International Journal of Mathematical, Engineering and Management Sciences
Vol. 8, No. 5, 1040-1055, 2023
https://doi.org/10.33889/IJMEMS.2023.8.5.060

An Algorithm for Constructing Support of Bent Functions by Extending a Set

Joseph Nelson
Department of Mathematics,
Amrita School of Physical Sciences, Coimbatore, Amrita Vishwa Vidyapeetham, India.
Corresponding author: p_josephnelson@cb.students.amrita.edu

Chungath Srinivasan
TIFAC-CORE in Cyber Security,
Amrita School of Engineering, Coimbatore, Amrita Vishwa Vidyapeetham, India.
E-mail: c_srinivasan@cb.amrita.edu

K. V. Lakshmy
TIFAC-CORE in Cyber Security,
Amrita School of Engineering, Coimbatore, Amrita Vishwa Vidyapeetham, India.
E-mail: kv_lakshmy@cb.amrita.edu

(Received on April 21, 2023; Accepted on August 3, 2023)

Abstract
Boolean functions form the fundamental components of symmetric cryptographic systems, serving as the building blocks. Modifying bent functions makes it feasible to design Boolean functions with desired properties that exhibit high non-linearity. The current study offers a comprehensive analysis of bent functions through its support, culminating in the introduction of an algorithm for the systematic construction of four variable bent functions. This algorithm enables the complete generation of all 896 four-variable bent functions. Furthermore, we introduce a methodology for constructing *n*-variable bent functions (where *n* > 4), leveraging both the algorithm and an established secondary technique for bent function construction. Lastly, we examine the estimation of the count of bent functions by utilising certain properties associated with the support of bent functions.

Keywords- Primary construction of bent functions, Algorithm to construct bent functions, Boolean functions, Number of bent functions, Auto correlation.

1. Introduction

In the field of cryptography, the utilisation of Boolean functions plays a crucial role in the advancement of ciphers (Neethu et al., 2018, Srinivasan et al., 2011). The pursuit of Boolean functions exhibiting maximum non-linearity has led to the development of bent functions, pioneered by Oscar Rothaus in the 1960s (Rothaus, 1976). Initially, bent functions introduced optimal non-linearity in cryptographic applications such as constructing ciphers CAST, Grain, and the hash function HAVAL. The construction of substitution boxes (S-boxes) is pivotal in block ciphers, as they provide essential cryptographic properties, such as non-linearity and security. Creating S-boxes with robust cryptographic strength hinges on the chosen construction technique. An S-box that exhibits strong cryptographic properties can be constructed by generating a set of bent functions as its output. This approach ensures that favourable cryptological properties for the S-box are achieved, providing a compelling incentive for the development of bent functions.

A substantial body of scholarly work has investigated the iterative, combinatorial, and exhaustive methodologies used to formulate bent functions (Carlet et al., 2014). Bent function constructions are broadly classified into two types: primary and secondary. Primary constructions entail the development of bent functions based on their innate properties, whereas secondary constructions use pre-existing bent functions as fundamental building blocks (Carlet et al., 2012). The Maiorana-McFarland construction, a significant primary construction technique, was independently introduced by Maiorana and McFarland. Through their research, these scholars determined the number of functions that can be generated using this construction, which remains a fundamental lower bound for the count of bent functions. Following that, a plethora of primary construction techniques have been developed. Carlet extensively explored primary construction methods for bent functions in a 1996 work (Carlet, 1996), while a subsequent publication in 2004 focused on the secondary construction of resilient and bent functions (Carlet, 2004).

Depending on the construction techniques employed and the structural characteristics of the functions, bent functions are further classified into distinct classes, including the Maiorana-McFarland class, the partial spread class, and the class H (Dillon, 1974, Carlet and Mesnager, 2011). Carlet identified that new classes of bent functions can be generated from pre-existing bent functions through a secondary construction technique (Carlet, 1994). The thorough exploration of primary and secondary construction techniques for bent functions boosts possibilities and deepens our understanding of their properties and practical applications in various cryptographic contexts.

In addition to the constructions mentioned above, bent functions include several noteworthy classes (Li et al., 2013, Mesnager et al., 2021), with Climent et al. (2012) contributing to this domain by developing a primary construction method for constructing bent functions that leverages a vector space basis. Secondary constructions, on the other hand, comprise techniques such as direct sum, Rothaus' construction, indirect sum, and their generalisations (Dillon, 1974).

The manipulation of diverse properties of bent functions results in the creation of functions with significant cryptographic relevance, rendering them indispensable for the comprehensive study of bent functions. The introduction of the construction of bent functions using Niho power functions by Dobbertin et al. (2006) has paved the way for various findings and outcomes achieved through the analysis of Niho bent functions (Budaghyan et al., 2012).

By imposing additional constraints and extending the capabilities of bent functions, it becomes possible to create bent functions well-suited for diverse cryptographic applications, with a notable example being the normal extension of bent functions (Carlet et al., 2004). Boolean functions exhibiting rotation symmetry invariance are particularly advantageous for hardware implementations as they offer cost-effectiveness (Cusick et al., 2016, Dalai et al., 2009). Within this context, rotation symmetric bent functions assume significant importance. Additionally, bent functions that can be represented as monomials hold desirable properties (Leander, 2006).

Apart from nonlinearity, the algebraic degree of Boolean functions is essential, and Eliseev and Stepchenkov's demonstration in 1962 revealed that for $n \geq 4$, the degree of an n-variable bent function cannot exceed $\frac{n}{2}$, setting the foundation for establishing an upper bound on the number of bent functions. Consequently, it was found that constructing Boolean functions with maximum nonlinearity and high algebraic degrees is impossible. A Boolean function f is considered perfect nonlinear if $f(x) \oplus f(x \oplus \boldsymbol{a})$ is balanced for all non-zero elements $\boldsymbol{a}$ as discussed by (Nyberg, 1991; Meier and Staffelbach, 1990). Importantly, all bent functions inherently exhibit perfect nonlinearity.

Rank serves as a measure for comparing two functions to ascertain their affine equivalence; if two bent functions possess different ranks, it indicates they are not affine equivalent. However, the converse is not always valid, as two bent functions with the same rank may not necessarily be affine equivalent. Gadouleau et al. (2023) presented the construction of partial spread bent functions using subspaces generated by linear recurring sequences and calculated the ranks of these generated functions, thereby contributing to a deeper comprehension of these constructions.

Notably, Langevin and Leander (2011) attempted to determine the count of all 8-variable bent functions by partitioning the space of all such functions. This approach closely resembles an exhaustive search in its nature and objective. Despite the existence of numerous construction methods for bent functions, the complete set of bent functions remains to be discovered, and there currently needs to be a comprehensive method to construct all possible bent functions. Consequently, any novel construction method in this area holds relevance. In this context, the process involves augmenting an initially empty set by progressively adding elements to form a support for a bent function. To facilitate the random generation of bent functions, we propose an algorithm.

A Boolean function's domain can be represented as a field or an n-dimensional vector space with 2^n elements and characteristic 2. While existing literature primarily adopts the field as the domain for construction techniques, our approach in this work considers the domain as an n-dimensional vector space over the field with two elements. In contrast to traditional construction procedures for bent functions that heavily rely on the properties of the algebraic normal form in Boolean functions, our approach utilises the properties of the support of bent functions for the construction process.

This study delves into several distinctive properties of bent functions and their support. Leveraging these properties, the authors aim to formulate an algorithm for constructing bent functions and provide an approximation for their count. Section 2 introduces the fundamental concepts defining Boolean functions. In section 3, a subset δ_f of $GF(2)^n$ is defined for each n-variable Boolean function f, along with establishing a relationship between δ_f and the weight of f. Furthermore, a characterization criterion for bent functions is described, forming the basis for an algorithm that facilitates their primary construction. Section 4 derives an equation to approximate the count of bent functions using the proposed primary construction algorithm. Additionally, we present a construction method for n-variable bent functions (where n is even and $n > 4$) by combining an existing secondary construction technique with the proposed primary construction algorithm. The practical usability of the proposed algorithm is discussed in section 5, while section 6 concludes the paper.

2. Preliminaries

It is to be noted that the notation $GF(2)$ is the field with two elements 0 and 1, under addition modulo 2 denoted by $\oplus$ and multiplication modulo 2 denoted by writing two elements conjointly. $GF(2)^n$ is used for representing the n-dimensional vector space over the field $GF(2)$, which consists of n-tuples with entries from $GF(2)$. The elements in $GF(2)^n$ have been denoted using bold letters, and 0 is used to denote the vector (0, 0, ..., 0). $GF(2)^{n*}$ consists of all elements of $GF(2)^n$ except 0.

For any two vectors $\boldsymbol{a} = (a_1, a_2, \ldots, a_n)$ and $\boldsymbol{b} = (b_1, b_2, \ldots, b_n)$, denote:

- Addition of $\boldsymbol{a}$ and $\boldsymbol{b}$ as $\boldsymbol{a} \oplus \boldsymbol{b} = (a_1 \oplus b_1, a_2 \oplus b_2, \ldots, a_n \oplus b_n)$.
- Scalar product of $\boldsymbol{a}$ and $\boldsymbol{b}$ as $\langle \boldsymbol{a}, \boldsymbol{b} \rangle$ defined by $\langle \boldsymbol{a}, \boldsymbol{b} \rangle = a_1 b_1 \oplus a_2 b_2 \oplus \ldots \oplus a_n b_n$.

Denote $\#A$ as the cardinality of a set A. A Boolean function in n-variable is a function from $GF(2)^n$ to $GF(2)$. B_n denotes the collection of all Boolean functions in n-variable.

For any $\mathbf{d} \in GF(2)^n$ and $c \in GF(2)$, the affine function is $l_{d,c}(\boldsymbol{x}) = \langle \boldsymbol{d}, \boldsymbol{x} \rangle \oplus c$.

There are many ways of representing Boolean functions. However, in this paper we use a truth table to represent Boolean functions. The truth table representation of an n-variable Boolean function f denoted by $f(x)$ is a 2^n bit array with elements from $\{0,1\}$, and is defined as $(f(\boldsymbol{x_1}), f(\boldsymbol{x_2}), \ldots, f(\boldsymbol{x_{2^n}}))$ where $\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_{2^n}}$ are the elements from $GF(2)^n$ in lexicographical order. Similarly, define $f(x \oplus \boldsymbol{a})$ as $(f(\boldsymbol{x_1} \oplus \boldsymbol{a}), f(\boldsymbol{x_2} \oplus \boldsymbol{a}), \ldots, f(\boldsymbol{x_{2^n}} \oplus \boldsymbol{a}))$.

The Algebraic normal form of f at $\boldsymbol{x} = (x_1, x_2, x_3, \ldots, x_n)$ is given by

$$f(\boldsymbol{x}) = \bigoplus_{\boldsymbol{a} \in GF(2)^n} b_a x_1^{a_1} x_2^{a_2} x_3^{a_3} \ldots x_n^{a_n} \tag{1}$$

where, $b_a \in GF(2)$ and $\boldsymbol{a} = (a_1, a_2, a_3, \ldots, a_n)$.

Hamming weight (or weight) of a vector, denoted by $wt(\boldsymbol{x})$, is the number of 1s in the vector $\boldsymbol{x}$. The hamming weight of a Boolean function is the weight of its truth table. An n-variable Boolean function f is balanced if $wt(f)$ is half of the cardinality of $GF(2)^n$. Hamming distance (or distance) between two functions $f, g : GF(2)^n \to GF(2)$ denoted by $d(f,g)$ is defined as $d(f,g) = wt(f \oplus g)$ (Ciungu, 2012). Support of a function f denoted by $Supp(f)$ is defined as the set, $\{\boldsymbol{x} \in GF(2)^n; f(\boldsymbol{x}) = 1\}$. The algebraic degree of a Boolean function is defined with respect to the algebraic normal form representation of that Boolean function. Algebraic degree of a function $f(x) = \bigoplus_{\boldsymbol{a} \in GF(2)^n} b_a x_1^{a_1} x_2^{a_2} x_3^{a_3} \ldots x_n^{a_n}$ is defined as

$$\max_{\boldsymbol{a} \epsilon GF(2)^n; b_a = 1} wt(\boldsymbol{a}) \tag{2}$$

Definition 1: The nonlinearity of f is given by

$$nl(f) = \min_{\boldsymbol{a} \epsilon GF(2)^n, c \in GF(2)} d(f, l_{\boldsymbol{a}}, c) \tag{3}$$

Definition 2: An n variable Boolean function f with $nl(f) = 2^{n-1} - 2^{\frac{n}{2}-1}$ is called a bent function.

These functions are defined only for positive even integer n.
The weight of these functions is either $2^{n-1} - 2^{\frac{n}{2}-1}$ or $2^{n-1} + 2^{\frac{n}{2}-1}$. So, corresponding to each n, we can partition the collection of all bent functions into two equal halves with weights $2^{n-1} - 2^{\frac{n}{2}-1}$ and$2^{n-1} + 2^{\frac{n}{2}-1}$. In this paper, theorems, lemmas, and corollaries concerning bent functions with weight $2^{n-1} - 2^{\frac{n}{2}-1}$ have been discussed.

Autocorrelation is an important tool for analysing the cryptographic properties of Boolean functions and is defined as follows.

Definition 3: Autocorrelation of f with shift $\boldsymbol{a}$ is

$$\Delta f(\boldsymbol{a}) = \sum_{\mathbf{x} \in GF(2)^n} (-1)^{f(x) \oplus f(x \oplus \boldsymbol{a})} \tag{4}$$

Boolean functions with a low absolute value of autocorrelation for all nonzero values of a are of great importance. Hence, an indicator (termed absolute indicator) for reflecting this property is defined.

Definition 4: Absolute indicator is the maximum of autocorrelation, it defined as

$$\Delta_f = \max_{\boldsymbol{a} \epsilon GF(2)^{n*}} |\Delta_f(\boldsymbol{a})| \tag{5}$$

3. Construction of Bent Function

The collection of all $\boldsymbol{x}$ that satisfy $f(\boldsymbol{x})f(\boldsymbol{x} \oplus \boldsymbol{a}) = 1$ (the point-wise multiplication results in 1) denoted by δ_f $(\boldsymbol{a})$, has an inevitable role in the construction of the special type of functions.

Define,

$$\delta_f(\boldsymbol{a}) = \{\boldsymbol{x} \in GF(2)^n; f(\boldsymbol{x}) = f(\boldsymbol{x} \oplus \boldsymbol{a}) = 1\}.$$

The following lemma explains a relation between $wt(f)$ and $\delta_f(\boldsymbol{0})$.

Lemma 1. *For an n-variable Boolean function f*

$$\#\delta_f(\boldsymbol{0}) = wt(f) \tag{6}$$

Proof. We have

$\delta_f(\boldsymbol{0}) = \{\boldsymbol{x} \in GF(2)^n; f(\boldsymbol{x}) = f(\boldsymbol{x} \oplus \boldsymbol{0}) = 1\}$,

$= \{\boldsymbol{x}; f(\boldsymbol{x}) = 1\}$.

Hence $\#\delta_f(\boldsymbol{0}) = wt(f)$.

Lemma 2. *Let f be an n-variable Boolean function with $\delta_f(\boldsymbol{a}) \neq \emptyset$ for some $\boldsymbol{a} \in GF(2)^{n*}$. Then*

$$Supp(f) = \cup_{\boldsymbol{a} \in GF(2)^{n*}} \delta_f(\boldsymbol{a}) \tag{7}$$

Proof. We need to prove that

$\cup_{\mathbf{a} \in GF(2)^{n*}} \delta_f(\boldsymbol{a}) \subseteq Supp(f)$ and $Supp(f) \subseteq \cup_{\boldsymbol{a} \in GF(2)^{n*}} \delta_f(\boldsymbol{a})$.

Let $\boldsymbol{x} \in \delta_f(\boldsymbol{a})$ for some $\boldsymbol{a} \in GF(2)^{n*}$.

$\Rightarrow f(\boldsymbol{x}) = f(\boldsymbol{x} \oplus \boldsymbol{a}) = 1$

$\Rightarrow \boldsymbol{x} \in Supp(f)$

$\Rightarrow \delta_f(\boldsymbol{a}) \subseteq Supp(f)$.

Since a is arbitrary,

$\cup_{\mathbf{a} \in GF(2)^{n*}} \delta_f(\boldsymbol{a}) \subseteq Supp(f)$.

Let **y** be an element from $Supp(f)$. Since $\delta_f(\boldsymbol{a}) \neq \emptyset$, the set $\delta_f(\boldsymbol{a})$ has at least two elements (if **x** is there, then $\boldsymbol{x} \oplus \boldsymbol{a}$). Let $\mathbf{x}_1$ and $\mathbf{x}_2$ are two different elements in $Supp(f)$. Also assume that $\boldsymbol{x_1} \neq \boldsymbol{y}$. If $\boldsymbol{b} = \boldsymbol{x_1} \oplus \boldsymbol{y}$, then $\boldsymbol{y} \in \delta_f(\boldsymbol{b})$. This is true for all elements from $Supp(f)$.

Hence,

$Supp(f) \subseteq \cup_{\mathbf{a} \in GF(2)^{n*}} \delta_f(\boldsymbol{a})$.

The next theorem explains a relation between $\delta_f(\boldsymbol{a})$ and weight of a Boolean function.

Theorem 1. *Relation between $wt(f)$ and $\#\delta_f(\boldsymbol{a})$) Let f be a Boolean function of n-variables with $\#\delta_f(\boldsymbol{a}) = C$ and $C \neq 0$ $\forall$ $\boldsymbol{a} \in GF(2)^{n*}$ then $\frac{C \times (2^n - 1)}{wt(f) - 1} = wt(f)$ (where C is a constant).*

Proof. $wt(f) = \#Supp(f) = \# \cup_{\mathbf{a} \in GF(2)^{n*}} \delta_f(\boldsymbol{a})$

Let $\#\delta_f(\boldsymbol{a}) = C$ for all $\boldsymbol{a} \in GF(2)^{n*}$ and $\#GF(2)^{n*}$ is $2^n - 1$.

Each $\boldsymbol{x}$ with $f(\boldsymbol{x}) = 1$ is contained in every $\delta_f(\boldsymbol{a})$ with $f(\boldsymbol{x} \oplus \boldsymbol{a}) = 1$. There are exactly $wt(f)$ number of $\delta_f(\boldsymbol{a})$ with $\boldsymbol{x} \in \delta_f(\boldsymbol{a})$**.** So, each **x** will be contained in exactly $wt(f)-1$ sets (excluding $\boldsymbol{a} = \mathbf{0}$).

$$\#Supp(f) = \frac{\#\delta_f(\boldsymbol{a})\times number\;of\;sets}{number\;of\;sets\;with\,each\,x\;contained} = \frac{C\times(2^n-1)}{wt(f)-1} \tag{8}$$

Hence, the relation.

Note: Suppose $C = 0$ in Theorem 1, then the function will be either with weight one or zero.

3.1 Characterization of Bent Functions

The following theorem gives the characterization of bent functions with respect to absolute indicator.

Theorem 2. *A Boolean function f is bent if and only if the absolute indicator of f is zero, that is* $\Delta_f = 0$ (Zhang and Zheng, 1996).

The set $\delta_f(\boldsymbol{a})$ and autocorrelation have some similarities. Theorem 3, Theorem 4 and Corollary 1 illustrate a connection between $\delta_f(\boldsymbol{a})$ and autocorrelation.

Theorem 3. For an n-variable Boolean function, following are equivalent.

i. $\#\delta_f(\boldsymbol{a}) = 2^{n-2} \pm 2^{\frac{n}{2}-1}\ \forall\, \boldsymbol{a} \in GF(2)^{n*}$.

ii. $wt(f) = 2^{n-1} \pm 2^{\frac{n}{2}-1}$ and f is bent.

Proof. Proof of the theorem can be demonstrated by considering the following two cases.
Case 1*:* Let f be a bent function of n-variables with weight $2^{n-1}-2^{\frac{n}{2}-1}$.
Since $wt(f(x)) = wt(f(x \oplus \boldsymbol{a}))\ \forall \boldsymbol{a} \in GF(2)^n$,
$wt(f(x \oplus \boldsymbol{a}))$ is also $2^{n-1}-2^{\frac{n}{2}-1}$.

Since f is bent,

$$wt(f(x) \oplus (f(x \oplus \boldsymbol{a})) = 2^{n-1}\ [Theorem\ 2] \tag{9}$$

$$= wt(f(x)) + wt(f(x \oplus \boldsymbol{a})) - 2 \times \#\delta_f(\boldsymbol{a})\ (Reid, 2021) \tag{10}$$

$$= \left(2^{n-1}-2^{\frac{n}{2}-1}\right) + (2^{n-1}-2^{\frac{n}{2}-1}) - 2 \times \#\delta_f(\boldsymbol{a}) \tag{11}$$

$\Rightarrow \#\delta_f(\boldsymbol{a}) = 2^{n-2} - 2^{\frac{n}{2}-1}$.

Let $\#\delta_f(\boldsymbol{a}) = 2^{n-2} - 2^{\frac{n}{2}-1}\ \forall\, \boldsymbol{a} \in GF(2)^{n*}$.

We get $wt(f) = 2^{n-1} - 2^{\frac{n}{2}-1}$. [Theorem 1]

$$wt(f(x) \oplus (f(x \oplus \boldsymbol{a})) = wt(f(x)) + wt(f(x \oplus \boldsymbol{a})) - 2 \times \#\delta_f(\boldsymbol{a}) \tag{12}$$

$$= \left(2^{n-1}-2^{\frac{n}{2}-1}\right) + \left(2^{n-1}-2^{\frac{n}{2}-1}\right) - \left(2^{n-1}-2^{\frac{n}{2}}\right) \tag{13}$$

$$= 2^{n-1} \forall\, \boldsymbol{a} \in GF(2)^n \tag{14}$$

$$\Rightarrow \Delta_f = 0 \tag{15}$$

$\Rightarrow f$ is bent and $wt(f) = 2^{n-1} - 2^{\frac{n}{2}-1}$.

Case 2: $\#\delta_f(\boldsymbol{a}) = 2^{n-2} + 2^{\frac{n}{2}-1} \forall\, \boldsymbol{a} \in GF(2)^{n*}$ if and only if f is a bent function with weight $2^{n-2} + 2^{\frac{n}{2}-1}$. The proof is similar to case 1.

Next theorem states the connection between support of a bent function and one of its properties.

Theorem 4. *Let f be an n-variable Boolean function with support X, then* $\#\delta_f(\boldsymbol{a}) = 2^{n-2} + 2^{\frac{n}{2}-1} \forall\, \boldsymbol{a} \in GF(2)^{n*}$ *if and only if each element in $GF(2)^{n*}$ can be expressed as the sum of two vectors of X in $\frac{2^{n-2} \pm 2^{\frac{n}{2}-1}}{2}$ ways.*

Proof. Let f be an n-variable Boolean function with support X. The theorem can be proved by considering two cases.

Case 1: Let $\#\delta_f(\boldsymbol{a}) = 2^{n-2} - 2^{\frac{n}{2}-1} \forall\, \boldsymbol{a} \in GF(2)^{n*}$.
$\#\,\delta_f(\boldsymbol{a}) = \{\boldsymbol{x} \in GF(2)^n; f(\boldsymbol{x}) = f(\boldsymbol{x} \oplus \boldsymbol{a}) = 1\}$
$\boldsymbol{x} \in \delta_f(\boldsymbol{a}) \Rightarrow \boldsymbol{x} \oplus \boldsymbol{a} \in \delta_f(\boldsymbol{a})$
$\Rightarrow \boldsymbol{x}, \boldsymbol{x} \oplus \boldsymbol{a} \in X$ *and* $\boldsymbol{x} \oplus (\boldsymbol{x} \oplus \boldsymbol{a}) = \boldsymbol{a}$.
Corresponding to each $\boldsymbol{a} \in GF(2)^{n*}$, there exist $\frac{2^{n-2} - 2^{\frac{n}{2}-1}}{2}$ pairs $\boldsymbol{x}, \boldsymbol{y}$ such $\boldsymbol{x} \oplus \boldsymbol{y} = \boldsymbol{a}$.
Suppose each element of $GF(2)^{n*}$ can be expressed as the sum of two elements of X in exactly $\frac{2^{n-2} - 2^{\frac{n}{2}-1}}{2}$ ways, then for any $\boldsymbol{a} \in GF(2)^{n*}$, $\boldsymbol{y} = \boldsymbol{x} \oplus \boldsymbol{y}$ for exactly $\frac{2^{n-2} - 2^{\frac{n}{2}-1}}{2}$ pairs $\boldsymbol{x}, \boldsymbol{y} \in X$, all these $\boldsymbol{x}, \boldsymbol{y} \in \delta_f(\boldsymbol{y})$, implies $\#\delta_f(\boldsymbol{y}) = 2^{n-2} - 2^{\frac{n}{2}-1} \forall\, \boldsymbol{y} \in GF(2)^{n*}$.
Case 2: $\#\delta_f(\boldsymbol{y}) = 2^{n-2} + 2^{\frac{n}{2}-1} \forall\, \boldsymbol{y} \in GF(2)^{n*}$ if and only if each element in $GF(2)^{n*}$ can be expressed as the sum of two vectors of X in $\frac{2^{n-2} + 2^{\frac{n}{2}-1}}{2}$ different ways. Proof is similar to case 1.

The following corollary is a direct consequence of Theorem 3 and Theorem 4.

Corollary 1 (Characterization for support of a bent function)**.** *For an n-variable Boolean function f with support X, f is bent if and only if each element in $GF(2)^{n*}$ can be expressed as the sum of two vectors of X in $\frac{2^{n-2} \pm 2^{\frac{n}{2}-1}}{2}$ different ways.*

Additionally, bent functions with $wt(f) = 2^{n-1} + 2^{\frac{n}{2}-1}$ can be created using bent functions with weight $wt(f) = 2^{n-1} - 2^{\frac{n}{2}-1}$ by adding $2^{\frac{n}{2}-1}$ elements to the support with respect to the proposed characterization criteria.

Suppose f is a bent function with weight $wt(f) = 2^{n-1} - 2^{\frac{n}{2}-1}$ then $f \oplus 1$ is a bent function with weight, $wt(f) = 2^{n-1} + 2^{\frac{n}{2}-1}$.

The following details explain the construction of support of bent function from a given support of bent function.

Definition 5. *Let* $X \subset GF(2)^n$. *Then for any* $\boldsymbol{a} \in GF(2)^n$,*we define* $X \oplus \boldsymbol{a} = \{\boldsymbol{x} \oplus \boldsymbol{a}; \boldsymbol{x} \in X\}$.

Lemma 3. *Suppose* $X \subset GF(2)^n$ *has the property that each element in* $GF(2)^{n*}$ *can be expressed as the sum of two different elements of* X *in exactly* $\frac{2^{n-2} \pm 2^{\frac{n}{2}-1}}{2}$ *ways, then* $X + \boldsymbol{a}$ *has also the same property.*

Proof. Let $X \subset GF(2)^n$. The lemma can be proved by considering the following two cases.
*Case*1: Suppose X has the property that each element in $GF(2)^{n*}$ can be expressed as the sum of two different elements of X in exactly $\frac{2^{n-2} \pm 2^{\frac{n}{2}-1}}{2}$ ways, and let $\boldsymbol{c} \in GF(2)^{n*}$ be an arbitrary element. By the property of X, there exists $2^{n-1} - 2^{\frac{n}{2}-1}$ number of pair of elements $\boldsymbol{x_1}, \boldsymbol{x_2} \in X$ with $\boldsymbol{c} = \boldsymbol{x_1} \oplus \boldsymbol{x_2}$, and subsequently $\boldsymbol{x_1} \oplus \boldsymbol{a}, \boldsymbol{x_2} \oplus \boldsymbol{a} \in X \oplus \boldsymbol{a}$. Clearly,

$$\begin{aligned}(\boldsymbol{x_1} \oplus \boldsymbol{a}) \oplus (\boldsymbol{x_2} \oplus \boldsymbol{a}) &= (\boldsymbol{x_1} \oplus \boldsymbol{x}_2) \oplus (\boldsymbol{a} \oplus \boldsymbol{a}) \\ &= \boldsymbol{x_1} \oplus \boldsymbol{x}_2 \\ &= \boldsymbol{c} \end{aligned} \tag{16}$$

Case 2: Suppose X has the property that each element in $GF(2)^{n*}$ can be expressed as the sum of two distinct elements of X in exactly $\frac{2^{n-2}+2^{\frac{n}{2}-1}}{2}$ ways, then $X \oplus \boldsymbol{a}$ also has the same property. Proof is similar to case 1.

Theorem 5. *Suppose* $X \subset GF(2)^n$ *has the property that each element in* $GF(2)^{n*}$ *can be expressed as the sum of two distinct elements of X in exactly* $\frac{2^{n-2} \pm 2^{\frac{n}{2}-1}}{2}$ *ways, then*

$$\#(X \oplus \boldsymbol{a}) \cap (X \oplus \boldsymbol{b}) = \begin{cases} 2^{n-1} \pm 2^{\frac{n}{2}-1}, if\ \boldsymbol{a} = \boldsymbol{b} \\ 2^{n-2} \pm 2^{\frac{n}{2}-1}, if\ \boldsymbol{a} \neq \boldsymbol{b} \end{cases} \tag{17}$$

Proof. Let $X \subset GF(2)^n$. The theorem can be proved by considering two cases.
Case 1: Each element in $GF(2)^{n*}$ can be expressed as the sum of two distinct elements of X in exactly $\frac{2^{n-2}-2^{\frac{n}{2}-1}}{2}$ ways.
Subcase 1: $\boldsymbol{a} = \boldsymbol{b}$.

Clearly

$X \oplus \boldsymbol{a} = X \oplus \boldsymbol{b}$
$\#X \oplus \boldsymbol{a} = \#X = 2^{n-1} - 2^{\frac{n}{2}-1}$

Subcase 2: $\boldsymbol{a} \neq \boldsymbol{b}$.

$\boldsymbol{x} \in (X \oplus \boldsymbol{a}) \cap (X \oplus \boldsymbol{b})$ *if* $\boldsymbol{x} = \boldsymbol{x_1} \oplus \boldsymbol{a}$ *and* $\boldsymbol{x} = \boldsymbol{x_2} \oplus \boldsymbol{b}$; $\boldsymbol{x_1}, \boldsymbol{x_2} \in X$
$\Rightarrow \boldsymbol{x_1} \oplus \boldsymbol{a} = \boldsymbol{x_2} \oplus \boldsymbol{b}$
$\Rightarrow \boldsymbol{x_1} \oplus \boldsymbol{x_2} = \boldsymbol{a} \oplus \boldsymbol{b}$

Since $\boldsymbol{a} \neq \boldsymbol{b}, \boldsymbol{a} \oplus \boldsymbol{b} \neq \boldsymbol{0}$
$\Rightarrow \boldsymbol{a} \oplus \boldsymbol{b} \in GF(2)^{n*}$.

Hence, $\boldsymbol{a} \oplus \boldsymbol{b}$ can be expressed as the sum of two distinct elements of X in exactly $\frac{2^{n-2}-2^{\frac{n}{2}-1}}{2}$ ways.

Corresponding to each representation, there will be two elements in $(\boldsymbol{x} \oplus \boldsymbol{a}) \cap (\boldsymbol{x} \oplus \boldsymbol{b})$ (Example: If $\boldsymbol{x_1} \oplus \boldsymbol{x_2} = \boldsymbol{a} \oplus \boldsymbol{b}$ then $\boldsymbol{x_1} \oplus \boldsymbol{a} = \boldsymbol{x_2} \oplus \boldsymbol{b}$ and $\boldsymbol{x_2} \oplus \boldsymbol{a} = \boldsymbol{x_1} \oplus \boldsymbol{b}$, hence two elements in $(X \oplus \boldsymbol{a}) \cap (X \oplus \boldsymbol{b})$ are $\boldsymbol{x_1} \oplus \boldsymbol{a}$ and $\boldsymbol{x_2} \oplus \boldsymbol{a}$)

Hence if $\boldsymbol{a} \neq \boldsymbol{b}$, we have
$\#(X \oplus \boldsymbol{a}) \cap (X \oplus \boldsymbol{b}) = 2^{n-2} - 2^{\frac{n}{2}-1}$.

Case 2: If X has the property that each element in $GF(2)^{n*}$ can be expressed as the sum of two different elements of X in exactly $\frac{2^{n-2}+2^{\frac{n}{2}-1}}{2}$ ways, then

$$\#(X \oplus \boldsymbol{a}) \cap (X \oplus \boldsymbol{b}) = \begin{cases} 2^{n-1} + 2^{\frac{n}{2}-1}, if\ \boldsymbol{a} = \boldsymbol{b} \\ 2^{n-2} + 2^{\frac{n}{2}-1}, if\ \boldsymbol{a} \neq \boldsymbol{b} \end{cases} \tag{18}$$

Proof is similar to case 1.

The following result is a consequence of Theorem 5.

Corollary 2. *Suppose $X \subset GF(2)^n$ has the property that each element in $GF(2)^{n*}$ can be expressed as the sum of two distinct elements of X in exactly $\frac{2^{n-2}\pm 2^{\frac{n}{2}-1}}{2}$ ways, then $X \oplus \boldsymbol{a} = X \oplus \boldsymbol{b}$ if and only $\boldsymbol{a} = \boldsymbol{b}$.*

Using the above corollary, corresponding to the support of each bent function, it is feasible to make 2^n different bent functions with the same weight. Furthermore, an n-variable function $f \oplus 1$ is bent if and only if f is bent. Effectively, it is possible to generate 2^{n+1}bent functions with each bent function.

The subsequent subsection is dedicated to the construction of bent functions, employing the characterization criteria described earlier.

3.2 Primary Construction of Bent Functions

By leveraging the properties related to the support of bent functions, it becomes feasible to construct bent functions. For this purpose, the following algorithm (Algorithm 1) is developed. In this context, our endeavour is to transform an initially empty set into a support structure for a bent function through a process of extension.

Hence $f(t) = \begin{cases} 1, if\ t \in X \\ 0, otherwise \end{cases}$

Here $(f(t))$ is the truth table representation of a bent function with $wt(f) = 2^{n-1} - 2^{\frac{n}{2}-1}$.

If f is a bent function of n-variables with weight $wt(f) = 2^{n-1} - 2^{\frac{n}{2}-1}$, then $f \oplus 1$ is a bent function with weight $wt(f) = 2^{n-1} + 2^{\frac{n}{2}-1}$. So, it is able to build bent functions with $wt(f) = 2^{n-1} + 2^{\frac{n}{2}-1}$ from bent functions with $wt(f) = 2^{n-1} - 2^{\frac{n}{2}-1}$ by XORing 1 to the determined function.

Algorithm 1 Construction of bent functions

```
Data: Even integer $n > 2$
Result: Support of a bent function $X$
$B \leftarrow GF(2)^n$
$X \leftarrow \emptyset$
$Z \leftarrow \emptyset$ is a multiset
$A \leftarrow \emptyset$
$k \leftarrow 2^{n-1} - 2^{\frac{n}{2}-1}$
$l \leftarrow \frac{2^{n-1}-2^{\frac{n}{2}-1}}{2}$
for $i\ in\ range\ (0, k)$ do
    choose $x_c \in B$ (random)
    $B.remove\ (x_c)$
    for $x \in X$ do
        $Z = Z.add(x \oplus x_c)$
    End
    $X = X.add(x_c)$
    for $z \in Z\ with\ frequency\ of\ z\ being\ l$ do
        $A = A.add(z)$
        for $x \in X$ do
            $B = B.remove(x \oplus z)$
        End
    End
    for $a \in A$ do
        $B = B.remove(x_c \oplus a)$
    End
    choose $comp \in \{0,1\}$ (random)
    if $comp == 0$ then
        $X = X$
    end
    Else
        $X = GF(2)^n - X$
    End
End
```

4. Count of Bent Functions

Finding the exact count of bent functions is still considered a herculean task when it is 8-variabled or more; this is because an exhaustive search is infeasible in this setting (Lakshmy et al.,2014). To overcome this issue, a lot of studies have been conducted on the approximations of bent functions (Langevin and Leander, 2011, Tokareva, 2011). The authors enumerated the count of bent functions using the counting principle and found that it agrees with the proposed algorithm.

Counting principle: Consider a process that consists of r stages. Suppose that:
(a) There are n_1 possible results for the first stage.
(b) For every possible result of the first stage, there are n_2 possible results at the second stage.

(c) More generally, for all possible results of the first $i-1$ stages, there are n_i possible results at the i^{th} stage. Then, the total number of possible results of the r-stage process is $n_1 n_2 \dots n_r$.

Let b_i be the cardinality of B at the beginning of each step in the algorithm. Since the sets are made with cardinality $2^{n-1} - 2^{\frac{n}{2}-1}$, it is conceivable to choose the first element in b_0 ways, the second element in b_1 ways, and similarly i^{th} element in b_{i-1} ways. Hence, by the counting principle, the number of functions become,

$$\prod_{i=0}^{2^{n-1}-2^{\frac{n}{2}-1}-1} b_i \tag{19}$$

Any change in the order of choosing an element will not affect the set. Therefore, it is necessary to divide the above term with $(2^{n-1} - 2^{\frac{n}{2}-1})!$. Since there are exactly the same number of bent functions with $2^{n-1} - 2^{\frac{n}{2}-1}$ and $2^{n-1} + 2^{\frac{n}{2}-1}$ elements in the support, it is required to multiply the entire equation with 2. The resulting formula is given by,

$$\frac{\prod_{i=0}^{2^{n-1}-2^{\frac{n}{2}-1}-1} b_i}{\left(2^{n-1}-2^{\frac{n}{2}-1}\right)!} \tag{20}$$

As n increases, count of bent functions will increase drastically. So, it is necessary to express count of bent functions as power of two. For this, the logarithm to the base 2 is taken on the equation (20), which yields

$$1 + \sum_{i=0}^{2^{n-1}-2^{\frac{n}{2}-1}} \log_2(b_i) - \sum_{i=0}^{2^{n-1}-2^{\frac{n}{2}-1}} \log_2(i+1) \tag{21}$$

Corresponding to this, an algorithm has been developed, which is given as (Algorithm 2).

Algorithm 2 Count of bent functions

```
Data: Even integer $n > 2$
Result: $\log_2$ of approximated count of $n$-variable bent functions
$B \leftarrow GF(2)^n$
$X \leftarrow \emptyset$
$Z \leftarrow \emptyset$ is a multiset
$A \leftarrow \emptyset$
$k \leftarrow 2^{n-1} - 2^{\frac{n}{2}-1}$
$l \leftarrow \frac{2^{n-1}-2^{\frac{n}{2}-1}}{2}$
$approx\_num \leftarrow 1$
for $i$ in range $(0,k)$ do
    $approx\_num = approx\_num + (\log_2(\#B) - \log_2(i+1))$
    choose $x_c \in B$ (random)  $B.remove(x_c)$
    for $x \in X$ do
        $Z = Z.add(x \oplus x_c)$
    End
    $X = X.add(x_c)$
    for $z \in Z$ with frequency of $z$ being $l$ do
        $A = A.add(z)$
        for $x \in X$ do
            $B = B.remove(x \oplus z)$
        End
    End
    for $a \in A$ do
        $B = B.remove(x_c \oplus a)$
    End
End
approx_num
```

The number of functions is obtained as the output of Algorithm 2. Values corresponding to each b_i (for $n = 4$) is given in Table 1.

Table 1. Number of possible elements.

b_0	16
b_1	15
b_2	14
b_3	12
b_4	8
b_5	1

4.1 Secondary Construction of Bent Functions

Since the proposed algorithm does not construct bent functions with a number of variables greater than 4, the authors introduce a procedure (Algorithm 3) to construct n-variable bent functions ($n > 4$) by applying the proposed algorithm (Algorithm 1) in combination with an existing secondary construction technique for bent functions introduced by Rothaus.

Rothaus's construction (Mesnager and Mesnager, 2016): Rothaus (1976) invented a secondary construction technique that builds a bent function of $n + 2$-variables using three bent functions of n-variables. For the n-variable bent functions g, h, k and $g \oplus h \oplus k$, the function $f(g, h, k)$ defined for every $(x_1, x_2, \boldsymbol{x}) \in GF(2)^{n+2}$ with $x_1, x_2 \in GF(2)$ and $\boldsymbol{x} \in GF(2)^n$ given by:

$$f'(g,h,k,x_1,x_2,\boldsymbol{x}) = g(\boldsymbol{x})h(\boldsymbol{x}) \oplus g(\boldsymbol{x})k(\boldsymbol{x}) \oplus h(\boldsymbol{x})k(\boldsymbol{x}) \oplus [g(\boldsymbol{x}) \oplus h(\boldsymbol{x})]x_1 \oplus [g(x) \oplus k(x)]x_2 \oplus x_1x_2 \tag{22}$$

is also a bent function.

Suppose $h(\boldsymbol{x}) = g(\boldsymbol{x})$, then $g \oplus h \oplus k = k$.

Let **Roth** be the function that maps (g, k) to $f(g, g, k)$ and denote the algorithm that produces 4-variable bent function as **func1**. The algorithm to construct an m-variable bent function ($m > 4$) is denoted as algorithm 3 and is described below.

```
Algorithm 3 Construction of $m$-variable bent functions
Data: $m$ number of variables required (even integer greater than 4)
Result: An $m$-variable bent function
$B \leftarrow GF(2)^4$
```

```
$X \leftarrow \emptyset$
$Z \leftarrow \emptyset$ is a multiset
$A \leftarrow \emptyset$
$n = \frac{m-4}{2}$ (number of iterations required)
$F \leftarrow ()$ (collection of truth table of functions)
**for** $i\ in\ range\ (0, m+1)$ **do**
    F.append(())
**End**
**for** $j\ in\ range\ (0,2^n)$ **do**
    F(0).append(**func1**)
**End**
**for** $j\ in\ range\ (0, n+1)$ **do**
    **for** $p\ in\ range\ (0, \frac{len(F[j-1])}{2})$ **do**
        F(j).append(**Roth**($F[j-1][2*p], F[j-1][2*p+1]$))
    **End**
**End**
**F**(m) (The $m$-variable bent function)
```

In algorithm 3, in the process of constructing an m-variable bent function, $2^{\frac{m-4}{2}}$ 4-variable bent functions are constructed. Using these 4-variable bent functions $\frac{2^{\frac{m-4}{2}}}{2}$ 6-variable bent functions are constructed. Similarly, using these functions it is possible to construct 8, 10, 12…, m variable bent functions.

5. Experimental Illustration

The authors make a novel attempt to create bent functions with number of variables 4, 6, 8, 10 and 12 using the Algorithm 1. However, this algorithm need not provide bent functions if the number of variables is greater than 4. For example, if the number of variables is chosen to be 6 and selected 21 elements from $GF(2)^6$ are picked as A = {000000, 000001, 000010, 000011, 000100, 000101, 000110, 000111, 001000, 001001, 001010, 001011, 001100, 010000, 010100, 011000, 011101, 100000, 100110, 110000, 110111}, then it is possible to represent each of the element in B = {000011, 000010, 000001, 000100, 001000, 001100, 001101, 001001, 000101, 000110, 000111} in six different ways as pairwise sum of two elements from A. Suppose an additional element is added to set A. In that case, it becomes possible to represent at least one of the elements in the set B in seven or more different ways, resulting in a support with 22 elements. However, it is essential to note that the support of a 6-variable bent function should ideally contain 28 elements, not 22. The algorithm will work properly for $n = 4$. Using this algorithm, it is possible to produce 4-variable bent functions, and the count of the functions that can be constructed using this technique is calculated using Algorithm 2. The count of bent functions in 4-variable that can be constructed using the algorithm is 896. It is equal to the total number of 4-variable bent functions. To date, the same could only be calculated using an exhaustive search.

To generate bent functions corresponding to each even integer n, in literature, a secondary construction is discussed. Accordingly, a combination of primary (refer to Algorithm 1) and secondary construction (refer to Algorithm 3) techniques leads to the generation of bent functions of any number of variables.

6. Conclusion

This paper presents an algorithm for constructing all 4-variable bent functions. Our proposed algorithm can precisely count the number of 4-variable bent functions, 896. Previously, determining this count required an exhaustive search, but our approach offers a more efficient and systematic method. It is also shown that this proposed algorithm can be combined with a secondary construction for constructing n variable bent functions with number of variables $n > 4$. The algorithm under consideration manifests a fundamental limitation, as it needs more direct extendibility to facilitate the principal construction of 6-variable bent functions. In this current investigation, we have undertaken a novel approach wherein the empty set is extended to encompass the underlying support structure of a bent function. It is noteworthy to highlight that a multitude of non-empty sets, as viable alternatives to the empty set, are also amenable to extension, thus serving as potential support structures for bent functions. However, the precise governing criteria dictating the feasibility of such extensions for non-empty sets remain enigmatic. Consequently, the pivotal research endeavour pertains to ascertaining the minimal prerequisites that render a non-empty set eligible for expansion into a support structure of a bent function, thus constituting an intellectually formidable challenge for the scientific community. A possible future work is to extend the proposed algorithm to 6 variables and above. The authors of this study are currently involved in developing an innovative methodology aimed at the primary construction of 6 or more variable bent functions. In parallel, they are actively engaged in a rigorous investigation to establish criteria that can effectively characterize whether a given set is eligible to be considered a subset of the support structure for a bent function. These combined efforts signify a substantial advancement in the field of bent functions, opening new avenues for further research and applications.

Conflict of Interest

The authors confirm that there is no conflict of interest to declare for this publication.

Acknowledgments

The first author would like to acknowledge the Council of Scientific and Industrial Research (CSIR) for providing research fellowship vide reference number 09/1034(0006)/2019-EMR-I. The authors would like to thank the editor and anonymous reviewers for their comments that help improve the quality of this work.

References

Budaghyan, L., Carlet, C., Helleseth, T., Kholosha, A., & Mesnager, S. (2012). Further results on Niho bent functions. *IEEE Transactions on Information Theory*, *58*(11), 6979-6985.

Carlet, C. (1994). Two new classes of bent functions. In: Helleseth, T. (ed) *Advances in Cryptology* — EUROCRYPT '93. EUROCRYPT 1993. Lecture Notes in Computer Science (Vol. 765, pp. 77-101). Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-48285-7_8.

Carlet, C. (1996). A construction of bent functions. *London Mathematical Society Lecture Note Series*, pp. 47-58.

Carlet, C. (2004). On the secondary constructions of resilient and bent functions. In: Feng, K., Niederreiter, H., Xing, C. (eds) *Coding, Cryptography and Combinatorics*. *Progress in Computer Science and Applied Logic* (Vol. 23, pp. 3-28). Birkhäuser, Basel. https://doi.org/10.1007/978-3-0348-7865-4_1.

Carlet, C., & Mesnager, S. (2011). On Dillon's class H of bent functions, Niho bent functions and o-polynomials. *Journal of Combinatorial Theory, Series A*, *118*(8), 2392-2410.

Carlet, C., Dobbertin, H., & Leander, G. (2004). Normal extensions of bent functions. *IEEE Transactions on Information Theory*, *50*(11), 2880-2885.

Carlet, C., Gao, G., & Liu, W. (2014). A secondary construction and a transformation on rotation symmetric functions, and their action on bent and semi-bent functions. *Journal of Combinatorial Theory, Series A*, *127*, 161-175.

Carlet, C., Zhang, F., & Hu, Y. (2012). Secondary constructions of bent functions and their enforcement. *Advances in Mathematics of Communications*, *6*(3), 305-314.

Ciungu, L.C. (2012). Weight and nonlinearity of Boolean functions. *Turkish Journal of Mathematics*, *36*(4), 520-529.

Climent, J.J., Garcia, F.J., & Requena, V. (2012). Construction of bent functions of 2 *k* variables from a basis of. *International Journal of Computer Mathematics, 89*(7), 863-880.

Cusick, T.W., Lakshmy, K.V., & Sethumadhavan, M. (2016). Affine equivalence of monomial rotation symmetric Boolean functions: A polya's theorem approach. *Journal of Mathematical Cryptology*, *10*(3-4), 145-156. https://doi.org/10.1515/jmc-2016-0042.

Dalai, D.K., Maitra, S., & Sarkar, S. (2009). Results on rotation symmetric bent functions. *Discrete Mathematics*, *309*(8), 2398-2409. https://doi.org/10.1016/j.disc.2008.05.017.

Dillon, J.F. (1974). *Elementary hadamard difference-sets*. University of Maryland, College Park (Doctoral Thesis).

Dobbertin, H., Leander, G., Canteaut, A., Carlet, C., Felke, P., & Gaborit, P. (2006). Construction of bent functions via Niho power functions. *Journal of Combinatorial Theory, Series A, 113*(5), 779-798. https://doi.org/10.1016/j.jcta.2005.07.009.

Gadouleau, M., Mariot, L., & Picek, S. (2023). Bent functions in the partial spread class generated by linear recurring sequences. *Designs, Codes and Cryptography*, *91*(1), 63-82. https://doi.org/10.1007/s10623-022-01097-1.

Lakshmy, K.V., Sethumadhavan, M., & Cusick, T.W. (2014). Counting rotation symmetric functions using Polya's theorem. *Discrete Applied Mathematics, 169*, 162-167. https://doi.org/10.1016/j.dam.2013.12.016.

Langevin, P., & Leander, G. (2011). Counting all bent functions in dimension eight 99270589265934370305785861242880. *Designs, Codes and Cryptography, 59*(1-3), 193-205. https://doi.org/10.1007/s10623-010-9455-z.

Leander, N.G. (2006). Monomial bent functions. *IEEE Transactions on Information Theory, 52*(2), 738-743. https://doi.org/10.1109/TIT.2005.862121.

Li, N., Helleseth, T., Tang, X., & Kholosha, A. (2013). Several new classes of bent functions from Dillon exponents. *IEEE Transactions on Information Theory, 59*(3), 1818-1831. https://doi.org/10.1109/TIT.2012.2229782.

Meier, W., & Staffelbach, O. (1990). Nonlinearity criteria for cryptographic functions. In: Quisquater, J.J., & Vandewalle, J. (eds) *Advances in Cryptology* — EUROCRYPT '89. EUROCRYPT 1989. Lecture Notes in Computer Science (Vol. 434, pp. 549-562). Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-46885-4_53.

Mesnager, S., & Mesnager (2016). *Bent functions*. Springer International Publishing, Switzerland. https://doi.org/10.1007/978-3-319-32595-8.

Mesnager, S., Ozbudak, F., & Sinak, A. (2021). Secondary constructions of (non) weakly regular plateaued functions over finite fields. *Turkish Journal of Mathematics, 45*(5), 2295-2306.https://doi.org/10.3906/mat-2104-5.

Neethu, R., Sindhu, M., & Srinivasan, C. (2018). XUBA: An authenticated encryption scheme. In: Satapathy, S., Bhateja, V., Raju, K., Janakiramaiah, B. (eds) *Data Engineering and Intelligent Computing*. Advances in Intelligent Systems and Computing (Vol. 542, pp. 647-655). Springer, Singapore. https://doi.org/10.1007/978-981-10-3223-3_62.

Nyberg, K. (1991). Constructions of bent functions and difference sets. In: Damgård, I.B. (ed) *Advances in Cryptology* — EUROCRYPT '90. EUROCRYPT 1990. Lecture Notes in Computer Science (Vol. 473, pp. 151-160). Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-46877-3_13.

Rothaus, O.S. (1976). On "bent" functions. *Journal of Combinatorial Theory, Series A, 20*(3), 300-305. https://doi.org/10.1016/0097-3165(76)90024-8.

Srinivasan, C., Lakshmy, K., & Sethumadhavan, M. (2011). Complexity measures of cryptographically secure boolean functions. In: Santanam, R., Sethumadhavan, M., & Virendra, M. (eds) *Cyber Security, Cyber Crime and Cyber Forensics: Applications and Perspectives*, IGI Global, pp. 220-230. https://doi.org/10.4018/978-1-60960-123-2.ch015.

Tokareva, N. (2011). On the number of bent functions from iterative constructions: Lower bounds and hypotheses. *Advances in Mathematics of Communications*, *5*(4), 609-621. https://doi.org/10.3934/amc.2011.5.609.

Zhang, X.M., & Zheng, Y. (1996). GAC-the Criterion for global avalanche characteristics of cryptographic functions. In: Maurer, H., Calude, C., Salomaa, A. (eds) *J.UCS The Journal of Universal Computer Science*. pp 320–337. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-80350-5_30

Publisher's Note- Ram Arti Publishers remains neutral regarding jurisdictional claims in published maps and institutional affiliations.