.), Advances in Cryptology - EUROCRYPT 90, LNCS, vol.473, pp.151–160, 1991.

[13] L. Poinsot, "Bent functions on a finite non-Abelian group," J. Discrete Mathematical Science and Cryptography, vol.9, no.2, pp.349–364, 2006.

[14] A. Pott, "Nonlinear functions in Abelian groups and relative difference sets," Discrete Applied Mathematics, vol.138, no.1-2, pp.177–193, March 2004.

[15] M.R. Stojić, M.S. Stanković, and R.S. Stanković, Discrete Transforms in Applications, Nauka, Belgrade, 1993, ISBN 86-7621-019 (in Serbian).

[16] N. Tokareva, "Generalizations of bent functions - A survey," J. Appl. Ind. Math., vol.5, no.1, pp.110–129, 2011.

[17] N. Tokareva, Bent Functions - Results and Applications to Cryptography, Elsevier, 2015.

[18] T. Wada, "Characteristic of bit sequences applicable to constant amplitude orthogonal multicode systems," IEICE Trans. Fundamentals, vol.E83-A, no.11, pp.2160–2164, Nov. 2000.

[19] G.B. Weng, W.S. Qiu, Z.Y. Wang, and Q. Xiang, "Pseudo-Paley graphs and skew Hadamard difference sets from presemifields," Des. Codes Cryptogr., vol.44, pp.49–62, 2007.

Radomir S. Stanković received the B.Sc. degree in electronic engineering from the Faculty of Electronics, University of Niš, Serbia, in 1976, and M.Sc. and Ph.D. degrees in applied mathematics from the Faculty of Electrical Engineering, University of Belgrade, Serbia, in 1984 and 1986, respectively. He was a Professor at the Department of Computer Science, Faculty of Electronics, University of Niš, Serbia, until 2017, and a Full research professor at the Mathematical Institute of SASA in Belgrade, Serbia, from 2017 until his retirement in 2019. In 1997, he was awarded by the Kyushu Institute of Technology Fellowship and worked as a visiting researcher at the Department of Computer Science and Electronics, Kyushu Institute of Technology, Iizuka, Fukuoka, Japan. In 2000 he was awarded by the Nokia Professorship by Nokia, Finland. From 1999 until 2017, he worked in part at the Tampere International Center for Signal Processing, Department of Signal Processing, Faculty of Computing and Electrical Engineering, Tampere University of Technology, Tampere, Finland, first as a visiting researcher and from 2007 as an adjunct professor. His research interests include switching theory, multiple-valued logic, spectral techniques, and signal processing. A special field of his interest is history of computing.

Milena Stanković received the B.Sc. degree in electronic engineering from the Faculty of Electronic Engineering University of Niš, Serbia, in 1976, and M.Sc. and Ph.D. degrees in Computer science and Informatics from the Faculty of Electronic Engineering, University of Niš, in 1982 and 1988, respectively. She was the Head of the Department of Computing, Faculty of Electronic Engineering and the Head of the Computational Intelligence and Information Technologies Laboratory (CIITLab). Her research interests include switching theory, multiple-valued logic, spectral techniques, and data mining.

Claudio Moraga received his B.Sc. in Electronic Engineering from the Catholic University of Valparaiso, Chile, in 1961, a M.Sc. in Electrical Engineering from the Massachusetts Institute of Technology in 1962, and a Ph.D. in Electrical Engineering from the Technical University “Federico Santa María”, Valparaiso, Chile, in 1972. In 1974 he was awarded an Alexander von Humboldt Research Fellowship at the Department of Computer Science of the University of Dortmund, Germany, where in 1986 he became a Professor in the area of Theoretical Computer Science until his retirement in 2002, remaining at the University as Researcher ad honorem. In February 2006 he received a Doctorate Honoris Causa from the University of Niš, Serbia. In March 2006 he became Founding Researcher Emeritus of the European Centre for Soft Computing, Asturias, Spain. This Centre was closed in 2016 and Moraga became a representative of Germany at the COST Action IC 1405 of the European Community, dedicated to Reversible Computing. The research interests of Dr. Moraga include Multiple-valued Logic, Bent functions, Reversible Computing, and Spectral Techniques.

Jaakko T. Astola has been working in signal processing and related fields for over 30 years. He is member of the Finnish Academy of Science and Letters, The Finnish Academy of Technology, and Academia Europaea. He is a Fellow of the IEEE, SPIE and EURASIP. He is on the editorial boards of several international journals in the field. He has served as the General Chair or Co-Chair of many International conferences, e.g. EUSIPCO 2000, SPIE Electronic Imaging 2001, GENSIPS 2005 and 2007. He has made significant contributions to image processing image coding and image analysis. The concept of Vector Median Filter, developed by him, is widely referenced. He has also studied image processing methods that are based on local approximation and statistical decision rules, genomic signal processing, e.g. microarray quality control, and information theoretic methods in the analysis of biological signals. He has also worked on algebraic coding theory, logic design and representation of discrete functions, and written several articles and textbooks in the field. Overall, Professor Astola has co-authored several books, published over 800 articles, about 200 of which are in archive journals, mostly on image processing and coding. He has supervised 40 of doctoral theses.