The 8th International Workshop on
Boolean Functions and their Applications
BFA2023

Book of Abstracts

September 3-8, 2023
Voss, Norway

IN MEMORIAM

Kai-Uwe Schmidt

Prof. Dr. Kai-Uwe Schmidt (1978 - 2023) worked at Paderborn University where he was head of the Discrete Mathematics research group. His research focused on topics in combinatorics, algebra, and number theory, and he also contributed significantly to the theory of Boolean functions.

Contents

Invited Talks

Uni/Multi variate polynomial embeddings for zkSNARKs

Guang Gong

University of Waterloo, Canada

A zero-knowledge proof is a cryptographic primitive that enables a prover to convince a verifier the validity of a mathematical statement (an NP statement) without reveal any secret inputs. A special case, called zero-knowledge Succinct Non-interactive ARgument of Knowledge (zkSNARK) is particularly designed for arithmetic circuit proof systems which have important applications in blockchain privacy. The major computations in the type of zkSNARK proofs with post-quantum security are polynomial evaluations and Lagrange interpolations over finite fields. In this talk, I will show our new work on deviation of the concrete complexities of provers, proof sizes and verifiers instead of just using big notation. Given a sequence over a finite field, in coding and sequences research, we understand that there are two representations of the sequence, one is a univariate polynomial and the other, a multivariate polynomial. This is exactly what is done in those proof systems to transform the proof of a R1CS system (more general than a circuit system) to evaluate uni/multi variate polynomials at some random points in the finite field. We will use two zkSNARK schemes, i.e., Polaris, univariate polynomial representation and Spartan, multivariate polynomial representation, as examples to show our analysis.

On Division Property and Degree Bounds

Aleksei Udovenko

Abstract

Computing or bounding the algebraic degree of iterated functions is a fundamental problem in Boolean functions. Especially important it is in symmetric-key cryptography, where a low algebraic degree of a cipher leads to the so-called integral distinguishers and key recovery attacks. Furthermore, fine-grained algebraic deficiencies such as missing monomials in the algebraic normal form can also be exploited and therefore need to be detected by the designers.

Techniques for estimating the algebraic degree and finding missing monomials significantly evolved in the recent decade. Classic approaches require only a small amount of information about the iterated functions, such as their algebraic degree, the algebraic degree of their compositional inverse, or the algebraic degree of their graph indicator. However, full knowledge of a function's structure leads to much more precise bounds. The state-of-the-art technique for exploiting this information is the so-called *division property*, alternatively described as *monomial trails*.

This talk will summarize the most influential degree bounds and show their relation to division property variants, as well as describe techniques for proving lower bounds.

An optimal universal construction of threshold implementation

Enrico Piccione

University of Bergen, Norway

Threshold implementation is a method based on secret sharing to secure the hardware implementation of cryptographic ciphers against differential power analysis (DPA) side-channel attacks. This method was proposed by Nikova, Rechberger, and Rijmen in 2006 to mitigate the leakage caused by glitches. Mathematically, a threshold implementation is a vectorial Boolean function $\mathcal{F}$ with some properties strictly related to another vectorial Boolean function F which is the target function we want to implement. There is a special interest in implementing permutations F over $\mathbb{F}_2^n$ because of their application in SPN ciphers. The need to satisfy those properties make constructing $\mathcal{F}$ a challenging problem especially when F is large in size. Another problem, is to provide threshold implementations with the theoretical minimum number of Boolean shares s, which must be greater or equal than $t+1$ where t is the algebraic degree of F. In this talk, we present the first universal threshold implementation with $t+2$ shares and we discuss some problems related to the construction of threshold implementations with $t+1$ shares.

Relevant classes of polynomial functions with applications to Cryptography

Daniele Bartoli
Università degli Studi di Perugia - Dipartimento di Matematica e Informatica

Abstract

A number of different polynomial functions over finite fields have relevant applications in applied areas of Mathematics, as Cryptography or Coding Theory. Among them, APN functions, PN functions, APN permutations, permutation polynomials have been widely studied in the recent years.

In order to investigate non-existence of such functions or to provide constructions of infinite families, algebraic varieties over finite fields are a useful tool. In this direction, a key ingredient is an estimate of the number of rational points of such algebraic varieties and therefore Hasse-Weil type theorems (Lang-Weil, Serre,...) play a fundamental role.

The aim of this talk is to summarize recent results in this direction.

Side-channel analysis of cryptographic implementations: Lessons learned and future directions

Lejla Batina

Radboud University, The Netherlands

Side-channel analysis has changed the field of cryptography and it became the most common cause of real-world security applications failing today. It has also shaped the way crypto competitions are run such as recently finished NIST Post-quantum and Lightweight crypto standardization processes. In this talk we give an overview of side-channel attacks on implementations of cryptography and countermeasures. We also discuss the ways in which Machine learning and AI changed the side-channel analysis landscape and attackers' capabilities in particular. We survey several examples of AI assisting with leakage evaluation and discuss the impact of it on the field and security evaluations. Finally, we also describe the way side-channel analysis threatens AI implementations e.g. neural nets architectures that are commonly used in practice. In the end, we identify some avenues for future research.

On round functions of permutation

Joan Daemen

Radboud University, The Netherlands

Permutation-based cryptography was successful the NIST SHA-3 competition and more recently also in the NIST lighweight cryptography competition. Building an efficient permutation is similar to building a good block cipher, but not quite. In this talk we take a closer look at the structure and components of round functions of successful permutations with a focus on symmetry properties.

Resemblance

Robert S. Coulter

Department of Mathematical Sciences
University of Delaware

joint work with Li-An Chen

One of the main problems in our area is that of constructing bijections with low differential uniformity. Indeed, the "Big APN problem" is familiar to all of us. In this talk we will introduce the notion of *resemblance*, which is a way of comparing how close two functions are from each other. One can apply the concept in a variety of situations. We will concentrate mostly (if not exclusively) on the concept of *permutation resemblance* (P-Res), which provides a new way of measuring how close a function is to being a permutation. P-Res provides some advantages over historical tools for measuring a function's "bijectiveness", especially in the case where one is concerned about constructing bijections with low differentially uniformity. However, it also offers some disadvantages, perhaps most notably being that it is not immediately clear how to compute the P-Res of a function. To this end, we will describe a linear programming method showing how to resolve this issue. We will also present some computational and theoretical results.

Accepted Abstracts

Truncated rotation symmetric Boolean functions Extended Abstract

Thomas W. Cusick[a] *, Younhwan Cheon[b] †

[a]Department of Mathematics, University at Buffalo
244 Mathematics Bldg., Buffalo, NY 14260
[b]Department of Defence System Science, Korea Army Academy at YeongCheon
135-9, Hoguk-ro, Gogyeong-myeon, Yeongcheon-si, Gyeongsangbuk-do, Republic of Korea,38900

June 26, 2023

1 Introduction

Let $\mathbf{V}_n$ denote the set of all $n-$tuples $(x_1, \ldots, x_n)$ with entries in $GF(2)$ and let B_n denote the set of all Boolean functions g_n in n variables. We use $wt(g)$ for the (Hamming) weight of a Boolean function g and we say that any function in B_n is balanced if its weight is 2^{n-1}.

Definition 1. *Let ρ denote the cyclic shift defined on $\mathbf{V}_n$ by $\rho((x_1, \ldots, x_n)) = (x_2, \ldots, x_n, x_1)$. A function $g \in B_n$ is called rotation symmetric (RS) if and only if for any $(x_1, \cdots, x_n) \in \mathbf{V}_n$, $g(x_1, \cdots, x_n) = g(\rho^k(x_1, \cdots, x_n))$ for any $1 \leq k \leq n$. It is called monomial rotation symmetric (MRS) if it is generated by a single monomial.*

Rotation symmetric functions are important because of their applications in cryptography (see [10, Section 6.2], which has about 16 pages devoted to the history of the research on these functions), and more generally in some algorithms using Boolean functions whose efficient evaluation is necessary.

Any quadratic MRS function $g(x)$ in n variables can be written as

$$(1, j)_n = g_{n,j}(x) = x_1 x_j + x_2 x_{j+1} + \cdots + x_n x_{j-1} \tag{1}$$

*email: cusick@buffalo.edu
†email: yhcrypt@gmail.com

for some j with $2 \le j \le \lceil \frac{n+1}{2} \rceil$, or, in the special case when n is even and $j = \frac{n}{2} + 1$, as

$$g_{n,\frac{n}{2}+1}(x) = x_1 x_{\frac{n}{2}+1} + x_2 x_{\frac{n}{2}+2} + \cdots + x_{\frac{n}{2}} x_n. \tag{2}$$

These functions g_n are called *bent functions* and this is equivalent to saying $wt(g_n) = 2^{n-1} \pm 2^{(n/2)-1}$ (see [10, Def. 5.1, p. 84]).

Definition 2. *A modified MRS function $f \in B_n$ is called truncated rotation symmetric (TRS) if the function stops the expansion for the n-variable MRS function at the term where x_n first occurs.*

Thus any quadratic TRS function $f(x)$ in n variables can be written as

$$[1, j]_n = f_{n,j}(x) = x_1 x_j + x_2 x_{j+1} + \cdots + x_{n-j+1} x_n \tag{3}$$

for some j with $2 \le j \le \lceil \frac{n+1}{2} \rceil$.

For example, $(1, 2)_5 = g_{5,2}(x) = x_1x_2 + x_2x_3 + x_3x_4 + x_4x_5 + x_5x_1$ and $[1, 2]_5 = f_{5,2}(x) = x_1x_2 + x_2x_3 + x_3x_4 + x_4x_5$.

The TRS functions are important because they play an important role in the algorithm that enables the computation of linear recursion relations for the weights of any MRS or TRS function. The algorithm is explained in detail in [9] and a Mathematica program which performs the algorithm is given in [8]. It turns out that the recursion relations for any MRS function also apply to the corresponding TRS function (with different weights for the two functions), but it is much simpler to describe (and program) the algorithm for the TRS case. This was first observed, for degree 3 MRS functions only, in [2], but the generalization to arbitrary RS and TRS functions of any degree was not achieved until [8, 9]. There has been much work on various extensions and generalizations of this work since 2012, for instance [3, 4, 5, 11, 12].

It seemed for some time that the algorithm of [9] was not needed in the quadratic MRS case, since the work of [13] in 2009 already gave easy ways to directly compute the weight and nonlinearity for the functions $(1, j)_n$ in (1). However it was shown in [6, 7], using new ideas, that combining the algorithm with the results of [13] leads to very complete information about the weight and nonlinearity of the quadratic MRS functions, and also a complete determination of those n for which any function $(1, j)_n$ is balanced. The purpose of this paper is to obtain new results about the TRS functions $[1, j]_n$ in order to more fully understand the connections between

those functions and the MRS functions. This paper shows that in some ways the TRS theory is more complicated than the MRS theory, but in other ways it is simpler. In particular we prove a precise formula for the generating function of the sequence of weights for the TRS functions which is simpler than the corresponding formula for the weights of the MRS functions. For details of the latter formula, see [7, Theorem 5.4].

2 Preliminaries

We shall also need the concept of *Walsh transform*. The Walsh transform of a function g in n variables is the map $W(g) : \mathbf{V}_n \to R$ defined for $w \in \mathbf{V}_n$ by

$$W(g)(w) = \sum_{x \in \mathbf{V}_n} (-1)^{g(x)+w \cdot x},$$

where the values of g are taken to be the real numbers 0 and 1. The integers $W(g)(w)$ are called *Walsh values*. We are especially interested in the Walsh values for $w = \mathbf{0} = (0, \ldots, 0)$ because of the well known [10, Lemma 2.10] fact

$$wt(g_n) = 2^{n-1} - \frac{1}{2} W(g_n)(\mathbf{0}). \tag{4}$$

We need the definition of a *plateaued* Boolean function (see [10, pp. 78-79] for some history). We say that a Boolean function function $g = g_n$ in n variables is *v-plateaued* if every Walsh value $W(g)(w)$ is either 0 or $\pm 2^{(n+v)/2}$ and we say that $v = v(n)$ is the v-value of g_n or that $v(n)$ is one of the v-values for g. It is well known that any quadratic Boolean function is plateaued. A discussion of v-values for ordinary RS quadratic functions is in [6, pp. 1310-1311] and a discussion for a much broader class of functions is in [1] (that paper uses s instead of our $v(n)$).

3 The v-values for quadratic TRS functions

In this section we find all of the v-values for the functions $[1, j]_n$ and we determine every element in the period for those values. Extending this work to other quadratic TRS functions seems to require new ideas. We first need the following lemma which gives the values of n for which $[1, j]_n$ is a bent function, and more.

Lemma 1. *The functions $f_{n,j} = [1, j]_n$ are bent, and in fact $W(f_{n,j})(\boldsymbol{0}) = 2^{n/2}$, for $n = (2j-2)k,\ k \geq 1$. The functions $f_{n,j}$ have $W(f_{n,j})(\boldsymbol{0}) = 2^{(n+j-1)/2}$ for $n = j - 1 + (2j-2)k,\ k \geq 1$.*

Theorem 1. *The sequence of the v-values for* $f_{n,j} = [1, j]_n$, *beginning at* $n = 2j - 2$, *has initial entries* $0, 1, 2, \ldots, j-2, j-1, j-2, j-3, \ldots, 2, 1$ *and is periodic with period* $2j - 2$.

Theorem 2. *The functions* $f_n(x) = [1, j]_n$ *are never balanced for* $n \geq 2j-2$.

We let $G(f)$ denote any closed formula for the generating function $gen(f)$ of f, where $gen(f) = \sum_{i=1}^{\infty} wt(f_n)x^{n-1}$. We shall only use this notation for truncated RS functions. The next theorem determines $G([1, t])$ for all $t \geq 2$.

Theorem 3. *For* $f_n = [1, t]_n, t \geq 2$, *We have*

$$G(f) = \frac{(\sum_{i=0}^{t-2} x^i)2^{t-2}x^{t-1}}{(1-2x)(1-2^{t-1}x^{2(t-1)})} = \frac{(1-x^{t-1})2^{t-2}x^{t-1}}{(1-x)(1-2x)(1-2^{t-1}x^{2(t-1)})}$$

The examples below include a sum of two TRS functions, though we cannot yet prove the extension of Theorem 3 to those cases. The obstacles include generalizing Theorem 1 and finding a formula for the numerator of the rational function $G(f)$ when f has more than one TRS function.

Example 1. *For* $f_n = [1, 2]_n$, *we have*

$$G(f) = \frac{x}{(1-2x)(1-2x^2)}$$

$gen([1, 2])(x) = x+2x^2+6x^3+12x^4+28x^5+56x^6+120x^7+240x^8+496x^9+\cdots$

Example 2. *For* $f_n = [1, 2]_n + [1, 3]_n$, *we have*

$$G(f) = \frac{4x^3(2-4x+5x^2-10x^3+8x^4)}{(1-2x)(1-2x+2x^2-4x^3+4x^4)}$$

$gen([1, 2] + [1, 3])(x) = 8x^3 + 16x^4 + 36x^5 + 72x^6 + 136x^7 + 272x^8 + 544x^9 + 1056x^{10} + 2080x^{11} + 4160x^{12} + 8256x^{13} + 16384x^{14} + \cdots$

References

[1] N. Anbar, W. Meidl and A. Topuzoglu, Idempotent and p-potent quadratic functions: distribution of nonlinearity and co-dimension, *Des. Codes Cryptogr.* 82 (2017), 265-291.

[2] A. Brown and T. W. Cusick, Recursive weights for some Boolean functions, *J. Math. Cryptol.* 6 (2012), 105-135.

[3] F. Castro, R. Chapman,L. Medina, L. Sepulveda and L. Brehsner, Recursions associated to trapezoid, symmetric and rotation symmetric functions over Galois fields, *Discrete Math.* 341 (2018), 1915–1931.

[4] F. Castro and L. Medina, Modular periodicity of exponential sums of symmetric Boolean functions, *Discrete Appl. Math* 217 (2017), 455-473.

[5] F. Castro, L. Medina and P. Stănică, Generalized Walsh transforms of symmetric and rotation symmetric boolean functions are linear recurrent, *Appl. Algebra Eng. Commun. Comput.* 29 (2018), 433-453.

[6] A. Chirvasitu and T. W. Cusick, Affine equivalence for quadratic rotation symmetric functions, *Des. Codes Cryptogr.* 88 (2020), 1301-1329.

[7] A. Chirvasitu and T. W. Cusick, Symbolic dynamics and rotation symmetric Boolean functions, *Cryptogr. Commun.* 14 (2022), 1091-1115.

[8] T. W. Cusick, Weight recursions for any rotation symmetric Boolean functions, https://arxiv.org/abs/1701.06648, 18 pp., 2017.

[9] T. W. Cusick, Weight recursions for any rotation symmetric Boolean functions, *IEEE Trans. Inform. Theory* 64 (2018), 2962-2968.

[10] T. W. Cusick and P. Stănică, *Cryptographic Boolean Functions and Applications*, second ed. (San Diego: Academic Press, 2017). First edition 2009.

[11] A. Gomez-Flores, L. Medina, L. Pomales and C. Santiago-Calderon, Recurrences in terms of special polynomials for exponential sums of elementary symmetric polynomials over finite fields, *Integers* 23 Paper No. A11, 17 pp., 2023.

[12] A. Gomez-Flores, L. Medina and P. Stănică, Recursions for modified Walsh transforms of some families of Boolean functions, *Rocky Mountain J. Math.* 52 (4) (2022), 1355-1373.

[13] H. Kim, S.-M. Park and S. G. Hahn, On the weight and nonlinearity of homogeneous rotation symmetric Boolean functions of degree 2, *Discr. Appl. Math.* 157 (2009), 428-432.

A new method to represent the inverse map as a composition of quadratics in a binary finite field

Florian Luca[1,2], Santanu Sarkar[3], Pantelimon Stănică[4]

[1] School of Mathematics, University of the Witwatersrand,
Private Bag X3, Wits 2050, Johannesburg, South Africa; and
[2] Centro de Ciencias Matemáticas, UNAM,
Morelia, Mexico; Florian.Luca@wits.ac.za,
[3] Department of Mathematics, Indian Institute of Technology Madras,
Sardar Patel Road, Chennai TN 600036, INDIA; santanu@iitm.ac.in,
[4] Applied Mathematics Department, Naval Postgraduate School,
Monterey 93943, USA; pstanica@nps.edu

June 24, 2023

1 Introduction

Carlitz [1] showed that all permutation polynomials over $\mathbb{F}_q$, where $q > 2$ is a power of a prime, are generated by the special permutation polynomials x^{q-2} (the inversion) and $ax + b$ (affine functions, where $0 \neq a, b \in \mathbb{F}_q$). The smallest number of inversions in such a decomposition is called the *Carlitz rank*.

Here, we ask whether the inverse in $\mathbb{F}_{2^n}$ (the finite field of dimension n over the two-element prime field $\mathbb{F}_2$) can be written as a composition of quadratics (and suggest an extension allowing quadratics and cubics). That is, we ask if there are integers $r \geq 1$ and $a_1 \geq 0, \ldots, a_r \geq 0$ such that $-1 \equiv \prod_{i=1}^{r}(2^{a_i}+1) \pmod{2^n - 1}$. Nikova, Nikov, Rijmen [8] proposed an algorithm to find such a decomposition. Via Carlitz [1], they were able to use the algorithm and show that for $n \leq 16$ any permutation can be decomposed in quadratic permutations, when n is not multiple of 4 and in cubic permutations, when n is multiple of 4. Petrides [9], in addition to a theoretical result, which we will discuss below, improved the complexity of the algorithm and presented a computational table of shortest decompositions for $n \leq 32$, allowing also cubic permutations in addition to quadratics. Here, we extend Petrides' result, as well as we propose a number theoretical approach, which allows us to cover easily all (surely, odd) exponents up to 100, at least, with weight 2 factorizations (in the full paper we will cover up to n a few hundred). Our method is based on some hard number theoretical conjectures we propose, which allow us some inferences in our algorithmic approach. The algorithm easily extends the table of Nikova, Nikov, Rijmen [8] and Petrides [9] that covered the mentioned factorizations up to $n = 32$.

2 Our results

Let ν_2 be the 2-valuation, that is, the largest power of 2 dividing the argument. We start with a proposition, extending one of Petrides' results [9], which stated that if n is an odd integer and $\frac{n-1}{2^{\nu_2(n-1)}} \equiv$

$2^k \pmod{2^n-1}$, for some k, then,

$$2^n-2 = 2\left(\left(2^{\frac{n-1}{2^{\nu_2(n-1)}}}\right)^{2^{\nu_2(n-1)}}-1\right) = 2\left(2^{\frac{n-1}{2^{\nu_2(n-1)}}}-1\right)\prod_{j=1}^{\nu_2(n-1)}\left(2^{\frac{n-1}{2^j}}+1\right)$$
$$\equiv 2\left(2^{2^k}-1\right)\prod_{j=1}^{\nu_2(n-1)}\left(2^{\frac{n-1}{2^j}}+1\right) = 2\prod_{j=0}^{k-1}\left(2^{2^j}+1\right)\prod_{j=1}^{\nu_2(n-1)}\left(2^{\frac{n-1}{2^j}}+1\right).$$

This implies, via Carlitz [1], that for all odd integers (coined *good integers*, with the counterparts named *bad integers* in [6]) satisfying the congruence $\frac{n-1}{2^{\nu_2(n-1)}} \equiv 2^k \pmod{2^n-1}$, one can decompose any permutation polynomial in $\mathbb{F}_{2^n}$ into affine and quadratic power permutations.

The smallest odd positive integer that is not *good* is $n=7$. We note however that in that case $2^7-2 = 2(2^6-1) = 2(2^2-1)(2^4+2^2+1) = 2(2+1)(2^4+2^2+1)$, and so, any permutation in $\mathbb{F}_{2^7}$ can be decomposed into affine, quadratic and cubic permutations. We are ready to generalize this observation.

Theorem 1. *Let n be an odd integer satisfying $\frac{n-1}{2^{\nu_2(n-1)}} \equiv 2^k3^s \pmod{2^n-1}$, for some non-negative integers r,s. Then, the inverse power permutation in $\mathbb{F}_{2^n}$ has a decomposition into affine, quadratic and cubic power permutations of length $k+s+\nu_2(n-1)$.*

Proof. We use the difference of cubes factorization, $a^3-b^3 = (a-b)(a^2+ab+b^2)$, and write

$$2^n-2 = 2\left(2^{\frac{n-1}{2^{\nu_2(n-1)}}}-1\right)\prod_{j=1}^{\nu_2(n-1)}\left(2^{\frac{n-1}{2^j}}+1\right) \equiv 2\left(2^{2^k3^s}-1\right)\prod_{j=1}^{\nu_2(n-1)}\left(2^{\frac{n-1}{2^j}}+1\right)$$
$$= 2\left(2^{2^k3^{s-1}}-1\right)\left(2^{2^{k+1}3^{s-1}}+2^{2^k3^{s-1}}+1\right)\prod_{j=1}^{\nu_2(n-1)}\left(2^{\frac{n-1}{2^j}}+1\right)$$
$$\cdots\cdots\cdots\cdots$$
$$= 2\left(2^{2^k}-1\right)\prod_{j=0}^{s-1}\left(2^{2^{k+1}3^j}+2^{2^k3^j}+1\right)\prod_{j=1}^{\nu_2(n-1)}\left(2^{\frac{n-1}{2^j}}+1\right)$$
$$\equiv 2\prod_{j=0}^{k-1}\left(2^{2^j}+1\right)\prod_{j=0}^{s-1}\left(2^{2^{k+1}3^j}+2^{2^k3^j}+1\right)\prod_{j=1}^{\nu_2(n-1)}\left(2^{\frac{n-1}{2^j}}+1\right).$$

The claim is shown. □

Example 1. *It is natural to investigate the counting function $\mathcal{B}(x)$ of* superbad integers *(that is, integers n such that $\frac{n-1}{2^{\nu_2(n-1)}} \not\equiv 2^k3^s \pmod{2^n-1}$), with $\mathcal{B}(x) = \{n \le x : n \text{ is superbad}\}$, or the complement $\mathcal{A}(x) = \{n \le x : \frac{n-1}{2^{\nu_2(n-1)}} \equiv 2^k3^s \pmod{2^n-1}\}$. As an example, $|\mathcal{B}(50)| = 16$, more precisely, $\mathcal{B}(50) = \{1,2,3,4,5,7,9,10,13,17,19,25,28,33,37,49\}$ (Petrides [9] noted that 25 integers up to 50 are bad, so our extension surely prunes the integers better).*

Let $p \ge 3$ be prime, $N := N_p = 2^p-1$. It is known that if $q \mid N_p$, then $q \equiv 1 \pmod{p}$. We ask if we can say anything about the number of distinct prime factors $\omega(N_p)$ of N_p. Recall that, via Mihailescu's theorem (which solves Catalan's conjecture from 1844) [5], we know that 2^p-1 is not a (nontrivial) prime power, if $p \ge 3$. In general, we propose the following conjecture.

Conjecture 1. *There exists p_0 such that for $p > p_0$, $\omega(N_p) < 1.36 \log p$.*

Similar type of heuristics regarding lower bounds for $\Omega(2^n-1)$ and $\omega(2^n-1)$ can be found in [3] and [4]. Conjecture 1 is based on statistical arguments originating from sieve methods. It is shown in [2, Exercise 04] that for fixed $\delta > 0$ we have

$$\#\{n \le x : \omega(n) \ge (1+\delta)\log\log x\} \ll_\delta \frac{x}{(\log x)^{Q(\delta)}},$$

where $Q(\delta) := (1+\delta)\log((1+\delta)/e)+1$. We apply such heuristics to $N_p = 2^p - 1$. Note that if $q \mid N_p$, then $2^p \equiv 1 \pmod q$. In particular, $\left(\frac{2}{q}\right) = 1$, so $q \equiv \pm 1 \pmod 8$. Using a similar approach as in [2, Exercise 04] we can infer that the probability that a number having only prime factors congruent to $\pm 1 \pmod 8$ to have more than $1.36\log\log n$ distinct prime factors is $O\left(\frac{1}{(\log n)^{1.00008}}\right)$. Applying this to N_p, we get $O\left(\frac{1}{(\log(2^p-1))^{1.0008}}\right) \ll \frac{1}{p^{1.0008}}$, and since the series $\sum_{p\ge 3}\frac{1}{p^{1.0008}}$ is convergent, we are led to believe that there are at most finitely many prime numbers p such that $\omega(N_p) \ge 1.36\log p$. Perhaps infinitely often $\omega(N_p) \ge 2$. For example, this is the case if $p \equiv 3 \pmod 4$ is such that $q = 2p+1$ is prime. Indeed, then 2 is a quadratic residue modulo q so $2^{(q-1)/2} \equiv 1 \pmod q$, showing that $q \mid N_p$. Since N_p is never a perfect power, in particular it cannot be a power of q, we get the desired conclusion that $\omega(N_p) \ge 2$. The next conjecture is proposed based upon some results of Murata and Pomerance, under the Generalized Riemann Hypothesis (GRH).

Conjecture 2. *There exists p_0 such that if $p > p_0$, then N_p is squarefree.*

So, assuming Conjecture 1 and 2, let $N_p := q_1 \cdots q_k$ for some distinct primes $q_1, \ldots, q_k$ with $k \le 1.36\log p$. We take numbers of the form 2^a+1 with an odd $a \in [5, p-2]$. We want to compute $\left(\frac{2^a+1}{2^p-1}\right)$, and use a method by Rotkiewicz [10]. Precisely, we write the Euclidean algorithm with even quotients and signed remainders:

$$\begin{aligned}
p &= (2k_1)a + \varepsilon_1 r_1, \quad \varepsilon_1 \in \{\pm 1\}, \quad 1 \le r_1 \le a-1\\
a &= (2k_2)r_1 + \varepsilon_2 r_2, \quad \varepsilon_2 \in \{\pm 1\}, \quad 1 \le r_2 \le r_1 - 1,\\
\ldots &= \ldots\\
r_{\ell-2} &= (2k_\ell)r_{\ell-1} + \varepsilon_\ell r_\ell, \quad \varepsilon_\ell \in \{\pm 1\}, \quad r_\ell = 1,
\end{aligned}$$

where $\ell := \ell(a,p)$ is minimal with $r_\ell = 1$. We show in the full paper that $\left(\frac{2^a+1}{2^p-1}\right) = (-1)^{\ell+1}$. We select the subset $\mathcal{A}(p)$ of odd a in the interval $[5, p-2]$ such that $\ell \equiv 0 \pmod 2$. We assume that there are a positive proportion of such, namely that there is a constant $c_1 > 0$ such that for large p, there are $> c_1 p$ odd numbers $a \in [5, p-2]$ such that $\ell(a,p) \equiv 0 \pmod 2$. So, we have $\prod_{i=1}^k \left(\frac{2^a+1}{q_i}\right) = -1 \quad \text{for} \quad a \in \mathcal{A}(p)$. We next conjecture that for such a, the values are $\left(\left(\frac{2^a+1}{q_i}\right), 1 \le i \le k\right)$ are uniformly distributed among the 2^k vectors $\underbrace{(\pm 1, \pm 1, \cdots, \pm 1)}_{k \text{ times}}$. That is, $2^{a_i}+1$ is a quadratic residue modulo p_j for all $j \ne i$ but it is not a quadratic residue modulo q_i. In the full paper we provide an argument why we expect to find it and under the previous two conjectures the following should hold. The rest of our method is unconditional and we summarize it in the next algorithm.

Algorithm 1 works for most primes (and odd integers), and we applied it for $n \le 100$. But there are a few primes like 47 for which there is no $a_j \in [5, p-2]$ such that $\left(\frac{2^{a_j}+1}{q_i}\right) = (-1)^{\delta_{ij}}$, with Kronecker symbols as exponents. If that happens, the system may not be solvable (it has even determinant). However, experimentally, we observed that if it fails, we can always get suitable a_i's such that the corresponding matrix has odd determinant, and is therefore invertible. The factorization of $2^n - 2$ with weight 2 factors for odd $33 \le n \le 100$ is given in Table 1.

Algorithm 1:

for *prime (or odd)* $p \leq B$ *(suitable bound)* **do**
  Factor $2^p - 1 = q_1 \cdots q_k$, where q_i is prime for $1 \leq i \leq k$;
  for $j = 1$ *to* k **do**
    Find odd $a_j \in [5, p-2]$ such that the Legendre symbol $\left(\frac{2^{a_j}+1}{q_i}\right) = (-1)^{\delta_{ij}}$ where δ_{ij} is the Kronecker symbol.
  end
  Take a primitive root ρ_i modulo q_i for $1 \leq i \leq k$;
  Find b_{ij} such that $2^{a_i} + 1 = \rho_j^{b_{ij}} \pmod{q_j}$ for $1 \leq i, j \leq k$;
  Find largest α_i such that 2^{α_i} is a divisor of $q_i - 1$ for $1 \leq i \leq k$;
  Calculate $\alpha = \max\{\alpha_i : 1 \leq i \leq k\}$;
  Solve the system of linear equations $\sum_{i=1}^{k} y_i b_{ij} = 2^{\alpha_j - 1} \quad \text{for} \quad j = 1, 2, \ldots, k.$ in $\mathbb{Z}_\alpha$
end

References

[1] L. Carlitz, "Permutations in a finite field", *Proc. Amer. Math. Soc.* **4** (1953), 538.

[2] R. T. Hall and G. Tenenbaum, *Divisors*, Cambridge Tracts in Mathematics, **90**. Cambridge University Press, Cambridge, 1988.

[3] A. Kontorovich and J. Lagarias, "On toric orbits in the affine sieve", *Exp. Math.* **30** (2021), 575–587.

[4] F. Luca and P. Stănică, "Prime divisors of Lucas sequences and a conjecture of Skałba", *Int. J. Number Theory* **1** (2005), no. 4, 583–591.

[5] P. Mihăilescu, Preda (2004), "Primary Cyclotomic Units and a Proof of Catalan's Conjecture", J. Reine Angew. Math. **572** (2004), 167–195.

[6] P. Moree, "On the divisors of $a^k + b^k$", *Acta Arith.* LXXX.3 (1997), 197–212.

[7] L. Murata and C. Pomerance, "On the largest prime factor of a Mersenne number", in *Number Theory*, 209–218, CRM Proc. Lecture Notes, 36, Amer. Math. Soc., Providence, RI, 2004.

[8] S. Nicoka, V. Nikov, V. Rijmen, "Decomposition of permutations in a finite field", *Cryptogr. Commun.* **11** (2019), 379–384.

[9] G. Petrides, "On decompositions of permutation polynomials into quadratic and cubic power permutations", *Cryptogr. Commun.* **15** (2023), 199–207.

[10] A. Rotkiewicz, "Applications of Jacobi's symbol to Lehmer's numbers", *Acta Arith.* **42** (1983), 163–187.

Table 1: Factorization of $2^n - 2 \pmod{2^n - 1}$ for odd $33 \leq n \leq 99$.

$n = 33$	$(2^5 + 1)^{599478} \cdot (2^{13} + 1)^{299739} \cdot (2^{29} + 1)^{1798434}$
$n = 35$	$\Big((2 + 1)(2^{17} + 1)\Big)^{967995} \cdot (2^{29} + 1)^{276570}$
$n = 37$	$(2^5 + 1)^{77039772} \cdot (2^{13} + 1)^{19259943}$
$n = 39$	$\Big((2^{11} + 1)(2^{21} + 1)\Big)^{1592955}$
$n = 41$	$(2^9 + 1)^{20111512782} \cdot (2^{13} + 1)^{3351918797}$
$n = 43$	$\Big((2^5 + 1)(2^{17} + 1)(2^{23} + 1)\Big)^{593211015}$
$n = 45$	$(2 + 1)^{407925} \cdot (2^{13} + 1)^{349650} \cdot \Big((2^{25} + 1)(2^{33} + 1)(2^{41} + 1)\Big)^{116550}$
$n = 47$	$(2^{11} + 1)^{1927501725} \cdot (2^{37} + 1)^{435242325} \cdot (2^{41} + 1)^{1616614350}$
$n = 49$	$(2^9 + 1)^{34630287489} \cdot (2^{11} + 1)^{3393768173922}$
$n = 51$	$(1 + 2^{29})^{150009615}$
$n = 53$	$(1 + 2^5)^{6512186850} \cdot (1 + 2^{15})^{3506562150} \cdot (1 + 2^{21})^{250468725}$
$n = 55$	$(1 + 2)^{6588945} \cdot (1 + 2^{11})^{5856840} \cdot (1 + 2^{17})^{732105}$ $\cdot (1 + 2^{25})^{1464210} \cdot (1 + 2^{33})^{10249470} \cdot (1 + 2^{47})^{732105}$
$n = 57$	$(1 + 2^5)^{396029391534} \cdot (1 + 2^{17})^{1188088174602} \cdot (1 + 2^{21})^{594044087301}$ $\cdot (1 + 2^{47})^{198014695767}$
$n = 59$	$(1 + 2^7)^{3663925098759300} \cdot (1 + 2^{13})^{305327091563275}$
$n = 61$	$(1 + 2^9)^{1152921504606846975}$
$n = 63$	$(1 + 2)^{42958503} \cdot (1 + 2^5)^{3735522} \cdot (1 + 2^{39})^{56032830} \cdot$ $(1 + 2^{43})^{44826264} \cdot (1 + 2^{47})^{29884176}$
$n = 65$	$(1 + 2^{17})^{72647571779055} \cdot (1 + 2^{23})^{72647571779055} \cdot (1 + 2^{29})^{72647571779055}$
$n = 67$	$(1 + 2^5)^{15295807610659665}$
$n = 69$	$(1 + 2^{11})^{36566619637113225} \cdot (1 + 2^{17})^{2437774642474215} \cdot$ $(1 + 2^{53})^{19502197139793720} \cdot (1 + 2^{67})^{21939971782267935}$
$n = 71$	$(1 + 2^{11})^{3659326099961865} \cdot (1 + 2^{13})^{14637304399847460}$
$n = 73$	$(1 + 2^{31})^{1726845200475585} \cdot (1 + 2^{45})^{107064402429486270}$
$n = 75$	$(1 + 2)^{36654975} \cdot (1 + 2^{39})^{17832150} \cdot (1 + 2^{41})^{9906750} \cdot$ $(1 + 2^{43})^{7925400} \cdot (1 + 2^{53})^{57459150} \cdot (1 + 2^{55})^{15850800} \cdot (1 + 2^{63})^{43589700}$
$n = 77$	$(1 + 2^{25})^{290641821624556479} \cdot (1 + 2^{31})^{290641821624556479} \cdot$ $(1 + 2^{41})^{290641821624556479} \cdot (1 + 2^{67})^{581283643249112958}$
$n = 79$	$(1 + 2^9)^{12102186118644337359} \cdot (1 + 2^{15})^{12102186118644337359} \cdot$ $(1 + 2^{41})^{12102186118644337359}$
$n = 81$	$(1 + 2)^{106331083505919} \cdot (1 + 2^{25})^{155626336778778} \cdot (1 + 2^{37})^{105108887143782} \cdot$ $(1 + 2^{39})^{155626336778778} \cdot (1 + 2^{43})^{4073987873790}$
$n = 83$	$(1 + 2^{11})^{7239076764159456135965}$
$n = 85$	$(1 + 2^9)^{4760486403166879215} \cdot (1 + 2^{13})^{4760486403166879215} \cdot$ $(1 + 2^{23})^{4760486403166879215}$
$n = 87$	$(1 + 2^{39})^{3371346107168004} \cdot (1 + 2^{41})^{280945508930667} \cdot (1 + 2^{53})^{2809455089306670} \cdot$ $(1 + 2^{61})^{4214182633960005} \cdot (1 + 2^{71})^{1685673053584002} \cdot (1 + 2^{83})^{280945508930667}$
$n = 89$	$(1 + 2^{13})^{309485009821345068724781055}$
$n = 91$	$(1 + 2^{59})^{280368506850705} \cdot (1 + 2^{67})^{1682211041104230} \cdot (1 + 2^{71})^{280368506850705} \cdot$ $(1 + 2^{73})^{280368506850705} \cdot (1 + 2^{81})^{3364422082208460}$
$n = 93$	$(1 + 2^{17})^{2305843010287435773}$
$n = 95$	$(1 + 2^{43})^{7354378117756963125} \cdot (1 + 2^{51})^{7354378117756963125}$
$n = 97$	$(1 + 2^5)^{61253537018541048982516284 6} \cdot (1 + 2^9)^{10208922836423508163752714 1}$
$n = 99$	$(1 + 2)^{160190876329840719} \cdot (1 + 2^{23})^{160190876329840719} \cdot (1 + 2^{35})^{58251227756305716} \cdot$ $(1 + 2^{57})^{29125613878152858} \cdot (1 + 2^{59})^{101939648573535003} \cdot (1 + 2^{75})^{58251227756305716}$

A Class of Weightwise Almost Perfectly Balanced Boolean Functions with High Weightwise Nonlinearity

Deepak Kumar Dalai[1] and Krishna Mallick[2]
[1]School of Mathematical Sciences,
[2]School of Computer Sciences,
National Institute of Science Education and Research,
An OCC of Homi Bhabha National Institute,
Bhubaneswar, Odisha 752050, India
Email: {deepak, krishna.mallick}@niser.ac.in

Abstract

A Boolean function with good cryptographic properties over a set of vectors with constant Hamming weight is significant for stream ciphers like FLIP [MJSC16]. This paper presents a construction for weightwise almost perfectly balanced (WAPB) Boolean functions with good nonlinearity and good weightwise nonlinearities. We have presented the comparison of nonlinearity and weightwise nonlinearities with other available WAPB Boolean functions, which shows that this class of WAPB functions has the highest nonlinearities.

Keywords— Boolean function, FLIP cipher, Weightwise perfectly balanced (WPB), Weightwise almost perfectly balanced (WAPB)

1 Introduction

An n-variable Boolean function f is a mapping from the n-dimensional vector space $\mathbb{F}_2^n$ to $\mathbb{F}_2$, where $\mathbb{F}_2$ is a finite field with two elements $\{0,1\}$. Depending upon the underlying algebraic structure, the '+' symbol is used for the addition operation in both $\mathbb{F}_2$ and $\mathbb{R}$. In stream ciphers, Boolean functions are used as a filter function for generating pseudorandom sequences; in some block ciphers, these functions are used to generate round keys. In these classical ciphers, the inputs to the function reach the whole space $\mathbb{F}_2^n$, whereas for reducing multiplicative depth in lightweight ciphers, the inputs can be restricted to some subsets of $\mathbb{F}_2^n$. The inputs to the filter function that has been used in the FLIP cipher introduced in [MJSC16] are restricted to the vectors of Hamming weight $\frac{n}{2}$. The analysis of different cryptographic criteria of Boolean functions over restricted domains arises after the work of Carlet, Méaux, and Rotella in [CMR17]. Therefore to avoid the biased output, one of the important cryptographic criteria for a Boolean function is balancedness over the defined domain. Moreover, it is desirable to construct Boolean functions over the set of vectors $E_{n,k} = \{x \in \mathbb{F}_2^n : w_H(x) = k\}$ for $1 \leq k \leq n-1$ with good cryptographic properties to avoid attacks. In [CMR17], Carlet et. al introduced the concepts of weightwise perfectly balanced (WPB) and weightwise almost perfectly balanced (WAPB) functions, which are balanced over $E_{n,k}$ for all k and its cryptographic criteria like nonlinearity and algebraic immunity over $E_{n,k}$.
There are several proposed methods for constructing WAPB and WPB (see [DLR16, CMR17, LM19, MZD19, TL19, LS20, MS21, MSL21, GM22, GS22, ZS22, ZS23, DM23]) in which the nonlinearity over $E_{n,k}$ of the defined functions have been discussed. Still, there is a noticeable gap in the upper bound of nonlinearity proposed in [CMR17] over $E_{n,k}$ (i.e., weightwise nonlinearity) and the known constructions. In our construction, we have attempted to reduce the gap in weightwise nonlinearity and also nonlinearity over $\mathbb{F}_2^n$.

2 Preliminaries

Let $\mathcal{B}_n$ be the set of all n-variable Boolean functions. Let us denote $[i,j]=\{i,i+1,\ldots,j\}$ for two integers i,j with $i\leq j$. For any $v=(v_1,v_2,\ldots,v_n)\in\mathbb{F}_2^n$, the Hamming weight of v is defined as $\mathtt{wt}(v)=|\{i\in[1,n]:v_i=1\}|$. The support of a Boolean function $f\in\mathcal{B}_n$ is $\mathtt{sup}(f)=\{v\in\mathbb{F}_2^n:f(v)=1\}$ and Hamming weight of f is $\mathtt{wt}(f)=|\mathtt{sup}(f)|$. Let us denote $E_{n,k}=\{v\in\mathbb{F}_2^n:\mathtt{wt}(v)=k\}$ for every $k\in[0,n]$. The support and Hamming weight of f restricted to $E_{n,k}$ are denoted as $\mathtt{sup}_k(f)=\{v\in E_{n,k}:f(v)=1\}$ and $\mathtt{wt}_k(f)=|\mathtt{sup}_k(f)|$ respectively. The Hamming distance between two functions $f,g\in\mathcal{B}_n$ is given as $\mathtt{d}(f,g)=|\{v\in\mathbb{F}_2^n:f(v)\neq g(v)\}|=\mathtt{wt}(f+g)$ and the Hamming distance between two functions f,g restricted to $E_{n,k}$ is given as $\mathtt{d}_k(f,g)=|\{v\in E_{n,k}:f(v)\neq g(v)\}|=\mathtt{wt}_k(f+g)$. The truth table representation of a Boolean function $f\in\mathcal{B}_n$ is a 2^n-dimensional vector representation, i.e., $f=\{f(0,0,\ldots,0),f(0,0,\ldots,1),\ldots,f(1,1,\ldots,1)\}$. The algebraic normal form (ANF) representation is defined as $f(x)=\sum_{u\in\mathbb{F}_2^n}a_ux^u$, where $a_u\in\mathbb{F}_2$ and $x^u=x_1^{u_1}x_2^{u_2}\cdots x_n^{u_n}$ for $x=(x_1,x_2,\ldots,x_n)$. The algebraic degree of the Boolean function $f\in\mathcal{B}_n$ is defined as $\deg(f)=\max\{\mathtt{wt}(u):u\in\mathbb{F}_2^n,a_u\neq 0\}$. Any $f\in\mathcal{B}_n$, with $\deg(f)\leq 1$, is said to be an affine Boolean function, and the set of all affine Boolean functions in $\mathcal{B}_n$ is denoted by $\mathcal{A}_n$. A Boolean function $f\in\mathcal{B}_n$ is balanced, if $\mathtt{wt}(f)=2^{n-1}$. The nonlinearity of $f\in\mathcal{B}_n$, denoted as $\mathtt{nl}(f)$, is the minimum Hamming distance of f to any affine function. That is, $\mathtt{nl}(f)=\min_{g\in\mathcal{A}_n}\mathtt{d}(f,g)$. Similarly, all these cryptographic criteria are also defined for the n-variable Boolean function when the inputs are restricted to $E_{n,k}$.

Definition 2.1. *[CMR17] A Boolean function $f\in\mathcal{B}_n$ is said to be weightwise almost perfectly balanced (WAPB) if, for every $k\in[0,n]$, $\mathtt{wt}_k(f)=\frac{\binom{n}{k}}{2}$ if $\binom{n}{k}$ is even and $\mathtt{wt}_k(f)=\frac{\binom{n}{k}\pm 1}{2}$ if $\binom{n}{k}$ is odd.*

Definition 2.2. *[CMR17] A Boolean function $f\in\mathcal{B}_n$ is said to be weightwise perfectly balanced (WPB) if the restriction of f to $E_{n,k}$, is balanced for all $k\in[1,n-1]$, i.e., $\binom{n}{k}$ is even and $\mathtt{wt}_k(f)=\frac{\binom{n}{k}}{2}$.*

Therefore, a WPB function $f_n\in\mathcal{B}_n$ exists if $n=2^m$ and a WAPB function $f\in\mathcal{B}_n$ is called WPB Boolean function for $n=2^m$ for a nonnegative integer m. A WPB Boolean function $f\in\mathcal{B}_n$ is balanced, if $f(0,0,\ldots,0)\neq f(1,1,\ldots,1)$. Hence, there are $2\prod_{k=1}^{n-1}\left(\binom{n}{k}\atop\binom{n}{k}/2\right)$ balanced WPB Boolean functions.

Definition 2.3. *[CMR17] The nonlinearity of $f\in\mathcal{B}_n$ over $E_{n,k}$, denoted as $\mathtt{nl}_k(f)$, is the Hamming distance of f to the set of all affine functions $\mathcal{A}_n$ when evaluated over $E_{n,k}$. That is, $\mathtt{nl}_k(f)=\min_{g\in\mathcal{A}_n}d_k(f,g)=\min_{g\in\mathcal{A}_n}\mathtt{wt}_k(f+g)$.*

Let $\triangle$ be the symbol represents the symmetric difference between two sets.

Proposition 2.4. *[MS21] For a positive integer $n=2^m$, let $f_n\in\mathcal{B}_n$ with support*

$$\mathtt{sup}(f_n)=\begin{cases}\{(x,1)\in\mathbb{F}_2^2:x\in\mathbb{F}_2\}=\{(0,1),(1,1)\} & \text{if } n=2,\\ \{(x,y):x,y\in\mathbb{F}_2^{\frac{n}{2}},\mathtt{wt}(x)\text{ is odd}\}\triangle\{(z,z):z\in\mathtt{sup}(f_{\frac{n}{2}})\} & \text{if } n>2.\end{cases}$$

Then f_n is a WPB Boolean function.

Proposition 2.5. *[DM23] For $n\geq 2$, let $f_n\in\mathcal{B}_n$ with support*

$$\mathtt{sup}(f_n)=\begin{cases}\{(x,1)\in\mathbb{F}_2^2:x\in\mathbb{F}_2\}=\{(0,1),(1,1)\} & \text{if } n=2,\\ \{(x,0)\in\mathbb{F}_2^n:x\in\mathtt{sup}(f_{n-1})\}\cup\{(x,1)\in\mathbb{F}_2^n:x\notin\mathtt{sup}(f_{n-1})\} & \text{if } n>2 \text{ and odd},\\ \{(x,y)\in\mathbb{F}_2^n:x,y\in\mathbb{F}_2^{\frac{n}{2}},\mathtt{wt}(x)\text{ is odd}\}\triangle\{(z,z)\in\mathbb{F}_2^n:z\in\mathtt{sup}(f_{\frac{n}{2}})\}, & \text{if } n>2 \text{ and even}.\end{cases}$$

Then f_n is a WAPB Boolean function.

The construction proposed in Proposition 2.5 is a generalization of the construction proposed in Proposition 2.4 to get a WAPB Boolean function. The construction proposed in Proposition 2.5 is important for our study as we will provide a construction that improves its nonlinearity.

Theorem 2.6. *[DM23] Let $f_n \in \mathcal{B}_n$ $(n > 2)$, defined as in Proposition 2.5. Then $\mathtt{nl}(f_n) = 2\mathtt{nl}(f_{n-1})$ if n is odd and $\mathtt{nl}(f_n) \leq \mathtt{wt}(f_{\frac{n}{2}})$ if n is even.*

For n even, the nonlinearity of f_n is very low as $X_1 = \{(x, y) \in \mathbb{F}_2^n : x, y \in \mathbb{F}_2^{\frac{n}{2}}, \mathtt{wt}(x)$ *is odd*$\}$ is the support of a linear function $\sum_{i=1}^{\frac{n}{2}} x_i$ and the cardinality of $X_2 = \{(z, z) \in \mathbb{F}_2^n : z \in \mathtt{sup}(f_{\frac{n}{2}})\}$ is $\mathtt{wt}(f_{\frac{n}{2}})$. Further, for n even and k odd, $\mathtt{sup}_k(f_n) = \mathtt{sup}(f_n) \cap E_{n,k} = \{(x, y) \in \mathbb{F}_2^n : x, y \in \mathbb{F}_2^{\frac{n}{2}}, \mathtt{wt}(x)$ *is odd*$\} \cap E_{n,k} = \mathtt{sup}_k(\sum_{i=1}^{\frac{n}{2}} x_i)$ and hence $\mathtt{nl}_k(f_n) = 0$. Therefore, in our technique, we attempt to permute the coordinates of the vectors of weight k in X_1 to improve the nonlinearity by avoiding the linear patterns and preserving the weightwise balancedness.

3 A class of WAPB Boolean functions with good nonlinearity

In this case, $\mathtt{nl}_k(f_n) = 0$ as described above. Here, we will present a class of WAPB Boolean functions by modifying $\mathtt{sup}(f_n)$ presented in Proposition 2.5. We observed that the nonlinearity becomes weak because the $\mathtt{sup}(f_n)$ when n is even is close to a linear function. In our technique, we attempt to increase the nonlinearity by permuting the coordinates of some vectors in $\mathtt{sup}(f_n)$ when n is even.

Therefore, it is assumed that $n > 2$ and is **even** in this section. Hence, when n is even, as Proposition 2.5, $\mathtt{sup}(f_n) = \{(x, y) \in \mathbb{F}_2^n : x, y \in \mathbb{F}_2^{\frac{n}{2}}, \mathtt{wt}(x)$ *is odd*$\} \triangle \{(z, z) \in \mathbb{F}_2^n : z \in \mathtt{sup}(f_{\frac{n}{2}})\}$. Then

$$\mathtt{sup}_k(f_n) = \begin{cases} \{(x, y) \in \mathbb{F}_2^n : x, y \in \mathbb{F}_2^{\frac{n}{2}}, \mathtt{wt}(x) \text{ is odd}, \mathtt{wt}(x) + \mathtt{wt}(y) = k\} & \\ \qquad \triangle \{(z, z) \in \mathbb{F}_2^n : z \in \mathtt{sup}_{\frac{k}{2}}(f_{\frac{n}{2}})\} & \text{if } k \text{ is even} \\ \{(x, y) \in \mathbb{F}_2^n : x, y \in \mathbb{F}_2^{\frac{n}{2}}, \mathtt{wt}(x) \text{ is odd}, \mathtt{wt}(x) + \mathtt{wt}(y) = k\} & \text{if } k \text{ is odd} \end{cases}$$

Now we will consider both cases of k (i.e., odd or even) and will propose to permute the coordinates of some vectors in $\mathtt{sup}_k(f_n)$.

3.1 When k is odd

In this case, $\mathtt{sup}_k(f_n) = \{(x, y) \in \mathbb{F}_2^n : x, y \in \mathbb{F}_2^{\frac{n}{2}}, \mathtt{wt}(x)$ *is odd*$, \mathtt{wt}(x) + \mathtt{wt}(y) = k\} = \mathtt{sup}_k(\sum_{i=1}^{\frac{n}{2}} x_i)$ as we discussed at the end of Section 2. The linear function $l = \sum_{i=1}^{\frac{n}{2}} x_i$ is independent of y. We attempt to break the independence and linearity on the cordinates in y using the support of a nonlinear function $a \in \mathcal{B}_{\frac{n}{2}}$. That is, for every $x \in \mathbb{F}_2^{\frac{n}{2}}$ satisfying l (i.e., $\mathtt{wt}(x)$ is odd), we keep (x, y) if $y \in \mathrm{sup}(a)$ otherwise we replace (x, y) by (y, x). If a is a highly nonlinear function, then the component y is expected to be far from the linear functions and results a high nonlinearity in f.

Here, if $\mathtt{wt}((x, y)) = k$ then $\mathtt{wt}((y, x)) = k$. Further, if $(x, y) \in \mathrm{sup}_k(f_n)$ then $\mathtt{wt}(y)$ is even as $\mathtt{wt}(x)$ is odd. So, $(y, x) \notin \mathrm{sup}_k(f_n)$ if $(x, y) \in \mathrm{sup}_k(f_n)$. Therefore, replacement of $(x, y) \in \mathrm{sup}_k(f_n)$ by (y, x) does not change the weight of the resultant function in the domain $E_{n,k}$.

Lemma 3.1. *Let $a \in \mathcal{B}_{\frac{n}{2}}$. A function $f \in \mathcal{B}_n$ such that for every $k \in [0, n]$ and odd,*

$$\begin{aligned} \mathtt{sup}_k(f^a) \quad = \quad & \{(x, y) \in \mathbb{F}_2^n : x, y \in \mathbb{F}_2^{\frac{n}{2}}, \mathtt{wt}(x) \text{ is odd}, y \in \mathtt{sup}(a), \mathtt{wt}(y) = k - \mathtt{wt}(x)\} \\ & \cup \{(y, x) \in \mathbb{F}_2^n : x, y \in \mathbb{F}_2^{\frac{n}{2}}, \mathtt{wt}(x) \text{ is odd}, y \notin \mathtt{sup}(a), \mathtt{wt}(y) = k - \mathtt{wt}(x)\}. \end{aligned} \tag{1}$$

Then $\mathtt{wt}_k(f^a) = \frac{1}{2}\binom{n}{k}$.

3.2 When k is even

In this case, $\mathtt{sup}_k(f_n) = \{(x, y) \in \mathbb{F}_2^n : x, y \in \mathbb{F}_2^{\frac{n}{2}}, \mathtt{wt}(x)$ *is odd*$, \mathtt{wt}(x) + \mathtt{wt}(y) = k\} \triangle \{(z, z) \in \mathbb{F}_2^n : z \in \mathtt{sup}_{\frac{k}{2}}(f_{\frac{n}{2}})\}$. Let us denote the set $L = \{(x, y) \in \mathbb{F}_2^n : x, y \in \mathbb{F}_2^{\frac{n}{2}}, \mathtt{wt}(x)$ *is odd*$, \mathtt{wt}(x) + \mathtt{wt}(y) = k\}$ and $M = \{(z, z) \in \mathbb{F}_2^n : z \in \mathtt{sup}_{\frac{k}{2}}(f_{\frac{n}{2}})\}$. In this case, the replacement of $(x, y) \in \mathrm{sup}_k(f_n)$ by (y, x) is not straight

forward as in Subsection 3.1. If $(x, y) \in L$ then $\mathtt{wt}(y)$ is odd as $\mathtt{wt}(x)$ is odd. As a result, (y, x) could be present in L. Therefore, replacement of $(x, y) \in \sup_k(f_n)$ by (y, x) can possibly duplicate an existing vector in L, which reduces the weight of the resultant function. Therefore, we attempt to swap two bits x_i and y_i in stead of swapping x and y as in the following lemma. For given $(x, y) \in \mathbb{F}_2^n$ where $x = (x_1, \ldots, x_{\frac{n}{2}}), y = (y_1, \ldots, y_{\frac{n}{2}}) \in \mathbb{F}_2^{\frac{n}{2}}$, denote $(x^i, y^i) = (x_1, \ldots, x_{i-1}, y_i, x_{i+1}, \ldots, x_{\frac{n}{2}}, y_1, \ldots, y_{i-1}, x_i, y_{i+1}, \ldots, y_{\frac{n}{2}})$. That is, (x^i, y^i) is obtained by swapping the i-th bits of x and y.

Lemma 3.2. *Let $f_n \in \mathcal{B}_n$ be the function defined in Proposition 2.5. For every $k \in [0, n]$ and even, let $W_k = \{(x, y) \in \mathtt{sup}_k(f_n) | \mathtt{wt}(x)$ is odd, and there is an $i \in [1, \frac{n}{2}]$ such that $x_j = y_j$ for $1 \leq j \leq i - 1$ and $y_i = 1, x_i = 0\}$ and*
$W'_k = \{(x^i, y^i) | (x, y) \in W_k$ and $i \in [1, \frac{n}{2}]$ such that $x_j = y_j$ for $1 \leq j \leq i - 1$ and $y_i = 1, x_i = 0$ i.e., the i obtained for(x, y) in $W_k\}$.
A function $g_n \in \mathcal{B}_n$ such that $\mathtt{sup}_k(g_n) = (\mathtt{sup}_k(f_n) \setminus W_k) \cup W'_k$ for every $k \in [0, n]$ and even.
Then $\mathtt{wt}_k(g_n) = \mathtt{wt}_k(f_n)$ if k is even.

Like in Lemma 3.1, now we will use the support of another Boolean function (possibly, a highly nonlinear) to swap x^i and y^i in some of $(x^i, y^i) \in W'_k$ as defined in Lemma 3.2.

Lemma 3.3. *Let $b \in \mathcal{B}_{\frac{n}{2}}$. Let $g_n \in \mathcal{B}_n$ as defined in Lemma 3.2 with W_k and W'_k. A function $h_n^b \in \mathcal{B}_n$ such that for every $k \in [0, n]$ and even,*
$\mathtt{sup}_k(h_n^b) = \{(x, y) \in \mathtt{sup}_k(g_n) : (x, y) \notin W'_k\} \cup \{(x, y) : (x, y) \in W'_k$ and $y \in \mathtt{sup}(b)\} \cup \{(y, x) : (x, y) \in W'_k$ and $y \notin \mathtt{sup}(b)\}$.
Then $\mathtt{wt}_k(h_n^b) = \mathtt{wt}_k(g_n)$.

3.3 A class of WAPB Boolean functions

Now we will apply Lemma 3.1 and Lemma 3.3 to construct a WAPB Boolean function with improved nonlinearity.

Theorem 3.4. *Let $a, b \in \mathcal{B}_{\frac{n}{2}}$. Let $f_n \in \mathcal{B}_n$ be the function defined in Proposition 2.5. Let $F_n \in \mathcal{B}_n$ with support $\mathtt{sup}_k(F_n) = \begin{cases} \mathtt{sup}_k(h_n^b) & \text{if } k \text{ is even} \\ \mathtt{sup}_k(f_n^a) & \text{if } k \text{ is odd,} \end{cases}$*
where f_n^a, h_n^b are as defined in Lemma 3.1 and Lemma 3.3 respectively. Then F_n is a WAPB Boolean function.

The following is a recursive construction of a WAPB Boolean function.

Construction 3.5. *For $n \geq 2$, let $F_n \in \mathcal{B}_n$ with support*

$$\mathtt{sup}(F_n) = \begin{cases} \{(x, 1) \in \mathbb{F}_2^2 : x \in \mathbb{F}_2\} = \{(0, 1), (1, 1)\} & \text{if } n = 2, \\ \{(x, 0) \in \mathbb{F}_2^n : x \in \mathtt{sup}(F_{n-1})\} \cup \{(x, 1) \in \mathbb{F}_2^n : x \notin \mathtt{sup}(F_{n-1})\} & \text{if } n > 2 \text{ and odd,} \\ S_n \triangle \{(z, z) \in \mathbb{F}_2^n : z \in \mathtt{sup}(F_{\frac{n}{2}})\} & \text{if } n > 2 \text{ and even.} \end{cases}$$

Here $S_n = \cup_{k=0}^n \mathtt{sup}_k(F_n)$ and $\mathtt{sup}_k(F_n) = \begin{cases} \mathtt{sup}_k(h_n^b) & \text{if } n > 2 \text{ and even and } k \text{ is even} \\ \mathtt{sup}_k(h_n^a) & \text{if } n > 2 \text{ and even and } k \text{ is odd.} \end{cases}$

3.4 Experimental results on nonlinearity

In this section, we have presented experimental results on the nonlinearity ($\mathtt{nl}(F_n)$) and weightwise nonlinearity ($\mathtt{nl}_k(F_n)$) of F_n. We have chosen $a, b \in \mathcal{B}_{\frac{n}{2}}$, a highly nonlinear function

$$a(y) = b(y) = \begin{cases} y_1 y_2 + \cdots + y_{\frac{n}{2}-1} y_{\frac{n}{2}} & \text{if } \frac{n}{2} \text{ is even} \\ y_1 y_2 + \cdots + y_{\frac{n}{2}-2} y_{\frac{n}{2}-1} + y_{\frac{n}{2}} & \text{if } \frac{n}{2} \text{ is odd.} \end{cases}$$

This function is a bent function when n is even and concatenation of two bent functions when n is odd. Further, these two functions are easy to compute which is helpful for implementation in light weight

n	$\mathtt{nl}$	$\mathtt{nl}_2$	$\mathtt{nl}_3$	$\mathtt{nl}_4$	$\mathtt{nl}_5$	$\mathtt{nl}_6$	$\mathtt{nl}_7$	$\mathtt{nl}_8$	$\mathtt{nl}_9$	$\mathtt{nl}_{10}$	$\mathtt{nl}_{11}$	$\mathtt{nl}_{12}$	$\mathtt{nl}_{13}$	$\mathtt{nl}_{14}$	$\sum_{k=0}^{n} \mathtt{nl}_k$
8	96	4	16	20	16	4	0	0	-	-	-	-	-	-	60
9	192	6	22	45	45	22	6	0	0	-	-	-	-	-	146
10	416	9	36	69	94	73	12	9	0	0	-	-	-	-	302
11	832	11	50	113	163	173	117	34	11	0	0	-	-	-	672
12	1596	12	36	146	264	286	264	148	36	14	0	0	-	-	1206
13	3192	15	69	219	507	660	660	495	240	69	17	0	0	-	2951
14	6904	19	102	336	764	1083	1484	1079	654	299	30	18	0	0	5868
15	13808	22	147	474	1155	2013	2735	2670	1965	1154	465	75	22	0	12897
16	28152	24	64	564	1216	2547	5036	4610	5036	2919	1216	516	64	24	23836

Table 1: Listing of $\mathtt{nl}(F_n), \mathtt{nl}_k(F_n)$ and $\sum_{k=0}^{n} \mathtt{nl}_k(F_n)$ for $8 \leq n \leq 16$.

cryptography. Table 1 presents the nonlinearity and weightwise nonlinearity of the functions F_n for $n = 8, 9, \ldots, 16$, which are generated using Construction 3.5.

We have presented a comparison of weightwise nonlinearities of F_n with the upper bound presented in [CMR17] in Table 2. Further, no upper bound is available for the nonlinearity of WAPB Boolean functions. Therefore, we have presented a comparison of the nonlinearity of F_n with the upper bound of the nonlinearity of n variable Boolean functions [dH97].

n	*function*	$\mathtt{nl}$	$\mathtt{nl}_2$	$\mathtt{nl}_3$	$\mathtt{nl}_4$	$\mathtt{nl}_5$	$\mathtt{nl}_6$	$\mathtt{nl}_7$	$\mathtt{nl}_8$	$\mathtt{nl}_9$	$\mathtt{nl}_{10}$	$\mathtt{nl}_{11}$	$\sum_{k=0}^{n} \mathtt{nl}_k$
8	*UB*	120	11	24	30	24	11	-	-	-	-	-	100
	F_8	96	4	16	20	16	4	-	-	-	-	-	60
9	*UB*	244	15	37	57	57	37	15	-	-	-	-	218
	F_9	192	6	22	45	45	22	6	-	-	-	-	146
10	*UB*	496	19	54	97	118	97	54	19	-	-	-	498
	F_{10}	416	9	36	69	94	73	12	9	-	-	-	302
11	*UB*	1000	23	76	155	220	220	155	76	23	-	-	948
	F_{11}	832	11	50	113	163	173	117	34	11	-	-	672
12	*UB*	2016	28	102	236	381	446	381	236	102	28	-	1940
	F_{12}	1596	12	36	146	264	286	264	148	36	14	-	1206
13	*UB*	4050	34	134	344	625	837	837	625	344	134	34	3948
	F_{13}	3192	15	69	219	507	660	660	495	240	69	17	2951

Table 2: Comparison of $\mathtt{nl}_k(F_n)$ with the upper bound(UB) presented in [CMR17]

We compare the nonlinearities of our result with some recent constructions for $n = 8$ in Table 3. The sum of the weightwise nonlinearity of our construction is highest for $n = 8$ among the available constructions.

4 Conclusions and Future work

We have presented constructing a class of WAPB Boolean functions in n variables from the idea of constructions presented in [MS21, DM23]. The experimental results on nonlinearity and weightwise nonlinearities show a good improvement and are the highest among the available constructions. For future work, we are studying the cryptographic properties of this class of WAPB functions and attempting to further improve the nonlinearities and weightwise nonlinearities by modifying this class of functions.

WPB/ WAPB functions	$\mathtt{nl}_2$	$\mathtt{nl}_3$	$\mathtt{nl}_4$	$\mathtt{nl}_5$	$\mathtt{nl}_6$	$\sum_{k=0}^{8} \mathtt{nl}_k$
Upper Bound [CMR17]	11	24	30	24	11	100
Carlet, Méaux, Rotella [CMR17]	2	12	19	12	2	47
Li and Su [LS20, g_{2q+2} Equation(9)]	2	12	19	12	2	47
Mesnager and Su [MS21, f_m Equation(13)]	2	0	3	0	2	7
Mesnager and Su [MS21, g_m Equation(22)]	2	14	19	14	2	51
Mesnager, Su and Li [MSL21, f_m Equation(2)]	2	8	8	8	2	28
Mesnager, Su and Li [MSL21, f_m Equation(3)]	6	8	26	8	6	54
Zhang and Su [ZS23, g_m Equation(11)]	2	12	19	12	6	51
F_n[Construction 3.5]	4	16	20	16	4	60

Table 3: Comparison of $\mathtt{nl}_k$ of 8-variable WPB constructions.

References

[CMR17] Claude Carlet, Pierrick Méaux, and Yann Rotella. Boolean functions with restricted input and their robustness; application to the FLIP cipher. *IACR Trans. Symmetric Cryptol.*, 2017(3):192–227, 2017.

[dH97] Xiang dong Hou. On the norm and covering radius of the first-order Reed-Muller codes. *IEEE Transactions on Information Theory*, 43(3):1025–1027, 1997.

[DLR16] Sébastien Duval, Virginie Lallemand, and Yann Rotella. Cryptanalysis of the FLIP family of stream ciphers. In *Advances in Cryptology - CRYPTO 2016*, volume 9814 of *Lecture Notes in Computer Science*, pages 457–475. Springer, 2016.

[DM23] Deepak Kumar Dalai and Krishna Mallick. A class of weightwise almost perfectly balanced boolean functions. In *ALgebraic and combinatorial methods for COding and CRYPTography-ALCOCRYPT*, 2023.

[GM22] Agnese Gini and Pierrick Méaux. On the weightwise nonlinearity of weightwise perfectly balanced functions. *Discrete Applied Mathematics*, 322:320–341, 2022.

[GS22] Xiaoqi Guo and Sihong Su. Construction of weightwise almost perfectly balanced boolean functions on an arbitrary number of variables. *Discrete Applied Mathematics*, 307:102–114, 2022.

[LM19] Jian Liu and Sihem Mesnager. Weightwise perfectly balanced functions with high weightwise nonlinearity profile. *Designs, Codes and Cryptography*, 87(8):1797–1813, 2019.

[LS20] Jingjing Li and Sihong Su. Construction of weightwise perfectly balanced boolean functions with high weightwise nonlinearity. *Discrete Applied Mathematics*, 279:218–227, 2020.

[MJSC16] Pierrick Méaux, Anthony Journault, François-Xavier Standaert, and Claude Carlet. Towards stream ciphers for efficient FHE with low-noise ciphertexts. In *Advances in Cryptology - EUROCRYPT 2016*, volume 9665 of *Lecture Notes in Computer Science*, pages 311–343. Springer, 2016.

[MS21] Sihem Mesnager and Sihong Su. On constructions of weightwise perfectly balanced boolean functions. *Cryptography and Communications*, 13(6):951–979, 2021.

[MSL21] Sihem Mesnager, Sihong Su, and Jingjing Li. On concrete constructions of weightwise perfectly balanced functions with optimal algebraic immunity and high weightwise nonlinearity. In *The 6th International Workshop on Boolean Functions and Applications*, 2021.

[MZD19] Sihem Mesnager, Zhengchun Zhou, and Cunsheng Ding. On the nonlinearity of boolean functions with restricted input. *Cryptography and Communications*, 11(1):63–76, 2019.

[TL19] Deng Tang and Jian Liu. A family of weightwise (almost) perfectly balanced boolean functions with optimal algebraic immunity. *Cryptography and Communications*, 11(6):1185–1197, 2019.

[ZS22] Linya Zhu and Sihong Su. A systematic method of constructing weightwise almost perfectly balanced boolean functions on an arbitrary number of variables. *Discrete Applied Mathematics*, 314:181–190, 2022.

[ZS23] Rui Zhang and Sihong Su. A new construction of weightwise perfectly balanced boolean functions. *Advances in Mathematics of Communications*, 17(4):757–770, 2023.

The second-order zero differential spectra of some power maps

Kirpa Garg[*], Sartaj Ul Hasan[*], Constanza Riera[**], and Pantelimon Stănică[***]

[*]Department of Mathematics, Indian Institute of Technology Jammu, Jammu 181221, India
[**]Department of Computer Science, Electrical Engineering and Mathematical Sciences, Western Norway University of Applied Sciences, 5020 Bergen, Norway
[***]Applied Mathematics Department, Naval Postgraduate School, Monterey, CA 93943, USA

Abstract

It was shown by Boukerrou et al. [3] that the F-boomerang uniformity (which is the same as the second-order zero differential uniformity in even characteristic) of perfect nonlinear functions is 0 on $\mathbb{F}_{p^n}$ (p prime) and the one of almost perfect nonlinear functions on $\mathbb{F}_{2^n}$ is also 0. It is natural to inquire what happens with APN or other low differential uniform functions in odd characteristics. As a by-product, our work implies that APN functions in odd characteristic may not have zero second-order zero differential spectra, as one might venture to conjecture. Here, we explicitly determine the second-order zero differential spectra of several maps with low differential uniformity. In particular, we compute the second-order zero differential spectra for some almost perfect nonlinear (APN) functions, and it turns out that these functions also have low second-order zero differential uniformity.

1 Introduction

Let n be a positive integer and p be a prime number. We denote by $\mathbb{F}_q$ the finite field with $q = p^n$ elements, by $\mathbb{F}_{p^n}^*$ the multiplicative cyclic group of non-zero elements of $\mathbb{F}_q$ and by $\mathbb{F}_q[X]$ the ring of polynomials in one variable X with coefficients in $\mathbb{F}_q$. It may be noted that functions over finite fields are very important objects due to their wide range of applications in coding theory and cryptography. For example, in cryptography, these functions (mostly, for $p = 2$, though there are some proposals in odd characteristic) are often used in designing what are known as substitution boxes (S-boxes) in modern block ciphers. One of the most effective attacks on block ciphers is differential cryptanalysis, which was first introduced by Biham and Shamir [1]. The resistance of a function against differential attacks is measured in terms of its differential uniformity – a notion introduced by Nyberg [11]. For a function $F : \mathbb{F}_q \to \mathbb{F}_q$, and any $a \in \mathbb{F}_q$, the derivative of F in the direction a is defined as $D_F(X, a) := F(X + a) - F(X)$ for all $X \in \mathbb{F}_q$. For any $a, b \in \mathbb{F}_q$, the Difference Distribution Table (DDT) entry $\Delta_F(a, b)$ at point (a, b) is the number of solutions $X \in \mathbb{F}_q$ of the equation $D_F(X, a) = b$. Further, the differential uniformity of F, denoted by Δ_F, is given by $\Delta_F := \max\{\Delta_F(a, b) : a \in \mathbb{F}_q^*, b \in \mathbb{F}_q\}$. We call the function F a perfect nonlinear (PN) function, respectively, an almost perfect nonlinear (APN) function, if $\Delta_F = 1$, respectively, $\Delta_F = 2$. Blondeau, Canteaut, and Charpin [2] introduced the idea of locally APN power functions as a generalization of the APN-ness property. A power function $F(X)$ over $\mathbb{F}_{2^n}$ is said to be locally-APN if $\max\{\mathrm{DDT}_F(1, b) : b \in \mathbb{F}_{2^n} \setminus \mathbb{F}_2\} = 2$.

The boomerang attack on block ciphers is another important cryptanalysis technique proposed by Wagner [13]. It can be considered as an extension of the classical differential attack. At Eurocrypt 2018, Cid et al. [5] introduced a systematic approach known as the Boomerang Connectivity Table (BCT), to analyze the boomerang style attack. Boura and Canteaut [4] further studied BCT and coined the term "boomerang uniformity", which is essentially the maximum value of nontrivial entries of the BCT, to quantify the resistance of a function against the boomerang attack. Boukerrou et al. [3] pointed out the need for the counterpart of the BCT by extending

the idea to Feistel ciphers. They introduced the Feistel Boomerang Connectivity Table (FBCT) as an extension for Feistel ciphers, where the S-boxes may not be permutations.

The authors in [3] investigated the properties of the FBCT for two classes of vectorial functions, namely, APN functions and functions based on inverse mapping over $\mathbb{F}_{2^n}$. They showed that all the non-trivial coefficients at FBCT are 0 for APN functions over $\mathbb{F}_{2^n}$ and are 0 and 4 for the inverse function over $\mathbb{F}_{2^n}$, where n is even. In fact, the coefficients of FBCT are related to the second-order zero differential spectra of the functions. Another important property of the FBCT is that F is an APN function over $\mathbb{F}_{2^n}$ if and only if the FBCT of F is 0 for $a, b \in \mathbb{F}_{2^n}$ with $ab(a+b) \neq 0$. Li et al. [10] further studied the second-order zero differential spectra of the inverse function and some APN functions in odd characteristic. The authors of [10] also show that these function also have low second-order zero differential uniformity. Although most of the block ciphers operate in even characteristic, there are proposals, which work in non-binary environments, and we mention here Schroeppel's Hasty Pudding cipher (a candidate for the AES competition) [12], defined on a set of arbitrary size.

We further extend their work by investigating the second-order zero differential spectra of some more classes of functions with low differential uniformity. In addition, these functions have low second-order zero differential uniformity. The paper is organized as follows. In Section 2, we recall some definitions. The second-order zero differential spectra of four power functions over finite fields of odd characteristic have been considered in Section 3. Further, in Section 4, second-order zero differential spectrum of a locally APN function has been studied. Finally, we conclude the paper in Section 5.

2 Preliminaries

In this section, we recall some definitions.

Definition 2.1 *For p an odd prime, n a positive integer, and $q = p^n$, we let η be the quadratic character of $\mathbb{F}_q$ defined by*

$$\eta(X) := \begin{cases} 1 & \text{if } X \text{ is square of an element of } \mathbb{F}_q^*, \\ -1 & \text{otherwise.} \end{cases}$$

Definition 2.2 [3, 10] *For $F : \mathbb{F}_{p^n} \to \mathbb{F}_{p^n}$ a function and $a, b \in \mathbb{F}_{p^n}$, the* second-order zero differential spectra *of F with respect to a, b is defined as*

$$\nabla_F(a,b) := \#\{X \in \mathbb{F}_{p^n} : F(X+a+b) - F(X+b) - F(X+a) + F(X) = 0\}. \tag{1}$$

If $p = 2$, we call $\nabla_F = \max\{\nabla_F(a,b) : a \neq b, a, b \in \mathbb{F}_{2^n} \setminus \{0\}\}$ the second-order zero differential uniformity of F. If $p > 2$, we call $\nabla_F = \max\{\nabla_F(a,b) : a, b \in \mathbb{F}_{p^n} \setminus \{0\}\}$ the second-order zero differential uniformity of F.

Definition 2.3 *(Feistel Boomerang Connectivity Table) [3] Let F be a function from $\mathbb{F}_{2^n}$ to $\mathbb{F}_{2^n}$ and $a, b \in \mathbb{F}_{2^n}$. The* Feistel Boomerang Connectivity Table (FBCT) *of F is given by a $2^n \times 2^n$ table T, in which the entry for the (a, b) position is given by:*

$$FBCT_F(a,b) = \#\{X \in \mathbb{F}_{2^n} : F(X+a+b) + F(X+b) + F(X+a) + F(X) = 0\}.$$

Definition 2.4 *(F-Boomerang Uniformity) [3, 10] The F-Boomerang uniformity corresponds to the highest value in the FBCT without considering the first row, the first column and the diagonal:*

$$\beta_F = \max_{a \neq 0, b \neq 0, a \neq b} FBCT_F(a,b).$$

Notice that the coefficients of FBCT are related to the second-order zero differential spectra of functions over $\mathbb{F}_{2^n}$. Note that the F-Boomerang uniformity is in fact the second-order zero differential uniformity of F in even characteristic.

3 The second-order zero differential spectrum for functions over finite fields of odd characteristic

Table 1 gives some of the known power functions with low second-order zero differential uniformity over finite fields of odd characteristic.

Table 1: Second-order differential uniformity for functions over finite fields of odd characteristic

p	d	condition	Δ_F	∇_F	Ref
any odd p	any d	any n	1	0	[10, Lemma 2.5]
$p>3$	3	any	2	1	[10, Theorem 3.1]
$p=3$	3^n-3	$n>1$ is odd	2	2	[10, Theorem 3.2]
$p>2$	p^n-2	$p^n\equiv 2 \pmod 3$	2	1	[10, Theorem 3.3]
$p>3$	p^m+2	$n=2m$, $p^m\equiv 1 \pmod 3$	2	1	[10, Theorem 3.4]
$p=3$	3^n-2	any	3	3	[10, Theorem 3.1]
p	p^n-2	$p^n\equiv 1 \pmod 3$	3	3	[10, Theorem 3.1]
$p>3$	4	$n>1$	3	2	This paper
p	$\frac{2p^n-1}{3}$	$p^n\equiv 2 \pmod 3$	2	1	This paper
$p>3$	$\frac{p^k+1}{2}$	$\gcd(2n,k)=1$	$\leq \gcd(\frac{p^k-1}{2}, p^{2n}-1)$	$\frac{p-3}{2}$	This paper
$p=3$	$\frac{3^n-1}{2}+2$	n is odd	4	3	This paper

In this section, we first deal with the computation of second-order zero differential spectrum of the function $F(X)=X^d$, where $d=\frac{2p^n-1}{3}$ over $\mathbb{F}_{p^n}$, for $p^n\equiv 2 \pmod 3$. Helleseth et al. [8] showed that F is an APN function over $\mathbb{F}_{p^n}$, for $p^n\equiv 2 \pmod 3$.

Theorem 3.1 *Let $F(X)=X^d$ be a function of $\mathbb{F}_{p^n}$, where $d=\frac{2p^n-1}{3}$, $p^n\equiv 2 \pmod 3$. Then for $a,b\in\mathbb{F}_{p^n}$,*

$$\nabla_F(a,b)=\begin{cases}1 & \text{if } ab\neq 0\\ p^n & \text{if } ab=0.\end{cases} \tag{2}$$

Moreover, F is second-order zero differential 1-uniform.

Next, we considered the power function $F(X)=X^{\frac{p^k+1}{2}}$, which was shown to be an APN power function by Helleseth et al. in [8]. We further compute its second-order zero differential spectrum over $\mathbb{F}_{p^n}$.

Theorem 3.2 *Let $F(X)=X^d$ be a power function of $\mathbb{F}_{p^n}$, where $d=\frac{p^k+1}{2}$, and $\gcd(k,2n)=1$. Let $p>3$. Then for $a,b\in\mathbb{F}_{p^n}$,*

$$\nabla_F(a,b)=\begin{cases}0 & \text{if } ab\neq 0, \text{and } \eta(D)=-1\\ 1 & \text{if } ab\neq 0, \text{and } \eta(D)=0\\ \frac{p-3}{2} & \text{if } ab\neq 0, \text{and } \eta(D)=1\\ p^n & \text{if } ab=0\end{cases} \tag{3}$$

where $D=\frac{4a^2}{(1-u^{2i})^2}+\frac{b^2}{u^{2i}}$, u is a primitive $(p-1)$-th root of unity in $\mathbb{F}^*_{p^{2n}}$. Moreover, F is second-order zero differential $\frac{p-3}{2}$-uniform.

Remark 3.3 *Helleseth et al. in [8] showed that $F(X)=X^d$ over $\mathbb{F}_{5^n}$, where $d=\frac{5^k+1}{2}$, and $\gcd(k,2n)=1$ is an APN power function. Hence, from the above Theorem 3.2, we get that F is second-order zero differential 1-uniform over $\mathbb{F}_{5^n}$.*

Remark 3.4 *Note that, if $p = 3$, then F is PN function [6]. Therefore, by [10], it is second-order zero differential 0-uniform over $\mathbb{F}_{3^n}$.*

Now, we considered some more functions with low differential uniformity, more precisely of differential uniformity 3 and 4. Dobbertin et al. in [7] show that $F(X) = X^4$ is differentially 3 uniform for all $p > 3$ and $n > 1$. In the following theorem, we show that $F(X) = X^4$ is second-order zero differential 1-uniform for all $p > 3$ and $n > 1$.

Theorem 3.5 *Let $F(X) = X^4$ be a power function of $\mathbb{F}_{p^n}$, where $p > 3, n > 1$. Then for $a, b \in \mathbb{F}_{p^n}$,*

$$\nabla_F(a,b) = \begin{cases} 0 & \text{if } \eta\left(\frac{-a^2-b^2}{3}\right) = -1 \\ 1 & \text{if } a^2+b^2 = 0 \\ 2 & \text{if } \eta\left(\frac{-a^2-b^2}{3}\right) = 1 \\ p^n & \text{if } ab = 0. \end{cases} \tag{4}$$

Moreover, F is second-order zero differential 2-uniform.

Helleseth et al. in [9] showed that $F(X) = X^d$, where $d = \frac{3^n-1}{2}+2$ is a differentially 4-uniform function over $\mathbb{F}_{p^n}$, for odd n. We compute its second-order zero differential spectrum and show that it is is second-order zero differential 3-uniform.

Theorem 3.6 *Let $F(X) = X^d$ be a function of $\mathbb{F}_{3^n}$, where $d = \frac{3^n-1}{2} + 2$ and n is odd. Then for $a, b \in \mathbb{F}_{3^n}$,*

$$\nabla_F(a,b) = \begin{cases} 1 & \text{if } \eta(ab) = 1 = \eta(a^2+b^2) \text{ or } \eta(ab) = -1 \text{ and } \eta(a^2+b^2) = 1 \\ 3 & \text{if } \eta(ab) = -1 = \eta(a^2+b^2) \text{ or } \eta(ab) = 1 \text{ and } \eta(a^2+b^2) = -1 \\ 3^n & \text{if } ab = 0. \end{cases} \tag{5}$$

Moreover, F is second-order zero differential 3-uniform.

4 The second-order zero differential spectrum for functions over finite fields of even characteristic

In this section, we compute the second-order zero differential spectrum of the locally APN function $F(X) = X^{2^m-1}$ over $\mathbb{F}_{2^{2m}}$, the DDT entries for which have already been computed by Blondeau et al. in [2].

Theorem 4.1 *Let $F(X) = X^{2^m-1} \in \mathbb{F}_{2^n}[X]$, where $n = 2m$. Then for any $a, b \in \mathbb{F}_{2^n}$,*

(1) *When m is odd,*

$$\nabla_F(a,b) = \begin{cases} 2^n & \text{if } a = 0, \text{ or } b = 0, \text{ or } a = b \\ 4 & \text{if } a \neq 0, b \neq 0, a \neq b, A \neq 0 \text{ and } B = 0 \\ 2^m - 4 & \text{if } a \neq 0, b \neq 0, a \neq b, A = 0 \text{ and } B \neq 0 \\ 0 & \text{if } a \neq 0, b \neq 0, a \neq b, A = 0 \text{ and } B = 0 \\ 0 & \text{if } a \neq 0, b \neq 0, a \neq b, A \neq 0 \text{ and } B \neq 0. \end{cases}$$

(2) *When m is even,*

$$\nabla_F(a,b) = \begin{cases} 2^n & \text{if } a = 0, \text{ or } b = 0, \text{ or } a = b \\ 0 & \text{if } a \neq 0, b \neq 0, a \neq b, A \neq 0 \text{ and } B = 0 \\ 2^m - 4 & \text{if } a \neq 0, b \neq 0, a \neq b, A = 0 \text{ and } B \neq 0 \\ 0 & \text{if } a \neq 0, b \neq 0, a \neq b, A = 0 \text{ and } B = 0 \\ 0 & \text{if } a \neq 0, b \neq 0, a \neq b, A \neq 0 \text{ and } B \neq 0, \end{cases}$$

where $A = \dfrac{ab^{2^m} + ba^{2^m}}{ab(a+b)}$ *and* $B = \dfrac{a^2b^{2^m} + b^2a^{2^m}}{ab(a+b)}$. *Moreover, the Feistel boomerang uniformity of* F *is* $\beta^F(F) = 2^m - 4$.

5 Conclusion

In this paper, we extended the work of Li et al. [10] by computing the second-order zero differential spectra of some APN power functions over finite fields of odd characteristic in order to derive additional cryptographic properties of APN functions. We also determined the second-order zero differential spectrum of some functions with low differential uniformity. Additionally, all of these functions exhibit a low second-order zero differential uniformity. In our future work, we will look into more functions with low differential uniformity and investigate their second-order zero differential spectrum.

References

[1] E. Biham, A. Shamir, *Differential cryptanalysis of DES-like cryptosystems,* J. Cryptol. 4:1 (1991), 3–72.

[2] C. Blondeau, A. Canteaut, P. Charpin,*Differential properties of* $X \to X^{2^t-1}$, IEEE Trans. Inf. Theory 57(12), (2011) 8127–8137.

[3] H. Boukerrou, P. Huynh, V. Lallemand, B. Mandal, M. Minier, *On the Feistel counterpart of the boomerang connectivity table,* IACR Trans. Symmetric Cryptol. 1 (2020), 331–362.

[4] C. Boura, A. Canteaut, *On the boomerang uniformity of cryptographic S-boxes,* IACR Trans. Symmetric Cryptol. 3 (2018) 290–310.

[5] C. Cid, T. Huang, T. Peyrin, Y. Sasaki, L. Song, *Boomerang connectivity table: a new cryptanalysis tool.* In: J. Nielsen, V. Rijmen (ed) Advances in Cryptology-EUROCRYPT'18, LNCS 10821 683-714, Springer, Cham (2018).

[6] R.S. Coulter, R.W. Matthews, *Planar functions and planes of Lenz-Barlotti class II,* Des. Codes Cryptogr. 10(2) (1997) 167–184.

[7] H. Dobbertin, D. Mills, E. N. Müller, A. Pott, W. Willems, *APN functions in odd characteristic*, Discrete Math. 267(1-3) (2003) 95–112.

[8] T. Helleseth, R. Chunming, S. Daniel, *New families of almost perfect nonlinear power mappings,* IEEE Trans. Inf. Theory 45.2 (1999) 475–485.

[9] T. Helleseth, D. Sandberg, *Some power mappings with low differential uniformity*, Appl. Algebra Eng. Commun. Comput. 8 (1997), 363–370.

[10] X. Li, Q. Yue, D. Tang, *The second-order zero differential spectra of almost perfect nonlinear functions and the inverse function in odd characteristic*, Cryptogr. Commun. 14(3) (2022), 653–662.

[11] K. Nyberg, *Differentially uniform mappings for cryptography,* In T. Helleseth (ed), Advances in Cryptology-EUROCRYPT'93, LNCS 765, pp. 55–64, Springer, Heidelberg (1994).

[12] R. Schroeppel, *Hasty Pudding Cipher Specifications*, `http://richard.schroeppel.name:8015/hpc/hpc-spec`; see also, https://en.wikipedia.org/wiki/Hasty_Pudding_cipher.

[13] D. Wagner, *The boomerang attack,* In: L. R. Knudsen (ed.) Fast Software Encryption-FSE 1999. LNCS 1636, Springer, Berlin, Heidelberg, (1999), pp. 156–170.

Optimizing Implementations of Boolean Functions

Meltem Sönmez Turan

National Institute of Standards and Technology

Abstract

Symmetric cryptography primitives are constructed by iterative applications of linear and nonlinear layers. Constructing efficient circuits for these layers, even for the linear one, is challenging. In 1997, Paar proposed a heuristic to minimize the number of XORs (modulo 2 addition) necessary to implement linear layers. In this study, we slightly modify Paar's heuristics to find implementations for nonlinear Boolean functions, in particular to homogeneous Boolean functions. Additionally, we show how this heuristic can be used to construct circuits for generic Boolean functions with small number of `AND` gates, by exploiting affine equivalence relations.

1 Introduction

Symmetric cryptography primitives are constructed by iterative applications of linear and nonlinear layers. Linear layers are typically composed of binary matrices, and are used for *diffusion*, whereas the nonlinear layers are composed of nonlinear substitution boxes (s-box), and are used for *confusion*. Constructing efficient circuits for these layers, even for the linear ones, is challenging. There are various metrics to measure the efficiency of the circuits such as number of specific gates (e.g., `AND`, `XOR`), or the depth of the circuits.

Multiplicative Complexity. The metric *Multiplicative Complexity* (MC) is defined as the minimum number of `AND` gates required to implement a function with a circuit over the basis {`AND, XOR, NOT`}. This complexity measure is relevant for many advanced cryptographic protocols (e.g., [1]), fully homomorphic encryption (e.g., [2]), and zero-knowledge proofs (e.g., [3]), where processing nonlinear gates such as `AND`, `NAND`, is more expensive than processing linear gates such as `XOR`. These protocols benefit from new symmetric-key primitives that can be implemented with small number of `AND` gates (e.g., Rasta [4], LowMC [5]).

There is no known asymptotically efficient technique to compute the MC of a random Boolean function. In 2000, Boyar et al. [6] showed that the MC of an n-variable random Boolean function is at least $2^{n/2} - \mathcal{O}(n)$ with high probability. For arbitrary n, it is known that under standard cryptographic assumptions, it is not possible to compute the MC in polynomial time in the length of the truth table [7]. The *degree bound* states that the MC of a Boolean function having degree d is at least $d - 1$ [8].

Although there are no efficient techniques to find MC of for random Boolean functions, the MC distribution has been established for Boolean functions having up to 6 variables [9, 10]. There are also known techniques specific for Boolean functions with low degree (e.g., less then or equal to three) or structure (e.g., symmetric). The MC of affine Boolean functions is zero. Mirwald and Schnorr [11] showed that the MC of a quadratic function f is k, iff f is affine equivalent to the canonical form $\bigoplus_{i=1}^{k} x_{2i-1}x_{2i}$. This implies the MC of quadratic functions is at most $\lfloor \frac{n}{2} \rfloor$. Turan and Peralta [12] improved the bounds on

MC of cubic Boolean functions. Brandão et al. [13] studied the MC of symmetric Boolean functions and constructed circuits for all such functions with up to 25 variables. In 2017, Find et al. [14] characterized the Boolean functions with MC 2 by using the fact that MC is invariant with respect to affine transformations. In 2020, Çalık et al. extended the result to Boolean functions with MC up to 4 [15]. In 2022, Häner and Soeken [16] showed the MC of interval checking.

XOR complexity. In addition to the optimization of `AND` gates for Boolean function, another line of research focuses on optimizing the implementations of linear matrices over $\mathbb{F}_2$, where the goal is to minimize the number of `XOR` gates necessary to implement the matrices. There are three metrics used while optimizing the number of `XOR` gates: direct `XOR` (`d-XOR`), sequential `XOR` (`s-XOR`) and general `XOR` (`g-XOR`). `d-XOR` is the direct `XOR` count and corresponds to the the number of 1's in the binary matrix representation of the linear layer. The `s-XOR` metric counts the number of `XOR` operations of the form $x_i = x_i \oplus x_j$, that updates the value of input x_i, whereas, `g-XOR` metric corresponds to the number of operations of the form $x_i = x_j \oplus x_k$. Determining optimal implementations for `s-XOR` and `g-XOR` is a hard problem. Boyar et al. [17] argue that minimizing the number of XORs to compute a binary matrix is equivalent to solving the Shortest Linear Program problem over GF(2), which is known to be NP-hard. One of the early heuristics for XOR optimization is by Paar[18] in 1997, which is cancellation-free, i.e., the circuits generated by Paar's heuristic does not include cancellation of identical input bits. Since ability to cancellation leads to better circuit, many new heuristics were suggested (e.g., [19, 20, 21]).

Contributions. In this study, we propose a modification to Paar's heuristics so that it can also be applied to nonlinear functions, in particular to homogeneous Boolean functions. Additionally, we show how this heuristic can be used to construct circuits for generic Boolean functions with small number of `AND` gates, by exploiting affine equivalance relations.

2 Preliminaries

2.1 Boolean functions

Let $\mathbb{F}_2$ be the finite field with two elements. An n-variable Boolean function f is a mapping from $\mathbb{F}_2^n$ to $\mathbb{F}_2$. Let B_n be the set of n-variable Boolean functions.The *algebraic normal form* (ANF) of f is the multivariate polynomial

$$f(x_1, \ldots, x_n) = \sum_{u \in \mathbb{F}_2^n} a_u x^u, \tag{1}$$

where $a_u \in \mathbb{F}_2$ and $x^u = x_1^{u_1} x_2^{u_2} \cdots x_n^{u_n}$ is a *monomial* containing the variables x_i where $u_i = 1$. The degree of the monomial x^u is the number of variables appearing in x^u. The *algebraic degree* of a Boolean function, denoted $deg(f)$, is the highest degree among the monomials appearing in its ANF. A Boolean function is called *homogeneous*, if all the monomials in its algebraic normal form have the same algebraic degree.

Two functions $f, g \in \mathcal{B}_n$ are *affine equivalent* if f can be written as

$$f(\mathbf{x}) = g(A\mathbf{x} + \mathbf{a}) + \mathbf{b}^\top \mathbf{x} + c, \tag{2}$$

where A is a non-singular $n \times n$ matrix over $\mathbb{F}_2$, $\mathbf{a}, \mathbf{b}$ are column vectors in $\mathbb{F}_2^n$, and $c \in \mathbb{F}_2$. We use $[f]$ to denote the affine equivalence class of the function f. Degree and MC are invariant under affine transformations.

2.2 Boolean Circuits

A *Boolean circuit* C with n inputs and m outputs is a directed acyclic graph, where the inputs and the gates are the nodes, and the edges correspond to the Boolean-valued *wires*. The *fanin* and *fanout* of a node is the number of wires going in and out of the node, respectively. The nodes with fanin zero are called the *input nodes* and are labeled with an input variable from $\{x_0, \ldots, x_{n-1}\}$. The circuits considered in this study only contain gates from the complete basis {`AND, XOR, NOT`} and have exactly one node with fanout zero (i.e., $m = 1$), which is called the *output node*. For our purposes, we assume `AND` gates have fan-in two, but `XOR` gates have arbitrary fan-in (i.e., > 0).

2.3 Paar's Heuristics

The linear layers of symmetric key primitives can be represented by a $m \times n$ binary matrix M, where there are n input variables $(x_0, \ldots, x_{n-1})$ and m output variables $(y_0, \ldots, y_{m-1})$. An upper bound for the number of `XOR` operations is $w - m$, where w is the *weight* of M (i.e., the number of ones).

Paar [18] proposed two heuristics to implement linear layers with small number of `XOR` operations. Both heuristics operate on the matrix representation of the linear layer. The heuristic determines the frequency for each possible pairs of input variable x_i, x_j $(i \neq j)$ that are XORed together in m linear functions. The pair with highest frequency is computed and placed to the matrix as a new variable. In the next iteration, the operation is repeated on the matrix of size $m \times (n+1)$. This procedure is repeated until all outputs have been computed (i.e., the weight of the resulting matrix is m).

Example. Let the linear layer to implement be given as follows:

$$\begin{aligned} x_0 + x_1 + x_2 &= y_0 \\ x_1 + x_3 + x_4 &= y_1 \\ x_0 + x_2 + x_3 + x_4 &= y_2 \\ x_1 + x_2 + x_3 &= y_3 \\ x_0 + x_1 + x_3 &= y_4 \\ x_1 + x_2 + x_3 + x_4 &= y_5 \end{aligned}$$

The matrix representation of the linear layer is

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{pmatrix}$$

Frequency of each pair of inputs appearing in the linear layer is

Pair	Frequency	Pair	Frequency
(x_0, x_1)	2	(x_1, x_3)	4
(x_0, x_2)	2	(x_1, x_4)	2
(x_0, x_3)	2	(x_2, x_3)	3
(x_0, x_4)	1	(x_2, x_4)	2
(x_1, x_2)	3	(x_3, x_4)	3

The first selected pair is (x_1, x_3) with frequency 4. So, the first step of the implementation is $t_0 = x_1 \oplus x_3$. Then the matrix is updated as follows.

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

and the updated frequency table is

Pair	Frequency	Pair	Frequency
(x_0, x_1)	1	(x_1, t_0)	0
(x_0, x_2)	2	(x_2, x_3)	1
(x_0, x_3)	1	(x_2, x_4)	2
(x_0, x_4)	1	(x_2, t_0)	2
(x_0, t_0)	1	(x_3, x_4)	1
(x_1, x_2)	1	(x_3, t_0)	0
(x_1, x_3)	0	(x_4, t_0)	2
(x_1, x_4)	0	-	-

There is a tie for the the pairs (x_0, x_2), (x_2, x_4), (x_2, t_0), and (x_4, t_0). For this example the next pair is selected randomly among these pairs as (x_0, x_2), and the next step of the implementation becomes $t_1 = x_0 \oplus x_2$. Continuing this way, the implementation of the layer is found as:

$$\begin{aligned} t_0 &= x_1 \oplus x_3 \\ t_1 &= x_0 \oplus x_2 \\ t_2 &= x_4 \oplus t_0 \\ t_3 &= x_1 \oplus t_1 \\ t_4 &= x_3 \oplus x_4 \\ t_5 &= t_1 \oplus t_4 \\ t_6 &= x_2 \oplus t_0 \\ t_7 &= x_0 \oplus t_0 \\ t_8 &= x_2 \oplus t_2 \end{aligned}$$

The output $(y_0, y_1, y_2, y_3, y_4, y_5)$ is obtained as $(t_3, t_2, t_5, t_6, t_7, t_8)$.

3 Application of Paar's Heuristic to Nonlinear Boolean Functions

Although Paar's heuristic is proposed to find implementations for linear layers, it can also be applied to nonlinear Boolean functions, with a slight modification. An n-variable Boolean function with m monomials can be represented by a $m \times n$ binary matrix, where each row corresponds to a monomial in the ANF of the function. For example, the following row (1 1 0 1 0 1) represents the monomial $x_0x_1x_3x_5$ for a 6-variable Boolean function. Instead of modulo 2 addition of each terms in the row, we are now interested in modulo 2 multiplication of each term. This method, in general, would not be efficient (in

terms of number of multiplications), especially for Boolean functions with large number of monomials, as the heuristic computes each monomials independently.

Next we propose a variation of the heuristic that decomposes Boolean functions into homogeneous Boolean functions and exploit affine equivalence relations to find efficient circuits.

Let $f \in B_n$, with degree d. The proposed heuristic to find efficient circuit for f is as follows:

1. Decompose f into d homogeneous Boolean functions,

$$f = a + f_1 \oplus f_2 \oplus \ldots \oplus f_d,$$

   where f_i is the sum of monomials of f with degree i, and a corresponds to the constant term.

2. Apply a number of affine equivalence transformations to the highest-degree homogeneous function, (i.e., f_d) to construct f'_d with smaller number of monomials with degree d. Note that if $d = n$, no affine transformation would decrease the number of monomials, as there is only one monomial with degree n. If f'_d includes monomials with degree smaller than d, those monomials are added to the corresponding f_i depending on their degree.

3. Apply modified Paar's heuristic to find an implementation for the degree d terms of f'_d. (Note that in modified Paar's heuristic each iteration corresponds to modulo 2 multiplication, instead of modulo 2 addition.) Apply the inverse affine transformation to the circuit to construct an implementation for the degree d monomials of f.

4. Repeat the procedure to find an implementation for f'_{d-1} where f'_{d-1} is the XOR of f_d and the new degree $d-1$ monomials generated during Step 2.

5. The procedure is repeated until implementations for each homogoeneous function is obtained and these sub-circuits are compined to find an implementation for f.

The combined implementations can further be improved by eliminating the common operations done in each independent implementations of the homogeneous functions.

4 Discussion

In this study, we proposed a modification of the Paar's heuristic to find efficient implementations for Boolean functions (in particular to reduce the number of nonlinear gates). In general, Paar's heuristic provides better solutions when the representation matrix has low weight, which may not be true for nonlinear Boolean functions. Decomposing the Boolean function into homogeneous Boolean functions, and applying affine transformations to the specific degree terms makes it easier to reduce the number of target monomials, since smaller degree terms are handled in the next iterations of the algorithm.

References

[1] Vladimir Kolesnikov and Thomas Schneider. Improved Garbled Circuit: Free XOR Gates and Applications. In Luca Aceto, Ivan Damgård, Leslie Ann Goldberg,

Magnús M. Halldórsson, Anna Ingólfsdóttir, and Igor Walukiewicz, editors, *Automata, Languages and Programming, 35th International Colloquium, ICALP*, volume 5126 of *Lecture Notes in Computer Science*, pages 486–498. Springer, 2008.

[2] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. In Shafi Goldwasser, editor, *Innovations in Theoretical Computer Science 2012, Cambridge, MA, USA, January 8-10, 2012*, pages 309–325. ACM, 2012.

[3] Joan Boyar, Ivan Damgård, and René Peralta. Short Non-Interactive Cryptographic Proofs. *J. Cryptology*, 13(4):449–472, 2000.

[4] Christoph Dobraunig, Maria Eichlseder, Lorenzo Grassi, Virginie Lallemand, Gregor Leander, Eik List, Florian Mendel, and Christian Rechberger. Rasta: A Cipher with Low ANDdepth and Few ANDs per Bit. In *CRYPTO (1)*, volume 10991 of *Lecture Notes in Computer Science*, pages 662–692. Springer, 2018.

[5] Martin R. Albrecht, Christian Rechberger, Thomas Schneider, Tyge Tiessen, and Michael Zohner. Ciphers for MPC and FHE. In Elisabeth Oswald and Marc Fischlin, editors, *Advances in Cryptology - EUROCRYPT 2015 - 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I*, volume 9056 of *Lecture Notes in Computer Science*, pages 430–454. Springer, 2015.

[6] Joan Boyar, René Peralta, and Denis Pochuev. On the Multiplicative Complexity of Boolean Functions over the Basis ($\wedge$, $\oplus$, 1). *Theor. Comput. Sci.*, 235(1):43–57, 2000.

[7] Magnus Gausdal Find. On the Complexity of Computing Two Nonlinearity Measures. In *Computer Science - Theory and Applications - 9th International Computer Science Symposium in Russia, CSR 2014, Moscow, Russia, June 7-11, 2014. Proceedings*, pages 167–175, 2014.

[8] C. P. Schnorr. The Multiplicative Complexity of Boolean Functions. In Teo Mora, editor, *Applied Algebra, Algebraic Algorithms and Error-Correcting Codes (AAECC 1988)*, volume 357 of *LNCS*, pages 45–58, Berlin, Heidelberg, 1989. Springer Berlin Heidelberg.

[9] Meltem Turan Sönmez and René Peralta. *The Multiplicative Complexity of Boolean Functions on Four and Five Variables*, pages 21–33. Springer International Publishing, Cham, 2015.

[10] Çağdaş Çalık, Meltem Sönmez Turan, and René Peralta. The Multiplicative Complexity of 6-variable Boolean Functions. *Cryptogr. Commun.*, 11(1):93–107, 2019.

[11] Roland Mirwald and Claus-Peter Schnorr. The Multiplicative Complexity of Quadratic Boolean Forms. *Theor. Comput. Sci.*, 102(2):307–328, 1992.

[12] Meltem Sonmez Turan and Rene Peralta. On the Multiplicative Complexity of Cubic Boolean Functions. The 6th International Workshop on Boolean Functions and their Applications (BFA), 2021.

[13] Luís T. A. N. Brandão, Çağdaş Çalık, Meltem Sönmez Turan, and René Peralta. Upper Bounds on the Multiplicative Complexity of Symmetric Boolean Functions. *Cryptogr. Commun.*, 11(6):1339–1362, 2019.

[14] Magnus Gausdal Find, Daniel Smith-Tone, and Meltem Sönmez Turan. The Number of Boolean Functions with Multiplicative Complexity 2. *IJICoT*, 4(4):222–236, 2017.

[15] Çağdaş Çalık, Meltem Sönmez Turan, and René Peralta. Boolean Functions with Multiplicative Complexity 3 and 4. *Cryptogr. Commun.*, 12(5):935–946, 2020.

[16] Thomas Häner and Mathias Soeken. The multiplicative complexity of interval checking, 2022.

[17] Joan Boyar, Philip Matthews, and René Peralta. Logic minimization techniques with applications to cryptology. *J. Cryptol.*, 26(2):280–312, 2013.

[18] C. Paar. Optimized arithmetic for Reed-Solomon encoders. *Proceedings of IEEE International Symposium on Information Theory*, pages 250–, 1997.

[19] Alexander Maximov and Patrik Ekdahl. New circuit minimization techniques for smaller and faster AES sboxes. *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, 2019(4):91–125, 2019.

[20] Subhadeep Banik, Yuki Funabiki, and Takanori Isobe. More results on shortest linear programs. In Nuttapong Attrapadung and Takeshi Yagi, editors, *Advances in Information and Computer Security - 14th International Workshop on Security, IWSEC 2019, Tokyo, Japan, August 28-30, 2019, Proceedings*, volume 11689 of *Lecture Notes in Computer Science*, pages 109–128. Springer, 2019.

[21] Zejun Xiang, Xiangyoung Zeng, Da Lin, Zhenzhen Bao, and Shasha Zhang. Optimizing implementations of linear layers. *IACR Transactions on Symmetric Cryptology*, 2020(2):120–145, Jul. 2020.

On the matrix equation $MX = \overline{X}$ and self-dual Butson bent sequences

J. A. Armario[1], R. Egan[2], and P. Ó Catháin[3]

[1]Departamento de Matemática Aplicada I, Universidad de Sevilla, Avda. Reina Mercedes s/n, 41012 Sevilla, Spain

[2]School of Mathematical Sciences, Dublin City University, Ireland

[3]Fiontar & Scoil na Gaeilge, Dublin City University, Ireland

Abstract

Let M be a square matrix of order n and X a vector of n components, each with complex entries. We are interested in studying $MX = \overline{X}$ for some particular M where $\overline{X}$ denotes the image of X under complex conjugation. If $X \in \mathbb{R}^n$, X is an eigenvector for M associated to the eigenvalue 1. Here we reduce our study to $M = \frac{1}{\sqrt{n}}H$ where $HH^* = nI$ and the entries of H and X are in set of the complex k^{th} roots of unity (i.e., H is a Butson Hadamard matrix). Connections to generalized bent functions are studied.

1 Introduction

A new notion of bent sequences was introduced in [3] as a solution in X, Y to the system

$$\frac{1}{\sqrt{n}}HX = Y,$$

where H is a real Hadamard matrix of order n and $X, Y \in \{\pm 1\}^n$. X is called *a bent sequence for* H. If H is the Sylvester Hadamard matrix of order $n = 2^m$ then any bent Boolean function $f\colon \mathbb{Z}_2^m \to \mathbb{Z}_2$ determines a bent sequence for H by the rule $X = (-1)^f$ (and vice versa).

Clearly, the vector Y can also be shown to be a bent sequence attached to H^T, called the dual of X. When $X = Y$ the sequence X is said to be *self-dual*. In [4] this notion of self-dual bent sequence for a real Hadamard matrix was further generalized to a $n \times n$ Butson-Hadamard matrix with entries in the set of complex 4-*th* roots of unity as a solution in X to the system

$$HX = \lambda X \tag{1}$$

where λ is an eigenvalue of H and $X \in \{\pm 1, \pm\sqrt{-1}\}^n$.

Bent functions are equivalent to certain Hadamard matrices and difference sets. The concept has been generalized, yielding equivalences between various associated objects. In Schmidt's survey [1] equivalences between generalized bent functions $f\colon \mathbb{Z}_k^m \to \mathbb{Z}_h$, group invariant Butson Hadamard matrices, and splitting relative difference sets are described.

In this paper, we extend the definition of self-dual bent sequence X for H to any Butson Hadamard matrix (not only for the 4-*th* roots of unity) which is "complementary" to the

definition given in [4]. That consists of considering, instead of (1), the system

$$\frac{1}{\sqrt{n}}HX = \overline{X} \quad \text{(or more generally, } HX = \lambda\overline{X}\text{)} \tag{2}$$

where the overline denotes complex conjugation, the entries of H and X belong to the set of complex k^{th} roots of unity. A solution X of the system (2) is what we understand in this paper for a self-dual bent sequence for H. We believe that it is a more natural extension from the real to the complex case. Furthermore, when H and X take values in the set $\{\pm 1\}$, we recover the definition of [3]. Some motivation for the study of this self-duality concept can also be found in this reference. Finally, it is easy to realize that if H is the complex conjugation of the m^{th} Kronecker power of the $q \times q$ Fourier matrix then any self-dual bent sequence for H determines a self-dual generalized bent function $f\colon \mathbb{Z}_q^m \to \mathbb{Z}_q$ by the rule $X = [\zeta_q^{f(\mathbf{a})}]_{\mathbf{a}\in\mathbb{Z}_q^m}^{\top}$ which we denote by $X = \zeta_q^f$ for convenience.

2 Preliminaries

Let m and k be positive integers, and $\zeta_k = \exp(2\pi\sqrt{-1}/k)$ be a complex k^{th} root of unity. We write $\langle\zeta_k\rangle = \{\zeta_k^j\}_{0\le j\le k-1}$. Let $\mathbb{Z}_k$ be the ring of integers modulo k with $k > 1$, and denote by $\mathbb{Z}_k^m$ the set of m-tuples over $\mathbb{Z}_k$. If k is a prime, then $\mathbb{Z}_k$ is the finite field of k elements. We use bold notation $\mathbf{x} = [x_1, \ldots, x_m] \in \mathbb{Z}_k^m$ to denote vectors (or codewords) in $\mathbb{Z}_k^m$. We denote the set of $n \times n$ matrices with entries in a set S by $\mathcal{M}_n(S)$ (and in general, the set of $m \times n$ matrices $\mathcal{M}_{m,n}(S)$). Finally, overline $\overline{a}$ denotes complex conjugation of the complex number a.

2.1 Butson Hadamard matrices

Let H be a matrix of order n with complex entries of modulus 1. If the rows of H are pairwise orthogonal under the Hermitian inner product, then H is a *Hadamard matrix*. The term Hadamard matrix is more commonly used in the literature to refer to the special case with entries in $\{\pm 1\}$. In this paper, such a matrix will be call a *real Hadamard matrix*. A *Butson Hadamard (or simply Butson) matrix of order n and phase k* is a matrix $H \in \mathcal{M}_n(\langle\zeta_k\rangle)$ such that $HH^* = nI_n$, where I_n denotes the identity matrix of order n and H^* denotes the conjugate transpose of H. We write $\mathrm{BH}(n,k)$ for the set of such matrices. The simplest examples of Butson matrices are the Fourier matrices $F_n = [\zeta_n^{(i-1)(j-1)}]_{i,j=1}^n \in \mathrm{BH}(n,n)$. Real Hadamard matrices of order n, as they are usually defined, are the elements of $\mathrm{BH}(n,2)$. Denote the set of monomial matrices in $\mathcal{M}_n(\langle\zeta_k\rangle)$ by $\mathrm{Mon}_n(\langle\zeta_k\rangle)$. The phase and orthogonality of a matrix $H \in \mathrm{BH}(n,k)$ is preserved by multiplication on the left or right by an element of $\mathrm{Mon}_n(\langle\zeta_k\rangle)$ as well as by complex conjugation, i.e., $\overline{H} \in \mathrm{BH}(n,k)$. The action of pairs $(P,Q) \in \mathrm{Mon}_n(\langle\zeta_k\rangle)^2$ is defined by $H(P,Q) = PHQ^*$, and this action induces an equivalence relation on $\mathrm{BH}(n,k)$. If $H(P,Q) = H'$, then H and H' are said to be *equivalent*.

A matrix is said to be in *dephased form* if every entry in its first row and first column is equal to 1. Every matrix can be dephased by using equivalence operations. Throughout this paper all matrices are assumed to be dephased.

Example 2.1 Let $D_{q,m}$ be the m^{th} Kronecker power of the $q\times q$ Fourier matrix, i.e., $(D_{q,m})_{i,j} = \zeta_q^{\alpha_{i-1}\cdot\alpha_{j-1}}$, where $\alpha_0 = (0,\ldots,0), \alpha_1 = (0,0,\ldots,1), \ldots, \alpha_{q^m-1} = (q-1,\ldots,q-1)$ with $\alpha_i \in \mathbb{Z}_q^m$. $D_{q,m} \in \mathrm{BH}(q^m, q)$. When $q = 2$ this is the well known Sylvester Hadamard matrix of order 2^n.

Let us mention that when q is a prime number, $D_{q,m}$ is related to the generalized first order Reed-Muller code $R_q(1,m)$.

2.2 Bent functions and generalizations

The notion of bentness admits various generalizations. We use the one in Schmidt's survey [1]. For positive integers q, m, h, a map $f\colon \mathbb{Z}_q^m \to \mathbb{Z}_h$ is a *generalized bent function (GBF)* if

$$\left|\mathcal{W}_f(w)\right|^2 = q^m \quad \forall\, w \in \mathbb{Z}_q^m,$$

where $|z|$ as usual denotes the modulus of $z \in \mathbb{C}$ and $\mathcal{W}_f(w) = \sum_{x\in\mathbb{Z}_q^m} \zeta_h^{f(x)}\zeta_q^{-w\cdot x}$ (the so-called *the Walsh-Hadamard transform* of f) where $w \cdot x$ is the inner product $wx^\top$ of w and x. Thus, a GBF for $q = h = 2$ and even m is a (Boolean) bent function. For $h = q$, GBFs exist if m is even or $q \not\equiv 2 \mod 4$. However, no GBF with $h = q$, m odd, and $q \equiv 2 \mod 4$ is known.

Remark 2.2 The map $f\colon \mathbb{Z}_q^m \to \mathbb{Z}_h$ is a GBF if, and only if, there exists $X \in \mathcal{M}_{q^m,1}(\langle\zeta_h\rangle)$ with $X_i = \zeta_h^{f(\alpha_i)}$ is a solution of the system $\frac{1}{q^{m/2}}\overline{D}_{q,m}\, X = Y$ for some $Y \in \mathcal{M}_{q^m,1}(\{y \in \mathbb{C}\colon |y| = 1\})$ (the α_i's and $D_{q,m}$ are defined in Example 2.1).

For Boolean functions, $\mathcal{W}_f(w)$ is always an integer and if it is also bent then $\mathcal{W}_f(w) = 2^{m/2}(-1)^{f^\star(w)}$ for $f^\star\colon \mathbb{Z}_2^m \to \mathbb{Z}_2$ called the *dual* of f. As is well-known, the dual $f^\star$ is a bent function as well, and $(f^\star)^\star = f$. If $f = f^\star$, the bent function f is called *self-dual*.

For $q = h$ an odd prime and $f\colon \mathbb{Z}_q^m \to \mathbb{Z}_q$ a GBF, the value of its Walsh-Hadamard transform satisfies

$$\mathcal{W}_f(w) = \begin{cases} \pm\zeta_q^{f^\star(w)} q^{m/2} & q^m = 1 \mod 4; \\ \pm\sqrt{-1}\,\zeta_q^{f^\star(w)} q^{m/2} & q^m = 3 \mod 4, \end{cases}$$

where $f^\star\colon \mathbb{Z}_q^m \to \mathbb{Z}_q$, which again is called the dual of f. A GBF f is said to be (γ, u)-*self dual* if for all $w \in \mathbb{Z}_q^m$, $\mathcal{W}_f(w) = \gamma q^{m/2}\zeta_q^{uf(w)}$ where $\gamma \in \langle\zeta_4\rangle$ and $u \in \mathbb{Z}_q^*$. Here we are interested in the case $\gamma = 1$ and $u = -1$.

Example 2.3 Let $f\colon \mathbb{Z}_q^m \to \mathbb{Z}_q$ with $m = 2t$ be the map

$$f(x_1, \ldots, x_{2t}) = x_1x_{t+1} + \ldots + x_tx_{2t}$$

is a $(1,-1)$-self dual GBF.

Remark 2.4 If we consider $X = \zeta_q^f$ where f is the function defined in Example 2.3, then X is a solution of the system $\frac{1}{q^{m/2}}\overline{D}_{q,m}\, X = \overline{X}$. In other words, X is a self-dual bent sequence for $\overline{D}_{q,m}$.

The *nonlinearity* of a map $f\colon \mathbb{Z}_q^m \to \mathbb{Z}_q$ is the Hamming distance between f and the set of the q^{m+1} affine functions from $\mathbb{Z}_q^m$ to $\mathbb{Z}_q$. When q is a prime, the largest possible nonlinearity, denoted by $\rho_q(m)$, is the covering radius of the (generalized) first order Reed-Muller code $R_q(1,m)$ over $\mathbb{Z}_q$. For m even and q a prime, we have (see [2])

$$\rho_q(m) = q^{m-1}(q-1) - q^{m/2-1}.$$

Boolean ($q = 2$) bent functions are characterized as the Boolean functions in even dimension with the largest possible nonlinearity. However, a similar characterization does not apply for GBF in general (even when $q = h$ an odd prime and m even). For $q = h$ an odd prime, the nonlinearity of a GBF is known. Here we only mention that the nonlinearity of a $(1, u)$-self dual GBF for m even is $(q-1)q^{m-1} - (q-1)q^{m/2-1}$ (different to $\rho_q(m)$). For m odd, the determination of $\rho_q(m)$ is an open problem in general.

3 Self-dual bent sequences for Butson matrices

In Remark 2.4, we have seen that for $n = q^m$ and $k = q$ there are self-dual bent sequences for $\overline{D}_{q,m}$ when m is even. In this Section, we show further progress on the study of self-dual bent sequences for Butson matrices.

Firstly, we study necessary conditions of existence for self-dual bent sequences over $\mathrm{BH}(n, k)$ for $k = 2, 3$ and 4.

Proposition 3.1 *If there exists at least one self-dual bent sequence for* $\mathrm{BH}(n, 3)$ *(resp.* $\mathrm{BH}(n, 4)$*), then* $n = 9m^2$ *(resp.* $n = 4m^2$*) with* m *a positive integer.*

We have checked by computer that there are self-dual bent sequences for, at least, one element of any of the three matrices in $\mathrm{BH}(9, 3)$ up to equivalence.

The necessary condition of existence for self-dual bent sequences for $\mathrm{BH}(n, 2)$ is also that $n = 4m^2$ (see [3]). Let us observe that our definition of self-dual in the real case and the one given in [3] are the same.

Proposition 3.2 *If* $H \in \mathrm{BH}(4m^2, 4)$ *is of Bush-type, then it has at least* 2^{2m} *self-dual bent sequences attached to* $-H$*.*

Secondly, we give more general results on the existence. The methods for obtaining them are based on some matrix analysis and the orthogonality relations in the matrices.

Proposition 3.3 *The map* $f\colon \mathbb{Z}_q^m \times \mathbb{Z}_q^m \to \mathbb{Z}_q$ *defined by* $\zeta_q^{f(\alpha_i,\alpha_j)} = \left(D_{q,m}\right)_{i,j}$ *is a* $(1, -1)$*-self dual GBF for any integer* $q > 1$*. In other words,* $X = \zeta_q^f$ *is a self-dual bent sequence for* $\overline{D}_{q,2m}$*.*

Remark 3.4 The GBF of Proposition 3.3 and Example 2.3 are the same.

Proposition 3.5 *If* $H \in \mathrm{BH}(n, k)$ *is symmetric then the sequence* $X_{(i-1)n+j} = (H)_{i,j}$ *is a self-dual bent sequence for* $H^* \otimes H^*$*.*

Example 3.6 Each of the representatives of the three classes of $\mathrm{BH}(9, 3)$ posted at `https://www.daneflannery.com/classifying-cocyclic-butson-hadamard-matrices` are symmetric. The Paley type II elements of $\mathrm{BH}(n, 2)$ are symmetric too.

Remark 3.7 The same argument of Proposition 3.5 runs for any symmetric Hadamard matrix. That is, if C is a Hadamard matrix of order n (i.e, the entries of C belong to the set of complex numbers of modulus 1 satisfying that $CC^* = nI$) which is symmetric, then $\frac{1}{n}\left(C^* \otimes C^*\right)X = \overline{X}$ where $X_{(i-1)n+j} = (C)_{i,j}$. Hence, X is a self-dual bent sequence for $C^* \otimes C^*$.

4 On the covering radius of Butson codes

For the remainder of this section we assume, for convenience, every Butson matrix is represented in logarithmic form and we are using the Hamming distance.

The *covering radius* of a $\mathbb{Z}_k$-code C of length n is defined by $r(C) = \max_{x\in\mathbb{Z}_k^n} \min_{y\in C} d(x,y)$. Let $H \in \mathrm{BH}(n,k)$. We denote by F_H the $\mathbb{Z}_k$-code of length n consisting of the rows of H, and we denote by C_H the $\mathbb{Z}_k$-code defined as $C_H = \cup_{\alpha\in\mathbb{Z}_k}(F_H + \alpha\mathbf{1})$ where $\mathbf{1}$ denotes the all-one vector (and $\alpha\mathbf{1}$ the all-α vector). The code C_H over $\mathbb{Z}_k$ is called a *Butson Hadamard code* (briefly, BH-code).

If $H \in \mathrm{BH}(n,k)$, then the *deviation* $\Theta(C_H,\mathbf{x})$ of an arbitrary vector $\mathbf{x} \in \mathbb{Z}_k^n$ from C_H is defined as

$$\Theta(C_H,\mathbf{x}) = \max\{|\langle \mathbf{x},\mathbf{y}\rangle| \colon \mathbf{y} \in C_H\},$$

where $\langle \mathbf{x},\mathbf{y}\rangle = (\zeta_k^{x_1},\ldots,\zeta_k^{x_n})(\zeta_k^{y_1},\ldots,\zeta_k^{y_n})^* = \sum_{i=1}^n \zeta_k^{x_i-y_i}$. Then the *total deviation* of C_H is

$$\Theta(C_H) = \min\{\Theta(C_H,\mathbf{x}) \colon \mathbf{x} \in \mathbb{Z}_k^n\}.$$

Proposition 4.1 *Let $H \in \mathrm{BH}(n,3)$. Then, C_H is a $(n, 3n, 2/3n)$ code and $r(C_H) \geq 2/3(n - \Theta(C_H))$. If there is a bent sequence for $H \in \mathrm{BH}(n,3)$, then $\Theta(C_H) = \sqrt{n}$.*

Example 4.2 We can always choose $H \in \mathrm{BH}(9,3)$ such that there is a self-dual bent sequence for H (this is always possible for the three equivalence classes). Then, $r(C_H) \geq 4$. On the other hand, the covering radius of the generalized Reed-Muller code $R_3(1,2)$ is 5. Let us point out that $R_3(1,2)$ and $C_{D_{3,2}}$ are equivalent.

Acknowledgement

The first author was supported by Spanish Strategic R+D project TED2021-130566B-I00.

References

[1] B. Schmidt, *A survey of group invariant Butson matrices and their relation to generalized bent functions and various other objects.* Radon Ser. Comput. Appl. Math. 23 (2019), 241–251.

[2] K-U Schmidt, *Highly nonlinear functions over finite fields.* Finite Fields Appl. 63 (2020), 101640.

[3] P. Solé, W. Cheng, S. Guilley, and O. Rioul, *Bent Sequences over Hadamard Codes for Physically Unclonable Functions.* IEEE International Symposion on Inf. Theory (2021), 801–806.

[4] M. Shi, Y. Li, W. Cheng, D. Crnkovic, D. Krotov, and P. Solé, *Self-dual bent sequences for complex Hadamard matrices.* Des. Codes Cryptogr. (2022). https://doi.org/10.1007/s10623-022-01157-6

Upper bounds on the numbers of binary plateaued and bent functions

V. N. Potapov
Sobolev Institute of Mathematics, vpotapov@math.nsc.ru

30th July 2023

1 Introduction

Bent functions are maximally nonlinear boolean functions with an even number of variables and are optimal combinatorial objects. In cryptography, bent functions are used in block ciphers. They are the source of nonlinearity and provide confusion in cryptosystems. Moreover, bent functions have many theoretical applications in discrete mathematics. Full classification of bent functions would be very useful for combinatorics and cryptography. But constructive classifications and enumerations of bent functions in n variables are likely impossible for large n.

The numbers of n-variable bent functions are only known for $n \le 8$. There exist 8 bent functions for $n = 2$, 896 for $n = 4$, approximately $2^{32.3}$ for $n = 6$ and $2^{106.3}$ for $n = 8$ [5]. Thus, lower and upper asymptotic bounds on the number of bent functions are very interesting. Currently, there exists a drastic gap between the upper and lower bounds of this number. Let $\mathcal{N}(n) = \log_2 |\mathcal{B}(n)|$, where $\mathcal{B}(n)$ is the set of boolean bent functions in n variables. The best known asymptotic lower bound on the number of boolean bent functions is proven in [9]. It holds $\mathcal{N}(n) \ge \frac{3n}{4} 2^{n/2}(1 + o(1))$ as n is even and $n \to \infty$. This bound is slightly better than the bound $\mathcal{N}(n) \ge \frac{n}{2} 2^{n/2}(1 + o(1))$ based on the Maiorana–McFarland construction of bent functions.

It is well known (see e.g. [2], [4], [6]) that the algebraic degree of a boolean bent function in n variables is at most $n/2$. Therefore, $\mathcal{N}(n) \le \sum_{i=0}^{n/2} \binom{n}{i} = 2^{n-1} + \frac{1}{2}\binom{n}{n/2}$. The bounds in [3] and [1] are of type $\mathcal{N}(n) \le 2^{n-1}(1+o(1))$. A better upper bound $\mathcal{N}(n) \le \frac{3}{4} \cdot 2^{n-1}(1+o(1))$ is proven in [7]. In this paper we improve it. We obtained that $\mathcal{N}(n) < \frac{11}{16} \cdot 2^{n-1}(1+o(1))$ (Theorem 2). Note that Tokareva's conjecture (see [10] and [6]) of the decomposition of boolean functions into sums of bent functions implies that $\mathcal{N}(n) \ge \frac{1}{2} 2^{n-1} + \frac{1}{4}\binom{n}{n/2}$.

The bounds mentioned above are asymptotic. We can use the suggested method to find a non-asymptotic upper bound. But for fixed $n = 6$ and $n = 8$ such bound is greater than the number of $\frac{2}{3} \cdot 2^{n-1}$ in two times. The main reason of this difference lies in the cardinality of the middle layer of the n-dimensional boolean cube. This cardinality is asymptotically negligible, but that is not the case for $n = 6$ and $n = 8$.

The new upper bound on the number of bent functions is based on new asymptotic upper bound on the number of s-plateaued boolean functions in n variables (Theorem 1). s-Plateaued functions are a generalization of bent functions, which are the same as 0-plateaued functions. Plateaued functions can combine important cryptographic properties of nonlinearity and correlation immunity.

The method of the proof of the listed above bounds implies a storage algorithm for bent and plateaued functions. The number of bits required by the algorithm is equal to the corresponding upper bound.

2 Walsh–Hadamard transform

Let $\mathbb{F} = \{0, 1\}$. The set $\mathbb{F}^n$ is called a boolean hypercube (or a boolean n-cube). $\mathbb{F}^n$ equipped with coordinate-wise modulo 2 addition $\oplus$ can be considered as an n-dimensional vector space. Define by $\langle x, y\rangle = x_1y_1 \oplus \cdots \oplus x_ny_n$ the inner product of vectors x and y.

Let G be a function that maps from the boolean hypercube to real numbers. Denote by $\widehat{G}(y) = \sum\limits_{x\in\mathbb{F}^n} G(x)(-1)^{\oplus\langle x,y\rangle}$ the Fourier transform of G. We can define the Walsh–Hadamard transform of a boolean function $f : \mathbb{F}^n \to \mathbb{F}$ by the formula $W_f(y) = \widehat{(-1)^f}(y)$. A boolean function b is called a bent function if $W_b(y) = \pm 2^{n/2}$ for all $y \in \mathbb{F}^n$. It is easy to see that n-variable bent functions exist only if n is even. A boolean function p is called an s-plateaued function if $W_p(y) = \pm 2^{(n+s)/2}$ or $W_p(y) = 0$ for all $y \in \mathbb{F}^n$. So, bent functions are 0-plateaued functions. 1-Plateaued functions are called near-bent.

From Parseval's identity $\sum\limits_{y\in\mathbb{F}^n} \widehat{H}^2(y) = 2^n \sum\limits_{x\in\mathbb{F}^n} H^2(x)$, where $H : \mathbb{F}^n \to \mathbb{C}$, it follows straightforwardly:

Proposition 1. *For every s-plateaued function, a proportion of nonzero values of its Walsh–Hadamard transform is equal to $\frac{1}{2^s}$.*

It is well known (see e.g. [2]) that for any function $H, G : \mathbb{F}^n \to \mathbb{C}$ it holds

$$\widehat{H * G} = \widehat{H} \cdot \widehat{G}, \qquad \widehat{(\widehat{H})} = 2^n H \qquad \text{and} \qquad 2^n H * G = \widehat{\widehat{H} \cdot \widehat{G}}, \tag{1}$$

where $H * G(z) = \sum\limits_{x\in\mathbb{F}^n} H(x)G(z \oplus x)$ is a convolution. Let Γ be a subspace of hypercube. Denote by $\Gamma^\perp$ a dual subspace, i.e., $\Gamma^\perp = \{y \in \mathbb{F}^n : \forall x \in \Gamma, \langle x, y\rangle = 0\}$. Let $\mathbf{1}_S$ be an indicator function for $S \subset \mathbb{F}^n$. It is easy to see that for every subspace Γ it holds $\widehat{\mathbf{1}_{\Gamma^\perp}} = 2^{n-\dim\Gamma}\mathbf{1}_\Gamma$. By (1) we have

$$H * \mathbf{1}_{\Gamma^\perp} = 2^{-\dim\Gamma}\widehat{\widehat{H} \cdot \mathbf{1}_\Gamma} \tag{2}$$

for any subspace $\Gamma \subset \mathbb{F}^n$.

Denote by $\mathrm{supp}(G) = \{x \in \mathbb{F}^n : G(x) \neq 0\}$ a support of G. We need the following known property of bent functions (see e.g. [6]).

Proposition 2. *Let f be an n-variable bent function and let Γ be a hyperplane. Consider $h = f \cdot \mathbf{1}_\Gamma$ as an $(n-1)$-variable function. Then h is a 1-plateaued function.*

3 Möbius transform

Denote by $\mathrm{wt}(z)$ a number of units in $z \in \mathbb{F}^n$. Every boolean function f can be represented as a polynomial

$$f(x_1, \ldots, x_n) = \bigoplus_{y \in \mathbb{F}^n} M[f](y) x_1^{y_1} \cdots x_n^{y_n},$$

where $x^0 = 1, x^1 = x$, and $M[f] : \mathbb{F}^n \to \mathbb{F}$ is the Möbius transform of f. Note that $M[M[f]] = f$ for each boolean function. The degree of this polynomial is called the algebraic degree of f.

Denote by $b(n, r)$ the cardinality of a ball $B_{n,r}$ with radius r in $\mathbb{F}^n$, i.e., $b(n, r) = |\{x \in \mathbb{F}^n : \mathrm{wt}(x) \le r\}|$. By properties of the Möbius transform, the number of n-variable boolean functions with degree $\deg f \le r$ is equal to $2^{b(n,r)}$.

Lemma 1 ([7]). *Suppose that f and g are n-variable boolean functions and $\max\{\deg(f), \deg(g)\} \le r$. If $f|_{B_{n,r}} = g|_{B_{n,r}}$ then $f = g$.*

Lemma 2 ([2], Theorem 2). *Let f be an n-variable boolean function. Suppose for every $v \in \mathbb{F}^n$ it holds $\widehat{(-1)^f}(v) = 2^k m(v)$, where $m(v)$ is integer. Then $\deg(f) \le n - k + 1$.*

Corollary 1 ([2], Proposition 96). *The degree of n-variable s-plateaued functions is not greater than $\frac{n-s}{2} + 1$.*

Note that degrees of bent (0-plateaued) functions is $n/2$ at most (see e.g. [2], [4], [6]). But for 1-plateaued function the bound $\frac{n+1}{2}$ is tight.

Proposition 3. *Let f be an n-variable bent function. Then for any hyperplane Γ the degree of the boolean function $h = \mathrm{supp}(\widehat{(-1)^f \cdot \mathbf{1}_\Gamma})$ is not greater than $n/2$.*

4 Subspace distribution

We will use the following well-known criterium (see, e.g. [2], Proposition 96).

Lemma 3. *An n-variable boolean function f is s-plateaued if and only if $(-1)^f * (-1)^f * (-1)^f = 2^{n+s}(-1)^f$.*

Consider an n-variable s-plateaued boolean function f and any fixed $x \in \mathbb{F}^n$. There are $V = \frac{(2^n-1)(2^n-2)}{6}$ 2-dimensional affine subspaces which contain x. Let $S(x)$ be a number of the subspaces that contain an odd number of zero values of f. By Lemma 3 we obtain

Corollary 2. *For any fixed $x \in \mathbb{F}^n$, $\frac{S(x)}{V} = \frac{1}{2} - \frac{1}{2} \cdot \frac{2^{n+s} - 3 \cdot 2^n + 2}{(2^n-1)(2^n-2)}$.*

Thus we have two equations: $\frac{S(x)}{V} = \frac{1}{2} + \frac{1}{2(2^{n-1}-1)}$ for every bent function and $\frac{S(x)}{V} = \frac{1}{2} + \frac{1}{2(2^n-1)}$ for every 1-plateaued function. We will use the following property of bent and plateaued functions.

Proposition 4 ([2], [4], [6]). *Let $f : \mathbb{F}^n \to \mathbb{F}$ be an s-plateaued function, let $A : \mathbb{F}^n \to \mathbb{F}^n$ be a non-degenerate affine transformation and let $\ell : \mathbb{F}^n \to \mathbb{F}$ be an affine function. Then $g = (f \circ A) \oplus \ell$ is an s-plateaued function.*

Functions f and g from Proposition 4 are called AE-equivalent. It is easy to see that the cardinality of any equivalence class is not greater than $a_n = 2^{n^2+n+1}(1+o(1))$. Note that two AE-equivalent functions f and g have the same algebraic degree as $\deg(f) > 1$.

There are 8 boolean 2-variable functions such that take value 0 even times. All of them are affine. 6 of them take value 0 two times and the other take value 0 four or zero times. Consider a 2-dimensional affine subspace Γ and an n-variable boolean function g. Let g take value 0 even times on Γ. It is easy to see that 3/4 among functions of the set $\{g \oplus \ell : \ell \text{ is an affine function}\}$ take value 0 two times and the other take value 0 four or zero times. Consequently, from Propositions 2 and 4 we deduced:

Corollary 3. *Let Γ be a 2-dimensional face (axes-aligned plane) of the hypercube and let $f : \mathbb{F}^n \to \mathbb{F}$ be an s-plateaued function. There exists a non-degenerate affine transformation A and an affine function ℓ such that the s-plateaued function $g = (f \circ A) \oplus \ell$ satisfies the following conditions.*

(a) *The number of faces $\Gamma \oplus y$, $y \in \mathbb{F}^n$, that contain an odd number of zero values of g, is less than 2^{n-3}.*

(b) *Among the faces $\Gamma \oplus y$, $y \in \mathbb{F}^n$, that contain an even number of zero values of g, not less than one fourth part contain four or zero values 0.*

Let p_0 be a probability of an even number of zero values in a 2-dimensional face and let p_1 be a probability of an odd number of zero values in a 2-dimensional face. Moreover, p'_0 is the probability of two zero value in a 2-dimensional face and $p'_0 < 3p_0/4$. How many bits on average we need to find four values $(-1)^{g(x)}$ from their sum in a 2-dimensional face? Under conditions (a) and (b) from the corollary, it is sufficient $p'_0 \log_2 6 + 2p_1 \leq 1 + \frac{3}{8}\log_2 6 = \alpha \approx 1.969$ bits by Shannon's theorem.

5 Main results

Denote by $\hbar$ Shannon's entropy function, i.e., $\hbar(p) = -p\log p - (1-p)\log(1-p)$ for $p \in (0,1)$. Let $\mathcal{N}(n,s)$ be the binary logarithm of the number of n-variable s-plateaued boolean functions. Since the Walsh–Hadamard transform is a bijection, $\mathcal{N}(n,s)$ is not greater than the number of bits such that is sufficient to identify W_f for an s-plateaued function f. Therefore, by Shannon's theorem and Proposition 1 we obtain inequality:

$$\mathcal{N}(n,s) \leq 2^n \left(\hbar(\frac{1}{2^s})(1+o(1)) + \frac{1}{2^s} \right). \tag{3}$$

Let $\mathcal{N}_0(n,1)$ be the binary logarithm of the number of n-variable 1-plateaued boolean functions which are obtained by a restriction of $(n+1)$-variable bent functions into hyperplanes.

Theorem 1. (a) $\mathcal{N}(n,s) \leq (\alpha b(n-2, \lceil \frac{n-s}{2} \rceil + 1) + 2^{n-2}(\hbar(\frac{1}{2^s}) + \frac{1}{2^s}))(1+o(1))$ *where $s > 0$ is fixed and $n \to \infty$.*

(b) $\mathcal{N}_0(n,1) \leq b(n-2, \frac{n+1}{2})(\alpha + \frac{3}{2})(1+o(1))$ *as $n \to \infty$.*

The main idea of the proof is the following. Let f be an s-plateaued function. We count the number of possible restrictions of W_f into $(n-2)$-dimensional face by (3). Let

we have such restrictions of W_f. By (2) we recover f on the ball with an appropriate radius. By Corollary 3 and the entropy estimation α we find the number of bits needed for this recovering. By Lemma 1 and Corollary 1 we restore f in full.

Theorem 2. $\mathcal{N}(n) \le \mathcal{N}_0(n-1,1) + 2^{n-3}(1+o(1)) \approx \frac{11}{32}2^n(1+o(1))$ *as* $n \to \infty$.

The proof is similar to the previous one. By Proposition 2 the restriction of a bent function into a hyperplane is a 1-plateaued function. We have counted these functions in Theorem 1 (b). Then we count the number of 1-plateaued function in $(n-1)$ variables corresponding to one n-variable bent function. Completed proofs are available in [8].

References

[1] S.V. Agievich, "On the continuation to bent functions and upper bounds on their number," Prikl. Diskr. Mat. Suppl., no. 13, 2020, pp. 18–21 (in Russian).

[2] C. Carlet, *Boolean Functions for Cryptography and Coding Theory*. Cambridge University Press, 562 pages, 2020.

[3] C. Carlet and A. Klapper, "Upper bounds on the number of resilient functions and of bent functions," Proceedings of the 23rd Symposium on Information Theory in the Benelux, Louvain-La-Neuve, Belgium. 2002.

[4] C. Carlet and S. Mesnager, "Four decades of research on bent functions," Des. Codes Cryptogr., vol. 78(1), 2016, pp. 5–50.

[5] P. Langevin, G. Leander, P. Rabizzoni, P. Veron, and J.-P. Zanotti. "Counting all bent functions in dimension eight 99270589265934370305785861242880," In Des. Codes Cryptography 59 (1-3), pages 193-205, 2011.

[6] S. Mesnager, *Bent Functions: Fundamentals and Results*. Springer International Publishing Switzerland, 2016.

[7] V.N. Potapov, "An Upper Bound on the Number of Bent Functions," 2021 XVII International Symposium on Problems of Redundancy in Information and Control Systems (25-29 October 2021 Moscow, Russia).IEEE, 2021. P. 95–96.

[8] V.N. Potapov, "Upper bounds on the numbers of binary plateaued and bent functions," DOI:10.48550/arXiv.2303.16547

[9] V.N. Potapov, A.A. Taranenko, Yu.V. Tarannikov, "Asymptotic bounds on numbers of bent functions and partitions of the Boolean hypercube into linear and affine subspaces," *Designs, Codes and Cryptography*, 2023. DOI: 10.1007/s10623-023-01239-z

[10] N. Tokareva, "On the number of bent functions from iterative constructions: lower bounds and hypothesis," Adv. Math. Commun., vol. 5(4), 2011, pp. 609–621.

On bent functions satisfying the dual bent condition

Alexandr Polujan[1], Enes Pasalic[2], Sadmir Kudin[2], Fengrong Zhang[3,4]

[1] Otto-von-Guericke-Universität, Universitätsplatz 2, 39106, Magdeburg, Germany
alexandr.polujan@gmail.com

[2] University of Primorska, FAMNIT & IAM, Glagoljaška 8, 6000 Koper, Slovenia
{enes.pasalic6@gmail.com, sadmir.kudin@iam.upr.si}

[3] State Key Laboratory of Integrated Services Networks,
Xidian University, Xian 710071, P.R. China

[4] Mine Digitization Engineering Research Center of Ministry of Education,
China University of Mining and Technology, Xuzhou, Jiangsu 221116, China
zhfl203@163.com

Abstract

For a concatenation of four bent functions $f = f_1||f_2||f_3||f_4$, the necessary and sufficient condition that f is bent is that the *dual bent condition* is satisfied [5, Theorem III.1], i.e., $f_1^*+f_2^*+f_3^*+f_4^* = 1$. However, specifying four bent functions satisfying this duality condition is in general quite a difficult task. Commonly, to simplify this problem, certain connections between f_i are assumed such as the one considered originally in [4] and later analyzed in [2]. Among them, is the construction method of bent functions satisfying the dual bent condition using the permutations of $\mathbb{F}_2^m$ with the $(\mathcal{A}_m)$ property [2, Theorem 7]. In this paper, we generalize this result and provide a construction of new permutations with the $(\mathcal{A}_m)$ property from the old ones. Combining these two results, we obtain a recursive construction method of bent functions satisfying the dual bent condition. Consequently, we provide a condition on the functions f_1, f_2, f_3, f_4, such that obtained with our approach bent functions are not equivalent to Maiorana-McFarland ones. Finally, with our construction method, we explain how one can construct homogeneous cubic bent functions, of which constructions only very few are known.

Keywords: Boolean bent function, dual bent condition, Maiorana-McFarland class, bent 4-concatenation, equivalence.

1 Preliminaries

Let $n = 2m$ and let $\mathcal{B}_n$ denote the set of Boolean functions in n variables. A function $f \in \mathcal{B}_n$ is called *bent*, if for all non-zero $a \in \mathbb{F}_2^n$ the first-order derivatives $D_a f(x) = f(x + a) + f(x)$ are balanced. Let $f_1, f_2, f_3, f_4 \in \mathcal{B}_n$ be four bent functions satisfying the dual bent condition. Then the function $f = f_1||f_2||f_3||f_4 \in \mathcal{B}_{n+2}$ defined by

$$f(z, z_{n+1}, z_{n+2}) = f_1(z)+z_{n+1}(f_1+f_3)(z)+z_{n+2}(f_1+f_2)(z)+z_{n+1}z_{n+2}(f_1+f_2+f_3+f_4)(z) \quad (1.1)$$

is bent and called the *bent 4-concatenation* of f_1, f_2, f_3, f_4, see [1]. As the following result shows, the dual bent condition could be satisfied [2] by using Maiorana-McFarland bent functions arising from permutations with the $(\mathcal{A}_m)$ property [6], which means that for three permutations π_i of $\mathbb{F}_2^m$, we have that $\pi_1 + \pi_2 + \pi_3 = \pi$ is also a permutation and $\pi^{-1} = \pi_1^{-1} + \pi_2^{-1} + \pi_3^{-1}$.

Theorem 1.1. *[2, Theorem 7] Let $f_j(x, y) = Tr(x\pi_j(y))+h_j(y)$ for $j \in \{1, 2, 3\}$ and $x, y \in \mathbb{F}_{2^m}$, where the permutations π_j satisfy the condition $(\mathcal{A}_m)$. If the functions h_j satisfy*

$$h_1(\pi_1^{-1}(x)) + h_2(\pi_2^{-1}(x)) + h_3(\pi_3^{-1}(x)) + (h_1 + h_2 + h_3)((\pi_1 + \pi_2 + \pi_3)^{-1}(x)) = 1, \quad (1.2)$$

then f_1, f_2, f_3 satisfy $f_1^* + f_2^* + f_3^* + f_4^* = 1$, where $f_1 + f_2 + f_3 = f_4$.

2 Constructing bent functions satisfying the dual bent condition recursively

First, we provide a generalization of Theorem 1.1. We omit the proof of this statement in order to explain in detail those results, which are more technical.

Theorem 2.1. *Let $f_j(x,y) = Tr\left(x\pi_j(y)\right) + h_j(y)$ for $j \in \{1,2,3\}$ and $x, y \in \mathbb{F}_{2^m}$ with $n = 2m$, where the permutations π_j satisfy the condition $(\mathcal{A}_m)$, and let $s \in \mathcal{B}_m$. Define a function $h_4 \in \mathcal{B}_m$ as $h_4 = h_1 + h_2 + h_3 + s$ and a bent function $f_4 \in \mathcal{B}_n$ as $f_4 = f_1 + f_2 + f_3 + s$. If the functions h_j satisfy*

$$h_1\left(\pi_1^{-1}(x)\right) + h_2\left(\pi_2^{-1}(x)\right) + h_3\left(\pi_3^{-1}(x)\right) + h_4\left(\left(\pi_1+\pi_2+\pi_3\right)^{-1}(x)\right) = 1, \tag{2.1}$$

then $f_1||f_2||f_3||f_4 \in \mathcal{B}_{n+2}$ is bent.

In the following example, we show the existence of permutations π_i and functions h_i with $h_4 \neq h_1 + h_2 + h_3$ satisfying the conditions of Theorem 2.1.

Example 2.2. Define the permutations π_i on $\mathbb{F}_2^4$ as follows:

$$\pi_1(y) = \begin{pmatrix} y_1 + y_2 + y_1y_4 + y_2y_4 + y_3y_4 \\ y_1 + y_1y_2 + y_3 + y_2y_3 + y_2y_4 \\ y_1y_2 + y_3 + y_1y_3 + y_2y_4 + y_3y_4 \\ y_1 + y_3 + y_1y_3 + y_2y_3 + y_4 + y_1y_4 + y_2y_4 \end{pmatrix}, \pi_2(y) = \pi_1(y) + \begin{pmatrix} y_2 + y_3 + y_4 \\ 1 + y_2 + y_3 + y_4 \\ y_1 + y_3 \\ y_1 + y_3 \end{pmatrix},$$

$$\pi_3(y) = \pi_1(y) + \begin{pmatrix} y_1 + y_4 \\ y_1 + y_2 \\ 1 + y_1 + y_2 \\ 1 + y_1 + y_4 \end{pmatrix}, \pi_4(y) = (\pi_1 + \pi_2 + \pi_3)(y).$$

The algebraic normal forms of the functions h_i are given as follows:

$$h_1(y) = y_1y_3y_4, \quad h_2(y) = y_2y_3 + y_1y_4 + y_2y_4 + y_3y_4 + y_1y_3y_4,$$
$$h_3(y) = y_1y_3 + y_2y_3 + y_3y_4 + y_1y_3y_4, \quad h_4(y) = (h_1 + h_2 + h_3)(y) + s(y),$$

where $s(y) = y_1 + y_2 + y_4$. One can check that the defined above permutations π_i of $\mathbb{F}_2^4$, satisfy the $(\mathcal{A}_4)$ property. Moreover, the condition (2.1) is satisfied as well, and thus by Theorem 2.1, we have that $f_1||f_2||f_3||f_4 \in \mathcal{B}_{10}$ is bent for bent functions $f_i(x,y) = x \cdot \pi_i(y) + h_i(y)$, where $x, y \in \mathbb{F}_2^4$.

Now, we show that as soon as a single example of such permutations π_i on $\mathbb{F}_2^m$ and Boolean functions h_i on $\mathbb{F}_2^m$ is found (here m is a fixed integer), then one can always construct many such examples on $\mathbb{F}_2^k$, where $k > m$ is an arbitrary integer.

Lemma 2.3. *Let σ_1, σ_2 be permutations of $\mathbb{F}_2^m$. Define the function $\pi\colon \mathbb{F}_2^{m+1} \to \mathbb{F}_2^{m+1}$ by*

$$\pi(y, y_{m+1}) = \left(y_{m+1}\sigma_1(y) + (1 + y_{m+1})\sigma_2(y), y_{m+1}\right), \text{ for all } y \in \mathbb{F}_2^m, y_{m+1} \in \mathbb{F}_2.$$

Then, π is a permutation, and its inverse on $\mathbb{F}_2^{m+1}$ is given by the permutation ρ on $\mathbb{F}_2^{m+1}$, defined by

$$\rho(y, y_{m+1}) = \left(y_{m+1}\sigma_1^{-1}(y) + (1 + y_{m+1})\sigma_2^{-1}(y), y_{m+1}\right), \text{ for all } y \in \mathbb{F}_2^m, y_{m+1} \in \mathbb{F}_2.$$

Now we are ready to provide a recursive construction of Maiorana-McFarland bent functions $f_1', f_2', f_3', f_4' \in \mathcal{B}_{n+2}$ satisfying the condition $(f_1')^* + (f_2')^* + (f_3')^* + (f_4')^* = 1$ from bent functions $f_1, f_2, f_3, f_4 \in \mathcal{B}_n$ satisfying the condition $f_1^* + f_2^* + f_3^* + f_4^* = 1$ using Theorem 2.1.

Proposition 2.4. *Let π_j for $j \in \{1,2,3\}$ be three permutations on $\mathbb{F}_2^m$ which satisfy the condition $(\mathcal{A}_m)$. Let σ be a permutation of $\mathbb{F}_2^m$. Denote by $\pi_4 = \pi_1 + \pi_2 + \pi_3$ and let Boolean functions h_j on $\mathbb{F}_2^m$ $j \in \{1,2,3,4\}$ satisfy*

$$h_1\left(\pi_1^{-1}(y)\right) + h_2\left(\pi_2^{-1}(y)\right) + h_3\left(\pi_3^{-1}(y)\right) + h_4\left(\pi_4^{-1}(y)\right) = 1.$$

Define four permutations ϕ_i on $\mathbb{F}_2^{m+1}$ as

$$\phi_i(y, y_{m+1}) = \begin{cases} (\pi_i(y), 1) & \text{if } y_{m+1} = 1 \\ (\sigma(y), 0) & \text{if } y_{m+1} = 0 \end{cases}, \quad \text{for all } y \in \mathbb{F}_2^m, y_{m+1} \in \mathbb{F}_2,$$

and four Boolean functions h_i' on $\mathbb{F}_2^{m+1}$ as follows

$$h_i'(y, y_{m+1}) = y_{m+1} h_i(y) \text{ for } i \in \{1,2,3\},$$
$$h_4'(y, y_{m+1}) = y_{m+1} h_4(y) + y_{m+1} + 1.$$

Then, the following hold.

1. *Permutations ϕ_1, ϕ_2, ϕ_3 satisfy the condition $(\mathcal{A}_m)$.*
2. *Functions h_j' satisfy*

$$h_1'\left(\phi_1^{-1}(y, y_{m+1})\right) + h_2'\left(\phi_2^{-1}(y, y_{m+1})\right) + h_3'\left(\phi_3^{-1}(y, y_{m+1})\right) + h_4'\left(\phi_4^{-1}(y, y_{m+1})\right) = 1,$$

   for all $y \in \mathbb{F}_2^m, y_{m+1} \in \mathbb{F}_2$, where $\phi_4 = \phi_1 + \phi_2 + \phi_3$.
3. *Boolean functions $f_j'(x', y') = Tr\left(x' \phi_j(y')\right) + h_j'(y')$ for $j \in \{1,2,3,4\}$ and $x', y' \in \mathbb{F}_2^{m+1}$ are bent, moreover, $f_1' || f_2' || f_3' || f_4' \in \mathcal{B}_{n+2}$ is bent as well.*

Proof. 1. The property $(\mathcal{A}_m)$ means that for three permutations ϕ_i on $\mathbb{F}_2^{m+1}$, we have that $\phi_1 + \phi_2 + \phi_3 = \phi_4$ is also a permutation and $\phi_4^{-1} = \phi_1^{-1} + \phi_2^{-1} + \phi_3^{-1}$. First, we show that ϕ_4 is a permutation. By definition of ϕ_4, we get that for all $y \in \mathbb{F}_2^m$, $y_{m+1} \in \mathbb{F}_2$ holds

$$\phi_4(y, y_{m+1}) = \begin{cases} ((\pi_1 + \pi_2 + \pi_3)(y), 1) & \text{if } y_{m+1} = 1 \\ (\sigma(y), 0) & \text{if } y_{m+1} = 0 \end{cases}.$$

Since $\pi_4 = \pi_1 + \pi_2 + \pi_3$ is a permutation, we get that ϕ_4 is a permutation as well. Now, we show that $\phi_4^{-1} = \phi_1^{-1} + \phi_2^{-1} + \phi_3^{-1}$. By Lemma 2.3, we have that for all $y \in \mathbb{F}_2^m$, $y_{m+1} \in \mathbb{F}_2$ holds

$$\phi_4^{-1}(y, y_{m+1}) = (\phi_1^{-1} + \phi_2^{-1} + \phi_3^{-1})(y, y_{m+1}),$$

from what follows that permutations ϕ_1, ϕ_2, ϕ_3 satisfy the condition $(\mathcal{A}_m)$.
2. Observe that for $j \in \{1,2,3\}$, we have that for all $y \in \mathbb{F}_2^m, y_{m+1} \in \mathbb{F}_2$ holds

$$h_i'(\phi_i^{-1}(y, y_{m+1})) = \begin{cases} h_i'(\phi_i^{-1}(y,1)) & \text{if } y_{m+1} = 1 \\ h_i'(\phi_i^{-1}(y,0)) & \text{if } y_{m+1} = 0 \end{cases} = \begin{cases} h_i(\pi_i^{-1}(y)) & \text{if } y_{m+1} = 1 \\ 0 & \text{if } y_{m+1} = 0 \end{cases}$$

Similarly, one can show that for all $y \in \mathbb{F}_2^m, y_{m+1} \in \mathbb{F}_2$ holds

$$h_4'(\phi_i^{-1}(y, y_{m+1})) = \begin{cases} h_4'(\phi_4^{-1}(y,1)) & \text{if } y_{m+1} = 1 \\ h_4'(\phi_4^{-1}(y,0)) & \text{if } y_{m+1} = 0 \end{cases} = \begin{cases} h_4((\pi_1 + \pi_2 + \pi_3)^{-1}(y)) & \text{if } y_{m+1} = 1 \\ 1 & \text{if } y_{m+1} = 0 \end{cases}.$$

Finally, for all $y \in \mathbb{F}_2^m, y_{m+1} \in \mathbb{F}_2$, we consider the sum

$$\sum_{i=1}^{4} h_i'\left(\phi_i^{-1}(y, y_{m+1})\right) = \begin{cases} \sum_{i=1}^{3} h_i\left(\phi_i^{-1}(y)\right) + h_4((\pi_1 + \pi_2 + \pi_3)^{-1}(y)) & \text{if } y_{m+1} = 1 \\ 1 & \text{if } y_{m+1} = 0 \end{cases} = 1,$$

since $h_1\left(\pi_1^{-1}(y)\right) + h_2\left(\pi_2^{-1}(y)\right) + h_3\left(\pi_3^{-1}(y)\right) + h_4((\pi_1 + \pi_2 + \pi_3)^{-1}(y)) = 1$ holds for all $y \in \mathbb{F}_2^m$.
3. The statement follows immediately from Theorem 2.1. □

3 Analysis of the obtained construction method

Recall that the set of all bent functions, which are extended-affine equivalent to functions of the form $f(x,y) = x \cdot \pi(y) + h(y)$ for $x, y \in \mathbb{F}_2^m$, where π is a permutation of $\mathbb{F}_2^m$, and $h \in \mathcal{B}_m$ is an arbitrary Boolean function is called the *completed Maiorana-McFarland class* and denoted by $\mathcal{M}^\#$. It is well-known [3] that a bent function $f \in \mathcal{B}_n$ belongs to the $\mathcal{M}^\#$ iff there exists a vector space U of dimension m, such that $D_a D_b f = 0$ for all $a, b \in U$; such a vector space is called [10] an *$\mathcal{M}$-subspace* of a bent function $f \in \mathcal{M}^\#$. Note that if $f \in \mathcal{M}$, then at least one $\mathcal{M}$-subspace of f has the form $U = \mathbb{F}_2^m \times \{0_m\}$, which we call the *canonical $\mathcal{M}$-subspace* of f.

Since in the bent 4-concatenation we consider bent functions $f_i \in \mathcal{B}_n$ in $\mathcal{M}^\#$, it is essential to specify the conditions on these functions such that the resulting function $f = f_1||f_2||f_3||f_4 \in \mathcal{B}_{n+2}$ is outside $\mathcal{M}^\#$. Otherwise one just gets a complicated construction method of bent functions in $\mathcal{M}^\#$. For this purpose, we will use the following description of $\mathcal{M}$-subspaces of $f = f_1||f_2||f_3||f_4 \in \mathcal{B}_{n+2}$.

Proposition 3.1. *[9] Let $f_1, f_2, f_3, f_4 \in \mathcal{B}_n$ be four Boolean functions (not necessarily bent), such that $f = f_1||f_2||f_3||f_4 \in \mathcal{B}_{n+2}$ is a bent function in $\mathcal{M}^\#$. Let $W \subset \mathbb{F}_2^{n+2}$ be an $\mathcal{M}$-subspace of f. Then, there exists an $(\frac{n}{2}-1)$-dimensional subspace V of $\mathbb{F}_2^n$ such that $V \times \{(0,0)\}$ is a subspace of W, and such that for all $i = 1, \ldots, 4$ the equality $D_a D_b f_i = 0$ holds for all $a, b \in V$.*

For the main result of this section, we will also need to define the (P_1) property, which was recently introduced in [9] for specifying Maiorana-McFarland bent functions with the unique canonical $\mathcal{M}$-subspace. We say that the mapping $\pi\colon \mathbb{F}_2^m \to \mathbb{F}_2^m$ has the property (P_1) if $D_v D_w \pi \neq 0_m$ for all linearly independent $v, w \in \mathbb{F}_2^m$.

Theorem 3.2. *Let $n = 2m$ for $m > 3$ and define three bent functions $f_i(x,y) = x \cdot \pi_i(y) + h_i(y)$, with $x, y \in \mathbb{F}_2^m$, for $i = 1, \ldots, 3$, where π_i satisfies the property (P_1) and additionally $\pi_1 + \pi_2$ satisfies the property (P_1), and furthermore we assume that the components of $\pi_1 + \pi_2$ do not admit linear structures. Define $f = f_1||f_2||f_3||f_4$ where $f_4(x,y) = f_1(x,y) + f_2(x,y) + f_3(x,y) + s(y)$ (consequently $h_4 = h_1 + h_2 + h_3 + s$) using suitable h_i so that the dual bent condition in (2.1) is satisfied. Then, the functions f_i share the unique canonical $\mathcal{M}$-subspace $U = \mathbb{F}_2^m \times \{0_m\}$ and furthermore bent function $f \in \mathcal{B}_{n+2}$ is outside $\mathcal{M}^\#$. In particular, the same conclusion is valid when $s(y) = 0$.*

Proof. Denoting $a = (a', a^{(1)}, a^{(2)})$ and $b = (b', b^{(1)}, b^{(2)})$ and $a', b' \in \mathbb{F}_2^n$ and $a^{(i)}, b^{(i)} \in \mathbb{F}_2$, the second-order derivative of f is given by $D_a D_b f(x, y_1, y_2) =$

$$\begin{aligned}
&= D_{a'} D_{b'} f_1(x) + y_1 D_{a'} D_{b'} f_{13}(x) + y_2 D_{a'} D_{b'} f_{12}(x) + y_1 y_2 D_{a'} D_{b'} f_{1234}(x) \\
&+ a^{(1)} D_{b'} f_{13}(x + a') + b^{(1)} D_{a'} f_{13}(x + b') + a^{(2)} D_{b'} f_{12}(x + a') + b^{(2)} D_{a'} f_{12}(x + b') \\
&+ (a^{(1)} y_2 + a^{(2)} y_1 + a^{(1)} a^{(2)}) D_{b'} f_{1234}(x + a') + (b^{(1)} y_2 + b^{(2)} y_1 + b^{(1)} b^{(2)}) \\
&\times D_{a'} f_{1234}(x + b') + (a^{(1)} b^{(2)} + b^{(1)} a^{(2)}) f_{1234}(x + a' + b'),
\end{aligned} \tag{3.1}$$

where $f_{i_1 \ldots i_k} := f_{i_1} + \cdots + f_{i_k}$. Since $D_u D_v \pi_i(y) \neq 0$ for any nonzero $u \neq v \in \mathbb{F}_2^m$ (as π_i satisfies the property (P_1), the functions f_i share the unique canonical $\mathcal{M}$-subspace $U = \mathbb{F}_2^m \times \{0_m\}$. For convenience, we denote $a' = (a_1, a_2)$ and $b' = (b_1, b_2)$, where $a_i, b_i \in \mathbb{F}_2^m$. W.l.o.g. we assume that $D_{a_2} D_{b_2}(\pi_1(y) + \pi_2(y)) \neq 0$ for any $a_2, b_2 \in \mathbb{F}_2^m$ ($a_2, b_2 \neq 0$ and distinct), and the term $y_2 D_{a'} D_{b'} f_{12}(x,y)$ in (3.1) cannot be canceled unless $a_2 = 0$ or $b_2 = 0$ or $a_2 = b_2$, which is due to the fact that (same can be deduced for $D_{(a_1,a_2)} D_{(b_1,b_2)} f_{13}(x,y)$)

$$\begin{aligned}
D_{(a_1,a_2)} D_{(b_1,b_2)} f_{12}(x,y) =& x \cdot (D_{a_2} D_{b_2}(\pi_1(y) + \pi_2(y))) + a_1 \cdot D_{b_2}(\pi_1 + \pi_2)(y + a_2) \\
&+ b_1 \cdot D_{a_2}(\pi_1 + \pi_2)(y + b_2) + D_{a_2} D_{b_2} h_{12}(y).
\end{aligned} \tag{3.2}$$

Thus, for any $a = (a_1, a_2, a^{(1)}, a^{(2)})$ and $b = (b_1, b_2, b^{(1)}, b^{(2)})$ in some $(m+1)$-dimensional subspace W of $\mathbb{F}_2^{2m+2}$, we necessarily have that either $a_2 = 0$ or $b_2 = 0$, alternatively $a_2 = b_2$.

Since the functions f_i share the unique canonical $\mathcal{M}$-subspace $U = \mathbb{F}_2^m \times \{0_m\}$, any other subspace V of $\mathbb{F}_2^m \times \mathbb{F}_2^m$ for which $D_{a'}D_{b'}f_i(x,y) = 0$ for all $a', b' \in V$ must have dimension less than m. By Proposition 3.1, if f defined on $\mathbb{F}_2^{2m+2}$ belongs to $\mathcal{M}^\#$ then for any $\mathcal{M}$-subspace W of f of dimension $m+1$ there must exist $V \subset \mathbb{F}_2^{2m}$ of dimension $m-1$ such that $D_aD_bf_i = 0$ for all $i = 1, \ldots, 4$ and any $a, b \in V$. Furthermore, $V \times (0,0)$ is a subspace of W. There are only two possibilities for V, i.e., either $V \subset U = \mathbb{F}_2^m \times \{0_m\}$ or $V \not\subset U$.

We first consider the case that $V \subset U = \mathbb{F}_2^m \times \{0_m\}$, where $\dim(V) = m-1$. Then, $V \times (0,0) \subset W$ and to extend this subspace to W, we need to adjoin two elements of $\mathbb{F}_2^{2m+2}$, say $u = (u_1, u_2, u^{(1)}, u^{(2)}), v = (v_1, v_2, v^{(1)}, v^{(2)}) \in \mathbb{F}_2^m \times \mathbb{F}_2^m \times \mathbb{F}_2 \times \mathbb{F}_2$, and $u' = (u_1, u_2)$, $v' = (v_1, v_2)$. Then, we cannot have the case that $u_2 = v_2 = 0_m$ since this would imply that f_{12} on $\mathbb{F}_2^n$ has an $\mathcal{M}$-subspace of dimension $n/2+1$ which is impossible (see for instance [8]). On the other hand, if $u_2 \neq v_2 \neq 0$ then again $y_1 D_{u'}D_{v'}f_{12}(x,y)$ cannot be canceled in (3.1). W.l.o.g. we assume that $u_2 = 0$ and $v_2 \neq 0$, which implies that $U \times (0,0) \subset W$. Hence, $W = \langle U \times (0,0), v \rangle$, where $v_2 \neq 0$. Notice that the case $u_2 = v_2$, which also might lead to $D_{u'}D_{v'}f_{12}(x,y) = 0$, reduces to this case since $u_2 + v_2 = 0$ and then $u' + v' \in U$. Now, we note that in $W = \langle U \times (0,0), v \rangle$ there must exist an element $z = (z', 0, 0)$ such that $z_1 = v_1$ and consequently $z' + v' = (0_m, v_2)$. Considering (3.2), and replacing $a' \to z' = (v_1, 0_m)$ and $b' \to (0_m, v_2)$, we have that only the term $v_1 \cdot D_{b_2}(\pi_1 + \pi_2)(y)$ remains, which cannot be zero due to our assumption that the components of $\pi_1 + \pi_2(y)$ do not admit linear structures.

The second case arises when $V \not\subset U$, where $\dim(V) = m-1$. Hence, V contains at least one element $a' = (a_1, a_2) \notin U$, so that $a_2 \neq 0$. If V contains one more element not in U, say b', then $D_{a'}D_{b'}f_{12}(x,y) \neq 0$ and consequently $D_aD_bf(x,y,y_1,y_2) \neq 0$. If V does not contain one more element which is not in U, then it can be extended to U (by replacing a' with some $(u_1, 0_m)$) and the above arguments apply. □

Monomial permutations satisfying the $(\mathcal{A}_m)$ property were specified in [7]. We show that in a small number of variables, it is possible to find suitable functions h_i, such that the conditions of Theorem 3.2 are satisfied.

Theorem 3.3. *[7] Let $m \geq 3$ be an integer and $d^2 \equiv 1 \mod 2^m - 1$. Let π_i be three permutations of $\mathbb{F}_2^m$ defined by $\pi_i(y) = \alpha_i y^d$, for $i = 1, 2, 3$, where $\alpha_i \in \mathbb{F}_{2^m}^*$ are pairwise distinct elements such that $\alpha_i^{d+1} = 1$ and $\alpha_4^{d+1} = 1$ where $\alpha_4 = \alpha_1 + \alpha_2 + \alpha_3$. Then, the permutations π_i satisfy the property $(\mathcal{A}_m)$ and furthermore π_i are involutions as well as $\pi_4 = \pi_1 + \pi_2 + \pi_3$.*

Example 3.4. Let $m = 4$ and the multiplicative group of $\mathbb{F}_{2^4}$ be given by $\mathbb{F}_{2^4}^* = \langle a \rangle$, where the primitive element a satisfies $a^4 + a + 1 = 0$. Let $d = 14$, which satisfies $d^2 \equiv 1 \mod 15$. Define $\alpha_1 = a, \alpha_2 = a^2, \alpha_3 = a^4$ and $\alpha_4 = \alpha_1 + \alpha_2 + \alpha_3 = a^8$. It is possible to check that for $i = 1, \ldots, 3$, the defined permutations π_i as well as $\pi_1 + \pi_2$ satisfy the property (P_1) and additionally the components of $\pi_1 + \pi_2$ do not admit linear structures. Define the following four Boolean functions $h_1(y) = 0, h_2(y) = Tr(y), h_3(y) = Tr(ay), h_4(y) = Tr(a^{13}y) + 1$, as well as four bent Maiorana-McFarland bent functions $f_i(x,y) = Tr(x\pi_i(y)) + h_i(y)$ for $i = 1, 2, 3, 4$, where $x, y \in \mathbb{F}_{2^3}$. Note that $h_1(y) + h_2(y) + h_3(y) + h_4(y) = s(y) = Tr(a^{11}y) + 1$, and hence, $f_4 = f_1 + f_2 + f_3 + s$. Since the functions h_i satisfy the condition (2.1) of Theorem 2.1, we have that $f = f_1||f_2||f_3||f_4 \in \mathcal{B}_8$. By Theorem 3.2, the function f is outside $\mathcal{M}^\#$.

Open Problem 3.5. 1. Find explicit infinite families of permutations π_i and Boolean functions h_i satisfying the conditions of Theorem 2.1. 2. Relax the conditions of Theorem 2.1. The latter question is motivated by the fact that even in $n = 6$ variables we were able to find permutations π_i and Boolean functions h_i in $m = 3$ variables, such that the concatenation of corresponding bent functions f_i is bent and outside $\mathcal{M}^\#$. These examples, however, cannot be covered by Theorem 2.1, since all permutations in 3 variables are quadratic, and hence, their components have linear structures.

4 An application to the design of homogeneous bent functions

A Boolean function is called *homogeneous* if all the monomials in its ANF have the same algebraic degree. Now, we show how bent functions satisfying the dual bent condition and permutations with the $(\mathcal{A}_m)$ property can be used for the construction of homogeneous bent functions.

Proposition 4.1. *Let $f_1 \in \mathcal{B}_n$ be a homogeneous cubic bent function. Let $q_1, q_2 \in \mathcal{B}_n$ be two homogeneous quadratic functions, such that $f_2 = f_1 + q_2$ and $f_3 = f_1 + q_3$ are bent, and additionally $f_1 + f_2 + f_3$ is also bent. Defining $f_4 = f_1 + f_2 + f_3 + s$ for $s \in \mathcal{B}_n$, the function $f = f_1||f_2||f_3||f_4 \in \mathcal{B}_{n+2}$ is homogeneous cubic bent iff $f_1^* + f_2^* + f_3^* = (f_1 + f_2 + f_3 + s)^* + 1$, where $s \in \mathcal{B}_n$ is a linear function.*

Example 4.2. Consider the following homogeneous functions $f_1, q_2, q_3, s \in \mathcal{B}_8$, which are given by their algebraic normal forms as follows:

$$\begin{aligned}
f_1(z) =& z_1z_2z_5 + z_1z_2z_8 + z_1z_3z_4 + z_1z_3z_5 + z_1z_3z_6 + z_1z_3z_7 + z_1z_4z_5 + z_1z_4z_7 + z_1z_4z_8 \\
&+ z_1z_5z_8 + z_1z_6z_8 + z_2z_3z_4 + z_2z_3z_5 + z_2z_4z_5 + z_2z_4z_6 + z_2z_4z_8 + z_2z_5z_6 + z_2z_6z_7 \\
&+ z_2z_6z_8 + z_2z_7z_8 + z_3z_4z_6 + z_3z_4z_8 + z_3z_5z_6 + z_3z_5z_7 + z_3z_6z_8 + z_4z_7z_8 + z_5z_6z_7 \\
&+ z_5z_6z_8, \\
q_2(z) =& z_1z_4 + z_1z_5 + z_1z_7 + z_5z_7 + z_1z_8 + z_4z_8 + z_6z_7 + z_6z_8 + z_7z_8, \\
q_3(z) =& z_1z_3 + z_1z_4 + z_1z_7 + z_1z_8 + z_2z_3 + z_2z_8 + z_3z_5 + z_3z_8 + z_4z_7 + z_5z_6 + z_6z_7 + z_7z_8, \\
s(z) =& z_1 + z_4 + z_6 + z_8.
\end{aligned}$$

One can check that the functions $f_1, q_2, q_3, s \in \mathcal{B}_8$ satisfy the conditions of Proposition 4.1, and hence $f = f_1||f_2||f_3||f_4 \in \mathcal{B}_{10}$ constructed as in Proposition 4.1 is homogeneous cubic bent. Notably, there exists a linear non-degenerate transformation $z \mapsto zA$ such that $f_i(zA) = x \cdot \pi_i(y) + h_i(y)$, where permutations π_i and Boolean functions h_i are defined in Example 2.2, and hence, permutations π_i have the $(\mathcal{A}_4)$ property. Finally, we note that the function $f \notin \mathcal{M}^{\#}$ since the functions f_i satisfy the conditions of [9, Theorem 5.11].

Open Problem 4.3. Find explicit infinite families of homogeneous bent functions using the dual bent condition and permutations with the $(\mathcal{A}_m)$ property.

References

[1] A. Canteaut, P. Charpin. "Decomposing bent functions". *IEEE Transactions on Information Theory*, vol. 49, no. 8, pp. 2004–2019 (2003). p. 1.

[2] N. Cepak, E. Pasalic, A. Muratović-Ribić. "Frobenius linear translators giving rise to new infinite classes of permutations and bent functions". *Cryptogr. Commun.* 11(6): 1275–1295 (2019). p. 1.

[3] J. F. Dillon. "Elementary Hadamard difference sets". Ph.D. dissertation. *University of Maryland, USA* (1974). p. 4.

[4] S. Hodžić, E. Pasalic, Y. Wei. "A general framework for secondary constructions of bent and plateaued functions". *Des. Codes Cryptogr.* 88(10): 2007–2035 (2020). p. 1.

[5] S. Hodžić, E. Pasalic, W. G. Zhang. "Generic constructions of five-valued spectra Boolean functions". *IEEE Trans. Inf. Theory* 65(11): 7554–7565 (2019). p. 1.

[6] S. Mesnager. "Further constructions of infinite families of bent functions from new permutations and their duals". *Cryptogr. Commun.* 8, 229–246 (2016). p. 1.

[7] S. Mesnager, G. D. Cohen, and D. Madore. "On existence (based on an arithmetical problem) and constructions of bent functions". In: Groth, J. (eds) Cryptography and Coding. IMACC 2015. Lecture Notes in Computer Science, vol 9496. Springer, Cham., (2015). p. 5.

[8] E. Pasalic, A. Bapic, F. Zhang, Y. Wei. "Explicit infinite families of bent functions outside the completed Maiorana-McFarland class". *Des. Codes Cryptogr.* (2023). p. 5.

[9] E. Pasalic, A. Polujan, S. Kudin, F. Zhang. "Design and analysis of bent functions using $\mathcal{M}$-subspaces". arXiv preprint arXiv:2304.13432 (2023) pp. 4 and 6.

[10] A. Polujan, A. Pott. "Cubic bent functions outside the completed Maiorana-McFarland class". *Des. Codes Cryptogr.* 88, 1701–1722 (2020). p. 4.

Asymptotic Lower Bounds On The Number Of Bent Functions Having Odd Many Variables Over Finite Fields of Odd Characteristic

V. N. Potapov * and Ferruh Özbudak**

*Sobolev Institute of Mathematics, Novosibirsk, Russia e-mail: vpotapov@math.nsc.ru
**Faculty of Engineering and Natural Sciences, Sabancı University, 34956, Istanbul, and Middle East Technical University, 06800, Ankara, Turkey, e-mail:ozbudak@metu.edu.tr

Abstract

Using recent deep results of Keevash et al. [8] and Eberhard et al. [6] together with further new detailed techniques in combinatorics, we present constructions of two concrete families of generalized Maiorana-McFarland bent functions. Our constructions improve the lower bounds on the number of bent functions in n variables over a finite field $\mathbb{F}_p$ if p is odd and n is odd in the limit as n tends to infinity.

Let p be a prime. Let $\mathbb{F}_p$ be the finite field with p elements. For a set A, let $|A|$ denote its cardinality. Let $\ln(\cdot)$ be the natural logarithm function.

Bent functions were first introduced by Rothaus in 1976 [14] over $\mathbb{F}_2$. In 1985, Kumar et al. generalized the notion of bent function to arbitrary finite fields [9]. We prefer to introduce bent functions as a special class of functions, namely, plateaued functions.

For a function $f : \mathbb{F}_p^n \to \mathbb{F}_p$ and $\alpha \in \mathbb{F}_p^n$, let $\hat{f} : \mathbb{F}_{p^n} \to \mathbb{C}$ be the Walsh Transform of f at α defined as

$$\hat{f}(\alpha) = \sum_{x \in \mathbb{F}_p^n} e^{\frac{2\pi\sqrt{-1}}{p}(f(x)-\alpha\cdot x)},$$

where $\alpha \cdot x$ is the inner product $\alpha_1 x_1 + \cdots + \alpha_n x_n$ of $\alpha = (\alpha_1, \ldots, \alpha_n)$ and $x = (x_1, \ldots, x_n)$.

Let $0 \leq m$ be an integer. We say that f is m-plateaued if

$$|\hat{f}(\alpha)| \in \{0, p^{\frac{n+m}{2}}\}$$

for all $\alpha \in \mathbb{F}_{p^n}$. Here $|\cdot|$ denotes the absolute value in complex numbers. Let $\mathrm{Supp}(\hat{f})$ denote the subset of $\mathbb{F}_{p^n}$ consisting of α such that $\hat{f}(\alpha) \neq 0$. The following facts (definitions) are well known (see, for example, [4], [12])

- f is bent if and only if f is 0-plateaued.
- If f is m-plateaued, then $|\mathrm{Supp}(\hat{f})| = p^{n-m}$.

It seems we have rather limited knowledge in construction of plateaued functions over arbitrary finite field (see, for example, [3], [7]). A direct, but still very powerful construction of a strict subclass of plateaued functions is for the class of partially bent functions [2]. If $f : \mathbb{F}_{p^s} \to \mathbb{F}_p$ is a bent function, then for any integer $m \geq 1$, the function

$$\begin{array}{rcl} g : \mathbb{F}_{p^s} \times \mathbb{F}_{p^m} & \to & \mathbb{F}_p \\ (x, y) & \mapsto & f(x) \end{array}$$

is a partially bent function and m-plateaued function in $m+s$ many variables over $\mathbb{F}_p$. Moreover, given any affine space U_1 of dimension s in $\mathbb{F}_q^{m+s}$, it is easy to modify g to g_1 such that $\mathrm{Supp}(\hat{g}_1)$ is U_1.

Bent functions and plateaued functions are central objects for a variety of topics related to cryptography, coding theory and combinatorics. We refer, for example, to [4], [11], [12] and the references therein for further information.

It is an interesting open problem to count bent functions, even for rather moderate values of n (see, [10], [13]). Hence the asymptotic number of bent functions is a natural and actually difficult problem to consider (see [13] and the references therein).

Let $\mathcal{M}^\sharp(p,n)$ denote the family of completed Maiorana-McFarland bent functions in n variables over $\mathbb{F}_p$. Note that n is even if $p=2$.

The following are well known (see, for example, [4], [12] and [13]):

- Case n is even:
$$\ln\left|\mathcal{M}^\sharp(p,n)\right| = \frac{n}{2}p^{n/2}\ln(p)\,(1+o(1)) \tag{1}$$
as $n\to\infty$ and n is even.
- Case n is odd:
$$\ln\left|\mathcal{M}^\sharp(p,n)\right| = \frac{n-1}{2}p^{(n-1)/2}\ln(p)\,(1+o(1)) \tag{2}$$
as $n\to\infty$ and n is odd.

Here and throughout the paper $o(\cdot)$ stands for the small o notation as $n\to\infty$.

Let $\mathcal{B}(p,n)$ denote the family of bent functions in n variables over $\mathbb{F}_p$. Let $\mathcal{GMM}(p,n)$ denote the family of generalized Maiorana-McFarland bent functions in n variables over $\mathbb{F}_p$ (see [1] and [5]). Note that the notions of completed Maiorana-McFarland bent functions (see [4]) and generalized Maiorana-McFarland bent functions are different.

We have the obvious bound that
$$|\mathcal{B}(p,n)| \geq |\mathcal{GMM}(p,n)|. \tag{3}$$

In [13], the authors obtain that, if $p=2$, then
$$\ln\left(|\mathcal{GMM}(p,n)|\right) \geq \frac{3}{4}np^{n/2}\ln(p)\,(1+o(1)) \tag{4}$$
as $n\to\infty$ and n is even.

In particular they improve the lower bound in (1) so that the coefficient of the main term $np^{n/2}\ln(p)$ is increased from $\frac{1}{2}$ to $\frac{3}{4}$.

Combining (3) and (4) we obtain an asymptotic lower bound on the number of bent functions over $\mathbb{F}_2$, which is the best known asymptotic lower bound on the number of bent functions over $\mathbb{F}_2$.

The methods of [13] do not generalize to odd characteristic. In this paper we improve (2) and we obtain an asymptotic lower bounds on the number of bent functions in odd n variables over $\mathbb{F}_p$ as $n\to\infty$ and p is odd.

We construct two families of generalized bent functions using two different methods related to the results of [8] and [6], respectively.

Using results of [8] and further detailed techniques we prove our first main result in the following.

Theorem 0.1 *Let p be an odd prime. There exists a sequence of odd integers n (moreover $n\equiv 3 \mod 4$), $n\to\infty$ and a corresponding sequence of families $\mathcal{F}_1(n)$ of generalized Maiorana-McFarland bent functions in n variables over $\mathbb{F}_p$ satisfying*
$$\ln\left(|\mathcal{F}_1(n)|\right) \geq \frac{np^{n/2}}{\sqrt{p}}\left(1-\frac{1}{2(p^2-1)}\right)\ln(p)(1+o(1))$$
as $n\to\infty$.

We present a sketch of the proof of Theorem 0.1 in Section 2 below.

Remark 0.2 *In Theorem 0.1, we improve the lower bound in (2) by increasing the coefficient of the main term* $np^{n/2}\ln(p)$ *from* $\frac{1}{2\sqrt{p}}$ *to* $\frac{1}{\sqrt{p}}\left(1-\frac{1}{2(p^2-1)}\right)$. *Note that if* $p=3$, *then* $\frac{1}{\sqrt{p}}\left(1-\frac{1}{2(p^2-1)}\right)=\frac{1}{\sqrt{3}}\frac{15}{16}$. *This also gives an improved lower bound in the number of bent functions over* $\mathbb{F}_p$ *for odd number of variables* n *using (3) in the limit as* $n\to\infty$ *if* $p>3$.

Using results of [6] and further different detailed techniques we prove our second main result in the following.

Theorem 0.3 *Recall that* $\mathbb{F}_3$ *is the finite field with* 3 *elements. There exists a sequence of odd integers* $n\to\infty$ *and a corresponding sequence of families* $\mathcal{F}_2(n)$ *of generalized Maiorana-McFarland bent functions in* n *variables over* $\mathbb{F}_3$ *satisfying*

$$\ln\left(|\mathcal{F}_2(n)|\right)\geq\frac{n3^{n/2}}{\sqrt{3}}\ln(3)(1+o(1))$$

as $n\to\infty$.

We present a sketch of the proof of Theorem 0.3 in Section 3 below.

Remark 0.4 *In Theorem 0.3, we improve the lower bound in Theorem 0.1 (and hence the lower bound in (2)) by increasing the coefficient of the main term* $n3^{n/2}\ln(3)$ *from* $\frac{1}{\sqrt{3}}\frac{15}{16}$ *to* $\frac{1}{\sqrt{3}}$. *This also gives an improved lower bound in the number of bent functions over* $\mathbb{F}_3$ *for odd number of variables* n *using (3) in the limit as* $n\to\infty$.

1 Why do we use only partially bent functions?

In this section we explain why we only use partially bent functions and not arbitrary plateaued functions shortly. Let $s\geq 1$ be an integer. Let $n_1\geq 1$ be a variable integer which runs and tends infinity over a sequence. We construct bent functions with $2n_1+s$ many variables over $\mathbb{F}_p$. Hence our number of variables tends to infinity as n_1 tends to infinity.

Let $\mathcal{P}=(A_1,\ldots,A_{p^{n_1}})$ be an ordered partition of $\mathbb{F}_{p^{n_1+s}}$ into subsets of size exactly p^s. We will need a huge number of such partitions that we can control.

By control we mean the following. Given such $\mathcal{P}$, we need to design a corresponding ordered set of n_1-plateaued functions $(g_1,\ldots,g_{p^{n_1}})$ such that $g_i:\mathbb{F}_{p^{s+n_1}}\to\mathbb{F}_p$ and

$$\mathrm{Supp}(\hat{g_i})=A_i \tag{5}$$

for each $1\leq i\leq p^{n_1}$.

Let $\phi:\mathbb{F}_{p^{n_1}}\to\{1,2,\ldots,p^{n_1}\}$ be a fixed bijection. A generalized Maiorana-McFarland bent function in $(2n_1+s)$ variables over $\mathbb{F}_p$ is defined as (see [1], [5])

$$\begin{array}{rccc} f:\mathbb{F}_p^{s+n_1}\times\mathbb{F}_p^{n_1} & \to & \mathbb{F}_p \\ (y,z) & \mapsto & g_{\phi(z)}(y). \end{array}$$

If $(A_1,\ldots,A_{p^{n_1}})$ and $(B_1,\ldots,B_{p^{n_1}})$ are two distinct ordered partitions of $\mathbb{F}_{p^{n_1+s}}$ into subsets of size exactly p^s, i.e. $A_i\neq B_i$ for at least one i, then independent from the corresponding ordered set of n_1-plateaued functions (provided they exist), the constructed bent functions f_A and f_B in $(2n_1+s)$ variables are distinct. Moreover assume that we fix an ordered partition $(A_1,\ldots,A_{p^{n_1}})$ of $\mathbb{F}_{p^{n_1+s}}$ into subsets of size exactly p^s. Assume also that there are two corresponding ordered set of n_1-plateaued functions $(g_1,\ldots,g_{p^{n_1}})$ and $(h_1,\ldots,h_{p^{n_1}})$ such that $g_i,h_i:\mathbb{F}_{p^{s+n_1}}\to\mathbb{F}_p$ and

$$\mathrm{Supp}(\hat{g_i})=\mathrm{Supp}(\hat{h_i})=A_i \tag{6}$$

for each $1 \leq i \leq p^{n_1}$. Then if $g_i \neq h_i$ for some i, then the constructed bent functions f_g and f_h in $(2n_1 + s)$ variables are distinct.

An important problem is to have a large number of such partitions $\mathcal{P}$ that we make sure existence of a large number of corresponding ordered sequences of n_1-plateaued functions.

We know sufficiently large number of such partitions using affine subspaces of $\mathbb{F}_{p^{n_1+s}}$ of dimension s. This implies that we use only partially bent functions [2]. It is still not an easy problem to count even this particular subject as n_1 tends to infinity. We use methods from [8], [6] together with many new and further techniques to have a good asymptotic lower bound. It seems difficult to improve these asymptotic lower bounds making also use of non partially bent but plateaued functions.

2 Sketch of proof of Theorem 0.1

Let $s \geq 1$ be an integer. Let m be an integer such that $(s+1) \mid m$. Recall that a spread $\mathbb{S}$ of dimension $(s+1)$ in $\mathbb{F}_{p^m}$ is a collection of $(s+1)$-dimensional subspaces of $\mathbb{F}_{p^m}$ such that any one dimensional subspace of $\mathbb{F}_{p^m}$ lies in exactly one of the elements of $\mathbb{S}$. Note that $\mathbb{S}$ should have exactly $\frac{1+p+\cdots+p^{m-1}}{1+p+\cdots+p^s}$ many elements. As $m \to \infty$ and $(s+1) \mid m$, Keevash et al. [8] proved existence of $M_1(s,m)$ many spreads such that

$$\ln\left(M_1(s,m)\right) = p^{m-s-1}(m-1)s\ln(p)(1+o(1))$$

as $m \to \infty$.

Take $m = n_1 + s + 1$. Using an hyperplane restriction of these spreads and using also more techniques from perfect matchings we obtain that the number $M_2(s,n_1)$ of ordered partitions of $\mathbb{F}_{p^{n_1+s}}$ into s dimensional affine subspaces satisfies

$$\ln\left(M_2(s,n_1)\right) \geq \left(p^{n_1} - \delta(s)p^{n_1-s-1}\right)(n_1+s)s\ln(p)(1+o(1)) + p^{n_1}n_1\ln(p)(1+o(1)) \tag{7}$$

as $n_1 \to \infty$. Here $\delta(s) = \frac{p^{s+1}}{(p^{s+1}-1)}$.

Using generalized Maiorana-McFarland construction and (7) we obtain that the number $M_3(s,n_1)$ of bent functions in $(2n_1+s)$ variables gives

$$\ln(M_3(s,n_1)) \geq p^{n_1}\left(n_1 s + n_1 + s^2 - \frac{(n_1+s)s\delta(s)}{p^{s+1}}\right)\ln(p)(1+o(1))$$

as $n_1 \to \infty$. Putting $s = 1$ we complete the proof.

3 Sketch of proof of Theorem 0.3

Using results of Eberhald et al. [6] we obtain exact number of transversals of the Cayley table of $\mathbb{F}_3^n$. This implies that the number $M_4(m)$ of unordered partitions of $\mathbb{F}_{3^m}$ into 1-dimensional affine subspaces satisfies

$$\ln(M_4(m)) \geq 3^{m-1}m\ln(3) - 2\cdot 3^{m-1}\ln(3)(1+o(1)) \tag{8}$$

as $m \to \infty$. Take $m = n_1 + 1$. Using generalized Maiorana-McFarland construction and (8) we obtain that the number $M_5(n_1)$ of $(2n_1+1)$-variable bent functions over $\mathbb{F}_3$ satisfies

$$\ln(M_5(n_1)) \geq 3^{n_1}2n_1\ln(3)(1+o(1))$$

as $n_1 \to \infty$. This completes the proof.

References

[1] S. Agievich. Bent rectangles. *Boolean functions in cryptology and information security*, NATO Sci. Peace Secur. Ser. D Inf. Commun. Secur., vol. 18, pp. 3–22, Amsterdam, 2008.

[2] C. Carlet. Partially-bent functions. *Advances in cryptology'CRYPTO '92 (Santa Barbara, CA, 1992)*, 280-291, Lecture Notes in Comput. Sci., 740, Springer, Berlin, 1993.

[3] C. Carlet. Boolean and vectorial plateaued functions and APN functions. *IEEE Transactions on Information Theory*, vol. 61, no. 11. pp. 6272–6289, 2015.

[4] C. Carlet. *Boolean Functions for Cryptography and Coding Theory*, Cambridge University Press, Cambridge, 2021.

[5] A.Çesmelioğlu, W. Meidl and A. Pott. Generalized Maiorana-McFarland class and normality of p-ary bent functions. *Finite Fields and Their Applications*, vol. 24, pp. 105–117, 2013.

[6] S. Eberhard, F. Manners and R. Mrazovic. An asymptotic for the Hall-Paige conjecture. *Advances in Mathematics*, Part A, Paper No. 108423, 73 pp, 2022.

[7] S. Hodžić, E. Pasalic, Y. Wei, F. Zhang. Designing plateaued Boolean functions in Spectral Domain and Their Classification. *IEEE Transactions on Information Theory*, vol. 65, no. 9. pp. 5865–5879, 2019.

[8] P. Keevash, M. Sah and M. Sawhney. The existence of subspace designs. arXiv: 2212.00870, 61 pp, 2022.

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

[10] P. Langevin and G. Leander. Counting all bent functions in dimension eight 99270589265934370305785861242880. *Designs, Codes and Cryptography*, vol. 59, no. 1-3, pp. 193–205, 2011.

[11] W. Meidl. A survey on p-ary and generalized bent functions. *Cryptography and Communications*, vol. 14, pp. 737–782, 2022.

[12] S. Mesnager. *Bent Functions. Fundamentals and Results*, Springer International Publishing, 2016.

[13] V. N. Potapov, A. A. Taranenko and Yu. V. Tarannikov. An asymptotic lower bound on the number of bent functions. *Designs, Codes and Cryptography*, 2023. DOI: 10.1007/s10623-023-01239-z

[14] O. S. Rothaus "On 'bent' functions" J. Combinatorial Theory Ser. A vol. 20, no. 3, pp. 300–305, 1976.

Normality of Boolean bent functions in eight variables, revisited

Alexandr Polujan[1], Luca Mariot[2], and Stjepan Picek[3]

[1]Otto von Guericke University Magdeburg
Universitätsplatz 2, 39106, Magdeburg, Germany
alexandr.polujan@gmail.com

[2]Semantics, Cybersecurity and Services Group, University of Twente,
Drienerlolaan 5, 7511GG Enschede, The Netherlands
l.mariot@utwente.nl

[3]Digital Security Group, Radboud University
Postbus 9010, 6500 GL Nijmegen, The Netherlands
stjepan.picek@ru.nl

Abstract

There are approximately 2^{106} bent functions in 8 variables, and the known constructions cover only a tiny part of all these functions [9]. However, finding *"rare"* bent functions, i.e., those which do not arise from generic classes of functions or those of which examples are only a few known, is still a non-trivial problem. In this paper, we give for the first time an example of a non-normal partial spread bent function in 8 variables by analyzing the list of all partial spread bent functions [8], thus solving two open problems by Charpin [4, Open problem 5] and Leander [10, p.17], respectively. Additionally, we show that all partial spread bent functions in $n = 8$ variables are either normal or weakly normal. Finally, using evolutionary algorithms, we show that it is possible to construct bent functions which do not belong, up to equivalence, to the Maiorana-McFarland class.

Keywords: Boolean bent function, partial spread class, normality, evolutionary computation.

1 Preliminaries

A mapping $f\colon \mathbb{F}_2^n \to \mathbb{F}_2$ is called a *Boolean function*. For $a \in \mathbb{F}_2^n$, the *Walsh transform* $\hat{\chi}_F\colon \mathbb{F}_2^n \to \mathbb{Z}$ is defined by $\hat{\chi}_f(a) = \sum_{x\in\mathbb{F}_2^n}(-1)^{f(x)+a\cdot x}$, where $a \cdot x = a_1x_1 + \cdots + a_nx_n$. A Boolean function $f\colon \mathbb{F}_2^n \to \mathbb{F}_2$ is called *bent* if its Walsh transform satisfies $\hat{\chi}_f(a) = \pm 2^{n/2}$ for all $a \in \mathbb{F}_2^n$.

Definition 1.1. A Boolean function f on $\mathbb{F}_2^n$ is said to be *normal* if it is constant on some affine subspace U of $\mathbb{F}_2^n$ of dimension $\lceil n/2 \rceil$. In this case, f is said to be normal with respect to the affine space U. If no such an affine space exists, f is said to be *non-normal*.

To prove theoretically that a given bent function f on $\mathbb{F}_2^n$ is non-normal is a very challenging task. Nevertheless, for small values of n (i.e., $n \leq 8$), one can check the normality of a given bent function with the help of Algorithm 1.1. With a recursive algorithm suggested in [2, Algorithm 1], several examples of non-normal bent functions in $n = 10, 12, 14$ variables were obtained. For example, the restriction of the Kasami–Welch function $x \in \mathbb{F}_{2^{11}} \mapsto Tr(x^{241})$ to the trace 0 (and trace 1) elements is a non-normal bent function in $n = 10$ variables [11, Fact 14]. Note that $n = 10$ is the smallest number of variables for which such a bent function is known. Using the direct sum construction, one can construct new non-normal bent functions in an arbitrary number of variables from the known in the following way.

Result 1.2. *[10, p. 24] Let f be a Boolean bent function on $\mathbb{F}_2^n$ and g be a quadratic Boolean bent function on $\mathbb{F}_2^m$. Then $h(x, y) = f(x) + g(y)$ is normal on $\mathbb{F}_2^n \times \mathbb{F}_2^m$ iff f is normal on $\mathbb{F}_2^n$.*

Despite the progress on the normality of bent functions in $n \geq 10$ variables, the following two questions (the first is due Charpin [4, Open problem 5] and the second due Leander [10, p.17]) still remain not answered:

Algorithm 1.1. Checking normality (according to [4, Theorem 1]).

Require: Bent function $f\colon \mathbb{F}_2^n \to \mathbb{F}_2$.

1: **for all** subspaces V of dimension $n/2$ **do**

2: **Check** the following condition: f is constant on $b+V$ if and only if

$$(-1)^{b\cdot v}\hat{\chi}_f(v) = \varepsilon 2^k, \text{ for all } v \in V^\perp = \{u \in \mathbb{F}_2^n \colon u \cdot v = 0 \text{ for all } v \in V\},$$

where ε is constant, equal either to $+1$ or -1.

3: **Output** affine subspaces $b+V$, on which f is constant.

4: **end for**

1. Do non-normal bent functions of 8 variables and degree 4 exist?
2. Do non-normal bent functions in the $\mathcal{PS}^- \setminus \mathcal{PS}_{ap}$ class exist?

In the following section, we positively answer both of the mentioned questions by finding among all $\mathcal{PS}$ bent functions in $n = 8$ variables [8] a non-normal bent function in $\mathcal{PS}^- \setminus \mathcal{PS}_{ap}$.

2 A non-normal partial spread bent function in eight variables

First, we give a definition of a partial spread and define its canonical representation.

Definition 2.1. A partial spread of order s in $\mathbb{F}_2^n$ with $n = 2k$ is a set of s vector subspaces $U_1, \ldots, U_s$ of $\mathbb{F}_2^n$ of dimension k each, such that $U_i \cap U_j = \{0\}$ for all $i \neq j$. The partial spread of order $s = 2^k + 1$ in $\mathbb{F}_2^n$ with $n = 2k$ is called a spread.

Following the notation in [8], for two matrices $A, B \in \mathbb{F}_2^{(k,k)}$ s.t. $\operatorname{rank}[A\ B] = k$, we denote by $[A : B]$ the linear span of the rows of $[A\ B]$. Let 0_k and I_k denote the all-zero and all-one matrix of order k, respectively. Any partial spread of order s is equivalent to one of the form

$$\mathcal{S} = \{\underbrace{[0_k : I_k]}_{U_1}, \underbrace{[I_k : 0_k]}_{U_2}, \underbrace{[I_k : I_k]}_{U_3}, \underbrace{[I_k : A_4]}_{U_4}, \ldots, \underbrace{[I_k : A_s]}_{U_s}\}, \tag{2.1}$$

where $A_2(= 0_k), A_3(= I_k), A_4, \ldots, A_s$ have the property that $A_i - A_j$ is invertible for all $2 \leq i < j \leq s$. In the following, we denote by $\mathbb{1}_U : \mathbb{F}_2^n \to \mathbb{F}_2$ the *indicator function* of $U \subseteq \mathbb{F}_2^n$, i.e., $\mathbb{1}_U(x) = 1$ if $x \in U$, and 0 otherwise. Let the vector spaces $U_1, \ldots, U_{2^{k-1}+1}$ of $\mathbb{F}_2^n$ form a partial spread in $\mathbb{F}_2^n$. The *partial spread class* $\mathcal{PS}$ of bent functions on $\mathbb{F}_2^n$ is the union of the following two classes [5]: the $\mathcal{PS}^+$ *class* is the set of Boolean bent functions of the form $f(x) = \sum_{i=1}^{2^{k-1}+1} \mathbb{1}_{U_i}(x)$; the $\mathcal{PS}^-$ *class* is the set of Boolean bent functions of the form $f(x) = \sum_{i=1}^{2^{k-1}} \mathbb{1}_{U_i^*}(x)$, where $U_i^* := U_i \setminus \{0\}$. The *Desarguesian partial spread* class $\mathcal{PS}_{ap} \subset \mathcal{PS}^-$ is the set of Boolean bent functions f on $\mathbb{F}_{2^k} \times \mathbb{F}_{2^k}$ of the form $f\colon (x, y) \in \mathbb{F}_{2^k} \times \mathbb{F}_{2^k} \mapsto h\,(x/y)$, where $\frac{x}{0} = 0$, for all $x \in \mathbb{F}_{2^k}$ and $h\colon \mathbb{F}_{2^k} \to \mathbb{F}_2$ is a balanced Boolean function with $h(0) = 0$.

Clearly, every $\mathcal{PS}^+$ bent function f on $\mathbb{F}_2^n$ is normal, since $f|_{U_i} = 1$ for every spread line U_i. Moreover, all functions in $\mathcal{PS}_{ap}$ class are normal, since they vanish on the k-dimensional subspace $\{0\} \times \mathbb{F}_{2^k}$. However, the question about the normality of bent functions in $\mathcal{PS}^- \setminus \mathcal{PS}_{ap}$, becomes non-trivial since, in this case, one deals with the sets U_i^*, which are not vector subspaces anymore.

Partial spreads on $\mathbb{F}_2^8$ were completely classified in [8]; the representatives of the corresponding bent functions are available (at the moment of submission of this article) at [7]. Remarkably, there exist 9,316 partial spreads of order 8 on $\mathbb{F}_2^8$, and each of them gives rise to a partial spread bent function in the $\mathcal{PS}^-$ class. Now, we give an example of such a bent function, which is non-normal.

Example 2.2. Let $n = 2k = 8$. Let us define invertible $k \times k$-matrices $A_4, \dots, A_8$, which, in turn, define the partial spread $\mathcal{S}$ of order $s = 8$, given by its canonical representation (2.1):

$$A_4 = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}, A_5 = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{pmatrix}, A_6 = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}, A_7 = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{pmatrix}, A_8 = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}.$$

The corresponding bent function $f(x) = \sum_{i=1}^{2^{k-1}} \mathbb{1}_{U_i^*}(x)$ is in the $\mathcal{PS}^- \setminus \mathcal{PS}_{ap}$ class (it is the function psf=970 in [7, psf-8.txt]). The ANF of this function is given by:

$$\begin{aligned} f(x) = {} & x_1 + x_2 + x_1x_2 + x_3 + x_1x_3 + x_2x_3 + x_1x_2x_3 + x_4 + x_1x_4 + x_2x_4 + x_1x_2x_4 + x_3x_4 \\ & + x_1x_3x_4 + x_2x_3x_4 + x_1x_2x_3x_4 + x_5 + x_1x_5 + x_1x_2x_5 + x_1x_3x_5 + x_2x_3x_5 + x_4x_5 + x_1x_4x_5 \\ & + x_2x_4x_5 + x_1x_2x_4x_5 + x_2x_3x_4x_5 + x_6 + x_1x_6 + x_2x_6 + x_3x_6 + x_1x_3x_6 + x_2x_3x_6 \\ & + x_1x_2x_3x_6 + x_1x_4x_6 + x_1x_2x_4x_6 + x_3x_4x_6 + x_1x_3x_4x_6 + x_5x_6 + x_2x_5x_6 + x_3x_5x_6 \\ & + x_2x_3x_5x_6 + x_4x_5x_6 + x_7 + x_2x_7 + x_1x_2x_7 + x_3x_7 + x_2x_3x_7 + x_2x_4x_7 + x_1x_2x_4x_7 \\ & + x_1x_3x_4x_7 + x_2x_3x_4x_7 + x_5x_7 + x_2x_5x_7 + x_1x_2x_5x_7 + x_3x_5x_7 + x_1x_3x_5x_7 + x_4x_5x_7 \\ & + x_1x_4x_5x_7 + x_2x_4x_5x_7 + x_6x_7 + x_1x_6x_7 + x_2x_6x_7 + x_3x_6x_7 + x_2x_3x_6x_7 + x_1x_4x_6x_7 \\ & + x_5x_6x_7 + x_1x_5x_6x_7 + x_2x_5x_6x_7 + x_4x_5x_6x_7 + x_8 + x_1x_8 + x_1x_2x_8 + x_4x_8 + x_1x_4x_8 \\ & + x_2x_4x_8 + x_3x_4x_8 + x_1x_3x_4x_8 + x_2x_3x_4x_8 + x_5x_8 + x_1x_2x_5x_8 + x_4x_5x_8 + x_2x_4x_5x_8 \\ & + x_6x_8 + x_1x_6x_8 + x_2x_6x_8 + x_1x_3x_6x_8 + x_4x_6x_8 + x_5x_6x_8 + x_1x_5x_6x_8 + x_4x_5x_6x_8 \\ & + x_7x_8 + x_1x_7x_8 + x_2x_7x_8 + x_1x_2x_7x_8 + x_3x_7x_8 + x_2x_3x_7x_8 + x_4x_7x_8 + x_5x_7x_8 \\ & + x_1x_5x_7x_8 + x_3x_5x_7x_8 + x_6x_7x_8 + x_1x_6x_7x_8 + x_3x_6x_7x_8 + x_5x_6x_7x_8. \end{aligned}$$

Using Algorithm 1.1, one can check that this function is non-normal. With this example, we give positive answers to both mentioned questions and also make the following conclusion (we give a short proof for completeness).

Corollary 2.3. *Let f be a non-normal bent function on $\mathbb{F}_2^n$. Then, $n \geq 8$.*

Proof. Since f is bent on $\mathbb{F}_2^n$ and $n \leq 6$, we have that f is either quadratic or cubic (the latter is only possible for $n = 6$). Every quadratic bent function f on $\mathbb{F}_2^n$ is normal, see [4, Theorem A.1]. For $n = 6$, every cubic bent function is equivalent, up to a nonsingular affine transformation on the variables, to the function $g(x, y) = g(x_1, x_2, x_3, y_1, y_2, y_3) = x \cdot \pi(y) + x_1x_2x_3$, where π is a permutation of $\mathbb{F}_2^3$, see [3, Proposition 4]. Clearly, $g|_V = 0$ for a vector space $V = \{0\} \times \mathbb{F}_2^3$, and hence $f|_{b+V'} = 0$ for some affine space $b + V'$. With Example 2.2 and Result 1.2, we conclude that non-normal bent functions exist on $\mathbb{F}_2^n$ for all even $n \geq 8$. □

Surprisingly, the function $f(x)$ in Example 2.2 is the only non-normal bent function from the list of all partial spread bent functions [7]. This function is, however, *weakly normal*, i.e., $f + l$ is normal for a non-zero linear function l on $\mathbb{F}_2^8$. Indeed, it is possible to verify that for a linear function $l(x) = x_8$ and an affine subspace $V = (0,1,0,1,0,0,0,0) + \langle (0,0,1,1,0,0,0,1), (0,0,0,0,1,0,0,1), (0,0,0,0,0,1,0,0), (0,0,0,0,0,0,1,1) \rangle$ the following holds $(f + l)|_V = 1$. With this observation, we make the following conclusion.

Result 2.4. *All $\mathcal{PS}$ bent functions in $n = 8$ variables are either normal or weakly normal.*

3 Computational construction methods of bent functions

Aimed to generate more non-normal bent functions and to find the first examples of non-weakly normal bent functions in 8 variables, we use two computational approaches for the generation of large sets of bent functions, based on the *cellular automata (CA)* and the *genetic programming (GP)*. In the following, we briefly discuss the used approaches and bent functions obtained with their help.

3.1 Generating bent functions with CA and GP

Cellular Automata (CA) can be seen as a particular kind of discrete dynamical system equipped with a shift-invariant update function that acts over a regular lattice of cells. When the state set of the cells is a finite field, and the local rule is linear, a cellular automaton can be interpreted as a linear recurring sequence (LRS). The authors of [6] studied families of LRS of order d, whose feedback polynomials are pairwise coprime. In this way, it is possible to define a partial spread by considering the projection of the LRS onto their first $2d$ coordinates. Such families exist only when the degree of the feedback polynomials is either 1 or 2. The former case corresponds to the Desarguesian spread. For degree 2, the authors of [6] found 273 $\mathcal{PS}^-$ functions of 8 variables, most of which are inequivalent to Maiorana-McFarland and Desarguesian spread-based functions. Therefore, they seem to be good candidates to test for non-normality.

Genetic Programming (GP) is an optimization algorithm loosely inspired by the principles of biological evolution. The underlying idea is to encode a Boolean function $f\colon \mathbb{F}_2^n \to \mathbb{F}_2$ as a syntactic tree where the leaves represent the input variables, the internal nodes are Boolean operators (such as AND, XOR, NOT, etc.) acting on the inputs received from their children, and the root node gives the output of the function. Therefore, one can define the truth table of the function by evaluating the circuit encoded by the tree over all 2^n input combinations. The GP algorithm randomly initializes a population of trees encoding n-variables Boolean functions, then evaluates their *fitness*, which measures the optimization criterion to optimize. In our case, the fitness function is defined as the nonlinearity of the functions to be maximized. Then, the GP algorithm iteratively evolves the population by applying mutation and crossover operators, which give a new population to be evaluated against the fitness function. The fittest individuals are then carried over to the next iteration. For our problem, we employed the GP algorithm proposed in [12], adopting the same experimental settings and parameters. In particular, the GP algorithm performed 10 000 optimization runs, where in each run, a population of 50 trees encoding Boolean functions of 8 variables is evolved for 500 000 iterations.

3.2 Analysis of generated bent functions

CA. Aimed to analyze whether it is possible to generate non-normal partial spread bent functions using CA, we revised all 273 $\mathcal{PS}^-$ bent functions generated with this approach in [6]. It turned out that all partial spread bent functions constructed with this approach are normal.

Genetic Programming. With this approach, we were able to generate 7,478 different bent functions. Among them, there are 4 690 quadratic, 2 367 cubic, and 421 of degree 4. Since all quadratic bent functions and all cubic functions in 8 variables are normal, it is enough to analyze only bent functions of degree 4. We note that all the generated bent functions of degree 4 turned out to be normal as well, which was reasonable to expect since most of them have only a few monomials of degree 4. For this reason, and due to the fact that the majority of generated bent functions are quadratic and cubic (and hence are equivalent to the Maiorana-McFarland class), it was essential to check whether these bent functions of degree 4 are equivalent to the Maiorana-McFarland class. Among 421 functions of degree 4, we identified a function inequivalent to a member of the Maiorana-McFarland class (this fact was checked with the corresponding algorithms described in [1, 13]). The ANF of this function is given by

$$\begin{aligned} g(x) = {} & 1 + x_2 + x_5 + x_6 + x_8 + x_1x_5 + x_1x_7 + x_1x_8 + x_2x_6 + x_2x_7 + x_3x_8 + x_4x_7 \\ & + x_2x_5x_8 + x_1x_3x_6x_7 + x_2x_5x_7x_8. \end{aligned} \tag{3.1}$$

Again, with Algorithm 1.1, one can check that the function g given in (3.1) is normal, since $g|_V = 0$ for the affine subspace $V = (1,1,0,0,0,0,0,0) + \langle (0,0,1,0,0,0,0,0), (0,0,0,1,0,0,0,0), (0,0,0,0,1,0,0,0), (0,0,0,0,0,1,0,0)\rangle$ of dimension 4.

4 Conclusion and open problems

In this paper, we completely analyzed all partial spread bent functions in $n = 8$ variables with respect to normality, thus providing the first example of a non-normal bent function in $n = 8$ variables. The next essential step is to find (if possible) the examples of non-weakly normal bent functions on $\mathbb{F}_2^8$, as well as non-weakly normal bent functions in the $\mathcal{PS}^- \setminus \mathcal{PS}_{ap}$ class; the latter question was essentially asked by Leander in [10, p.17].

Aimed to generate more non-normal bent functions and even to find non-weakly normal ones, we used evolutionary algorithms to construct such functions. Being unable to find such examples (mostly due to the reason that we evolved only non-linearity), we still, however, were able to find bent functions, which, up to equivalence, do not belong to the Maiorana-McFarland class. This finding indicates, that using suitably chosen evolutionary algorithms (e.g., by additionally minimizing the number of flats on which a bent function is affine), it might be possible to construct "rare" bent functions.

Finally, we want to underline that future research on generating new bent functions should be focused on the construction of algorithms 1) generating bent functions outside the known classes with a high probability, 2) generating non-normal bent functions, and 3) generating non-weakly normal bent functions. We believe that based on the analysis of big sets of bent functions not coming from the known analytic constructions, it should be possible to develop generic theoretical construction methods of new families of bent functions.

References

[1] Bapić, A., Pasalic, E., Polujan, A., Pott, A.: Vectorial Boolean functions with the maximum number of bent components beyond the Nyberg's bound. Designs, Codes and Cryptography (2023). p. 4.

[2] Canteaut, A., Daum, M., Dobbertin, H., Leander, G.: Finding nonnormal bent functions. Discrete Applied Mathematics **154**(2), 202–218 (2006). p. 1.

[3] Carlet, C.: Two new classes of bent functions. In: T. Helleseth (ed.) Advances in Cryptology — EUROCRYPT '93, pp. 77–101. Springer Berlin Heidelberg, Berlin, Heidelberg (1994). p. 3.

[4] Charpin, P.: Normal Boolean functions. J. Complexity **20**(2-3), 245–265 (2004). pp. 1, 2, and 3.

[5] Dillon, J.F.: Elementary Hadamard difference sets. Ph.D. thesis, University of Maryland (1974). p. 2.

[6] Gadouleau, M., Mariot, L., Picek, S.: Bent functions in the partial spread class generated by linear recurring sequences. Designs, Codes and Cryptography **91**(1), 63–82 (2023). p. 4.

[7] Langevin, P.: Classification of partial spread functions in eight variables. Philippe Langevin's numerical project page (2010). pp. 2 and 3.

[8] Langevin, P., Hou, X.D.: Counting partial spread functions in eight variables. IEEE Transactions on Information Theory **57**, 2263–2269 (2011). pp. 1 and 2.

[9] Langevin, P., Leander, G.: Counting all bent functions in dimension eight 99270589265934370305785861242880. Designs, Codes and Cryptography **59**(1), 193–205 (2011). p. 1.

[10] Leander, G.: Normality of bent functions. Monomial- and binomial-bent functions. Ph.D thesis, Ruhr-Universität Bochum, Universitätsbibliothek (2005) pp. 1 and 5.

[11] Leander, G., McGuire, G.: Construction of bent functions from near-bent functions. Journal of Combinatorial Theory, Series A **116**(4), 960–970 (2009) p. 1.

[12] Picek, S., Jakobovic, D., Miller, J.F., Batina, L. Cupic, M.: Cryptographic Boolean functions: One output, many design criteria. Appl. Soft Comput. 40: 635-653 (2016) p. 4.

[13] Polujan, A., Pott, A.: Cubic bent functions outside the completed Maiorana-McFarland class. Designs, Codes and Cryptography **88**(9), 1701–1722 (2020). p. 4.

$\mathcal{S}_0$-equivalent classes, a new direction to find better weightwise perfectly balanced functions, and more

Agnese Gini[0009−0001−9565−380X], Pierrick Méaux[0000−0001−5733−4341]

University of Luxembourg, Luxembourg
agnese.gini@uni.lu, pierrick.meaux@uni.lu

Abstract. We investigate the concept of $\mathcal{S}_0$ equivalent class, n-variable Boolean functions up to the addition of a symmetric function null in 0_n and 1_n, as a tool to study weightwise perfectly balanced functions. On the one hand we show that weightwise properties, such as being weightwise perfectly balanced, the weightwise nonlinearity and weightwise algebraic immunity, are invariants of these classes. On the other hand we analyze the variation of global parameters inside the same class, showing for example that there is always a function with high degree, algebraic immunity, or nonlinearity in the $\mathcal{S}_0$ equivalent class of a function. Finally, we discuss how these results extend to other equivalence relations and their applications in cryptography.

1 Introduction

Weightwise Perfectly Balanced (WPB) functions have been introduced by Carlet *et al.* in [CMR17] while studying the cryptographic properties of Boolean functions when the input is restricted to a subset of $\mathbb{F}_2^n$, motivated by the analysis of FLIP stream cipher [MJSC16]. These objects are the functions $f\colon \mathbb{F}_2^n \to \mathbb{F}_2$, such that $|\{x \in \mathsf{E}_{k,n} \mid f(x) = 0\}| = |\{x \in \mathsf{E}_{k,n} \mid f(x) = 1\}|$ for each $1 \leq k \leq n-1$ where the slice $\mathsf{E}_{k,n}$ denotes the set of $\mathbb{F}_2^n$ with all vectors of Hamming weight k, f globally balanced, and $f(0_n) = 0$. Since then, several articles studied the properties on restricted sets, and multiple articles focused on WPB functions such as [LM19, TL19, LS20, MS21, ZS21, MSL21, GS22, ZS22, MPJ+22, GM22a, GM22b, MKCL22, MSLZ22, GM22c, ZJZQ23, ZLC+23, GM23].

In this paper we study their parameters relatively to the concept of $\mathcal{S}_0$ equivalent class, which considers two n-variable Boolean functions being in the same class if they are equal up to the addition of a symmetric function null in 0_n and 1_n. The interest for WPB functions is that being WPB is an invariant of $\mathcal{S}_0$-classes. Hence, by stabilizing the WPB functions, the notion of $\mathcal{S}_0$-equivalence gives a new direction to find WPB functions.

Since for every practical application it is crucial to have a WPB function with both good weightwise and global parameters, this work aims to suggest a new strategy to construct a WPB function satisfying this assumption. Indeed, the results of this article imply that in order to find such a function, we can first search for one with suitable weightwise properties and later improve the global properties by looking directly inside its $\mathcal{S}_0$-class.

Indeed, in this paper we show that the weightwise parameters such as weightwise nonlinearity and weightwise algebraic immunity stay unchanged inside the $\mathcal{S}_0$-class. Then, we investigate the variation of the global parameters such as the degree, algebraic immunity and nonlinearity, inside an $\mathcal{S}_0$-class and we prove bounds on the maximal parameters in all classes. We demonstrate, for example, that from WPB functions with algebraic immunity as low as 2 (*e.g.* , in [GM23]), we can find a function with algebraic immunity at least $t+1$ in its $\mathcal{S}_0$-class provided $\log_2(n) \geq \log_2(2t+1)+t+2$; while, for those whose nonlinearity is as low as $2^{n/2-1}$ (as exhibited in [GM22c]), we can find a function with nonlinearity at least $2^{n-2} - 2^{\frac{n}{2}-2}$ in its $\mathcal{S}_0$-class. We show that in every class we can find a function with degree $n-1$.

Using this framework are also able to prove that for every degree between $n/2$ and $n-1$ we can exhibit a WPB function with such a degree. Finally, we discuss how these results can be extended to other equivalence relations defined up to the addition of functions from of family $\mathcal{T}$. In different context of cryptography where a family $\mathcal{T}$ is easy to compute, and the addition is cheap, finding a Boolean function with good cryptographic parameters could then be reduced to finding the best function inside its $\mathcal{T}$-class.

We complement our investigation performing experimental analyses on equivalence classes for WPB functions in a small number of variables. Specifically, we are able to provide an exhaustive taxonomy of 4-variables classes. For 8-variables we selected some function from know families, *e.g.* [CMR17, LM19, TL19, GM22c, GM23], and computed statistics over the properties in their classes. The result of these experiments is provided in the full version of the paper.

2 Some preliminaries

A *Boolean function* f in n variables is a function from $\mathbb{F}_2^n$ to $\mathbb{F}_2$. We recall here general concepts on Boolean functions and their weightwise properties, we refer to *e.g.* [Car21] and to [CMR17] respectively for further details. The set of all Boolean functions in n variables is denoted by $\mathcal{B}_n$, and we denote $\mathcal{B}_n^*$ the set without the null function. We call *Algebraic Normal Form* of a Boolean n-variable polynomial representation over $\mathbb{F}_2$ (*i.e.* in $\mathbb{F}_2[x_1,\dots,x_n]/(x_1^2+x_1,\dots,x_n^2+x_n)$): $f(x_1,\dots,x_n)=\sum_{I\subseteq[1,n]} a_I \left(\prod_{i\in I} x_i\right)$ where $a_I\in\mathbb{F}_2$. The *(algebraic) degree* of f, denoted $\mathsf{deg}(f)$ is $\mathsf{deg}(f)=\max_{I\subseteq[1,n]}\{|I|\,|\,a_I=1\}$ if f is not null, 0 otherwise.

To denote when a property or a definition is restricted to a slice we use the subscript k. For example, for a n-variable Boolean function f we denote its support $\mathsf{supp}(f)=\{x\in\mathbb{F}_2^n\,|\,f(x)=1\}$ and we denote $\mathsf{supp}_k(f)$ its support restricted to a slice, that is $\mathsf{supp}(f)\cap\mathsf{E}_{k,n}$.

A Boolean function $f\in\mathcal{B}_n$ is called *balanced* if $|\mathsf{supp}(f)|=2^{n-1}=|\mathsf{supp}(f+1)|$. For $k\in[0,n]$ the function is said *balanced on the slice* k if $||\mathsf{supp}_k(f)|-|\mathsf{supp}_k(f+1)||\leq 1$. In particular when $|\mathsf{E}_{k,n}|$ is even $|\mathsf{supp}_k(f)|=|\mathsf{supp}_k(f+1)|=|\mathsf{E}_{k,n}|/2$.

Let $m\in\mathbb{N}^*$ and $n=2^m$, f is called *weightwise perfectly balanced* (WPB) if, for every $k\in[1,n-1]$, f is balanced on the slice k, that is $\forall k\in[1,n-1], |\mathsf{supp}_k(f)|=\binom{n}{k}/2$, and $f(0_n)=0$ and $f(1_n)=1$. The set of WPB functions in 2^m variables is denoted $\mathcal{WPB}_m$. When n is not a power of 2, other weights than $k=0$ and n give slices of odd cardinality, in this case we call $f\in\mathcal{B}_n$ *weightwise almost perfectly balanced* (WAPB) if $|\mathsf{supp}_k(f)|$ is either $|\mathsf{E}_{k,n}|/2$ if $|\mathsf{E}_{k,n}|$ is even, or $(|\mathsf{E}_{k,n}|\pm 1)/2$, otherwise. The set of WAPB functions in n variables is denoted $\mathcal{WAPB}_n$. The first WAPB family of function has been exhibited in [CMR17, Proposition 5] and it is usually referred as CMR functions.

The *nonlinearity* $\mathsf{NL}(f)$ of $f\in\mathcal{B}_n$ is the minimum Hamming distance between f and all the affine functions in $\mathcal{B}_n$, *i.e.* $\mathsf{NL}(f)=\min_{g,\,\mathsf{deg}(g)\leq 1}\{\mathsf{d_H}(f,g)\}$. For $k\in[0,n]$ we denote NL_k the *nonlinearity on the slice* k, the minimum Hamming distance between f restricted to $\mathsf{E}_{k,n}$ and the restrictions to $\mathsf{E}_{k,n}$ of affine functions over $\mathbb{F}_2^n$, *i.e.* $\mathsf{NL}_k(f)=\min_{g,\,\mathsf{deg}(g)\leq 1}|\mathsf{supp}_k(f+g)|$.

The *algebraic immunity* (AI) of a Boolean function $f\in\mathcal{B}_n$, denoted as $\mathsf{AI}(f)$, is defined as: $\mathsf{AI}(f)=\min_{g\neq 0}\{\mathsf{deg}(g)\mid fg=0 \text{ or } (f+1)g=0\}$. The function g is called an *annihilator* of f (or $f+1$). The *weightwise algebraic immunity* on the slice $\mathsf{E}_{k,n}$, denoted by $\mathsf{AI}_k(f)$, is defined as: $\min\{\mathsf{deg}(g)\mid fg=0 \text{ or } (f+1)g=0 \text{ over } \mathsf{E}_{k,n}\}$ where g is non null on $\mathsf{E}_{k,n}$.

The n-variable Boolean symmetric functions are those that are constant on each slice $\mathsf{E}_{k,n}$ for $k\in[0,n]$. The set of n-variable symmetric functions is denoted $\mathcal{SYM}_n$. Let $i\in[0,n]$, the *elementary symmetric function* of degree i in n variables, denoted $\sigma_{i,n}$, is the function which ANF contains all monomials of degree i and no monomials of other degrees; while, the *indicator functions* of the slice of weight k is the such that $\forall x\in\mathbb{F}_2^n$, $\varphi_{k,n}(x)=1$ if and only if $\mathsf{w_H}(x)=k$.

3 The $\mathcal{S}_0$-equivalence relation

For a fixed $n=2^m$ we consider the set of symmetric functions null in 0_n and 1_n:

$$\mathcal{S}_0=\{\sigma\in\mathcal{SYM}_n\colon \sigma(0_n)=\sigma(1_n)=0\}\,,$$

and the sets of Boolean functions in $\mathcal{B}_n$ up to addition of an element of $\mathcal{S}_0$:

Definition 1 **($\mathcal{S}_0$-equivalent functions).** *Let $m\in\mathbb{N}^*$ and $f,g\in\mathcal{B}_n$ Boolean functions in $n=2^m$ variables. f,g are called $\mathcal{S}_0$-equivalent if there exists a symmetric function $\sigma\in\mathcal{S}_0$ such that $f=g+\sigma$. We call $\mathcal{S}_0$-class of f the set of functions $\mathcal{S}_0$-equivalent to f and we denote it by $\mathcal{S}_0(f)$.*

Remark 1. Being $\mathcal{S}_0$-equivalent is an equivalence relation.

Lemma 1. *Let $m\in\mathbb{N}^*$ and $n=2^m$,*

1. *$\mathcal{S}_0$ is a $\mathbb{F}_2$-vector space of dimension $n-1$. In particular, $\mathcal{S}_0=\langle\varphi_{k,n}\colon k\in[1,n-1]\rangle_{\mathbb{F}_2}$ where we denote by $\varphi_{k,n}$'s the slice indicator functions.*

2. *For all* $f \in \mathcal{B}_n$, $\mathcal{S}_0(f) = f + \mathcal{S}_0$ *and* $|\mathcal{S}_0(f)| = 2^{n-1}$.
3. $\mathcal{S}_0 = \langle \sigma_{d,n} : d \in [1, n-1] \rangle_{\mathbb{F}_2}$ *where we denote by* $\sigma_{d,n}$*'s the elementary symmetric functions.*

Both $\mathcal{S}_0$-classes of weightwise almost perfectly balanced functions and weightwise perfectly balanced functions consist of functions having the same W(A)PB property.

Proposition 1. *Let* $m \in \mathbb{N}^*$ *and* $n = 2^m$,

1. *For all* $f \in \mathcal{WAPB}_n$, $\mathcal{S}_0(f) \subseteq \mathcal{WAPB}_n$.
2. *For all* $f \in \mathcal{WPB}_m$, $\mathcal{S}_0(f) \subseteq \mathcal{WPB}_m$.
3. *Let* $v = (v_1, \ldots, v_{n-1})$ *be a tuple such that* $\forall k \in [1, n-1]$, $v_k \in \mathsf{E}_{k,n}$. *For any* $f \in \mathcal{B}_n$, *there exists a unique* $g_v \in \mathcal{S}_0(f)$ *such that for all* $k \in [1, n-1]$, $g_v(v_k) = 1$. *We call* g_v *the canonical representative of its class respectively to* v.

As a consequence of Proposition 1 we obtain that $\mathcal{S}_0$-classes form a partition of $\mathcal{WAPB}_n$ and $\mathcal{WPB}_m$ and that for every tuple v we can represent the partition using canonical representatives. We prove that $\mathcal{S}_0$-equivalent classes have invariant restricted weightwise nonlinearity and restricted algebraic immunity:

Theorem 1. *Let* $m \in \mathbb{N}^*$, $n = 2^m$ *and* $f, g \in \mathcal{B}_n$ $\mathcal{S}_0$*-equivalent functions. For every* $k \in [0, n]$ *it holds* $\mathsf{NL}_k(f) = \mathsf{NL}_k(g)$.

Theorem 2. *Let* $m \in \mathbb{N}^*$, $n = 2^m$ *and* $f, g \in \mathcal{WPB}_m$ $\mathcal{S}_0$*-equivalent functions. For every* $k \in [0, n]$ *it holds* $\mathsf{AI}_k(f) = \mathsf{AI}_k(g)$.

While functions in the same $\mathcal{S}_0$-class have the same restricted weightwise nonlinearities and restricted algebraic immunities, they do not necessarily share the global properties such as the degree, nonlinearity and algebraic immunity. Working with $\mathcal{S}_0$-classes provides us a different principle for the construction of new functions. In fact, suppose we have a WPB function h with certain NL_k 's and AI_k 's and we are interested in increasing, for instance, its algebraic immunity, we can start our search for a new function inside $\mathcal{S}_0(h)$. Additionally, if h is a WPB function, we are guaranteed to obtain a function that is also WPB.

In the rest of this article we study the behavior of degree, nonlinearity and algebraic immunity inside $\mathcal{S}_0$-classes. Specifically, we are interested in the following edge quantities for WPB functions that characterize the best guaranteed value, for degree, algebraic immunity and nonlinearity, achievable by modifying a function in $\mathcal{WPB}_m$, while staying within its $\mathcal{S}_0$-class:

Definition 2. *Let* $m \in \mathbb{N}^*$ *and* $n = 2^m$, *we define:*

$$\mathsf{mdeg}\mathcal{S}_0(m) = \min_{f \in \mathcal{WPB}_m} \max_{g \in \mathcal{S}_0(f)} \mathsf{deg}(g),$$

$$\mathsf{mAI}\mathcal{S}_0(m) = \min_{f \in \mathcal{WPB}_m} \max_{g \in \mathcal{S}_0(f)} \mathsf{AI}(g),$$

$$\mathsf{mNL}\mathcal{S}_0(m) = \min_{f \in \mathcal{WPB}_m} \max_{g \in \mathcal{S}_0(f)} \mathsf{NL}(g).$$

4 Degree in $\mathcal{S}_0$-classes

In this part we study the potential algebraic degree inside $\mathcal{S}_0$-classes. We prove that we can preview the behavior of the degree inside the $\mathcal{S}_0$-class $\mathcal{S}_0(f)$ by looking at the ANF of f. As a consequence, we show that for any value between $n/2$ and $n-1$ (included) there exist WPB functions reaching this degree. The proof is constructive, we exhibit a new family of WPB functions with prescribed degree for all $n = 2^m$ (with $m \in \mathbb{N}^*$).

Definition 3 (Sigma-degree $\sigma\mathsf{deg}(f)$). *Let* $n \in \mathbb{N}^*$, *and* $f \in \mathcal{B}_n$. *Let* D_f *be the set of* $d \in [1, n-1]$ *such that the ANF of* f *contains at least a degree* d *monomial but not all of them. We define:* $\sigma\mathsf{deg}(f) = \max D_f$ *if* $D_f \neq \emptyset, 0$ *otherwise.*

Lemma 2. *Let* $m \in \mathbb{N}^*$ *and* $n = 2^m$. *Let* f, g $\mathcal{S}_0$*-equivalent Boolean functions in* n *variables. Then,* $\sigma\mathsf{deg}(f) = \sigma\mathsf{deg}(g)$.

Hence, $\sigma\mathsf{deg}(f)$ is an invariant of the $\mathcal{S}_0$-class and it is in fact the minimum degree in the class when f is not a symmetric function:

Theorem 3. *Let $m \in \mathbb{N}^*$ and $n = 2^m$. Let $f \in \mathcal{B}_n$ such that $f \notin \mathcal{SYM}_n$ and $\delta \in \mathbb{N}$.*

- *there exist exactly $2^{\sigma\mathsf{deg}(f)}$ functions $g \in \mathcal{S}_0(f)$ such that $\mathsf{deg}(g) = \sigma\mathsf{deg}(f)$.*
- *if $\sigma\mathsf{deg}(f) < \delta < n$, there exist exactly $2^{\delta-1}$ functions $g \in \mathcal{S}_0(f)$ such that $\mathsf{deg}(g) = \delta$.*
- *if $\delta < \sigma\mathsf{deg}(f)$, there does not exist $g \in \mathcal{S}_0(f)$ such that $\mathsf{deg}(g) = \delta$.*

Therefore, in the $\mathcal{S}_0$ class of every WPB function there exists at least a function of degree $n-1$, *i.e.* the minimum of the maximal degree inside an $\mathcal{S}_0$-class of $\mathcal{WPB}_m$ is $n-1$:

Corollary 1. *Let $m \in \mathbb{N}^*$. $\mathsf{mdeg}\mathcal{S}_0(m) = n-1$.*

We can specialize the argument of Theorem 3 to explicitly construct WPB functions having for degree any value between $n/2$ and $n-1$ included, from CMR family.

Corollary 2 (WPB functions with prescribed degree). *Let $m \in \mathbb{N}^*$, $n = 2^m$, and $d \in [\frac{n}{2}, n-1]$. We define $f_{n,n/2} = f_n$ as in [CMR17, Proposition 5], and for all $\frac{n}{2} < d < n$, $f_{n,d} = f_n + \sigma_{d,n}$. The function $f_{n,d}$ is weightwise perfectly balanced and $\mathsf{deg}(f_{n,d}) = d$.*

Degree distribution in $\mathcal{WPB}_m$. Let $m \in \mathbb{N}^*$ and $n = 2^m$. We observe that $\mathcal{S}_0$-classes form a partition of $\mathcal{WPB}_m$ from Proposition 1. Denoting by $\theta_{d,m}$ the number of $\mathcal{S}_0$-classes such that $\sigma\mathsf{deg}(f) = d$ and setting $D_{d,m} = |\{f \in \mathcal{WPB}_m : \deg f = d\}|$, from Theorem 3 we have that:

$$D_{d,m} = 2^d \cdot \theta_{d,m} + 2^{d-1} \cdot \sum_{k=0}^{d-1} \theta_{k,m} = 2^{d-1} \cdot \theta_{d,m} + 2^{d-1} \cdot \sum_{k=0}^{d} \theta_{k,m}.$$

Theorem 4. *Let $m \in \mathbb{N}^*$, $n = 2^m$, the probability of a WPB function from $\mathcal{WPB}_m$ having degree $n-1$ is:*

$$\frac{D_{n-1,m}}{|\mathcal{WPB}_m|} = \frac{2^{n-2}\theta_{n-1,m}}{|\mathcal{WPB}_m|} + \frac{1}{2} > 1/2. \tag{1}$$

Practical experiments. To complement this investigation on the degree, we perform an experimental study of the degree distribution for WPB functions in a small number of variables. The results will be displayed in the full version of the paper.

5 Minimal parameters inside the $\mathcal{S}_0$-classes of WPB functions

For a WPB function reaching a very small algebraic immunity or nonlinearity, there always exists a function with better parameters in its $\mathcal{S}_0$-class. On the experimental side, it allows to optimize the parameters of a WPB while staying in the class.

Algebraic immunity inside an $\mathcal{S}_0$ class. In this part we focus on the $\mathsf{mAI}\mathcal{S}_0(m)$ parameter (Definition 2). In [GM23], the minimal AI that a WPB function can have is proven to be 2. In the following we show that $\mathsf{mAI}\mathcal{S}_0(m) > 2$ (for $m \geq 6$), which means that for such WPB functions exhibited in [GM23], there always exist functions with better AI in their $\mathcal{S}_0$-class, more adequate to be used in a cipher. We begin by demonstrating a general lemma:

Lemma 3. *Let $m \in \mathbb{N}^*$ and $n = 2^m$, let $t \in \mathbb{N}^*$, if there exist 2^t functions s_i in $\mathcal{S}_0$ such that $\mathsf{AI}(s_i) > 2t$, and $\mathsf{AI}(s_i + s_j) > 2t$ for all $i \neq j$, then for all $f \in \mathcal{B}_n$ there exists $g \in \mathcal{S}_0(f)$ such that $\mathsf{AI}(g) \geq t+1$.*

Then, we need a result on the AI of some symmetric functions, to show the existence of 2^t functions satisfying the conditions of Lemma 3 in $\mathcal{S}_0$.

Proposition 2. *Let $m \in \mathbb{N}^*$ and $n = 2^m$, let $r \in \mathbb{N}^*$, $r < m$, for all vector $v \in (\mathbb{F}_2^r)^*$ the symmetric function f defined as: $f = \sum_{i=1}^{r} v_i \sigma_{2^m - 2^{m-i}, 2^m}$ is such that $\mathsf{AI}(f) \geq 2^{m-r} - 1$.*

It allows to derive a first lower bound on $\mathsf{mAI}\mathcal{S}_0(m)$:

Theorem 5 (Lower bound on $\mathsf{mAI}\mathcal{S}_0(m)$). *Let $t, m \in \mathbb{N}$, $t \geq 2$, if $m > \log(2t+1) + t + 1 + (t \mod 2)$ then $\mathsf{mAI}\mathcal{S}_0(m) \geq t + 1$.*

Taking the first m satisfying the condition of Theorem 5, $m_t = \lfloor \log(2t+1) \rfloor + t + 2 + (t \mod 2)$, the first values are $m_2 = 6$, $m_3 = 8$, $m_4 = 9$, and $m_5 = 11$.

Theorem 5 shows that for $m \geq 6$ there are functions with AI at least 3 in each $\mathcal{S}_0$-class of $\mathcal{WPB}_m$. An interesting research direction is to determine if $\mathsf{mAI}\mathcal{S}_0(m) = 2^{m-1}$. If it holds, there are functions with optimal AI in each $\mathcal{S}_0$-class, and then finding a WPB function with good AI together with good NL_k and AI_k boils down to determining the adequate representative. If it does not hold, it is appealing to characterize the classes where optimal AI is not reachable.

Nonlinearity inside an $\mathcal{S}_0$-class. In this part we focus on $\mathsf{mNL}\mathcal{S}_0(m)$, as defined in Definition 2. In [GM22c], WPB functions with a nonlinearity as low as $2^{n/2-1}$ have been exhibited. In this part we demonstrate that $\mathsf{mNL}\mathcal{S}_0(m) \geq 2^{n-2} - 2^{\frac{n}{2}-2}$.

Theorem 6 (Lower bound on $\mathsf{mNL}\mathcal{S}_0(m)$). *Let $m \in \mathbb{N}$, $m \geq 2$ and $n = 2^m$, the following holds:*

$$\mathsf{mNL}\mathcal{S}_0(m) \geq 2^{n-2} - 2^{\frac{n}{2}-2}.$$

6 Beyond parameters in $\mathcal{S}_0$-classes

These results have more implications for cryptographic applications: for example in the (improved) filter permutator context [MJSC16, MCJS19], for hybrid homomorphic encryption, there are efficient ways to evaluate symmetric functions (as illustrated in [HMR20]), and doing one addition is cheap, therefore it is interesting to consider the best function in the $\mathcal{S}_0$-class of a filter function. In that case, for all contexts where adding one function is cheap, the hunt for optimized functions could be split into finding a cheap function to evaluate, and then determining the one with best cryptographic parameters in its $\mathcal{T}$-class. The $\mathcal{T}$-class would be the class given by an equivalence relation up to the addition of a fixed family of functions, at the same time efficiently computable in the context and enabling good cryptographic parameters.

Different results we presented can be generalized to $\mathcal{T}$-classes, in particular denoting $\mathsf{mdeg}\mathcal{T}$, $\mathsf{mAI}\mathcal{T}$ and $\mathsf{mNL}\mathcal{T}$, the minimum over the maximum degree, AI and nonlinearity parameter inside a $\mathcal{T}$-class:

- Similarly to Corollary 1, denoting by D the maximum degree of functions inside $\mathcal{T}$, we obtain that $\mathsf{mdeg}\mathcal{T} \geq D$.
- Lemma 3 can be generalized to any family $\mathcal{T}$, hence for any family $\mathcal{T}$ with functions with high AI and such that the sum of two elements still have high AI, we can obtain a bound on $\mathsf{mAI}\mathcal{T}$ similarly to the one of Theorem 5.
- The bound on $\mathsf{mNL}\mathcal{S}_0(m)$ from Theorem 6 comes from the fact that a bent function belongs to $\mathcal{S}_0$. Then, the same bound applies for each family $\mathcal{T}$ containing a bent function. More generally, denoting B the maximal nonlinearity for a function in $\mathcal{T}$, the bound $\mathsf{mNL}\mathcal{T} \geq B/2$ holds.

Acknowledgments. The two authors were supported by the ERC Advanced Grant no. 787390.

References

BP05. An Braeken and Bart Preneel. On the algebraic immunity of symmetric boolean functions. In *Progress in Cryptology - INDOCRYPT 2005, 6th International Conference on Cryptology in India, Bangalore, India, December 10-12, 2005, Proceedings*, pages 35–48, 2005.

Car04. Claude Carlet. On the degree, nonlinearity, algebraic thickness, and nonnormality of boolean functions, with developments on symmetric functions. *IEEE Trans. Information Theory*, pages 2178–2185, 2004.

Car21. Claude Carlet. *Boolean Functions for Cryptography and Coding Theory*. Cambridge University Press, 2021.

CM22. Claude Carlet and Pierrick Méaux. A complete study of two classes of boolean functions: direct sums of monomials and threshold functions. *IEEE Transactions on Information Theory*, 68(5):3404–3425, 2022.

CMR17. Claude Carlet, Pierrick Méaux, and Yann Rotella. Boolean functions with restricted input and their robustness; application to the FLIP cipher. *IACR Trans. Symmetric Cryptol.*, 2017(3), 2017.

CV05. Anne Canteaut and Marion Videau. Symmetric boolean functions. *IEEE Trans. Information Theory*, pages 2791–2811, 2005.

DMS06. Deepak Kumar Dalai, Subhamoy Maitra, and Sumanta Sarkar. Basic theory in construction of boolean functions with maximum possible annihilator immunity. *Designs, Codes and Cryptography*, 2006.

Fin47. N. J. Fine. Binomial coefficients modulo a prime. *The American Mathematical Monthly*, 54(10):589–592, 1947.

GM22a. Agnese Gini and Pierrick Méaux. On the weightwise nonlinearity of weightwise perfectly balanced functions. *Discret. Appl. Math.*, 322:320–341, 2022.

GM22b. Agnese Gini and Pierrick Méaux. Weightwise almost perfectly balanced functions: Secondary constructions for all n and better weightwise nonlinearities. In Takanori Isobe and Santanu Sarkar, editors, *Progress in Cryptology - INDOCRYPT*, volume 13774 of *Lecture Notes in Computer Science*, pages 492–514. Springer, 2022.

GM22c. Agnese Gini and Pierrick Maux. Weightwise perfectly balanced functions and nonlinearity. Cryptology ePrint Archive, Paper 2022/1777, 2022.

GM23. Agnese Gini and Pierrick Maux. On the algebraic immunity of weightwise perfectly balanced functions. Cryptology ePrint Archive, Paper 2023/495, 2023. https://eprint.iacr.org/2023/495

GS22. Xiaoqi Guo and Sihong Su. Construction of weightwise almost perfectly balanced boolean functions on an arbitrary number of variables. *Discrete Applied Mathematics*, 307:102–114, 2022.

HMR20. Clément Hoffmann, Pierrick Méaux, and Thomas Ricosset. Transciphering, using filip and TFHE for an efficient delegation of computation. In Karthikeyan Bhargavan, Elisabeth Oswald, and Manoj Prabhakaran, editors, *Progress in Cryptology - INDOCRYPT 2020 - 21st International Conference on Cryptology in India, Bangalore, India, December 13-16, 2020, Proceedings*, volume 12578 of *Lecture Notes in Computer Science*, pages 39–61. Springer, 2020.

LM19. Jian Liu and Sihem Mesnager. Weightwise perfectly balanced functions with high weightwise nonlinearity profile. *Des. Codes Cryptogr.*, 87(8):1797–1813, 2019.

LS20. Jingjing Li and Sihong Su. Construction of weightwise perfectly balanced boolean functions with high weightwise nonlinearity. *Discret. Appl. Math.*, 279:218–227, 2020.

MCJS19. Pierrick Méaux, Claude Carlet, Anthony Journault, and François-Xavier Standaert. Improved filter permutators for efficient FHE: better instances and implementations. In Feng Hao, Sushmita Ruj, and Sourav Sen Gupta, editors, *Progress in Cryptology - INDOCRYPT*, volume 11898 of *LNCS*, pages 68–91. Springer, 2019.

Méa21. Pierrick Méaux. On the fast algebraic immunity of threshold functions. *Cryptogr. Commun.*, 13(5):741–762, 2021.

MJSC16. Pierrick Méaux, Anthony Journault, François-Xavier Standaert, and Claude Carlet. Towards stream ciphers for efficient FHE with low-noise ciphertexts. In Marc Fischlin and Jean-Sébastien Coron, editors, *EUROCRYPT 2016, Part I*, volume 9665 of *LNCS*, pages 311–343. Springer, Heidelberg, May 2016.

MKCL22. Sara Mandujano, Juan Carlos Ku Cauich, and Adriana Lara. Studying special operators fortheapplication ofevolutionary algorithms intheseek ofoptimal boolean functions forcryptography. In Obdulia Pichardo Lagunas, Juan Martínez-Miranda, and Bella Martínez Seis, editors, *Advances in Computational Intelligence*, pages 383–396, Cham, 2022. Springer Nature Switzerland.

MPJ+22. Luca Mariot, Stjepan Picek, Domagoj Jakobovic, Marko Djurasevic, and Alberto Leporati. Evolutionary construction of perfectly balanced boolean functions. In *2022 IEEE Congress on Evolutionary Computation (CEC)*, page 18. IEEE Press, 2022.

MS78. F.J. MacWilliams and N.J.A. Sloane. *The Theory of Error-Correcting Codes*. North-holland Publishing Company, 2nd edition, 1978.

MS21. Sihem Mesnager and Sihong Su. On constructions of weightwise perfectly balanced boolean functions. *Cryptography and Communications*, 2021.

MSL21. Sihem Mesnager, Sihong Su, and Jingjing Li. On concrete constructions of weightwise perfectly balanced functions with optimal algebraic immunity and high weightwise nonlinearity. *Boolean Functions and Applications*, 2021.

MSLZ22. Sihem Mesnager, Sihong Su, Jingjing Li, and Linya Zhu. Concrete constructions of weightwise perfectly balanced (2-rotation symmetric) functions with optimal algebraic immunity and high weightwise nonlinearity. *Cryptogr. Commun.*, 14(6):1371–1389, 2022.

MT21. Sihem Mesnager and Chunming Tang. Fast algebraic immunity of boolean functions and LCD codes. *IEEE Trans. Inf. Theory*, 67(7):4828–4837, 2021.

QFLW09. Longjiang Qu, Keqin Feng, Feng Liu, and Lei Wang. Constructing symmetric boolean functions with maximum algebraic immunity. *IEEE Transactions on Information Theory*, 55:2406–2412, 05 2009.

SM07. Palash Sarkar and Subhamoy Maitra. Balancedness and correlation immunity of symmetric boolean functions. *Discrete Mathematics*, pages 2351 – 2358, 2007.

TL19. Deng Tang and Jian Liu. A family of weightwise (almost) perfectly balanced boolean functions with optimal algebraic immunity. *Cryptogr. Commun.*, 11(6):1185–1197, 2019.

ZJZQ23. Qinglan Zhao, Yu Jia, Dong Zheng, and Baodong Qin. A new construction of weightwise perfectly balanced functions with high weightwise nonlinearity. *Mathematics*, 11(5), 2023.

ZLC+23. Qinglan Zhao, Mengran Li, Zhixiong Chen, Baodong Qin, and Dong Zheng. A unified construction of weightwise perfectly balanced boolean functions. Cryptology ePrint Archive, Paper 2023/460, 2023. https://eprint.iacr.org/2023/460.

ZS21. Rui Zhang and Sihong Su. A new construction of weightwise perfectly balanced boolean functions. *Advances in Mathematics of Communications*, 0:–, 2021.

ZS22. Linya Zhu and Sihong Su. A systematic method of constructing weightwise almost perfectly balanced boolean functions on an arbitrary number of variables. *Discrete Applied Mathematics*, 314:181–190, 2022.

Orientable sequences over nonbinary alphabet

Abbas Alhakim, Chris J. Mitchell, Janusz Szmidt, Peter R. Wild

April 15, 2023

Abstract

We consider orientable sequences over the residue group $\mathbb{Z}_q$. We prove properties of a generalized Lempel homomorphism and give an upper bound on the periods of orientable sequences. We generalize the results of [6].

1 Introduction

For positive integers n and q greater than one, let $\mathbb{Z}_q^n$ be the set of all q^n vectors of length n with entries in the group $\mathbb{Z}_q$ of residues modulo q. An order n de Bruijn sequence with alphabet in $\mathbb{Z}_q$ is a periodic sequence that includes every possible string of size n exactly once as a subsequence of consecutive symbols in one period of the sequence.

The order n de Bruijn digraph, $B_n(q)$, is a directed graph with $\mathbb{Z}_q^n$ as its vertex set and for any two vectors $\mathbf{x} = (x_1, \ldots, x_n)$ and $\mathbf{y} = (y_1, \ldots, y_n)$, $(\mathbf{x}; \mathbf{y})$ is an edge if and only if $y_i = x_{i+1}$ for every i $(1 \leq i < n)$. We then say that $\mathbf{x}$ is a predecessor of $\mathbf{y}$ and $\mathbf{y}$ is a successor of $\mathbf{x}$. Evidently, every vertex has exactly q successors and q predecessors. Furthermore, two vertices are said to be conjugates if they have the same successors.

A cycle in $B_n(q)$ is a path that starts and ends at the same vertex. It is called vertex disjoint if it does not visit any vertex more than once. Two cycles or two paths in the digraph are vertex disjoint if they do not have a common vertex. A cycle in $B_n(q)$ is primitive if it does not simultaneously contain a word and any of its translates.

A function $d : \mathbb{Z}_q^n \to Z_q$ is said to be translation invariant if $d(w+\lambda) = d(w)$ for all $w \in \mathbb{Z}_q^n$ and all $\lambda \in \mathbb{Z}_q$. The weight $w(s)$ of a word or sequence s is the sum of all elements in s (not taken modulo q). Similarly, the weight of a cycle is the weight of the ring sequence that represents it. Obviously a de Bruijn sequence of order n defines a Hamiltonian cycle in $B_n(q)$, i.e., a cycle that visits each vertex exactly once and which we call a de Bruijn cycle.

For an integer $n > 1$ define a map $D : B_n(2) \to B_{n-1}(2)$ by

$$D(a_1, \ldots, a_n) = (a_1 + a_2, a_2 + a_3, \ldots, a_{n-1} + a_n)$$

where addition is modulo 2. This function defines a graph homomorphism and is known as Lempel's D-morphism since it was studied in [4].

We present a generalization to nonbinary alphabets [1]. For a nonzero $\beta \in \mathbb{Z}_q$, we define a function D_β from $B_n(q)$ to $B_{n-1}(q)$ as follows. For $a = (a_1, \ldots, a_n)$ and $b = (b_1, \ldots, b_{n-1})$, $D_\beta(a) = b$ if and only if $b_i = d_\beta(a_i, a_{i+1})$ for $i = 1$ to $n-1$, where $d_\beta(a_i, a_{i+1}) = \beta(a_{i+1} - a_i) \mod q$. Clearly D_β is translation invariant. It is also onto if $gcd(\beta, q) = 1$.

2 Orientable sequences

Definition 1
We define an n-window sequence $S = (s_i)$ (see, for example, [5]) to be a periodic sequence of period m with the property that no n-tuple appears more than once in a period of the sequence, i.e. with the property that if $s_n(i) = s_n(j)$ for some i, j, then $i = j \mod m$, where $s_n(i) = (s_i, s_{i+1}, \ldots, s_{i+n-1})$.

A de Bruijn sequence of order n over alphabet $\mathbb{Z}_q$ is then simply an n-window sequence of period q^n (i.e. of maximal period), and has the property that every possible n-tuple appears exactly once in a period. Since we are interested in tuples occurring either forwards or backwards in a sequence, we also introduce the notion of a reversed tuple, so that if $u = (u_0, u_1, \ldots, u_{n-1})$ is a q-ary n-tuple, i.e. if $u \in B_n(q)$, then $u^R = (u_{n-1}, u_{n-2}, \ldots, u_0)$ is its reverse. If a tuple u satisfies $u = u^R$ then we say it is symmetric.

A translate of a tuple involves switching $u = (u_0, u_1, \ldots, u_{n-1}) \in B_n(q)$ to $\overline{u} = (u_0 + \lambda, u_1 + \lambda, \ldots, u_{n-1} + \lambda)$, where $\lambda \in \mathbb{Z}_q$. In a similar way, we refer to sequences being translates if one can be obtained from the other by the addition of a nonzero constant λ. We define the conjugate of an n-tuple to be the tuple obtained by adding λ to the first bit for some non-zero λ, i.e. if $u = (u_0, u_1, \ldots, u_{n-1}) \in B_n(q)$, then a conjugate $\widehat{u}$ of u is an n-tuple $(u_0 + \lambda, u_1, \ldots, u_{n-1})$, where $\lambda \in \mathbb{Z}_q$.

Two n-window sequences $S = (s_i)$ and $T = (t_i)$ are said to be disjoint if they do not share an n-tuple, i.e. if $s_n(i) \neq t_n(j)$ for every i, j. An n-window sequence is said to be primitive if it is disjoint from its complement. We next give a well known result showing how two disjoint n-window sequences can be *joined* to create a single n-window sequence, if they contain conjugate n-tuples.

Lemma 1
Suppose $S = (s_i)$ and $T = (t_i)$ are disjoint n-window sequences of periods l and m respectively. Moreover suppose S and T contain the conjugate n-tuples u and v at positions i and j, respectively. Then

$$[s_0, s_1, \ldots, s_{i+n-1}, t_{j+n}, t_{j+n+1}, \ldots, t_{m-1}, t_0, \ldots, t_{j+n-1}, s_{i+n}, s_{i+n+1}, \ldots, s_{l-1}]$$

is a generating cycle for an n-window sequence of period $l + m$.

Definition 2
An n-window sequence $S = (s_i)$ of period m is said to be an q-orientable sequence of order n (an $\mathcal{OS}_q(n)$) if, for any i, j , $s_n(i) \neq s_n(j)^R$.

Definition 3
A pair of disjoint orientable sequences of order n, $S = (s_i)$ and $S' = (s'_i)$, are said to be orientable disjoint (or simply o-disjoint) if, for any i, j, $s_n(i) \neq s'_n(j)^R$.

We extend the notation to allow the Lempel morphism D_β to be applied to periodic sequences in the natural way. That is, D_β is a map from the set of periodic sequences to itself; the image of a sequence of period m will clearly have period dividing m. In the natural way we can define D_β^{-1} to be the *inverse* of D_β, i.e. if S is a periodic sequence than $D_\beta^{-1}(S)$ is the set of all sequences T with the property that $D_\beta(T) = S$.

Theorem 1
Suppose $S = (s_i)$ is an orientable sequence of order n and period m with the property that

if $[s_1, \ldots, s_n]$ is a word in S then $[-s_n, -s_{n-1}, \ldots, -s_1]$ is not a word of S. $(*)$

Then
(a) If $w(S) = 0 \mod q$ then $D_\beta^{-1}(S)$ consists of a disjoint set of q primitive orientable sequences of order $n+1$ and period m satisfying the condition $(*)$.
(b) If $gcd(w(S), q) = 1$ then $D_\beta^{-1}(S)$ is one sequence made of q shifts $T_0, T_1, \ldots, T_{q-1}$, where $T_i = T_{i-1} + c$.

3 An upper bound

We present here the results from the paper [7]. We first introduce a special type of symmetry for q-ary n-tuples.

Definition 4
An n-tuple $u = (u_0, u_1, \ldots, u_{n-1})$, $u_i \in \mathbb{Z}_q$ $(0 \leq i \leq n-1)$, is m-symmetric for some $m \leq n$ if and only if $u_i = u_{m-1-i}$ for every i $(0 \leq i \leq m-1)$.

An n-tuple is simply said to be symmetric if it is n-symmetric. We also need the notions of uniformity and alternating.

Definition 5
An n-tuple $\mathbf{u} = (u_0, u_1, \ldots, u_{n-1})$, $u_i \in \mathbb{Z}_q$ $(0 \leq i \leq n-1)$, is *uniform* if and only if $u_i = c$ for every i $(0 \leq i \leq n-1)$ for some $c \in \mathbb{Z}_q$. An n-tuple $\mathbf{u} = (u_0, u_1, \ldots, u_{n-1})$, $u_i \in \mathbb{Z}_q$ $(0 \leq i \leq n-1)$, is *alternating* if and only if $u_0 = u_{2i}$ and $u_1 = u_{2i+1}$ for every i $(0 \leq i \leq \lfloor (n-1)/2 \rfloor)$, where $u_0 \neq u_1$.

We can then state the following elementary results.

Lemma 2
If $n \geq 2$ and $\mathbf{u} = (u_0, u_1, \ldots, u_{n-1})$ is a q-ary n-tuple that is both symmetric and $(n-1)$-symmetric, then $\mathbf{u}$ is uniform.

Lemma 3
If $n \geq 2$ and $\mathbf{u} = (u_0, u_1, \ldots, u_{n-1})$ is a q-ary n-tuple that is both symmetric and $(n-2)$-symmetric then either $\mathbf{u}$ is uniform or n is odd and $\mathbf{u}$ is alternating.

The following definition leads to a simple upper bound on the period of an $\mathcal{OS}_q(n)$.

Definition 6
Let $N_q(n)$ be the set of all non-symmetric q-ary n-tuples.

Clearly, if an n-tuple occurs in an $\mathcal{OS}_q(n)$ then it must belong to $N_q(n)$; moreover it is also immediate that $|N_q(n)| = q^n - q^{\lceil n/2 \rceil}$. Observing that all the tuples in $\mathcal{OS}_q(n)$ and its reverse must be distinct, this immediately give the following well-known result.

Lemma 4 ([2])
The period of an $\mathcal{OS}_q(n)$ is at most $(q^n - q^{\lceil n/2 \rceil})/2$.

As a first step towards establishing our bound we need to define a special set of n-tuples, as follows.

Definition 7
Suppose $n \geq 2$, and that $\mathbf{v} = (v_0, v_1, \ldots, v_{n-r-1})$ is a q-ary $(n-r)$-tuple ($r \geq 1$). Then let $L_n(\mathbf{v})$ be the following set of q-ary n-tuples:

$$L_n(\mathbf{v}) = \{\mathbf{u} = (u_0, u_1, \ldots, u_{n-1}) : \ u_i = v_i, \ \ 0 \leq i \leq n-r-1\}.$$

That is $L_n(\mathbf{v})$ is simply the set of n-tuples whose first $n-r-1$ entries equal $\mathbf{v}$. Clearly, for fixed r, the sets $L_n(\mathbf{v})$ for all $(n-r)$-tuples $\mathbf{v}$ are disjoint. We have the following simple result.

Lemma 5
Suppose $\mathbf{v}$ and $\mathbf{w}$ are symmetric tuples of lengths $n-1$ and $n-2$, respectively, and they are not both uniform. Then

$$L_n(\mathbf{v}) \cap L_n(\mathbf{w}) = \emptyset.$$

We are particularly interested in how the sets $L_n(\mathbf{v})$ intersect with the sets of n-tuples occurring in either S or S^R, when S is an $\mathcal{OS}_q(n)$ and $\mathbf{v}$ is symmetric. To this end we make the following definition.

Definition 8
Suppose $n \geq 2$, $r \geq 1$, $S = (s_i)$ is an $\mathcal{OS}_q(n)$, and $\mathbf{v} = (v_0, v_1, \ldots, v_{n-r-1})$ is a k-ary $(n-r)$-tuple. Then let

$$L_S(\mathbf{v}) = \{\mathbf{u} \in L_n(\mathbf{v}) : \ \mathbf{u} \text{ appears in } S \text{ or } S^R\}.$$

We can now state the first result towards deriving our bound.

Lemma 6
Suppose $n \geq 2$, $r \geq 1$, $S = (s_i)$ is an $\mathcal{OS}_q(n)$, and $\mathbf{v} = (v_0, v_1, \ldots, v_{n-r-1})$ is a q-ary symmetric $(n-r)$-tuple. Then $|L_S(\mathbf{v})|$ is even.

That is, if $|L_n(\mathbf{v})|$ is odd, this shows that S and S^R combined must omit at least one of the n-tuples in $L_n(\mathbf{v})$. We can now state our main result.

Observe that, although the theorem below applies in the case $q = 2$, the bound is much weaker than the bound of Dai et al. [3] which is specific to the binary case. This latter bound uses arguments that only apply for $q = 2$. The fact that $q = 2$ is a special case can be seen by observing that, unlike the case for larger q, no string of $n-2$ consecutive zeros or ones can occur in an $\mathcal{OS}_d(n)$.

Theorem 2 (Generalization of Theorem from [3])
Suppose that $S = (s_i)$ is an $\mathcal{OS}_q(n)$ ($q \geq 2$, $n \geq 2$). Then the period of S is at most

$$\begin{array}{rl} (q^n - q^{\lceil n/2 \rceil} - q^{\lceil (n-1)/2 \rceil} + q)/2 & \text{if } q \text{ is odd,} \\ (q^n - q^{\lceil n/2 \rceil} - q)/2 & \text{if } q \text{ is even.} \end{array}$$

Table 1 provides the values of the bounds in the above theorem for small q and n. We also give an example of a recursively constructed sequence in $\mathcal{OS}_5(4)$ using $S = 0001112$ which is of order 3. In the notation of Theorem 1(a) with $\beta = 1$:
$T_0 = \underline{0}\ 0\ 0\ 0\ 1\ 2\ 3\ \not{0}$
$T_1 = 1\ 1\ 1\ 1\ 2\ 3\ 4\ \not{1}$
$T_2 = 2\ 2\ 2\ 2\ 3\ 4\ 0\ \not{2}$
$T_3 = 3\ 3\ 3\ 3\ 4\ 0\ 1\ \not{3}$
$T_4 = 4\ 4\ 4\ 4\ 0\ 1\ 2\ \not{4}$
These 4-window cycles are o-disjoint and can therefore be stitched together into one orientable sequence of order 4, by applying the construction given in [1].

Table 1: Bounds on the period of an $\mathcal{OS}_q(n)$ (from Theorem 2)

Order	$q = 2$	$q = 3$	$q = 4$	$q = 5$
$n = 2$	0	3	4	10
$n = 3$	1	9	22	50
$n = 4$	5	33	118	290
$n = 5$	11	105	478	1490

References

[1] A. Alhakim and M. Akinwande. A recursive construction of nonbinary de Bruijn sequences. Design, Codes and Cryptography. 60:155–169, (2011).

[2] J. Burns and C. J. Mitchell. Coding schemes for two-dimensional position sensing. Cryptography and Coding III (M. J. Ganley, ed.), Oxford University Press, pp. 31–66, 1993.

[3] Z.-D. Dai, K. M. Martin, M. J. B. Robshaw, and P. R. Wild. Orientable sequences. Cryptography and Coding III (M. J. Ganley, ed.), Oxford University Press, Oxford, pp. 97–115, 1993.

[4] A. Lempel. On a homomorphism of the de Bruijn graph and its applications to the design of feedback shift registers. IEEE Trans. Comput. C 19, 1204–1209 (1970).

[5] C. J. Mitchell, T. Etzion, and K. G. Paterson. A method for constructing decodable de Bruijn sequences, IEEE Transactions on Information Theory 42 (1996), 1472–1478.

[6] C. J. Mitchell. and P. R. Wild. Constructing Orientable Sequences. IEEE Transations on Information Theory, Vol. 68, no. 7, July 2022.

[7] C. J. Mitchell. and P. R. Wild. Bounds on the period of k-ary orientable sequences. Preprint, January 2022.

Improving differential properties of S-boxes with local changes of DDT

(Extended Abstract)

Pavol Zajac[1*]

[1*]Department of Computer Science and Mathematics, Slovak University of Technology in Bratislava, Ilkovicova 3, Bratislava, 81219, Slovakia.

Corresponding author(s). E-mail(s): pavol.zajac@stuba.sk;

Keywords: differential uniformity, S-box generation algorithm, cryptographic applications of Boolean functions

1 Introduction

In a recent article [1] we have proposed an algorithm to construct S-boxes with prescribed differential properties. The main principle is to produce the function assignment in discrete increments while checking the restrictions on a partially constructed difference distribution table. Although the algorithm can find any S-box with prescribed differential properties, it is impractical due to its high complexity. On the other hand, it can produce S-boxes with better differential properties than a random selection. We can naturally ask whether it is possible to obtain better S-boxes by manipulating the vector of values in a systematic way by taking into account the differential properties of the S-box.

We already know that the answer is positive. There is a large number of results concerning heuristic methods (e.g., evolutionary algorithms) that produce better S-boxes than a random search. However, these methods are typically generic, focusing on improving some objective function with a stochastic search directed by an objective function. The objective function typically includes but is not restricted to, the differential properties of the S-box.

In our current research, we take a different approach to the problem. We start with a random S-box and specify the operation we are allowed to do with the S-box value vector. We are concerned with bijective S-boxes, where the value vector is

a permutation. Our permitted operations are either swaps (interchange of values of $y_1 = S(x_1)$ and $y_2 = S(x_2)$) or application of cycles. We select the operation based on the difference distribution table properties in a systematic way. Our experimental results show that it is possible to significantly improve the differential properties of an S-box, but there seems to be a limit to which this strategy converges.

2 Methods

Let $S : \mathbb{Z}_2^n \to \mathbb{Z}_2^n$ be a bijective vectorial Boolean function (an S-box). The difference distribution table DDT of S contains values

$$DDT(dx, dy) = |\{x \in \mathbb{Z}_2^n : S(x) \oplus S(x \oplus dx) = dy\}| .$$

S-box S is δ-differentially uniform, if $\delta = \max_{dx \neq 0} \{DDT(dx, dy)\}$. Let c_0 denote the number of zero elements of DDT of S (for nonzero dx),

$$c_0 = |\{(dx, dy) \in \mathbb{Z}_2^n \times \mathbb{Z}_2^n : DDT(dx, dy) = 0 \wedge dx \neq 0\}| .$$

In ideal case, $\delta = 2$ and $c_0 = 2^n(2^n - 1)/2$ (for APN function S). Randomly selected S-box will have higher δ and c_0 due to collisions of differences: for some dx, randomly selected x_1 and $x_2 \neq x_1$, $x_2 \neq x_1 \oplus dx$ will produce the same difference $dy = S(x_i) \oplus S(x_i \oplus dx)$. Such collisions increase the maximum value in DDT and the number of zero positions in DDT (if there is a colliding pair, there will not be enough remaining x values to produce some dy).

Suppose that there exists a bijective APN S-box (or some S-box of desired quality), and consider it as a permutation. By composing the S-box with some other permutation, we will change the properties of the DDT in some way, typically decreasing the quality of the S-box. On the other hand, with systematic compositions (e.g., by using a swap of two elements, a transposition), it is possible to construct any other permutation from it. This process can work in the opposite direction: starting from a random S-box, we might reach an S-box with high quality. Unfortunately, in each step we can choose from a large number of transpositions, and the complexity of constructing the desired permutation blindly is super-exponential.

Our main research question is as follows: Starting from a random S-box, can we use a greedy selection of transpositions directed by DDT properties to find the best possible S-box? If not, how far can we improve our initial random selection? We do not know the theoretical answers to this question, but we present some experimental results for consideration.

In our experiments, we use the following method to improve S-boxes:

1. Start from a randomly selected S-box S.
2. For each dy compute list $D(dy)$ which contains quadruples (y_1, y_2, y_3, y_4), such that $dy = y_1 \oplus y_2 \oplus y_3 \oplus y_4$, with $y1 = S(x_1)$, $y_2 = S(x_1 \oplus dx)$, $y3 = S(x_2)$, $y_4 = S(x_2 \oplus dx)$, with $dx \neq 0$, $x_2 \neq x_1$ and $x_2 \neq x_1 \oplus dx$.

	Iterations			Changes	Initial $D(0)$			Final $D(0)$		
	min	avg	max	avg	min	avg	max	min	avg	max
Swaps	55	70.46	93	140.92	7734	8208.63	8601	4983	5161.08	5397
3-cycles	34	48.28	64	144.84	7710	8184.72	8571	4887	5082.78	5358
4-cycles	26	36.91	50	147.85	7776	8184.96	8661	4926	5053.53	5271

Table 1 The results of the experiments: number of iterations (and estimated average number of changes in S-box), initial, and final sizes of $D(0)$ sets.

3. List $D(0)$ contains "DDT collissions". In the ideal case, we want this list to be empty. We terminate the algorithm if $D(0)$ is empty, or if we cannot decrease its size by any transposition (see the next step).
4. For each pair (y, z), $y \neq z$, compute the effect of applying transposition $y \leftrightarrow z$ to S on the size of $D(0)$ (see discussion below).
5. Apply the transposition that minimizes the next $D(0)$. Repeat the algorithm with the new S-box.

The crucial step in the algorithm is the estimation of the size of $D(0)$ in the next step. In the case of transpositions, we can compute the value exactly. Each element of type (y, z, y_3, y_4) will remain in the corresponding set $D(dy)$, as swapping y and z will not change the sum dy. Each element of type (y, y_2, y_3, y_4) or (z, y_2, y_3, y_4) will change its position in sets $D(dy)$: If $y \oplus y_2 \oplus y_3 \oplus y_4 = dy$, than $z \oplus y_2 \oplus y_3 \oplus y_4 = dy \oplus (y \oplus z)$. Thus, all elements of this type will move from set $D(0)$ to set $D(y \oplus z)$, decreasing the size of new $D(0)$. On the other hand, such elements from $D(y \oplus z)$ will move to $D(0)$, increasing its size. By summing all such contributions (with a minus sign for elements in $D(0)$ and a plus sign for elements in $D(y \oplus z)$), we can assign a score (expected change of $|D(0)|$) to each pair (y, z).

Using the greedy approach, we first select such pair with the lowest score, if the score is less than 0. If there is no pair (y, z) with a negative score, we end the algorithm to ensure that it stops.

The method can be generalized in multiple ways. One generalization is to apply multiple transpositions at once (e.g., all that produce a negative score). In such a case, however, we cannot predict the change of $D(0)$ exactly because of the interactions between contributions of elements containing values from multiple transposition pairs. The preliminary experiments with multiple transpositions were unsuccessful.

The second generalization is to replace transpositions with more general permutations. We have used small cycles of sizes 3 and 4. In the experiments, we computed the estimated score for all possible cycles of a given size. This means that the complexity quickly grows with the cycle size and becomes impractical for longer cycles. Note that in case of more cycles, we have computed the score just as the unidirectional contributions of individual elements in the cycle (e.g., $a \leftarrow b$, then $b \leftarrow c$, then $c \leftarrow a$). This score is then only an estimate, as it ignores the effects of elements that contain multiple elements from the cycles. Even if the estimated change of $D(0)$ is negative, sometimes the contribution from the shared elements causes the $D(0)$ in the next step to grow. In such a case we again terminate the algorithm to ensure that it stops.

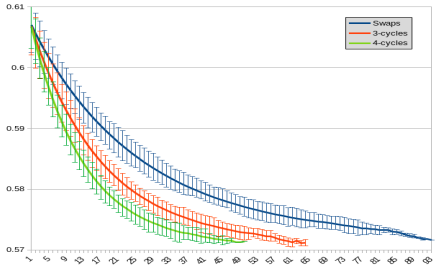

Fig. 1 Average fraction of zeroes (y) in the DDT (of 100 randomly generated S-boxes) after x iterations of the algorithm.

3 Results and Discussion

We have conducted 3 experiments with custom software implementing three methods from Section 2: swap, 3-cycles, and 4-cycles. For each experiment, we generated 100 random bijective 8-bit S-boxes. In the initial set, the S-boxes had differential uniformity between 10 and 16, with 60.7% of zeroes in the DDT. The "smoothing algorithm" improved the DDT of S-boxes gradually (see Figure 1), reaching a minimum of 57.1% of zeroes in the DDT (consistently for the 3 different methods). The final differential uniformity was between 8 and 10, the 3-cycle method produced two S-boxes with $\delta = 6$ (but not as a final step), and the 4-cycle method produced 1 S-box with $\delta = 6$ (in a final step).

While the number of steps depends on the chosen method, the expected number of changes in the S-box table, and the final results seem independent of the method chosen. From the computational perspective, it is better to implement only the "swap" method, which exchanges two values at a time, and each iteration is faster than the other methods.

An interesting observation is that the DDT-smoothing method also improved the non-linearity of the S-boxes. From the initial values between 86 and 96, we have reached S-boxes with non-linearity between 98 and 102. Interestingly, these values are comparable to results of advanced evolutionary techniques (see [2]) while using only a simple algorithm focusing on a completely different S-box characteristic.

Acknowledgments. This research was supported in part by the NATO Science for Peace and Security Programme under Project G5985 and in part by the Slovak Scientific Grant Agency, Grant Number VEGA 1/0105/23.

References

[1] Marochok, S., Zajac, P.: Algorithm for generating s-boxes with prescribed differential properties. Algorithms **16**(3), 157 (2023)

[2] Picek, S., Cupic, M., Rotim, L.: A new cost function for evolution of s-boxes. Evolutionary computation **24**(4), 695–718 (2016)

Counting unate and balanced monotone Boolean functions (Extended abstract)

Aniruddha Biswas and Palash Sarkar
Indian Statistical Institute
203, B.T.Road, Kolkata
India 700108.
Email: {anib_r, palash}@isical.ac.in

April 27, 2023

Abstract

For $n \leq 6$, we provide the number of n-variable unate and monotone Boolean functions under various restrictions. Additionally, we provide the number of balanced 7-variable monotone Boolean functions.
Keywords: Boolean function, unate Boolean function, monotone Boolean function, Dedekind number.
MSC: 05A99.

1 Introduction

For a positive integer n, an n-variable Boolean function f is a map $f : \{0,1\}^n \to \{0,1\}$. A Boolean function f is said to be monotone increasing (resp. decreasing) in the i-th variable if

$$f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n) \leq f(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n)$$

$$(\text{resp. } f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n) \geq f(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n))$$

for all possible $x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n \in \{0,1\}$. The function f is said to be locally monotone or unate, if for each $i \in \{1, \ldots, n\}$, it is either monotone increasing or monotone decreasing in the i-th variable. The function f is said to be monotone increasing (or, simply monotone) if for each $i \in \{1, \ldots, n\}$, it is monotone increasing in the i-th variable.

Unate functions have been studied in the literature from various viewpoints such as switching theory, combinatorial aspects, and complexity theoretic aspects. Monotone Boolean functions have been studied much more extensively than unate functions and have many applications so much so that it is difficult to mention a few representative works. The focus of the present work is on counting unate and monotone Boolean functions under various restrictions.

A Boolean function is said to be balanced if it takes the values 0 and 1 equal number of times. Two Boolean functions are said to be equivalent, if one can be obtained from the other by a permutation of the variables. We say that two functions are inequivalent if they are not equivalent.

The number of n-variable Boolean functions is 2^{2^n} and the number of n-variable balanced Boolean functions is $\binom{2^n}{2^{n-1}}$. For $n \leq 5$, it is possible to enumerate all n-variable Boolean functions. Consequently, the problem of counting various sub-classes of n-variable Boolean functions becomes a reasonably simple problem. Non-triviality of counting Boolean functions arises for $n \geq 6$. Often though, it becomes difficult to obtain results for n more than 7 or 8.

The number of n-variable monotone Boolean functions is called the Dedekind number, denoted $D(n)$, after Dedekind who posed the problem in 1897. Till date $D(n)$ is known only up to $n = 8$ (see [6]). A closed form summation formula for $D(n)$ was given in [2], though it was pointed out in [3] that using the formula to compute $D(n)$ has the same complexity as direct enumeration of all n-variable monotone Boolean functions. The numbers of n-variable inequivalent monotone Boolean functions are known for n up to 8 (see [7, 4]). To the best of our knowledge, there is no work in the literature on counting the number of n-variable (inequivalent) balanced monotone Boolean functions.

The number of NPN-equivalence classes[1] of unate Boolean functions has been studied (see $A003183$ in [6]). Even though the problem is to count NPN inequivalent unate functions, the entry for $A003183$ in [6] shows that by using simple operations, the problem can be reduced to that of counting monotone functions under certain restrictions. A proper subclass of unate functions is the class of unate cascade functions which have been studied in [5]. Entry $A005612$ in [6] provides counts of unate cascade functions. To the best of our knowledge, there is no work in the literature on counting the number of n-variable (inequivalent) unate functions and the number of n-variable (inequivalent) balanced unate functions.

The following notation will be used.

UBF_n	: The set of all n-variable *unate* Boolean functions (UBFs).
MBF_n	: The set of all n-variable *monotone* Boolean functions (MBFs).
$U(n)$,$V(n)$	: Number of all n-variable UBFs and balanced UBFs respectively.
$W(n)$,$X(n)$	: Number of all n-variable inequivalent UBFs and balanced inequivalent UBFs respectively.
$D(n)$,$E(n)$	: Number of all n-variable MBFs and balanced MBFs respectively.
$F(n)$	: Number of all n-variable balanced inequivalent MBFs.

Our Contributions. We obtain the values of $U(n), V(n), W(n), X(n)$ and $F(n)$ for $n \leq 6$, and the values of $E(n)$ for $n \leq 7$. None of these values were previously known.

2 Mathematical Results

Let f be an n-variable Boolean function. By $\overline{f}$, we will denote the negation of f, i.e. $\overline{f}(\mathbf{x}) = 1$ if and only if $f(\mathbf{x}) = 0$. The weight $\mathsf{wt}(f)$ of f is the size of its support, i.e. $\mathsf{wt}(f) = \#\{\mathbf{x} : f(\mathbf{x}) = 1\}$. An n-variable Boolean function f can be uniquely represented by a binary string of length 2^n in the following manner: for $0 \leq i < 2^n$, the i-th bit of the string is the value of f on the n-bit binary representation of i. We will use the same notation f to denote the string representation of f. So $f_0 \cdots f_{2^n-1}$ is the bit string of length 2^n which represents f.

[1] Two Boolean functions are said to be NPN equivalent, if one can be obtained from the other by some combination of the following operations: a permutation of the variables, negation of a subset of the variables, and negation of the output. We say that two functions are NPN inequivalent if they are not NPN equivalent.

Let g and h be two n-variable Boolean functions having string representations $g_0 \cdots g_{2^n-1}$ and $h_0 \cdots h_{2^n-1}$. We write $g \leq h$ if $g_i \leq h_i$ for $i = 0, \ldots, 2^n - 1$. From g and h, it is possible to construct an $(n+1)$-variable function f whose string representation is obtained by concatenating the string representations of g and h. We denote this construction as $f = g||h$.

In addition to the previous notation, we will also require the following notation.

$\mathcal{U}_{n,w}$, $\mathcal{M}_{n,w}$: Number of n-variable UBFs and MBFs of weight w respectively.

We record a known fact about MBFs.

Fact 1 [1] *Let g and h be n-variable Boolean functions and $f = g||h$. Then f is an MBF if and only if g and h are both MBFs and $g \leq h$.*

Next we present some new results on unate and monotone Boolean functions which will be useful in our enumeration strategy. However, the complete proofs will be presented in the full version.

Proposition 1 *Let g and h be n-variable Boolean functions and $f = g||h$. Then f is a UBF if and only if g and h are both UBFs satisfying the following two conditions.*

1. *For each variable, g and h are either both monotone increasing, or both monotone decreasing.*
2. *Either $g \leq h$ or $h \leq g$.*

Proposition 2 *If f is a UBF then $\overline{f}$ is also a UBF.*

Proposition 3 *For any $n \geq 1$ and weight $w \in [0, 2^n]$, $\mathcal{U}_{n,w} = \mathcal{U}_{n,2^n-w}$ and $\mathcal{M}_{n,w} = \mathcal{M}_{n,2^n-w}$.*

3 Enumeration Strategies

To generate all n-variable unate Boolean functions, the direct method is to generate all n-variable Boolean functions and then check each function for unateness. The problem with this approach is that there is no easy method to check whether a function is unate. So we adopted the recursive strategy which follows from Proposition 1 to generate unate functions which does not require the generation of all Boolean functions and hence is more efficient than the naive generate-and-check strategy. To generate all $(n+1)$-variable unate functions, our recursive algorithm requires considering all pairs of n-variable unate functions, i.e. a total of $(U(n))^2$ options. This is feasible for $n \leq 5$, but not for higher values of n. However, there is some subtlety in developing a recursive generation algorithm based on Proposition 1 which will be described in the full version. To obtain balanced functions, from the set of all unate functions, we filter out the ones that are unbalanced.

A similar and somewhat simpler method to recursively generate all n-variable monotone functions can be obtained from Fact 1. This method also becomes infeasible for $n > 6$. It is, however, possible to generate all 7-variable balanced monotone functions using a modified version of the recursive enumeration. We provide a general description of the method.

Suppose the set of all n-variable monotone functions have already been generated. Partition these functions into weight classes, where the number of n-variable monotone functions in weight class w is $\mathcal{M}_{n,w}$, $w = 0, \ldots, 2^n$. The method for generating all $(n+1)$-variable monotone functions

based on Fact 1 is modified as follows. Choose g from weight class w and h from weight class $2^n - w$ and check whether $g \leq h$. If the check passes, then generate $f = g||h$. The procedure ensures that the weight of f is 2^n, so that f is an $(n+1)$-variable monotone balanced function. The number of pairs of n-variable unate functions that need to be considered is $\sum_{w=0}^{2^n} \mathcal{M}_{n,w}\mathcal{M}_{n,2^n-w} = \sum_{w=0}^{2^n} (\mathcal{M}_{n,w})^2$, where the equality follows from Proposition 3. This is a substantial reduction from $(D(n))^2$ that would be otherwise required. To generate all balanced 7-variable monotone functions, the generate-and-filter strategy would have required considering $(D(6))^2 \approx 2^{45}$ pairs. The modified strategy requires considering $\sum_{w=0}^{64} (\mathcal{M}_{6,w})^2 \approx 2^{40}$ pairs, which makes the enumeration much faster. The modified method for generating balanced monotone functions can also be adapted to generate balanced unate functions and for generating all $(n+1)$-variable balanced unate functions requires considering $\sum_{w=0}^{2^n} (\mathcal{U}_{n,w})^2$ pairs.

Along with enumerating all unate functions, we also enumerate inequivalent unate functions. Similarly for the other classes of functions we enumerate inequivalent functions in the respective classes. Given a permutation π of $\{1, \ldots, n\}$ and an n-variable function f, let f^π denote the function such that for all $(x_1, \ldots, x_n) \in \{0,1\}^n$, $f^\pi(x_1, \ldots, x_n) = f(x_{\pi(1)}, \ldots, x_{\pi(n)})$. Consider the set $\mathcal{S}$ to be filterd is given as a list of functions. We incrementally generate $\mathcal{T}$ as follows. The first function in $\mathcal{S}$ is moved to $\mathcal{T}$. We iterate over the other functions in $\mathcal{S}$. For a function f in $\mathcal{S}$, we generate f^π for all permutations π of $\{1, \ldots, n\}$ using the technique described above. For each such f^π, we check whether it is present in $\mathcal{T}$. If none of the f^π's are present in $\mathcal{T}$, then we append f to $\mathcal{T}$. At the end of the procedure, $\mathcal{T}$ is the desired set of functions.

We are interested in the number of inequivalent n-variable unate functions. So we may apply the above inequivalent filtering procedure to UBF_n. It is possible to gain efficiency by noting that the weight of a function is invariant under permutation of variables. So instead of applying the inequivalent filtering procedure to UBF_n, we apply it to the weight-wise partition of UBF_n. This leads to a gain in efficiency, since a function of weight w is checked for equivalence only with other functions of weight w. The strategy for inequivalent filtering works in the same way for balanced unate functions, monotone functions and balanced monotone functions. This allows us to also find the number of inequivalent functions in these classes.

The results of the above enumeration procedures for the different classes of functions are shown in Tables 1 to 3.

n	1	2	3	4	5	6
$U(n)$	4	14	104	2170	230540	499596550
$V(n)$	2	4	14	296	18202	31392428

Table 1: Number of n-variable (balanced) UBFs for $n \leq 6$

n	1	2	3	4	5	6
$W(n)$	4	10	34	200	3466	829774
$X(n)$	2	2	6	24	254	50172

Table 2: Number of n-variable (balanced) inequivalent UBFs for $n \leq 6$

n	1	2	3	4	5	6	7
$E(n)$	1	2	4	24	621	492288	81203064840
$F(n)$	1	1	2	4	16	951	–

Table 3: Number of balanced (inequivalent) monotone functions.

4 Concluding Remarks

With access to a adequate computing resources, it should be possible to obtain $F(7)$, i.e. the number of inequivalent balanced monotone functions, and $V(7)$, i.e. the number of balanced unate functions. Both of these require computations which is somewhat more that 2^{50}. The computations can be parallelised and distributed across a large number of cores. With access to a sufficiently large computational cluster, the computations will be feasible. On the other hand, obtaining the values of $U(n), W(n)$ and $X(n)$ for $n \geq 7$ and the values of $V(n), E(n)$ and $F(n)$ for $n \geq 8$ will require new ideas. The running times for the enumeration of the corresponding sets using the techniques used in the present work are likely to remain infeasible in the forseeable future.

References

[1] Valentin Bakoev. Generating and identification of monotone Boolean functions. In *Mathematics and Education in Mathematics, Sofia*, pages 226–232, 2003.

[2] Andrzej Kisielewicz. A solution of Dedekind's problem on the number of isotone Boolean functions. *Journal fur die Reine und Angewandte Mathematik*, 1988(386):139 – 144, 1988.

[3] Aleksej D. Korshunov. Monotone Boolean functions. *Russian Mathematical Surveys*, 58(5):929 – 1001, 2003.

[4] Bartłomiej Pawelski. On the number of inequivalent monotone Boolean functions of 8 variables. `https://arxiv.org/pdf/2108.13997.pdf`, 2021.

[5] Tsutomu Sasao and Kozo Kinoshita. On the number of fanout-free functions and unate cascade functions. *IEEE Transactions on Computers*, 28(1):66–72, 1979.

[6] Neil J.A. Sloane. The online encyclopedia of integer sequences. `https://oeis.org/`, 1964.

[7] Tamon Stephen and Timothy Yusun. Counting inequivalent monotone Boolean functions. *Discrete Applied Mathematics*, 167:15–24, 2014.

MORE DE BRUIJN SEQUENCES AS CONCATENATION OF LYNDON WORDS

ABBAS ALHAKIM
DEPARTMENT OF MATHEMATICS
AMERICAN UNIVERSITY OF BEIRUT
BEIRUT, LEBANON

Abstract. We consider a de Bruijn sequence dB over a finite alphabet that is constructed via a preference function P. We use P to introduce a total order on the set of all sequences and show that it lists the de Bruijn sequences of a given order so that dB is the minimal sequence. We also show that an appropriate bijective image of the binary, prefer-opposite de Bruijn sequence is uniquely factored as a concatenation of Lyndon words. This presents a second example to the well known prefer-one de Bruijn sequence, both in terms of minimality, and in terms of concatenation of Lyndon words. We also present other examples that suggest that the concatenation property is universal for all de Bruijn sequences.

1. Introduction

The lexicographically smallest de Bruijn sequence is by far the most studied of all de Bruijn sequences. One reason may be that it is generated by the well known prefer one greedy algorithm, first discovered by Martin [10], and rediscovered several times later, see Fredriscksen [7] (note that we consider that 1 is *less* than 0). Another method of generating this smallest sequence (say, of order n) is via concatenating all Lyndon words, of lengths that divide n, in increasing lexicographical order. Donald Knuth [9] calls this construction "almost magical". It is due to this construction that many authors claim that one of the many applications of Lyndon words is to construct de Bruijn sequences. In this paper we show that the relationship between Lyndon words and de Bruijn sequences extends much further than the prefer one sequence. The main tool to do this is a transform that encodes a sequence by a sequence with the same alphabet and that is defined via a preference function. Firstly, we establish a fundamental result that every de Bruijn sequence is minimal with respect to a lexicographical order defined by the preference function that creates this de Bruijn sequence. More specifically, given a preference function that produces a de Bruijn sequence, we encode every de Bruijn sequence by keeping track of the levels of preference taken all along the sequence. We then compare these *trail sequences* via lexicographical order. It is then the de Bruijn sequence generated by this preference function that receives the lexicographically smallest encoding. The 'minimality' of the prefer one sequence is thus revealed as a special case of this general result.

Furthermore, we study two relatives of the prefer one sequence, the prefer same and the prefer opposite sequences. These two sequences have, respectively, the lexicographically smallest and largest run length encoding, see [3] for definition and proof. These optimality results follow easily from our main result. More importantly, we show that their preference trails essentially consist of a concatenation of Lyndon words, when they are encoded with respect to their own preference functions. We conclude with a conjecture that every trail sequence of a de Bruijn sequence is *essentially* a concatenation of Lyndon words, laid in some order that depends on the underlying preference function.

The rest of the paper is organized as follows. In Section 2 we give basic definitions and background about preference functions and Lyndon words with preliminary lemmas that will be essential for the rest of the paper.

[1]This research was partially supported by the University Research Board (URB) of the American University of Beirut. Project Number 104107

Key words and phrases. De Bruijn sequence, Lyndon words, preference function, prefer-one sequence, prefer-opposite sequence, prefer-same sequence.

2. PRELIMINARIES

For an integer $n \geq 1$, $\mathcal{A}^n$ refers to the set of all strings of n bits, taken from an ordered alphabet $\mathcal{A}$ with q symbols. We will denote these symbols as $\{0, 1, \ldots, q-1\}$. These strings will be referred to as *n-words*, and denoted as $a_1 \cdots a_n$ and often as $(a_1, \ldots, a_n)$ for notational clarity. α^n denotes the word obtained by concatenating the word α n times.

For an integer $n \geq 1$, a de Bruijn sequence of order n, over the alphabet $\mathcal{A}$, is defined such that every string of n consecutive bits occurs exactly one time as a substring. For example, 0001011100 is a binary De Bruijn sequence of order 3, observing that all 3-strings occur in the order 000, 001, 010, 101, 011, 111, 110, 100. It is customary to consider cyclic rotations of a de Bruijn sequence as equivalent. With this equivalence class interpretation, we usually remove the last $n-1$ bits and wrap the remaining bits on a circle. The above sequence is represented as [00010111]. The only other binary sequence of order 3 is represented as [00011101]. While there are 16 sequences of order 4. In fact, for a general n, the number of non-rotationally equivalent de Bruijn sequences is $2^{2^{n-1}-n}$. The formula for nonbinary has an even higher rate of growth, it can be found in [7], together with a historical reference of the early development and applications of de Bruijn sequences. The first part in the next definition follows Golomb [8]. The *span* was defined in Alhakim [2].

Definition 2.1. For an integer $n \geq 1$, a *preference function* is a function P from $\mathcal{A}^n$ to S, where S is the set of all permutations of the elements of $\mathcal{A}$. We write $P(\mathbf{x}) = (P_0, \ldots, P_{q-1})$ for every n-word $\mathbf{x} = (x_1, \ldots, x_n)$; where the right hand side is an arrangement of $0, \ldots, q-1$. Furthermore, the *span* of P is the smallest integer s, $0 \leq s \leq n$, such that $P(x_1, \ldots, x_n)$ is fully determined by $(x_{n-s+1}, \ldots, x_n)$, for all n-words $(x_1, \ldots, x_n)$.

The following recursive construction produces a unique finite binary sequence $\{a_i\}$, provided that a preference function of span s and an arbitrary initial n-word $(I_1, \cdots, I_n)$ with $n > s$ are given. We denote the unique resulting sequence by (P, I).

1. For $i = 1, \cdots, n$ let $a_i = I_i$.

2. Suppose that $a_1, \cdots, a_k$ for some integer $k \geq n$ have been defined. Let $a_{k+1} = P_i(a_{k-s+1}, \ldots, a_k)$ where i, $0 \leq i \leq q-1$ is the smallest integer such that $(a_{k-n+2}, \ldots, a_{k+1})$ has not appeared in the sequence as a substring, if such an i exists.

3. If no such i exists, halt the program (the construction is complete.)

The following lemma is a slight generalization of Lemma 2 of Chapter 3 in Golomb [8]. The proof is essentially the same.

Lemma 2.2. *Consider an arbitrary preference function P of span $s \geq 0$ and initial word $I = (I_1, \cdots, I_n)$; $n > s$. Then every n-word occurs at most once in (P, I). Furthermore, (P, I) ends with the pattern $(I_1, \cdots, I_{n-1})$.*

It follows that the sequence (P, I) can be identified with a cyclic string, by removing the last pattern $(I_1, \ldots, I_{n-1})$ and wrapping the rest around a circle. A preference function P is said to be *complete* if there exists an initial word I such that (P, I) is a de Bruijn sequence.

Definition 2.3. For an integer i such that $0 \leq i < q$, the i^{th} column function induced by P is a function from $\mathbf{A}^s$ to $\mathbf{A}^s$ defined as

$$g_i(x_1, \ldots, x_s) = (x_2, \ldots, x_s, P_i(x_1, \ldots, x_s)).$$

Clearly, g_i defines at least one cycle of length $k \geq 1$. That is, a sequence of k s-words $v_1, \ldots, v_k$ in $\mathbf{A}^s$ such that $g_i(v_j) = v_{j+1}$ for $j = 1, \ldots, k-1$ and $g_i(v_k) = v_1$.

Theorem 3.1 and Corollary 3.2 in Alhakim [4] provide a characterization of complete preference functions, along with legitimate initial words I. We will refer to these initial words as de Bruijn seeds. Briefly, in a complete preference function, the column function g_{q-1} must have exactly one cycle. Also a de Bruijn seed $(I_1, \ldots, I_n)$ must be such that $(I_1, \ldots, I_{n-1})$ is a path on g_{q-1}. For example, the corresponding cycles of the complete preference functions of Table 1 are respectively $2 \to 2$, $0 \to 1 \to 0$, $0 \to 1 \to 2 \to 0$ and $00 \to 01 \to 10 \to 00$. The first cycle means that $2 \cdots 20 = 2^{n-1}0$, $2 \cdots 21 = 2^{n-1}1$ and 2^n are all

0	→	0,	1,	2	0	→	2,	0,	1	0	→	0,	2,	1
1	→	0,	1,	2	1	→	1,	2,	0	1	→	0,	1,	2
2	→	0,	1,	2	2	→	0,	2,	1	2	→	2,	1,	0
00	→	0,	2,	1	10	→	1,	2,	0	20	→	1,	2,	0
01	→	1,	2,	0	11	→	0,	1,	2	21	→	0,	2,	1
02	→	2,	1,	0	12	→	1,	2,	0	22	→	0,	2,	1

TABLE 1. Top: complete preference diagrams of span 0 (left), and span 1 (middle and right). Bottom: *one* complete preference diagram of span 2.

de Bruijn seeds of length $n > 1$. Likewise, 01,010, 0101 and 01010 are de Bruijn seeds of various lengths for the second preference function, while 01201201 and $001100110\cdots$ are examples of seeds for the last two preference functions. Proofs and more details are given in [4].

3. The minimality of a de Bruijn sequence

We begin this section with the following definition.

Definition 3.1. Let P be a preference function of span s and $n \geq s$. Then

(a) P defines an operator $T_P^{(n)}$ that acts on arbitrary sequence $S = d_1 \dots d_l$ of length $l > s$ as $T_P^{(n)}(S) = d_1 \dots d_n | c_1 \dots c_{l-n}$, where for $i = 1, \dots, l-n$ $d_{n+i} = P_{c_i}(d_{n+i-s} \dots d_{n+i-1})$.

We refer to the the first n digits as the leading digits, and to the digits c_i as the preference trail digits, or simply the trail digits of S.

(b) We define the P-lex order, denoted $\prec_P$, as the total order on the set of sequences: for two sequences $S_1 = d_1 \dots d_l$ and $S_2 = d'_1 \dots d'_m$, $S_1 \prec_P S_2$ if and only if $c_1 \dots c_{l-s}$ is lexicographically smaller than $c'_1 \dots c'_{m-s}$, where $c_1 \dots c_{l-s}$ and $c'_1 \dots c'_{m-s}$ are resp. the trail sequences of S_1 and S_2 without the leading digits.

As an example, using the matrices in Table 1, the base 3 sequence 012210 is encoded using $n = s$ as $|012210$, $0|21122$, $0|22010$ and $01|1120$ respectively. Observe that the first preference function has no leading digits and it outputs the same input sequence. It is also evident that, for all preference functions, the initial sequence can be recovered uniquely by tracing the corresponding matrix, thanks to the leading digits. Another obvious but important observation is that the same sequence can have various lexicographical orders depending on the underlying function P.

In order to compare two de Bruijn sequences using the P-lex order, we will exclude the trail digits within the initial words and compare the trail of the $d_{n+1} \dots d_{q^n}$.

Theorem 3.2. *Let P be a complete preference function of span s and $I = d_1 \dots d_n$, $n > s$ be a de Bruijn seed such that $T_P^{(s)}(I) = d_1 \dots d_s | (q-1)^{n-s}$. For an arbitrary de Bruijn sequence dB'_n that is not rotationally equivalent to dB_n we have $dB_n \prec_P dB'_n$ where $dB_n = (P, I)$.*

For convenience of notation, we will denote $T_P^{(n)}(dB'_n) = d'_1 \dots d'_n | c'_1 \dots c'_{q^n}$ for any de Bruijn sequence. That is, we apply $T_P^{(n)}$ to a version of dB'_n that begins and ends with $d'_1 \dots d'_n$. In the case of dB_n, this amounts to appending the trail digits $(q-1)^s$ at the end of the sequence, which obviously has no effect on the P-lex order of dB_n.

Proof. Denote dB_n and dB'_n respectively by $d_1 \dots d_{q^n}$ and $d'_1 \dots d'_{q^n}$ and let

$T_P^{(n)}(dB_n) = d_1 \dots d_n | c_1 \dots c_{q^n}$ and $T_P^{(n)}(dB'_n) = d'_1 \dots d'_n | c'_1 \dots c'_{q^n}$.

We begin by establishing the inequality when $d'_1 \dots d'_n = d_1 \dots d_n$. Suppose, for a contradiction, that $dB'_n \prec_P dB_n$. Then there exists a minimal $i \geq 1$ such that $c'_i < c_i$. Since dB_n follows the preference strategy of P, the pattern $c_{i-n+s+1} \dots c_{i-1} c'_i$ must have appeared earlier, preceded by the same s leading digits $d_{i-n+1} \dots d_{i-n+s}$. By the minimality of i, all trail digits of dB_n and dB'_n are identical up to $i-1$. Thus by the assumption that $d'_1 \dots d'_n = d_1 \dots d_n = I$, dB'_n includes a repeated n-word, contradicting the fact that it is a de Bruijn sequence.

We will now tackle the case when $d'_1 \dots d'_n \neq I$, that is, when dB' is rotated to start at any word other than I. We do this in two steps. First, consider the sequence $S = (P, d'_1 \dots d'_n)$ whose trail sequence is

$d'_1 \dots d'_n | b_1 \dots b_l$, and which may or may not be a de Bruijn sequence. Since both S and dB'_n begin with the same initial word, the same argument as above establishes that $S \prec_P dB'_n$. Furthermore, since S follows the preference strategy of P, any digit placed after the terminal trail digit b_l leads to a repetition. So the first disagreement ($b_i < c'_i$) occurs at some $i < l$ or else dB'_n cannot be continued into a de Bruijn sequence.

Next we compare S and dB_n. Since $d'_1 \dots d'_n \neq I$, it is not of the form $d_1 \dots d_s | (q-1)^{n-s}$. Thus, one or more of the *wrap-around* words $d'_1 \dots d'_n$, $d'_2 \dots d'_{n+1}, \dots, d'_n \dots d'_{2n-1}$ is not a wrap-around word of dB_n, and therefore an internal word of the latter. Let j be the smallest index where a wrap-around word w of S is encountered for a second time upon proposing a trail digit $b_j = c$, it is avoided in S by either using higher preference $b_j > c$ if possible, or else S is stopped at b_{j-1} (i.e., $l = j-1$). In dB_n however, w is not a wrap-around word so that $c_j = c$.

Clearly, if $j < l$ then $dB_n \prec_P S$, and by the earlier proof above $S \prec_P dB'_n$, which shows that $dB_n \prec_P dB'_n$. If $j - 1 = l$ then $S \prec_P dB_n$, as $b_1 \dots b_l = c_1 \dots c_l$ and $l < q^n$. However, we proved above that the first disagreement between S and dB'_n occurs at $i < l$, implying that $c_i = b_i < c'_i$. This completes the proof. □

All complete binary preference functions of span 1 are represented by the matrices

$$F = \begin{bmatrix} 1 & 0 \\ 1 & 0 \end{bmatrix}; O = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}; S = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}; \quad \text{and } Z = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}.$$

Observe that The Ford sequence, or prefer-one sequence is $(F, 0^n)$, first attributed to Martin [10], and $(Z, 1^n)$ is clearly its bitwise complement. $(O, 0^n)$ is the prefer-opposite sequence, see Alhakim [1], while $(S, 010\cdots)$ is the prefer-same sequence, where $010\cdots$ is the alternating string of length n. Alhakim *et. al.* [3] shows that the last two sequences respectively have the lexicographically smallest and largest representation in *run length encoding*. These results follow almost immediately from Theorem 3.2, the proofs are omitted for brevity.

4. Factoring Into Lyndon words

Recall that a Lyndon word is a finite word that is smaller than all of its rotations. For example, 0012 and 0021 are Lyndon words of size 4 but 0101 is not because it is equal to one of its rotations. Note that single symbols are Lyndon words. It is well known that the lexicographically least de Bruijn sequence is a concatenation of Lyndon words of lengths diving n and arranged in increasing lexicographical order. In this section we present a Lyndon decomposition of the trail sequence of the prefer-opposite sequence $\mathbf{o}_n = (O, I = 0^n)$, where the preference function O is given at the end of the previous section.

For a Lyndon word η, we define the *weight* $w(\eta)$ to be the number of *zeros* in η. The following theorem states that the preference trail of the prefer-opposite sequence is a concatenation of words that are essentially Lyndon words except for few exceptions, depending on the size n, that are well-defined rotations of Lyndon words. Let $\bar{n} = n - 1$ and $L(\bar{n})$ be the set of all Lyndon words with a length that divides $\bar{n}$. Recall that η^2 is a concatenation of two copies of the word η.

Theorem 4.1. *The trail sequence part of $T_O^{(n)}(\mathbf{o}_n)$ is a concatenation of all Lyndon words in $L(\bar{n})$, such that each word appears twice, starting with two consecutive* 0*, with the other words appended inductively as follows. Suppose η_0 has just been appended and let $\eta = \tau 01^j$ be the lexicographically next word in $L(\bar{n})$, where $\tau = c_1 \dots c_{\lambda-j-1}$ and λ is the length of η. Let $w = w(\eta) \mod 2$. Then*

(1) If either $w = j = 1$ or $w = 1$, $j = 2$ and $\tau = 0^{\lambda-j-1}$ then append η^2.

(2) If $w = 1$, $j > 2$ and $\tau = 0^{\lambda-j-1}$ then append $\eta \cdot \tilde{\eta}_1 \dots \tilde{\eta}_{j-2} \cdot \eta$ where $\tilde{\eta}_1 = 0^{\lambda-j}101^{j-2}$, $\tilde{\eta}_2 = 0^{\lambda-j}1101^{j-3}, \dots, \tilde{\eta}_{j-2} = 0^{\lambda-j}1^{j-2}01$.

(3) If $w = 1$, $j > 1$ and $\tau \neq 0^{\lambda-j-1}$ then if $\tilde{\eta}_1$ is a Lyndon word append $\eta \cdot \tilde{\eta}_1 \dots \tilde{\eta}_{j-1} \cdot \eta$ otherwise (if $\tilde{\eta}_1$ is not a Lyndon word) append η^2, where $\tilde{\eta}_1 = \tau 001^{j-1}$, $\tilde{\eta}_2 = \tau 0101^{j-2}, \dots, \tilde{\eta}_{j-1} = \tau 01^{j-2}01$.

(4) If $w = 0$ and $\tau \neq 0^{\lambda-j-1}$ then append η.

(5) If $w = 0$ and $\tau = 0^{\lambda-j-1}$ then append $\eta_1 \star \eta_2 \star \cdots \star \eta_{j+1}$, where $\eta_1 = \eta = 0^{\lambda-j}1^j$, $\eta_2 = 0^{\lambda-j-1}101^{j-1}$, $\eta_3 = 0^{\lambda-j-1}1101^{j-2}, \dots, \eta_j = 0^{\lambda-j-1}1^{j-2}01$, $\eta_{j+1} = 0^{\lambda-j-1}1^j0$. The stars ($\star$) indicate segments of the sequence that contain the possible Lyndon words which are lexicographically ordered between η_i and η_{i+1}, arranged according to (1)-(4).

We will give a proof of this in the extended paper, due to the lack of space. We also omit a similar factorization theorem for the prefer-same sequence and only present some examples. We first list the factorization of the prefer-opposite sequence for orders 4 to 7. the Lyndon words are separated by one dot, blocks of types (1)-(4) are separated by an asterik ($*$), while words of type (5) are in bold. Note that there is a missing 1 relating to the missing word 1^n in $(O, 0^n)$.

$n = 4:\ 0000|0 \cdot 0 * \mathbf{001} \cdot \mathbf{010} * 011 \cdot 011 * 1$

$n = 5:\ 00000|0 \cdot 0 * 0001 \cdot 0001 * \mathbf{0011} \cdot 01 \cdot 01 * \mathbf{0110} * 0111 \cdot 0111 * 1$

$n = 6:\ 000000|0 \cdot 0 * \mathbf{00001} \cdot \mathbf{00010} * 00011 \cdot 00011 * 00101 \cdot 00101 * \mathbf{00111} * \mathbf{01011} * \mathbf{01101} * \mathbf{01110} * 01111 \cdot 01111 * 1$

$n = 7:\ 0000000|0 \cdot 0 * 000001 \cdot 000001 * \mathbf{000011} * \mathbf{000101} * \mathbf{000110} * 000111 \cdot 000101 \cdot 000111 * 001*$
$*001011 \cdot 001 \cdot 001011 * 001101 \cdot 001101 * \mathbf{001111} * 01 \cdot 01 * \mathbf{010111} * 011 \cdot 011 * \mathbf{011101} * \mathbf{011110} * 011111 \cdot 011111 * 1$

The following is a factorization of trail sequences of the prefer-same sequence with $n = 4$ and 5.

$n = 4:\ 0101|0 * 001 \cdot 0 \cdot 001 * 011 * 1 \cdot 011 * 1$

$n = 5:\ 01010|0 * 0001 \cdot 0 \cdot 0001 * 0011 * 01 \cdot 01 * 0111\dot{0}011 \cdot 0111 * 1$

Finally, letting P be the preference function of span 2 given in Table 1, which was arbitrarily chosen, we give a factorization of $(P, 0010)$. Note that Lyndon words of sizes 1 and 2, that divide $n - 2$ are each repeated $3^2 = 9$ times. Also note that there is only one *rotated* Lyndon word (underlined).

$n = 4, q = 3:\ 0010|0 \cdot 0 \cdot 0 \cdot 01 \cdot 0 \cdot 0 \cdot 02 \cdot 1 \cdot 0 \cdot 0 \cdot 0 \cdot 01 \cdot 01 \cdot \underline{10} \cdot 12 \cdot 01 \cdot 1 \cdot 01 \cdot 01 \cdot 02 \cdot 0 \cdot 01 \cdot 02 \cdot 02 \cdot 02 \cdot 02 \cdot 01 \cdot 02 \cdot 02 \cdot 1 \cdot 1 \cdot 1 \cdot 1 \cdot 1 \cdot 12 \cdot 12 \cdot 02 \cdot 2 \cdot 2 \cdot 12 \cdot 1 \cdot 12 \cdot 1 \cdot 12 \cdot 12 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 12 \cdot 12 \cdot 2 \cdot 2 \cdot 2$

5. Discussion and Conclusion

We introduced a transform that maps a de Bruijn sequence (or any sequence) to a trail sequence using an arbitrary but fixed preference function. Observe that the prefer-zero sequence $(Z, 1^n)$ is identical to its trail sequence when the leading digits 1^n are not considered. It is a concatenation of Lyndon words that appear one time each. In this paper we have presented binary de Bruijn sequences of span 1 whose trail sequence is a concatenation of Lyndon words, appearing twice each. More numerical experimentation strongly suggest that the trail sequence of any q-ary de Bruijn sequence generated by a preference function of span s is equally a concatenation of Lyndon words that divide $n - s$, and each appearing q^s times, in a way that if η_1 is less than η_2 then the first appearance of η_1 occurs before the first appearance of η_2. This is a subject of further research.

References

[1] A. Alhakim. A Simple Combinatorial Algorithm for de Bruijn Sequences. The American Mathematical Monthly, **117**, Number 8, (2010) 728-732.

[2] A. Alhakim. Spans of preference functions. Discrete Applied Mathematics, Vol. 160, 7-8, (2012) 992-998.

[3] A. Alhakim, E. Sala, J. Sawada. Revisiting the prefer-same and prefer-opposite de Bruijn Sequence Constructions. Theoretical Computer Science, (2020).

[4] A. Alhakim, Designing preference functions for de Bruijn sequences with forbidden words. Des. Codes Cryptogr. 90, 2319-2335, (2022).

[5] C. Eldert, H. J. Gray, H. M. Gurk, M. Rubinoff. Shifting Counters. AIEE Trans., 77, (1958), 70-74.

[6] L. R. Ford, A Cyclic Arrangement of m-tuples, Report P-1071, Rand Corp., 1957.

[7] H. Fredricksen, A Survey of Full Length Nonlinear Shift Register Cycle Algorithms, *SIAM Review*, **24** (1982) 195-221.

[8] S. Golomb, *Shift Register Sequences*, Holden-Day, San Francisco, 1967.

[9] D. Knuth, *The Art of Computer Programming*, vol. 4, Addison Wesley, 2011.

[10] M. H. Martin, A Problem in Arrangements, *Bulletin of the American Mathematical Society*, **40** (1934) 859-864.

[11] Evan Sala, E., Sawada, J., Alhakim, A. Efficient constructions of the Prefer-same and Prefer-opposite de Bruijn sequences. Submitted Manuscript.

E-mail address: aa145@aub.edu.lb

A Nonlinear Mapping Based on Squaring

Denise Verbakel[1], Daniel Kuijsters[1], Silvia Mella[1], Stjepan Picek[1], Luca Mariot[2] and Joan Daemen[1]

[1] Radboud University, Digital Security Department, Nijmegen, The Netherlands
[2] Semantics, Cybersecurity and Services Group, University of Twente, The Netherlands

Abstract. Many modern symmetric cryptographic primitives operate in an iterated way: they consist of the repeated application of a relatively simple round function over a state, alternated with the addition of secret round keys or round constants. A crucial component of the round function is the nonlinear layer, usually defined via an invertible map. However, many modes of operations do not require invertibility of the underlying primitive and recently Grassi proposed the usage of non-invertible nonlinear mappings in MPC-/FHE-/ZK-friendly symmetric cryptographic primitives. In this work, we consider one of these maps. It is a simple yet efficient nonlinear map, that we call γ, based on squaring over $\mathbb{F}_q$, with q an odd prime power. We discuss for the first time the differential and linear propagation properties of such a nonlinear map and observe that they follow the same rules. This is an intriguing property that, as far as we know, only occurs with γ and the binary mapping χ_3 used in Xoodoo.

Keywords: Nonlinear layer, Squaring, Finite fields

1 Introduction

The round functions in most of the modern symmetric cryptographic primitives usually consist of a non-linear mapping and a number of linear mappings. These mappings are chosen and combined so that there is no exploitable differential propagation from input to output or exploitable correlations between input and output. The relevant properties of these mappings over binary fields have been studied extensively by an expert community of mathematicians, leading to solid designs. But, this community does not stop at the binary case and also studies similar functions over $\mathbb{F}_p$ and its extensions, with p an odd prime. For instance, Kölbl et al. designed a ternary cryptographic hash function called Troika [KTDB19]. Other examples are the MPC-/FHE-/ZK-friendly symmetric primitives defined over $\mathbb{F}_p^n$ like MiMC [AGR+], Poseidon [GKR+21], and many others.

There are interesting differences between the binary case and the odd-prime case, and to a certain extent, the fields of odd characteristics are richer in functionality than binary fields. For example, addition and subtraction are the same in $\mathbb{F}_2$. In $\mathbb{F}_p$, this is no longer the case. In $\mathbb{F}_{2^d}$, squaring is a linear operation. In $\mathbb{F}_{p^d}$ squaring is, in a certain sense, an optimally nonlinear operation. In $\mathbb{F}_2$, correlations between input and output bits have values that are rational and range from -1 to $+1$. In $\mathbb{F}_p$, correlations are complex numbers in the unit disk.

This work investigates a mapping over $\mathbb{F}_q^n$ recently proposed by Grassi [Gra22], that we call γ. We investigate the differential and linear propagation properties of such mapping, both in the forward and backward direction. Our results are useful in determining the maximum probabilities of differentials and trails and correlations of linear approximations and trails over transformations making use of this mapping in their round function, as in computer-assisted trail search [DA].

2 Preliminaries

Let $\mathbb{F}_q$ be a finite field with $q = p^d$ an odd prime power. Let $\mathbb{F}_q^n$ be a vector space of dimension n over the finite field $\mathbb{F}_q$. We denote the coordinates of a vector $x \in \mathbb{F}_q^n$ by x_i with $i \in \{0, 1, \dots n-1\}$ and call them *digits*. We denote by e_i the vector with all coordinates equal to 0 except coordinate i equal to 1. The Hamming weight $\mathrm{HW}(x)$ of a vector $x \in \mathbb{F}_q^n$ is the number of non-zero digits in the vector.

Given two vectors $x, y \in \mathbb{F}_q^n$, we denote their vector subtraction by $x - y$, hence $x - y = x + (-1)y$. We denote by $x^{\mathrm{T}}y$ the value $\sum_i x_i y_i \in \mathbb{F}_q$.

Given a vector $x \in \mathbb{F}_q^n$ its activity pattern $\widetilde{x}$ is a vector in $\mathbb{F}_q^n$ with $\widetilde{x}_i = 1$ if $x_i \neq 0$ and 0 otherwise.

3 Our non-linear mapping γ

In this work, we consider a mapping defined in [Gra22] that we will denote by $\gamma : \mathbb{F}_q^n \to \mathbb{F}_q^n$ as

$$\gamma(x) = y \text{ with } y_i = x_i + x_{i+1 \bmod n}^2 \forall i.$$

From now on, we will omit the modular reduction in the index and always assume it is reduced modulo n.

4 Differential properties of γ

We analyzed the differential properties of the map γ. We will first define differential probability and weight for the non-binary case and then summarize our findings for γ.

4.1 Differentials, differential probability and weight

Let $x \in \mathbb{F}_q^n$ and $x^* \in \mathbb{F}_q^n$ be inputs of a transformation $\alpha : \mathbb{F}_q^n \to \mathbb{F}_q^n$ and let their difference be $a = x^* - x$. Likewise, let $y \in \mathbb{F}_q^n$ and $y^* \in \mathbb{F}_q^n$ be outputs of α and let their difference be $b = y^* - y$. The (ordered) pair $(a, b) \in \mathbb{F}_q^n \times \mathbb{F}_q^n$ containing the input and output difference is called a *differential over* α.

The *differential probability (DP)* of a differential (a, b) over the transformation α is defined as

$$\mathrm{DP}_\alpha(a, b) = \frac{\left|x \in \mathbb{F}_q^n \colon \alpha(x + a) - \alpha(x) = b\right|}{q^n}.$$

If $\mathrm{DP}_\alpha(a, b) > 0$, we say that a and b are *compatible* differences over α. We define the weight of a differential (a, b) over α with a and b compatible as:

$$\mathrm{w}_\alpha(a, b) = -\log_q(\mathrm{DP}_\alpha(a, b)).$$

4.2 Forward propagation from a given input difference

Consider the function $\beta \colon \mathbb{F}_q \to \mathbb{F}_q : x \mapsto x^2$. Given an input pair $(x + a, x)$, the corresponding output difference b is given by

$$b = (x + a)^2 - x^2 = x^2 + 2ax + a^2 - x^2 = 2ax + a^2. \tag{1}$$

This is a linear equation and for any output difference $b \in \mathbb{F}_q$ there is exactly one input pair $(x + a, x)$. Solving (2) gives $x = (2a)^{-1}(b - a^2)$ yielding the pair

$$\left(\frac{b}{2a} + \frac{a}{2}, \frac{b}{2a} - \frac{a}{2}\right).$$

It follows that the set of output differences b compatible over β with a non-zero input difference a coincides with $\mathbb{F}_q$ and they all have $\mathrm{DP}_\beta(a,b) = q^{-1}$.

For the map γ, we have

$$b_i = x_i + a_i + (x_{i+1} + a_{i+1})^2 - x_i - x_{i+1}^2 = a_i + a_{i+1}^2 + 2a_{i+1}x_{i+1}\,, \tag{2}$$

From (2) we can characterize the full difference distribution table (DDT) of γ.

Lemma 1. *An output difference b is compatible to an input difference a over γ if for every i, $b_i = a_i$ or $a_{i+1} \neq 0$, and, if so, $\mathrm{DP}(a,b) = q^{-\mathrm{HW}(a)}$.*

Therefore, for an input difference $a \in \mathbb{F}_q^n$, the compatible output differences over γ form an affine space with dimension $\mathrm{HW}(a)$. The offset and a basis with minimal Hamming weight for such affine space is given by:

- the i-th digit of the offset is equal to a_i if $a_{i+1} \neq 0$ and 0 otherwise;
- for each non-zero digit in the input difference a, the basis contains the vector e_{i-1}.

It follows that for all b compatible with an input difference a we have $\mathrm{DP}_\gamma(a,b) = q^{-\mathrm{HW}(a)}$ and likewise $\mathrm{w}_\gamma(a,b) = \mathrm{HW}(a)$ and therefore only depends on the input difference.

4.3 Backward propagation from a given output difference

For a given output difference b, the compatible input differences do not form an affine space. However, we will show in this section how to efficiently generate all compatible input differences a with $\mathrm{DP}_\gamma(a,b) \leq W$ with W some limit weight.

To this end, we introduce the concept of compatible activity pattern. We say that an activity pattern $\widetilde{a}$ is compatible with b if there exists an input difference a compatible with b that has activity pattern $\widetilde{a}$.

The generation of all compatible input differences is done in two phases: in the first phase, we generate the set of activity patterns compatible with b, and in the second phase, we determine for each compatible activity pattern the set of compatible input differences with that pattern.

We generate the compatible activity patterns in a recursive way making use of the following facts:

- if $a_i = 0$ and $b_{i-1} = 0$ then $a_{i-1} = 0$;
- if $a_i = 0$ and $b_{i-1} \neq 0$ then $a_{i-1} \neq 0$.

We specify our algorithm in Algorithm 1. We start with a fully unspecified activity pattern k. Then we specify whether a_{n-1} is active or not (and thus whether $k_{n-1} = 1$ or 0) and based on this we incrementally determine the activity of all other digits from a_{n-2} to a_0 using the rules given above.

Given an output difference b and a compatible input activity pattern k, all compatible input differences a with activity pattern k can be determined as follows:

- if $k_i = 0$, then $a_i = 0$;
- if $k_i = 1$ and $k_{i+1} = 0$, then $a_i = b_i$;
- if $k_i = 1$ and $k_{i+1} = 1$, then a_i can have all values.

The differentials (a,b) with given output difference b and input differences a compatible with b do not all have the same weight. We define the *minimum reverse weight* of an output difference b as:

$$\mathrm{w}_\gamma^{\mathrm{rev}}(b) = \min_{a\colon \mathrm{DP}_\gamma(a,b)>0} \mathrm{w}_\gamma(a,b)\,.$$

4.4 Computing the minimum reverse weight of an output difference

The minimum reverse weight of an output difference b is fully determined by its activity vector $\widetilde{b}$ and is given by the compatible activity patterns with minimum Hamming weight.

Algorithm 1 Generation of input activity patterns compatible with output difference b

Input: difference $b \in \mathbb{F}_q^n$ at output of γ and limit weight W
Output: list L of activity patterns k compatible with b at input of γ
Coordinates in k: $*$ denotes unspecified, 0 denotes passive, 1 denotes active

$L \leftarrow$ empty
$k \leftarrow *^n$
$k_{n-1} \leftarrow 0$; buildA$(n-1, k, b, W)$
$k_{n-1} \leftarrow 1$; buildA$(n-1, k, b, W)$

procedure buildA(i, k, b, W)
  if $\mathrm{HW}(k) > W$ **then return**
  if $(i = 0)$ **then**
    if $(k_{n-1} = 1)$ OR $(\widetilde{b}_0 = k_0)$ **then** add k to L
    return
  $k' \leftarrow k$
  if $(k_i = 1)$ OR $(\widetilde{b}_{i-1} = 1)$ **then** $k'_{i-1} \leftarrow 1$; buildA$(i-1, k', b, W)$
  if $(k_i = 1)$ OR $(\widetilde{b}_{i-1} = 0)$ **then** $k'_{i-1} \leftarrow 0$; buildA$(i-1, k', b, W)$
  return

Let a 1-run of length ℓ in $\widetilde{b}$ be a sequence of ℓ coordinates $b_i, b_{i+1}, \cdots, b_{i+\ell-1}$ with activity 1 and such that $b_{i-1} = 0 = b_{i+\ell}$ (where indexes are considered modulo n). Namely, the sequence is preceded by at least one coordinate 0 and followed by at least one coordinate 0.

We see that for each 1-run of length ℓ in $\widetilde{b}$, the digit $\widetilde{a}_{i+\ell-1}$ must be 1 and in the sequence $\widetilde{a}_i, \widetilde{a}_{i+1}, \cdots, \widetilde{a}_{i+\ell-1}$ there can be at most a single zero digit in between two active digits. It follows that for each 1-run in $\widetilde{b}$ of length ℓ, a has at least $\ell/2$ active digits if ℓ is even and $(\ell+1)/2$ if ℓ is odd. So to determine the minimum reverse weight, we decompose its output activity pattern in a sequence of 1-runs of lengths ℓ_j yielding minimum reverse weight $\sum_j \lceil \ell_j/2 \rceil$.

5 Input-output correlation properties of γ

We analyzed the correlation properties of the map γ. We will first define linear approximations and their correlations and then summarize our findings for γ.

5.1 Linear approximations, correlation and weight

Given a complex number x, we write its complex conjugate as $\overline{x}$. In the following section we will write ω as shorthand for $e^{\frac{2\pi i}{p}}$. We will also make use of the *trace* function $\mathrm{Tr}\colon \mathbb{F}_{p^d} \to \mathbb{F}_p$ as $\mathrm{Tr}(x) = \sum_{i=0}^{d-1} x^{p^i}$.

The *correlation* between two functions $f, g\colon \mathbb{F}_{p^d}^n \to \mathbb{F}_p$ is defined as:

$$\mathrm{C}(f, g) = q^{-n} \sum_{x \in \mathbb{F}_q^n} \omega^{g(x) - f(x)}\,.$$

For correlations of functions $f, g\colon \mathbb{F}_q^n \to \mathbb{F}_q$ we first must project the output from $\mathbb{F}_q$ to $\mathbb{F}_p$. A way to do that in a basis-agnostic way is by using the trace function:

$$\mathrm{C}(\mathrm{Tr}(uf), \mathrm{Tr}(vg)) = q^{-n} \sum_{x \in \mathbb{F}_q^n} \omega^{\mathrm{Tr}(vg(x) - uf(x))}\,.$$

Let α be a transformation : $\mathbb{F}_q^n \to \mathbb{F}_q^n$ with $q = p^d$. We call a pair of masks (u, v), with $u \in \mathbb{F}_q^n$ and $v \in \mathbb{F}_q^n$ a *linear approximation* over α, with u the input mask and v the output mask. The correlation of this linear approximation is the correlation between the functions $\mathrm{Tr}(u^{\mathrm{T}}x)$ and $\mathrm{Tr}(v^{\mathrm{T}}\alpha(x))$:

$$\mathrm{C}_\alpha(u, v) = q^{-n} \sum_{x \in \mathbb{F}_q^n} \omega^{\mathrm{Tr}(v^{\mathrm{T}}\alpha(x) - u^{\mathrm{T}}x)} .$$

If $\mathrm{C}_\alpha(u, v) \neq 0$, we say that masks u and v are *compatible* over α.

Correlations are, in general, complex numbers. The *linear potential (LP)* is real and related to a correlation by $\mathrm{LP}(u, v) = \mathrm{C}(u, v)\overline{\mathrm{C}}(u, v)$.

We define the weight of a linear approximation (u, v) over α with u and v compatible as

$$\mathrm{w}_\alpha(u, v) = -\log_q(\mathrm{LP}_\alpha(u, v)) .$$

5.2 Correlation properties of γ

Consider again the function β: $\mathbb{F}_{p^d} \to \mathbb{F}_{p^d} : x \mapsto x^2$. By applying Theorem 5.33 from [LN97], we obtain that the correlation between $x \mapsto vx^2$ and $x \mapsto ux$ (where $u, v \in \mathbb{F}_q$) is equal to:

$$\mathrm{C}_\beta(u, v) = \frac{1}{q} \sum_{x \in \mathbb{F}_q} \omega^{\mathrm{Tr}(vx^2 - ux)} = \begin{cases} \frac{(-1)^{d-1}}{\sqrt{q}} \omega^{\mathrm{Tr}(-u^2(4v)^{-1})} \eta(v) & \text{if } p \equiv 1 \pmod 4 \\ \frac{(-1)^{d-1}}{\sqrt{q}} i^d \omega^{\mathrm{Tr}(-u^2(4v)^{-1})} \eta(v) & \text{if } p \equiv 3 \pmod 4 \end{cases}$$

with $\eta(v) = 1$ if v is a square in $\mathbb{F}_q$ and -1 otherwise. It follows that for all $u, v \in \mathbb{F}_{p^d}$ and $v \neq 0$ we have $\mathrm{LP}_\beta(u, v) = q^{-1}$.

We can compute the correlation of linear approximations over γ from those over β:

$$\begin{aligned} \mathrm{C}_\gamma(u, v) &= q^{-n} \sum_{x \in \mathbb{F}_q^n} \omega^{\mathrm{Tr}\left(v^{\mathrm{T}}\gamma(x) - u^{\mathrm{T}}x\right)} && (3) \\ &= q^{-n} \sum_{x \in \mathbb{F}_q^n} \omega^{\mathrm{Tr}\left(\sum_i v_i(x_i + x_{i+1}^2) - u_i x_i\right)} && (4) \\ &= q^{-n} \sum_{x \in \mathbb{F}_q^n} \prod_i \omega^{\mathrm{Tr}\left((v_i - u_i)x_i + v_{i-1}x_i^2\right)} && (5) \\ &= \prod_i q^{-1} \sum_{x_i \in \mathbb{F}_q} \omega^{\mathrm{Tr}\left((v_i - u_i)x_i + v_{i-1}x_i^2\right)} && (6) \\ &= \prod_i \mathrm{C}_\beta(v_i - u_i, v_{i-1}) . && (7) \end{aligned}$$

From (3) we can characterize the full table of LPs of γ.

Lemma 2. *An input mask u is compatible to an output mask v over γ if for every i, $u_i = v_i$ or $v_{i-1} \neq 0$, and, if so,* $\mathrm{LP}(u, v) = q^{-\mathrm{HW}(v)}$.

Clearly, Lemma 1 and Lemma 2 are very alike and therefore propagation of differences and masks over γ follow similar laws. Concretely, let $\pi : \mathbb{F}_q^n \to \mathbb{F}_q^n : x \mapsto y$ with $\forall i$: $y_{-i} = x_i$. Then we have

$$\text{for } v = \pi(a), u = \pi(b) : \mathrm{LP}_\gamma(u, v) = \mathrm{DP}_\gamma(a, b) .$$

So masks propagate as differences, taking into account following correspondence:

- output masks play the role of input differences and vice versa;
- indexes shall be reversed: index i in a mask corresponds to index $-i$ in a difference.

For a nonlinear mapping this is an intriguing property that, as far as we know, occurs only in γ and the mapping χ_3 [DHVK18].

It follows that we can extend the results obtained in Section 4 to masks. In particular, for a given output mask, we can build the affine space of compatible input masks as in Section 4.2. Moreover, for a given input mask, the compatible output masks can be found by applying Algorithm 1. For a given input masks u, there can be several compatible output masks v. Among them, there will be one realizing the minimum value of $\mathrm{w}(u, v)$. The *minimum reverse weight* of u is defined as

$$\mathrm{w}^{\mathrm{rev}}_{\gamma}(u) = \min_{v\colon \mathrm{LP}_{\gamma}(u,v)>0} \mathrm{w}_{\gamma}(u, v)\,.$$

and is determined by the number of 1-runs in u and their weight, as in Section 4.4.

6 Non-invertibility and imbalance

A non-zero input difference a can lead to a zero output difference if 0 is in the affine space of compatible output differences, or equivalently, if its offset is 0. This can only happen if, for all positions, both a_i and a_{i+1} are active. Therefore, the input differences a that can lead to a collision are those with all coordinates active. There are $(q-1)^n$ such differences and for all of them $\mathrm{DP}(a, 0) = q^{-n}$.

Similarly, a non-zero output mask v can only be imbalanced if 0 is in the affine space of compatible input masks, or equivalently, if its offset is 0. This can only happen if, for all positions, both v_i and v_{i+1} are active. Therefore the output masks v that can lead to a collision are those with all coordinates active. There are $(q-1)^n$ such masks and for all of them $\mathrm{LP}(a, 0) = q^{-n}$.

The collision probability of a mapping is the probability that when randomly choosing two different inputs, the outputs collide. A permutation naturally has collision probability 0. A random transformation over $\mathbb{F}_q^n$ has collision probability q^{-n}: the probability that the two chosen inputs have the same image. For γ, the collision probability is the number of colliding pairs divided by the total number of pairs:

$$\frac{(q-1)^n}{\binom{q^n}{2}} = \frac{2(q-1)^n}{q^n(q^n-1)} \approx \frac{2(q-1)^n}{q^{2n}}$$

So the collision probability of γ is that of a random transformation times a factor $2\left(1-\frac{1}{q}\right)^n$.

References

[AGR+] M. R. Albrecht, L. Grassi, C. Rechberger, A. Roy, and T. Tiessen. MiMC: Efficient encryption and cryptographic hashing with minimal multiplicative complexity. Advances in Cryptology - ASIACRYPT 2016.

[DA] J. Daemen and G. Van Assche. Differential propagation analysis of keccak. Fast Software Encryption - 19th International Workshop, FSE 2012.

[DHVK18] J. Daemen, S. Hoffert, G. Van Assche, and R. Van Keer. The design of Xoodoo and Xoofff. *IACR Transactions on Symmetric Cryptology*, 2018(4):1–38, December 2018.

[GKR+21] L. Grassi, D. Khovratovich, C. Rechberger, A. Roy, and M. Schofnegger. Poseidon: A new hash function for zero-knowledge proof systems. 30th USENIX Security Symposium, 2021.

[Gra22] L. Grassi. Bounded surjective quadratic functions over f_p^n for mpc-/zk-/fhe-friendly symmetric primitives. Cryptology ePrint Archive, Paper 2022/1313, 2022. https://eprint.iacr.org/2022/1313.

[KTDB19] S. Kölbl, E. Tischhauser, P. Derbez, and A. Bogdanov. Troika: a ternary cryptographic hash function. *Designs, Codes and Cryptography*, 88(1):91–117, August 2019.

[LN97] Rudolf Lidl and Harald Niederreiter. *Finite Fields*. Cambridge University Press, 1997.

On quadratic APN functions $F(x) + \mathrm{Tr}(x)L(x)$

Hiroaki Taniguchi[1*]

[1*]Department of Education, Yamato University, 2-5-1, Katayamacho, Suita City, 564-0082, Japan.

Corresponding author(s). E-mail(s): taniguchi.hiroaki@yamato-u.ac.jp;

Abstract

We first characterize how two $(n-1, m)$ functions f and g can be combined into an APN (n, m)-function F of the form $F(x) = f(x)$ and $F(x + e_0) = g(x)$ for $x \in \mathbb{F}_2^{n-1}$ with $e_0 \in \mathbb{F}_2^n \backslash \mathbb{F}_2^{n-1}$. Next we specialize this cahracterization to the case when f is quadratic and $g(x) = f(x)+L(x)$ for some linearized polynomial L. Lastly for a qudratic APN (n, n)-function F and a linearized polynomial L, we give a characterization of APN-ness for (n, n)-function $F(x) + \mathrm{Tr}(x)L(x)$. With some computational experiments, we see that CCZ-inequivalent APN functions $F(x)+\mathrm{Tr}(x)L(x)$ can be obtained from F using this construction.

1 Preliminaries

Let $\mathbb{F}_2$ be the binary field, and n, m positive integers. A function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ is called an almost perfect nonlinear (APN) function if the cardinality $|\{x \mid F(x+a)+F(x) = b\}|$ is less than or equal to 2 for any nonzero $a \in \mathbb{F}_2^n$ and for any $b \in \mathbb{F}_2^m$. APN functions have been studied for many years because of their applications in cryptography. See [1], [2] or [5] for known APN functions. We call a function F *quadratic* if $F(x + y) + F(x) + F(y) + F(0)$ is $\mathbb{F}_2$-bilinear. Two functions F_1 and F_2 from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$ are called *CCZ-equivalent* if the graphs $G_{F_1} := \{(x, F_1(x)) \mid x \in \mathbb{F}_2^n\}$ and $G_{F_2} := \{(x, F_2(x)) \mid x \in \mathbb{F}_2^n\}$ in $\mathbb{F}_2^n \oplus \mathbb{F}_2^m$ are affine equivalent, that is, if there exists an $\mathbb{F}_2$-linear isomorphism $l \in GL_2(\mathbb{F}_2^n \oplus \mathbb{F}_2^m)$ and an element $v \in \mathbb{F}_2^n \oplus \mathbb{F}_2^m$ such that $l(G_{F_1}) + v = G_{F_2}$. *The* Γ*-rank* of a function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ is the rank of the incidence matrix over

$\mathbb{F}_2$ of the incidence structure $\{\mathcal{P}, \mathcal{B}, I\}$, where $\mathcal{P} = \mathbb{F}_2^n \oplus \mathbb{F}_2^m$, $\mathcal{B} = \mathbb{F}_2^n \oplus \mathbb{F}_2^m$ and $(a,b)I(u,v)$ for $(a,b) \in \mathcal{P}$ and $(u,v) \in \mathcal{B}$ if and only if $F(a+u) = b+v$. We know that if two functions F_1 and F_2 from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$ are CCZ-equivalent, then they have the same Γ-rank (see [3]). Let $\mathbb{F}_{2^n}$ be the finite field of 2^n elements. We sometimes identify $\mathbb{F}_{2^n}$ with $\mathbb{F}_2^n$ as an $\mathbb{F}_2$-vector space. We denote the set $\mathbb{F}_{2^n}\backslash\{0\}$ by $\mathbb{F}_{2^n}^\times$ and $\mathbb{F}_2^n\backslash\{0\}$ by $(\mathbb{F}_2^n)^\times$. For finite fields $K \supset F$ of characteristic 2, we denote the trace function from K to F by Tr_F^K. We denote $\mathrm{Tr}_{\mathbb{F}_2}^K$ by Tr and call it the absolute trace of K.

For a function F on $\mathbb{F}_{2^n}$, the value at $a \in \mathbb{F}_{2^n}$ of the Walsh transformation of the Boolean function $\mathbb{F}_{2^n} \ni x \mapsto \mathrm{Tr}(bF(x)) \in \mathbb{F}_2$ for $b \in \mathbb{F}_{2^n}^\times$ is defined by

$$W_F(a,b) = \sum_{x \in \mathbb{F}_{2^n}} (-1)^{\mathrm{Tr}(bF(x)+ax)}.$$

The Walsh spectrum of F is defined by $\mathcal{W}_F = \{W_F(a,b) \mid a \in \mathbb{F}_{2^n}, b \in \mathbb{F}_{2^n}^\times\}$. For a quadratic APN function F on $\mathbb{F}_{2^n}$, it is known that $W_F \in \{0, \pm 2^{(n+1)/2}\}$ if n is odd. For the case n is even, it is said that a quadratic APN function F has the classical Walsh spectrum if $\mathcal{W}_F = \{0, \pm 2^{n/2}, \pm 2^{(n+2)/2}\}$, and F has the non-classical Walsh spectrum if otherwise (see [4]).

2 A condition to have an APN function F from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$ using APN functions f, g from $\mathbb{F}_2^{n-1}$ to $\mathbb{F}_2^m$

Let f, g be functions from $\mathbb{F}_2^{n-1}$ to $\mathbb{F}_2^m$. We regard $\mathbb{F}_2^{n-1} \subset \mathbb{F}_2^n$ as an F_2-linear subspace. Let $e_0 \in \mathbb{F}_2^n$ with $e_0 \notin \mathbb{F}_2^{n-1}$ and $\mathbb{F}_2^{n-1} + e_0 := \{x + e_0 \mid x \in \mathbb{F}_2^{n-1}\}$. Then $\mathbb{F}_2^n = \mathbb{F}_2^{n-1} \cup (\mathbb{F}_2^{n-1} + e_0)$. , We want to have an APN function F from $\mathbb{F}_2^n = \mathbb{F}_2^{n-1} \cup (\mathbb{F}_2^{n-1} + e_0)$ to $\mathbb{F}_2^m$ defined by $F(x) = f(x)$ and $F(x + e_0) = g(x)$ for $x \in \mathbb{F}_2^{n-1}$.

Proposition 1 *F defined above is an APN function if and only if*

(1) *f and g are APN functions from $\mathbb{F}_2^{n-1}$ to $\mathbb{F}_2^m$,*
(2) *$f(x+a) + f(x) \neq g(y+a) + g(y)$ for any $x, y \in \mathbb{F}_2^{n-1}$ and for any nonzero $a \in \mathbb{F}_2^{n-1}$, and*
(3) *$G_a : \mathbb{F}_2^{n-1} \ni x \mapsto f(x+a) + g(x) \in \mathbb{F}_2^m$ are one-to-one mappings for any $a \in \mathbb{F}_2^{n-1}$.*

Proof Recall that F is an APN function if and only if, for any nonzero $A \in \mathbb{F}_2^n$ and for $X, Y \in \mathbb{F}_2^n$, $F(X+A)+F(X) = F(Y+A)+F(Y)$ implies $X = Y$ or $X = Y + A$.

Firstly assume that F is an APN function, and we will see that f and g must satisfy the conditions (1), (2) and (3).

Let $A = a \in (\mathbb{F}_2^{n-1})^\times$. For any $Y = y \in \mathbb{F}_2^{n-1}$, we must have $X = y \in \mathbb{F}_2^{n-1}$ or $X = y + a \in \mathbb{F}_2^{n-1}$ from $F(X+a) + F(X) = F(y+a) + F(y)$. Since $X \in \mathbb{F}_2^{n-1}$, we

have $f(X+a)+f(X) = f(y+a)+f(y)$ from $F(X+a)+F(X) = F(y+a)+F(y)$. Thus f must be an APN function. Next, for any $Y = y+e_0$ with $y \in \mathbb{F}_2^{n-1}$ we must have $X = y+e_0$ or $X = y+a+e_0$ from $F(X+a)+F(X) = F(y+e_0+a)+F(y+e_0)$. Since $X = x+e_0$ for some $x \in \mathbb{F}_2^{n-1}$, we have $g(x+a)+g(x) = g(y+a)+g(y)$ from $F(X+a)+F(X) = F(y+e_0+a)+F(y+e_0)$. Hence g must be an APN function. Thus the condition (1) must be satisfied.

Let $A = a \in (\mathbb{F}_2^{n-1})^\times$. For any $Y = y \in \mathbb{F}_2^{n-1}$, since $X = y$ or $X = y+a$, $F(X+a)+F(X) = F(y+a)+F(y)$ does not have a solution $X = x+e_0$ for $x \in \mathbb{F}_2^{n-1}$. Thus $F(x+e_0+a)+F(x+e_0) \neq F(y+a)+F(y)$ for any $x, y \in \mathbb{F}_2^{n-1}$, therefore we must have $g(x+a)+g(x) \neq f(y+a)+f(y)$ for any $x, y \in \mathbb{F}_2^{n-1}$. Thus the condition (2) must be satisfied.

Let $A = a+e_0$ with $a \in \mathbb{F}_2^{n-1}$ and $Y = y \in \mathbb{F}_2^{n-1}$. We have $X = y \in \mathbb{F}_2^{n-1}$ or $X = y+a+e_0$ with $y+a \in \mathbb{F}_2^{n-1}$. For $X \in \mathbb{F}_2^{n-1}$, we have $g(X+a)+f(X) = g(y+a)+f(y)$ from $F(X+a+e_0)+F(X) = F(y+a+e_0)+F(y)$, hence $g(X+a)+f(X) = g(y+a)+f(y)$ must have only one solution $X = y$ for any $y, a \in \mathbb{F}_2^{n-1}$. For $X \notin \mathbb{F}_2^{n-1}$, we have $f(X+a)+g(X) = g(y+a)+f(y)$ from $F(X+a)+F(X+e_0) = F(y+a+e_0)+F(y)$, hence $f(X+a)+g(X) = g(y+a)+f(y)$ must have only one solution $X = y+a$. Thus we see that the condition (3) must be satisfied.

Conversely, let us assume the conditions (1), (2) and (3). Assume $F(X+A)+F(X) = F(Y+A)+F(Y)$ with $A \neq 0$. We will prove that $X = Y$ or $X = Y+A$. We divide the case into the following four cases (i) $A = a \in (\mathbb{F}_2^{n-1})^\times$ and $Y = y \in \mathbb{F}_2^{n-1}$, (ii) $A = a \in (\mathbb{F}_2^{n-1})^\times$ and $Y = y+e_0$ with $y \in \mathbb{F}_2^{n-1}$, (iii) $A = a+e_0$ with $a \in \mathbb{F}_2^{n-1}$ and $Y = y$ with $y \in \mathbb{F}_2^{n-1}$, and (iv) $A = a+e_0$ with $a \in \mathbb{F}_2^{n-1}$ and $Y = y+e_0$ with $y \in \mathbb{F}_2^{n-1}$.

Firstly let us consider the case (i). If $X = x \in \mathbb{F}_2^{n-1}$, then we have $f(x+a)+f(x) = f(y+a)+f(y)$ hence $x = y$ or $x = y+a$ by (1). Let $X = x+e_0$ with $x \in \mathbb{F}_2^{n-1}$, then we have $g(x+a)+g(x) = f(y+a)+f(y)$ which has no solution by (2). Therefore, $X = Y$ or $X = Y+A$ in case (i).

Next, we consider the case (ii). Assume $X = x \in \mathbb{F}_2^{n-1}$, then we have $f(x+a)+f(x) = g(y+a)+g(y)$ which has no solution by (2). If $X = x+e_0$ with $x \in \mathbb{F}_2^{n-1}$, then we have $g(x+a)+g(x) = g(y+a)+g(y)$, hence $x+e_0 = y+e_0$ or $x+e_0 = y+e_0+a$ by (1). Thus we have $X = Y$ or $X = Y+A$ in case (ii).

Let us consider the case (iii). If $X = x \in \mathbb{F}_2^{n-1}$, then we have $g(x+a)+f(x) = g(y+a)+f(y)$. Since $G_a : x+a \mapsto f(x)+g(x+a)$ is a one-to-one mapping by (3), we have $x = y$. If $X = x+e_0$ with $x \in \mathbb{F}_2^{n-1}$, then we have $f(x+a)+g(x) = g(y+a)+f(y)$. By the same reason as above, we have $x+e_0 = y+(a+e_0)$. Thus we have $X = Y$ or $X = Y+A$ in case (iii).

Lastly we consider the case (iv). If $X = x \in \mathbb{F}_2^{n-1}$, then we have $g(x+a)+f(x) = f(y+a)+g(y)$. Since $G_a : x \mapsto f(x+a)+g(x)$ is a one-to-one mapping by (3), we have $x = (y+e_0)+(a+e_0)$. If $X = x+e_0$ with $x \in \mathbb{F}_2^{n-1}$, then we have $f(x+a)+g(x) = f(y+a)+g(y)$. By the same reason as above, we have $x+e_0 = y+e_0$. Thus we also have $X = Y$ or $X = Y+A$ in case (iv).

Hence F must be an APN function under the conditions (1), (2) and (3). □

3 The case f is a quadratic APN function and $g(x) = f(x) + L'(x)$ with L' a linear mapping

Let f be a function from $\mathbb{F}_2^{n-1}$ to $\mathbb{F}_2^m$ and $B_f(x,a) := f(x+a)+f(x)+f(a)+f(0)$. Recall that f is quadratic if $B_f(x,a)$ is an $\mathbb{F}_2$-bilinear mapping. In this section, we consider the case that f is a quadratic APN function from $\mathbb{F}_2^{n-1}$ to $\mathbb{F}_2^m$, and $g(x) = f(x)+L'(x)$ for $x \in \mathbb{F}_2^{n-1}$ with L' an $\mathbb{F}_2$-linear mappings from $\mathbb{F}_2^{n-1}$ to $\mathbb{F}_2^m$. We note that, if f is quadratic, $\mathbb{F}_2^{n-1} \ni x \mapsto L'(x)+B_f(x,a) \in \mathbb{F}_2^m$ are linear mappings for any $a \in \mathbb{F}_2^{n-1}$. We check the conditions (1), (2) and (3) in Proposition 1. We regard $\mathbb{F}_2^{n-1}$ as an $(n-1)$-dimensional subspace of $\mathbb{F}_2^n$.

Proposition 2 *Let f be a quadratic APN function from $\mathbb{F}_2^{n-1}$ to $\mathbb{F}_2^m$, and $g(x) = f(x)+L'(x)$ with L' an $\mathbb{F}_2$-linear mapping from $\mathbb{F}_2^{n-1}$ to $\mathbb{F}_2^m$. Let F be a function from $\mathbb{F}_2^n$ to $\mathbb{F}_2^m$ defined in Section 2, that is, $F(x) := f(x)$ and $F(x+e_0) := f(x) + L'(x)$ for some fixed $e_0 \in \mathbb{F}_2^n \backslash \mathbb{F}_2^{n-1}$ for $x \in \mathbb{F}_2^{n-1}$. Then F is an APN function if and only if $\mathbb{F}_2^{n-1} \ni x \mapsto L'(x) + B_f(x,a) \in \mathbb{F}_2^m$ are one-to-one mappings for any $a \in \mathbb{F}_2^{n-1}$.*

Proof Since f and $g = f + L'$ are APN functions, the condition (1) is satisfied. The condition (2) implies $f(x+a)+f(x) \neq f(y+a)+f(y)+L'(a)$ for any $x, y \in \mathbb{F}_2^{n-1}$ if $a \neq 0$, that is, $L'(a)+(f(x+a)+f(x))+(f(y+a)+f(y)) \neq 0$ for any $x, y \in \mathbb{F}_2^{n-1}$ if $a \neq 0$, which means $L'(a)+B_f(a, x+y) \neq 0$ if $a \neq 0$, $a \in \mathbb{F}_2^{n-1}$. The condition (3) implies $G_a : \mathbb{F}_2^{n-1} \ni x \mapsto f(x+a)+g(x) = L'(x)+(f(x+a)+f(x)) \in \mathbb{F}_2^m$ are one-to-one mappings for any $a \in \mathbb{F}_2^{n-1}$, that is, $\mathbb{F}_2^{n-1} \ni x \mapsto L'(x)+B_f(x,a)+f(a) \in \mathbb{F}_2^m$ are one-to-one mappings for any $a \in \mathbb{F}_2^{n-1}$. Thus we see that the conditions (1), (2) and (3) in Proposition 1 are satisfied if and only if $\mathbb{F}_2^{n-1} \ni x \mapsto L'(x)+B_f(x,a) \in \mathbb{F}_2^m$ are one-to-one mappings for any $a \in \mathbb{F}_2^{n-1}$. □

4 $F(x) + \mathrm{Tr}(x)L(x)$ for a quadratic APN function F on $\mathbb{F}_{2^n}$

Let $T_0 := \{x \in \mathbb{F}_{2^n} \mid \mathrm{Tr}(x) = 0\}$ and $e_0 \in \mathbb{F}_{2^n}$ with $\mathrm{Tr}(e_0) = 1$. Let F be a quadratic APN function on $\mathbb{F}_{2^n}$ and $B_F(x,a) := F(x+a)+F(x)+F(a)+F(0)$ for $x, a \in \mathbb{F}_{2^n}$. Let L be an $\mathbb{F}_2$-linear mapping on $\mathbb{F}_{2^n}$.

Theorem 3 *Let F be a quadratic APN function on $\mathbb{F}_{2^n}$ and L an $\mathbb{F}_2$-linear mapping on $\mathbb{F}_{2^n}$. Let $e_0 \in \mathbb{F}_{2^n}$ with $\mathrm{Tr}(e_0) = 1$. Then, $F(x) + \mathrm{Tr}(x)L(x)$ is a quadratic APN function on $\mathbb{F}_{2^n}$ if and only if $L_a : T_0 \ni x \mapsto L(x) + B_F(x, a+e_0) \in \mathbb{F}_{2^n}$ are one-to-one mappings from T_0 to $\mathbb{F}_{2^n}$ for any $a \in T_0$. (Hence, $F(x)+\mathrm{Tr}(x)L(x)$ is a quadratic APN function on $\mathbb{F}_{2^n}$ if, and only if, $L_a(x) = 0$ implies $x = 0$ for any $a \in T_0$).*

Proof Let $f := F|_{T_0}$ be the restriction of F to T_0; f is a quadratic APN function from T_0 to $\mathbb{F}_{2^n}$. For $x \in T_0$, we have $F(x) + \mathrm{Tr}(x)L(x) = f(x)$ and $F(x+e_0)+$

$\mathrm{Tr}(x+e_0)L(x+e_0) = f(x) + L(x) + B_F(x, e_0) + L(e_0) + F(e_0)$. Let G be a function on $\mathbb{F}_{2^n}$ defined by $G(x) := f(x)$ and $G(x + e_0) := f(x) + L(x) + B_F(e_0, x)$ for $x \in T_0$, then $G(x) = F(x) + \mathrm{Tr}(x)(L(x) + L(e_0) + F(e_0))$ for $x \in \mathbb{F}_{2^n}$, which is CCZ equivalent to $F(x) + \mathrm{Tr}(x)L(x)$. By Proposition 2, G is an APN function if and only if $T_0 \ni x \mapsto L(x) + B_F(x, e_0) + B_F(x, a) \in \mathbb{F}_{2^n}$ are one-to-one mappings for any $a \in T_0$. Thus $F(x) + \mathrm{Tr}(x)L(x)$ is a quadratic APN function on $\mathbb{F}_{2^n}$ if and only if $L_a : T_0 \ni x \mapsto L(x) + B_F(x, a + e_0) \in \mathbb{F}_{2^n}$ are one-to-one mappings from T_0 to $\mathbb{F}_{2^n}$ for any $a \in T_0$. □

Let e_0 be some fixed element of $\mathbb{F}_{2^n}$ with $\mathrm{Tr}(e_0) = 1$. Using a computer, for linear mappings L on $\mathbb{F}_{2^n}$ such that $L_a : T_0 \ni x \mapsto L(x) + B(x, a + e_0) \in \mathbb{F}_{2^n}$ are one-to-one mappings from T_0 to $\mathbb{F}_{2^n}$ for any $a \in T_0$, we have 448 L's with $L(e_0) = 0$ for $F(x) = x^3$ on $\mathbb{F}_{2^4}$, 4608 L's with $L(e_0) = 0$ for $F(x) = x^3$ on $\mathbb{F}_{2^5}$, and many (about $40,000$) L's with $L(e_0) = 0$ for $F(x) = x^3$ on $\mathbb{F}_{2^6}$.

Example 1 Let $F(x) = x^3$ on $\mathbb{F}_{2^6}$. The Γ-rank of F is 1102. Using a computer, we see that there are linear mappings L satisfying the conditions in Theorem 3 such that the Γ-ranks of $F(x) + \mathrm{Tr}(x)L(x)$ are 1144, 1146, 1158, 1166, 1168, 1170, 1172 and 1174. We also see that $F(x) + \mathrm{Tr}(x)L(x)$ with $L(x) = \alpha^{42}x + \alpha^{19}x^2 + \alpha^{51}x^{2^2} + \alpha^{59}x^{2^3} + \alpha^{26}x^{2^4} + \alpha^{38}x^{2^5}$, where α is a primitive element of $\mathbb{F}_{2^6}$, has non-classical Walsh spectrum $\mathcal{W}_F = \{0, \pm 8, \pm 16, \pm 32\}$ with the Γ-rank 1170. Since $F(x) + \mathrm{Tr}(x)L(x)$ with $L(x) = \alpha^{42}x + \alpha^{47}x^2 + \alpha^{35}x^{2^2} + \alpha^{54}x^{2^3} + \alpha^{23}x^{2^4} + \alpha^{27}x^{2^5}$ has classical Walsh spectrum $\mathcal{W}_F = \{0, \pm 8, \pm 16\}$ with the Γ-rank 1170, we see that there are inequivalent APN functions $F(x) + \mathrm{Tr}(x)L(x)$ with the same Γ-rank.

Let $F(x) = x^3$ on $\mathbb{F}_{2^7}$. The Γ-rank of F is 3610. Using a computer, we find that the linear mapping $L(x) := x + x^{2^3} + x^{2^5} + x^{2^6}$ satisfies the conditions in Theorem 3 and the Γ-rank of $F(x) + \mathrm{Tr}(x)L(x)$ is 4048.

References

[1] M. Calderini, L. Budaghyan and C. Carlet, On known constructions of APN and AB functions and their relation to each other, Proceedings of the 20th Central European Conference on Cryptography, Matematicke znanosti 25, pp. 79–105 (2021).

[2] C. Carlet, Boolean Functions for Cryptography and Coding Theory, Cambridge University Press, Cambridge (2021).

[3] Y. Edel and A. Pott, A new almost perfect nonlinear function which is not quadratic, Advances in Mathematics of Communications 3, pp. 59–81 (2009).

[4] A. Pott, Almost perfect and planar functions, Designs, Codes and Cryptography 78, pp. 141–195 (2016).

[5] https://boolean.h.uib.no/mediawiki/index.php/ .

On the Spread Sets of Planar Dembowski-Ostrom Monomials*

Christof Beierle[1] and Patrick Felke[2]

[1]Faculty of Computer Science, Ruhr University Bochum, Bochum, Germany
[2]University of Applied Sciences Emden-Leer, Emden, Germany

Abstract

Let $g \in \mathbb{F}_{p^n}[x]$ be a planar Dembowski-Ostrom (DO) polynomial, where p is an odd prime and n a positive integer. Let $\mathrm{Quot}(\mathcal{D}_g)$ be the set of quotients XY^{-1} with $Y \neq 0, X$ being elements from the spread set of the commutative presemifield corresponding to g. We analyze the algebraic structure of $\mathrm{Quot}(\mathcal{D}_g)$ for all planar DO *monomials*. More precisely, for g being CCZ-equivalent to a planar DO monomial, we show that every non-zero element $X \in \mathrm{Quot}(\mathcal{D}_g)$ generates a field $\mathbb{F}_p[X] \subseteq \mathrm{Quot}(\mathcal{D}_g)$. In particular, $\mathrm{Quot}(\mathcal{D}_g)$ contains the field $\mathbb{F}_{p^n}$.

1 Introduction and Preliminaries

Let p be an odd prime and n a positive integer. By $\mathrm{Mat}_{\mathbb{F}_p}(n,n)$, we denote the ring of all $n \times n$ matrices with coefficients in the prime field $\mathbb{F}_p$ and by $\mathrm{GL}(n,\mathbb{F}_p)$ the subgroup of all invertible matrices in $\mathrm{Mat}_{\mathbb{F}_p}(n,n)$. Given $A \in \mathrm{Mat}_{\mathbb{F}_p}(n,n)$, we denote by $\mathbb{F}_p[A]$ the $\mathbb{F}_p$-algebra generated by A, i.e., $\mathbb{F}_p[A] = \{\sum_i a_i A^i \mid a_i \in \mathbb{F}_p\}$. A polynomial $g \in \mathbb{F}_{p^n}[x]$ is called *planar* if, for all $\alpha \in \mathbb{F}_{p^n}^*$,

$$\Delta_{g,\alpha}(x) \coloneqq g(x+\alpha) - g(x) - g(\alpha)$$

is a permutation polynomial in $\mathbb{F}_{p^n}[x]$ i.e., its evaluation map $\mathbb{F}_{p^n} \to \mathbb{F}_{p^n}, y \mapsto \Delta_{g,\alpha}(y)$ is 1-to-1. Planar polynomials were introduced by Dembowski and Ostrom in [5]. Since we only study properties of evaluation maps in $\mathbb{F}_{p^n}$, we assume that $g \in \mathbb{F}_{p^n}[x]/(x^{p^n} - x)$, i.e., g has degree at most $p^n - 1$. A special type of polynomials in $\mathbb{F}_{p^n}[x]$ are *Dembowski-Ostrom* (DO) polynomials, which are those of the form

$$\sum_{0 \leq i \leq j \leq n-1} u_{i,j} \cdot x^{p^i + p^j}, \quad u_{i,j} \in \mathbb{F}_{p^n}.$$

If g is DO, $\Delta_{g,\alpha}$ is a linearized polynomial (i.e., its evaluation map is linear) for every $\alpha \in \mathbb{F}_{p^n}$. Let us denote by $M_{g,\alpha}$ the matrix (after fixing a choice of basis) associated to the evaluation map of $\Delta_{g,\alpha}$. For a planar DO polynomial g, we define its *spread set* $\mathcal{D}_g$ as

$$\mathcal{D}_g \coloneqq \{M_{g,\alpha} \mid \alpha \in \mathbb{F}_{p^n}\} \subseteq \mathrm{GL}(n,\mathbb{F}_p) \cup \{0\}.$$

*This extended abstract is extracted from the full article available at https://arxiv.org/abs/2211.17103

Remark 1. In [3], Coulter and Henderson showed a one-to-one correspondence between commutative presemifields of odd order and planar Dembowski-Ostrom polynomials. $\mathcal{D}_g$ is equal to the set of matrices corresponding to the mappings $x \to a \star x$ of left-multiplications with elements a in the corresponding commutative presemifield $\mathcal{R}_g$, hence $\mathcal{D}_g$ is equal to the spread set of $\mathcal{R}_g$ (see e.g., [6, Sec. 2.1]).

An equivalence relation between two polynomials that leaves the planarity property invariant is *CCZ-equivalence* [2]. CCZ-equivalence of two planar DO polynomials coincides with *linear equivalence* [1].

We study the *set of quotients in* $\mathcal{D}_g$, defined as

$$\mathrm{Quot}(\mathcal{D}_g) \coloneqq \bigcup_{Y \in \mathcal{D}_g \setminus \{0\}} \mathcal{D}_g Y^{-1} = \{XY^{-1} \mid X, Y \in \mathcal{D}_g \text{ and } Y \neq 0\}.$$

The following observation is immediate from the fact that $g(x+y)-g(x)-g(y)$ is symmetric in x and y and bilinear.

Lemma 1. *Let $g \in \mathbb{F}_{p^n}[x]$ be a DO polynomial and $\{\alpha_1, \alpha_2, \ldots, \alpha_n\}$ be an $\mathbb{F}_p$-basis of $\mathbb{F}_{p^n}$. For each $Y \in \mathrm{GL}(n, \mathbb{F}_p)$, the set $\mathcal{D}_g Y^{-1}$ is an n-dimensional $\mathbb{F}_p$-vector space with basis*

$$\{M_{g,\alpha_1} Y^{-1}, M_{g,\alpha_2} Y^{-1}, \ldots, M_{g,\alpha_n} Y^{-1}\}.$$

The reason we are interested in the set $\mathrm{Quot}(\mathcal{D}_g)$ is that it stays invariant up to a different choice of basis under linear-equivalence of g, hence yielding an invariant for the CCZ-equivalence of DO planar functions.

Proposition 1. *Let $g, g' \in \mathbb{F}_{p^n}[x]$ be two planar DO polynomials within the same linear-equivalence class. Then, $\mathrm{Quot}(\mathcal{D}_{g'}) = A^{-1} \cdot \mathrm{Quot}(\mathcal{D}_g) \cdot A$ for an element $A \in \mathrm{GL}(n, \mathbb{F}_p)$.*

Proof. This immediately follows from the fact that the spread sets of g and g' are related via $\mathcal{D}_{g'} = X^{-1} \cdot \mathcal{D}_g \cdot Y$ for some $X, Y \in \mathrm{GL}(n, \mathbb{F}_p)$ (see also [6, Sec. 2.1]). □

We would like to recall that any finite field $\mathbb{F}_{p^n}$ (resp., a proper subfield $\mathbb{F}_{p^m}$) is isomorphic to $\mathbb{F}_p[T_\beta]$, where T_β denotes a matrix corresponding to the linear mapping $x \mapsto \beta x$ over $\mathbb{F}_{p^n}$, for $\beta \in \mathbb{F}_{p^n}^*$ defining a polynomial basis of $\mathbb{F}_{p^n}$ (resp., of $\mathbb{F}_{p^m}$). For more details on *matrix representations* of finite fields, we refer to, e.g., [7] or [8]. Applying a change of basis transformation to all elements of a matrix algebra $\mathbb{F}_p[T]$ does not affect the property of being a field, hence $\mathbb{F}_p[T]$ is a finite field if and only if $A^{-1} \cdot \mathbb{F}_p[T] \cdot A$ is for all $A \in \mathrm{GL}(n, \mathbb{F}_p)$.

2 The Structure of $\mathrm{Quot}(\mathcal{D}_g)$ for a planar DO monomial g

In [4], Coulter and Matthews showed that any planar DO monomial in $\mathbb{F}_{p^n}[x]$ is CCZ-equivalent to $x^{p^k+1} \in \mathbb{F}_{p^n}[x]$ with $n/\gcd(k,n)$ being odd. We show that for any DO polynomial $h \in \mathbb{F}_{p^n}[x]$ CCZ-equivalent to a planar monomial, the set $\mathrm{Quot}(\mathcal{D}_h)$ always contains the finite field of order p^n. More precisely, we show the following.

Theorem 1. *Let p be an odd prime and n a positive integer. Let $g(x) \in \mathbb{F}_{p^n}[x]$ be a planar DO monomial. For any $\alpha, \beta \in \mathbb{F}_{p^n}^*$, the element $X \coloneqq M_{g,\beta} M_{g,\alpha}^{-1} \in \mathrm{Quot}(\mathcal{D}_g)$ generates a field isomorphic to $\mathbb{F}_p(\alpha^{-1}\beta)$ viz. $\mathbb{F}_p[X]$, and $\mathbb{F}_p[X] \subseteq \mathrm{Quot}(\mathcal{D}_g)$.*

Let us denote by $\phi_\alpha \colon \mathbb{F}_{p^n} \to \mathbb{F}_{p^n}, x \mapsto \alpha x^{p^k} + \alpha^{p^k} x$ the evaluation map of $\Delta_{x^{p^k+1},\alpha} \in \mathbb{F}_{p^n}[x]$. It is well known that ϕ_α is invertible if and only if $n/\gcd(k,n)$ is odd (see [4]). We have the following for the inverse, which is a special case of of Thm. 2.1 of [10]. It can also be proven by straightforward calculation of $\phi_\alpha^{-1}(\phi_\alpha(x))$.

Lemma 2 (Special case of Thm. 2.1 of [10]). *Let k be such that $n/\gcd(k,n)$ is odd. Let $d \coloneqq n/\gcd(k,n)$. For $\alpha \in \mathbb{F}_{p^n}^*$, the inverse of $\phi_\alpha \colon x \mapsto \alpha x^{p^k} + \alpha^{p^k} x$ is given by*

$$\phi_\alpha^{-1} \colon x \mapsto \frac{\alpha}{2} \cdot \sum_{i=0}^{d-1} (-1)^i \alpha^{-(p^k+1)p^{ki}} x^{p^{ki}}.$$

The following lemma is immediate.

Lemma 3. *Let k be such that $n/\gcd(k,n)$ is odd and let $\phi_\alpha \colon x \mapsto \alpha x^{p^k} + \alpha^{p^k} x$. For any $\alpha, \beta \in \mathbb{F}_{p^n}^*$, we have $\phi_\beta(\phi_\alpha^{-1}(x)) = (\beta^{p^k} - \alpha^{p^k-1}\beta) \cdot \phi_\alpha^{-1}(x) + \alpha^{-1}\beta x$.*

The monomial $g(x) = x^{p^k+1}$ admits a non-trivial self equivalence via $g(x) = \gamma^{-(p^k+1)} \cdot g(\gamma x)$, where γ is an arbitrary non-zero element of $\mathbb{F}_{p^n}$. From this, we obtain the following.

Lemma 4. *Let k be such that $n/\gcd(k,n)$ is odd and let $\phi_\alpha \colon x \mapsto \alpha x^{p^k} + \alpha^{p^k} x$. For any $\alpha, \beta, \gamma \in \mathbb{F}_{p^n}$, $\alpha, \gamma \neq 0$, we have $\phi_\beta(\phi_\alpha^{-1}(x)) = \gamma^{-(p^k+1)} \cdot \phi_{\gamma\beta}(\phi_{\gamma\alpha}^{-1}(\gamma^{p^k+1}x))$.*

To show Theorem 1, we will first deduce that each element in $\mathrm{Quot}(\mathcal{D}_g)$ generates (a subfield of) $\mathbb{F}_{p^n}$. To do so, we show that each element in $\mathrm{Quot}(\mathcal{D}_g)$ corresponds (up to a choice of basis) to a multiplication with an element of $\mathbb{F}_{p^n}$.

Lemma 5. *Let k be such that $n/\gcd(k,n)$ is odd. Let $\alpha, \beta \in \mathbb{F}_{p^n}, \alpha \neq 0$. If $\alpha^{-1}\beta \in \mathbb{F}_{p^{\gcd(k,n)}}$, the mapping $\phi_\beta \circ \phi_\alpha^{-1}$ is equal to $x \mapsto \alpha^{-1}\beta x$. If $\alpha^{-1}\beta$ lies not in $\mathbb{F}_{p^{\gcd(k,n)}}$, the mapping $\psi_{\alpha,\beta} \circ \phi_\beta \circ \phi_\alpha^{-1} \circ \psi_{\alpha,\beta}^{-1}$ is equal to $x \mapsto (\alpha^{-1}\beta)^{p^k} x$, where*

$$\psi_{\alpha,\beta} \colon x \mapsto \alpha^{p^k} \cdot \phi_\alpha \left(\frac{1}{\beta^{p^k} - \alpha^{p^k-1}\beta} \cdot x \right).$$

Proof. We first observe that $\beta^{p^k} - \alpha^{p^k-1}\beta$ is equal to zero if and only if $\beta = 0$ or $(\alpha^{-1}\beta)^{p^k-1} = 1$, i.e., if and only if $\alpha^{-1}\beta$ is contained in the subfield $\mathbb{F}_{p^{\gcd(k,n)}} \subseteq \mathbb{F}_{p^n}$. Hence, by Lemma 3, the statement is trivial for the case of $\alpha^{-1}\beta \in \mathbb{F}_{p^{\gcd(k,n)}} \subseteq \mathbb{F}_{p^n}$.

In the other case, the mapping $\psi_{\alpha,\beta}$ is well defined and we can decompose $\psi_{\alpha,\beta}$ as $C \circ B \circ A$, where A is a multiplication by $(\beta^{p^k} - \alpha^{p^k-1}\beta)^{-1}$, $B = \phi_\alpha$, and C is a multiplication by α^{p^k}. For all $x \in \mathbb{F}_{p^n}$, we then have:

$$L_1(x) \coloneqq A(\phi_\beta(\phi_\alpha^{-1}(A^{-1}(x)))) = \phi_\alpha^{-1}\left((\beta^{p^k} - \alpha^{p^k-1}\beta)x\right) + \alpha^{-1}\beta x.$$

$$\begin{aligned} L_2(x) \coloneqq B(L_1(B^{-1}(x))) &= (\beta^{p^k} - \alpha^{p^k-1}\beta) \cdot \phi_\alpha^{-1}(x) + \phi_\alpha(\alpha^{-1}\beta \cdot \phi_\alpha^{-1}(x)) \\ &= \beta^{p^k} \cdot \left(\phi_\alpha^{-1}(x) + \alpha^{-p^k+1}(\phi_\alpha^{-1}(x))^{p^k}\right). \end{aligned}$$

$$\begin{aligned} L_3(x) \coloneqq C(L_2(C^{-1}(x))) &= \beta^{p^k} \cdot \left(\alpha^{p^k}\phi_\alpha^{-1}(\alpha^{-p^k}x) + \alpha(\phi_\alpha^{-1}(\alpha^{-p^k}x))^{p^k}\right) \\ &= \beta^{p^k} \cdot \phi_\alpha(\phi_\alpha^{-1}(\alpha^{-p^k}x)) = (\alpha^{-1}\beta)^{p^k}x. \end{aligned}$$

The proof is complete since $L_3 = \psi_{\alpha,\beta} \circ \phi_\beta \circ \phi_\alpha^{-1} \circ \psi_{\alpha,\beta}^{-1}$. □

The more complicated part is to show that, for any $X \in \mathrm{Quot}(\mathcal{D}_g)$, the matrix algebra $\mathbb{F}_p[X]$ is indeed a subset of $\mathrm{Quot}(\mathcal{D}_g)$. We do this in the following.

Proof of Theorem 1. Let $\alpha, \beta \in \mathbb{F}_{p^n}^*$ and let $X \coloneqq M_{g,\beta} M_{g,\alpha}^{-1}$. By Lemma 5, the linear mapping $\phi_\beta \circ \phi_\alpha^{-1}$ is similar to $x \mapsto \alpha^{-1}\beta x$. Hence, the $\mathbb{F}_p$-algebra $\mathbb{F}_p[X]$ is isomorphic to $\mathbb{F}_p(\alpha^{-1}\beta)$ and thus a field. It is left to show that $\mathbb{F}_p[X] \subseteq \mathrm{Quot}(\mathcal{D}_g)$. The case of $\alpha^{-1}\beta \in \mathbb{F}_{p^{\gcd(k,n)}}$ is trivial and we therefore assume in the following that $\alpha^{-1}\beta \notin \mathbb{F}_{p^{\gcd(k,n)}}$. We will first handle the case of $\alpha = 1$ and show that $\left(M_{g,\beta} M_{g,1}^{-1}\right)^r \in \mathrm{Quot}(\mathcal{D}_g)$ for any integer $r \geq 2$. By Lemma 5, we have

$$\psi_{1,\beta} \circ \left(\phi_\beta \circ \phi_1^{-1}\right)^r \circ \psi_{1,\beta}^{-1}(x) = \left(\psi_{1,\beta} \circ \phi_\beta \circ \phi_1^{-1} \circ \psi_{1,\beta}^{-1}\right)^r (x) = \beta^{rp^k} x.$$

Further,

$$\beta^{rp^k} x = \begin{cases} \psi_{1,\beta^r} \circ \phi_{\beta^r} \circ \phi_1^{-1} \circ \psi_{1,\beta^r}^{-1}(x) & \text{if } \beta^r \notin \mathbb{F}_{p^{\gcd(k,n)}} \\ \beta^r x = \phi_{\beta^r} \circ \phi_1^{-1}(x) & \text{otherwise} \end{cases},$$

and thus

$$\left(\phi_\beta \circ \phi_1^{-1}\right)^r = \begin{cases} \psi_{1,\beta}^{-1} \circ \psi_{1,\beta^r} \circ \phi_{\beta^r} \circ \phi_1^{-1} \circ \psi_{1,\beta^r}^{-1} \circ \psi_{1,\beta} & \text{if } \beta^r \notin \mathbb{F}_{p^{\gcd(k,n)}} \\ \psi_{1,\beta}^{-1} \circ \phi_{\beta^r} \circ \phi_1^{-1} \circ \psi_{1,\beta} & \text{otherwise} \end{cases}. \tag{1}$$

We will now prove that the latter composition is equal to $\phi_\delta \circ \phi_\gamma^{-1}$ for properly chosen field elements δ, γ.

Case $\beta^r \in \mathbb{F}_{p^{\gcd(k,n)}}$**.** In this case, $\left(\phi_\beta \circ \phi_1^{-1}\right)^r (x) = \psi_{1,\beta}^{-1} \circ \phi_{\beta^r} \circ \phi_1^{-1} \circ \psi_{1,\beta}(x) = \psi_{1,\beta}^{-1}(\beta^r \cdot \psi_{1,\beta}(x)) = \beta^r \cdot \psi_{1,\beta}^{-1}(\psi_{1,\beta}(x)) = \beta^r x = \phi_{\beta^r} \circ \phi_1^{-1}(x)$, since $\psi_{1,\beta}$ is $\mathbb{F}_{p^{\gcd(k,n)}}$-linear.

Case $\beta^r \notin \mathbb{F}_{p^{\gcd(k,n)}}$**.** We first observe that $\psi_{1,\beta}^{-1} \circ \psi_{1,\beta^r}(x) = \frac{\beta^{p^k} - \beta}{\beta^{rp^k} - \beta^r} x$. Let us define $\lambda \coloneqq \frac{\beta^{p^k} - \beta}{\beta^{rp^k} - \beta^r} \in \mathbb{F}_{p^n}^*$. The image of the mapping $x \mapsto x^{p^k+1}$ over $\mathbb{F}_{p^n}$ is equal to the set of squares in $\mathbb{F}_{p^n}$. Indeed, every element in the image is a square as $p^k + 1$ is even, and $x \mapsto x^{p^k+1}$ is 2-to-1 as a DO planar function [9]. Hence, if λ is a square, we have $\lambda = \gamma^{p^k+1}$ for an element $\gamma \in \mathbb{F}_{p^n}^*$ and, otherwise, we have $\lambda = u\gamma^{p^k+1}$ with $u \in \mathbb{F}_{p^n}^*$ being an arbitrary non-square. Note that we can always choose $u \in \mathbb{F}_{p^{\gcd(k,n)}}^*$. Indeed, let $n = 2^m \ell$ and $k = 2^{m'} \ell'$ with ℓ, ℓ' being odd, we necessarily have $m' \geq m$, as otherwise $n/\gcd(k,n)$ would be even. So, $\mathbb{F}_{p^{\gcd(k,n)}}$ contains $\mathbb{F}_{p^{2^m}}$ as a subfield and the extension degree $[\mathbb{F}_{p^n} : \mathbb{F}_{p^{\gcd(k,n)}}]$ is odd. The claim then follows as a non-square in a finite field stays a non-square in any extension field of odd extension degree.

Let us therefore assume that $\lambda = u\gamma^{p^k+1}$ with $\gamma \in \mathbb{F}_{p^n}^*$ and $u \in \mathbb{F}_{p^{\gcd(k,n)}}^*$. We have

$$\begin{aligned} \psi_{1,\beta}^{-1} \circ \psi_{1,\beta^r} \circ \phi_{\beta^r} \circ \phi_1^{-1} \circ \psi_{1,\beta^r}^{-1} \circ \psi_{1,\beta}(x) &= \lambda \cdot \left(\phi_{\beta^r} \circ \phi_1^{-1}\right)\left(\lambda^{-1} x\right) \\ &= \gamma^{p^k+1} \cdot \left(\phi_{\beta^r} \circ \phi_1^{-1}\right)\left(\gamma^{-(p^k+1)} x\right), \end{aligned} \tag{2}$$

where the last equality follows from the fact that $u \in \mathbb{F}_{p^{\gcd(k,n)}}^*$. By Lemma 4, we have $\gamma^{p^k+1} \cdot \left(\phi_{\beta^r} \circ \phi_1^{-1}\right)\left(\gamma^{-(p^k+1)} x\right) = \phi_{\gamma\beta^r} \circ \phi_\gamma^{-1}(x)$.

To handle the case of $\alpha \neq 1$, we apply Lemma 4 with $\gamma = \alpha^{-1}$ and obtain $\phi_\beta(\phi_\alpha^{-1}(x)) = \alpha^{p^k+1} \cdot \phi_{\alpha^{-1}\beta}(\phi_1^{-1}(\alpha^{-(p^k+1)} x))$, hence,

$$\begin{aligned} (\phi_\beta \circ \phi_\alpha^{-1})^r(x) &= \alpha^{p^k+1} \cdot (\phi_{\alpha^{-1}\beta} \circ \phi_1^{-1})^r(\alpha^{-(p^k+1)} x) \\ &= \alpha^{p^k+1} \cdot \left(\phi_{\delta'} \circ \phi_{\gamma'}^{-1}(\alpha^{-(p^k+1)} x)\right) = \phi_{\alpha\delta'} \circ \phi_{\alpha\gamma'}^{-1}(x) \end{aligned}$$

for appropriate elements γ', δ'. We have now established that, for $\alpha^{-1}\beta$ being a generator of $\mathbb{F}_{p^n}^*$, the algebra $\mathbb{F}_p[X]$ is a field of order p^n contained in $\mathrm{Quot}(\mathcal{D}_g)$.

To handle the general case where $\alpha^{-1}\beta$ is not a generator of $\mathbb{F}_{p^n}^*$, we will show that X is equal to $(M_{g,\beta'}M_{g,\alpha'}^{-1})^r$ for some generator $\alpha'^{-1}\beta'$ of $\mathbb{F}_{p^n}^*$ and some non-negative integer r. Then, it would immediately follow that $\mathbb{F}_p[X] \subseteq \mathbb{F}_p[M_{g,\beta'}M_{g,\alpha'}^{-1}] \subseteq \mathrm{Quot}(\mathcal{D}_g)$. Indeed, let $\bar{\beta}$ be a generator of $\mathbb{F}_{p^n}^*$ such that $\bar{\beta}^r = \alpha^{-1}\beta$ and let

$$\frac{\bar{\beta}^{p^k} - \bar{\beta}}{\bar{\beta}^{rp^k} - \bar{\beta}^r} = u\gamma^{p^k+1}$$

with $\gamma \in \mathbb{F}_{p^n}^*$ and $u \in \mathbb{F}_{p^{\gcd(k,n)}}^*$. By extensively applying Lemma 4 and the result we established above, we obtain

$$\begin{aligned}
(\phi_{\alpha\gamma^{-1}\bar{\beta}} \circ \phi_{\alpha\gamma^{-1}}^{-1})^r(x) &= \left((\alpha^{-1}\gamma)^{-(p^k+1)} \cdot \phi_{\bar{\beta}} \circ \phi_1^{-1}((\alpha^{-1}\gamma)^{p^k+1}x)\right)^r \\
&= (\alpha^{-1}\gamma)^{-(p^k+1)} \cdot \left(\phi_{\bar{\beta}} \circ \phi_1^{-1}\right)^r ((\alpha^{-1}\gamma)^{p^k+1}x) \\
&= (\alpha^{-1}\gamma)^{-(p^k+1)} \cdot \phi_{\gamma\bar{\beta}^r} \circ \phi_\gamma^{-1}((\alpha^{-1}\gamma)^{p^k+1}x) \\
&= (\alpha^{-1}\gamma)^{-(p^k+1)} \cdot \phi_{\alpha^{-1}\gamma\beta} \circ \phi_\gamma^{-1}((\alpha^{-1}\gamma)^{p^k+1}x) = \phi_\beta \circ \phi_\alpha^{-1}(x).
\end{aligned}$$

□

Remark 2. For $g(x) = x^{p^k+1} \in \mathbb{F}_{p^n}[x]$ planar, we have $|\mathrm{Quot}(\mathcal{D}_g)| = \frac{(p^n - p^{\gcd(k,n)})\cdot(p^n-1)}{p^{\gcd(k,n)}-1} + p^{\gcd(k,n)}$.

References

[1] L. Budaghyan and T. Helleseth. New commutative semifields defined by new PN multinomials. *Cryptogr. Commun.*, 3(1):1–16, 2011.

[2] C. Carlet, P. Charpin, and V. A. Zinoviev. Codes, bent functions and permutations suitable for des-like cryptosystems. *Des. Codes Cryptogr.*, 15(2):125–156, 1998.

[3] R. S. Coulter and M. Henderson. Commutative presemifields and semifields. *Adv. Math.*, 217(1):282–304, 2008.

[4] R. S. Coulter and R. W. Matthews. Planar functions and planes of lenz-barlotti class II. *Des. Codes Cryptogr.*, 10(2):167–184, 1997.

[5] P. Dembowski and T. G. Ostrom. Planes of order n with collineation groups of order n^2. *Math. Z.*, 103(3):239–258, 1968.

[6] U. Dempwolff. Semifield planes of order 81. *J. Geom.*, 89:1–16, 2008.

[7] D. Hachenberger and D. Jungnickel. *Topics in Galois fields*. Springer, 2020.

[8] R. Lidl and H. Niederreiter. *Introduction to finite fields and their applications*. Cambridge university press, 1994.

[9] G. Weng and X. Zeng. Further results on planar DO functions and commutative semifields. *Des. Codes Cryptogr.*, 63(3):413–423, 2012.

[10] B. Wu. The compositional inverses of linearized permutation binomials over finite fields. *arXiv preprint arXiv:1311.2154*, 2013.

A computation of D(9) using FPGA Supercomputing

Lennart Van Hirtum[1,2,3], Patrick De Causmaecker[1], Jens Goemaere[1], Tobias Kenter[2,3], Heinrich Riebler[2,3], Michael Lass[2,3], and Christian Plessl[2,3]

[1]KU Leuven, Department of Computer Science, KULAK
[2]Department of Computer Science, Paderborn University
[3]Paderborn Center for Parallel Computing, Paderborn University
e-mail adresses in footnote*

April 2023

Abstract

This paper reports on the first computation the 9^{th} Dedekind Number. This was done by building an efficient FPGA Accelerator for the core operation of the process, and parallelizing it on the Noctua 2 Supercluster at Paderborn University. The resulting value is

286386577668298411128469151667598498812366

This value can be verified in two steps. We have made the data file containing the 490M subresults available upon request, each of which can be verified separately on CPU, and the whole file sums to our proposed value.

1 Introduction

Let us consider the finite set $A = \{1, \ldots, n\}$, which we will call the *base set*, and let us denote the set of subsets of A by $\mathcal{P}(A)$. Dedekind numbers count the number of monotone Boolean functions on $\mathcal{P}(A)$. The set of monotone Boolean functions with respect to inclusion on $\mathcal{P}(A)$ is denoted by $\mathcal{D}_n$. The number of such monotone Boolean functions is denoted by $D(n)$ and this is called *the* n^{th} *Dedekind number*.

*lennart.vanhirtum@gmail.com, patrick.decausmaecker@kuleuven.be, jens.goemaere@kuleuven.be, kenter@uni-paderborn.de, heinrich.riebler@uni-paderborn.de, michael.lass@uni-paderborn.de, christian.plessl@uni-paderborn.de

The set of permutations of the elements of base set A generates an equivalence relation on $\mathcal{D}_n$. The set of equivalence classes of this relation are denoted by $\mathcal{R}_n$ and the number of such equivalence classes is denoted by $R(n)$.
Richard Dedekind first defined the numbers $D(n)$ in 1897 [1]. Over the previous century, Dedekind numbers have been a challenge for computational power in the evolving domain of computer science. Computing the numbers proved exceptionally hard, and so far only formula's with a double exponential time complexity are known. Until recently, the largest known Dedekind number was $D(8)$. In this paper, we report on a computation of $D(9)$. Table 1 shows the known numbers, including the result of our computation. As we explain below, some uncertainty about the correctness of the number existed at the time of the first computation and we planned a verification run. In the mean time however, results of an independent computation were reported [2], confirming our result. Since computational methods as well as hardware implementation differ significantly between the two computations, we can conclude that the result is correct with a probability very close to 1.
Table 2 shows the known numbers $R(n)$ of equivalence classes of monotone Boolean functions under permutation of the elements of the base set. Note that the last result dates from 2023.

D(0)	2	Dedekind (1897)
D(1)	3	Dedekind (1897)
D(2)	6	Dedekind (1897)
D(3)	20	Dedekind (1897)
D(4)	168	Dedekind (1897)
D(5)	7581	Church (1940)
D(6)	7828354	Ward (1946)
D(7)	2414682040998	Church (1965)
D(8)	56130437228687557907788	Wiedemann (1991)
D(9)	286386577668298411128469151667598498812366	Our result (2023)

Table 1: Known Dedekind Numbers [3] and our first result.

R(0)	2	
R(1)	3	
R(2)	5	
R(3)	10	
R(4)	30	
R(5)	210	
R(6)	16353	
R(7)	490013148	Tamon Stephen & Timothy Yusun (2014) [4]
R(8)	1392195548889993358	Bartłomiej Pawelski (2021) [5]
R(9)	7892046358420350405277408463002526 80	Bartłomiej Pawelski (2023) [6]

Table 2: Known Equivalence Class Counts

For clarity of this paragraph, let us assume that we consider monotonically decreasing Boolean functions. Note that a monotonically decreasing Boolean function is completely defined by the set of sets which are maximal among the sets for which the function value is true. For any monotone Boolean function, no two of its maximal sets include one another. Such a set of sets is called an anti-chain. A monotone Boolean function is completely determined by its associated anti-chain, and any anti-chain is completely determined by its associated monotone Boolean function. We will use any of the two representations whichever is more convenient. We will represent monotone Boolean functions or anti-chains by letters from the Greek alphabet. If we say that $X \in \alpha$, we mean that X is a maximal set among the sets for which α is $True$, in other words

$$\forall Y \subseteq X : \alpha(Y) = True \; and \; \forall Z \supsetneq X : \alpha(Z) = False$$

If we say that $\alpha = \{X, Y, Z\}$, we mean that the sets $X, Y, Z \subseteq A$ are the maximal sets among the sets for which α is $True$. For the set D_n of monotone Boolean functions on the base set a natural partial order $\leq$ is defined by

$$\forall \alpha, \beta \in D_n : \alpha \leq \beta \Leftrightarrow \forall X \subseteq A : \alpha(X) \Rightarrow \beta(X) \tag{1}$$

This partial ordering defines a complete lattice on D_n. We denote by $\bot$ and $\top$ the smallest, respectively the largest, element of D_n:

$$\forall X \subseteq A : \bot(X) = False, \top(X) = True \tag{2}$$

$$\bot(X) = \{\}, \top(X) = \{A\} \tag{3}$$

Intervals in D_n are denoted by

$$\forall \alpha, \beta \in D_n : [\alpha, \beta] = \{\chi \in D_n : \alpha \leq \chi \leq \beta\} \tag{4}$$

For $\alpha, \beta \in D_n$, the *join* $\alpha \vee \beta$ and the *meet* $\alpha \wedge \beta$ are the monotone Boolean functions defined by

$$\forall X \subseteq A : (\alpha \vee \beta)(X) = \alpha(X) \; or \; \beta(X) \tag{5}$$

$$\forall X \subseteq A : (\alpha \wedge \beta)(X) = \alpha(X) \; and \; \beta(X) \tag{6}$$

Finally, in the formulas below, a number defined for each pair $\alpha \leq \beta \in D_n$ plays an important role. We refer to this number as the *connector number* $C_{\alpha,\beta}$ of α and β. It counts the number of connected components of the anti-chain β with respect to α. Two such sets $X, Y \in \beta$ are connected if $\alpha(X \cap Y) = False$ or if there is a path $X, Z_1, ..., Z_n, Y$ of such subsets $X, Z_1, ..., Z_n, Y \subseteq A$ in which for every two subsequent sets $\alpha(X \cap Z_1) = \alpha(Z_1 \cap Z_2) = ... = \alpha(Z_n \cap Y) = False$. It turns out that the number of solutions of

$$\chi \vee \upsilon = \beta \tag{7}$$

$$\chi \wedge \upsilon = \alpha \tag{8}$$

for $\chi, \upsilon \in D_n$ is given by $2^{C_{\alpha,\beta}}$. This is called the *P-Coefficient* [7, 8].

2 Method, Theory

We start from the original P-Coefficient Formula as taken from [7].

$$D(n+2) = \sum_{\alpha,\beta \in D_n} |[\bot, \alpha]| 2^{C_{\alpha,\beta}} |[\beta, \top]| \tag{9}$$

In the master thesis of the first author of the current paper, Lennart Van Hirtum [9], the author reworked this formula to a form making use of equivalence classes to reduce the total number of terms.

$$D(n+2) = \sum_{\alpha \in R_n} |[\bot, \alpha]| D_\alpha \sum_{\substack{\beta \in R_n \\ \exists \delta \simeq \beta : \alpha \leq \delta}} |[\beta, \top]| \frac{D_\beta}{n!} \sum_{\substack{\gamma \in Permut_\beta \\ \alpha \leq \gamma}} 2^{C_{\alpha,\gamma}} \tag{10}$$

The $Permut_\beta$ term is the collection of all $n!$ equivalents of β under permutation of the base set. D_β is the number of different equivalents, and hence, $Permut_\beta$ contains duplicates iff $D_\beta < n!$. These duplicates are divided out by the $\frac{D_\beta}{n!}$ factor.
For $D(9)$, this means iterating through D_7. That would require iterating over an estimated $4.59 * 10^{16}$ α, β pairs. The total number of P-Coefficients ($C_{\alpha,\gamma}$) that needed to be computed was $1.148 * 10^{19}$. However we were able to improve on this further using the process of 'deduplication', where we can halve the total amount of work again, by noticing that pairs of α, β give identical results to their dual pair $\overline{\beta}, \overline{\alpha}$. As per Equation 11. This allowed us to halve the total amount of work to $5.574 * 10^{18}$ P-Coefficients. [1]

$$|[\bot, \alpha]| 2^{C_{\alpha,\beta}} |[\beta, \top]| = |[\overline{\alpha}, \top]| 2^{C_{\overline{\beta},\overline{\alpha}}} |[\bot, \overline{\beta}]| \tag{11}$$

3 Computing P-Coefficients on FPGA

Computing P-Coefficients is uniquely well-suited for hardware implementation. Computing these terms requires solving the problem of counting the number of distinct connected components within a standard graph structure. An example of such a graph with its distinct connected components colored is shown in Figure 1. The standard depth first search algorithm for this problem is linear in the sum of the number of vertices (sets) and the number of edges between these vertices. Given the number of P-coefficients to be evaluated, it is clear that traditional instruction-based computing methods, particularly Single Instruction Multiple Data (SIMD), fare poorly on it. Since counting connected components in such fixed-sizes graphs (in this case 128-node 7-d hypercubes) consists almost purely of plain Boolean operations, it translates very well to a hardware implementation and provides a highly efficient implementation of the

[1] We made sure not to deduplicate pairs that were their own dual, ie when $\beta = \overline{\alpha}$

algorithm. A simple schematic implementation is shown in Figure 2. A detailed explanation of how it works is provided in the first author's master thesis [9]. In this thesis, some optimizations are derived that bring the average number of iterations down to 4.061. This corresponds to the number of cycles in the hardware design.

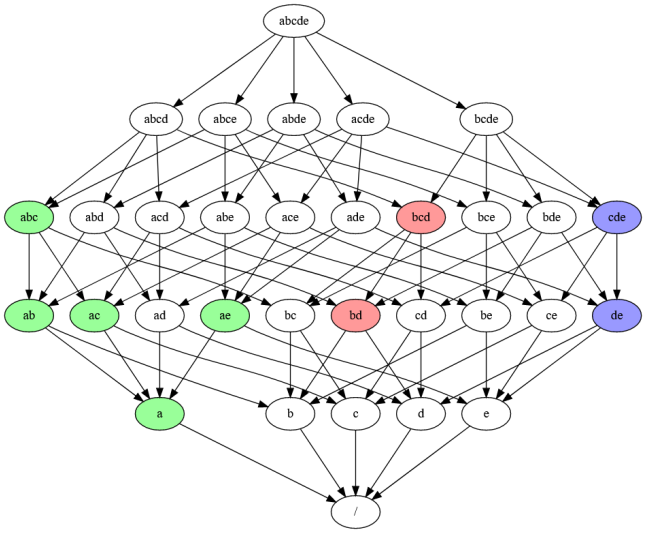

Figure 1: Connected components of an example graph. In this case there are 3 connected components.

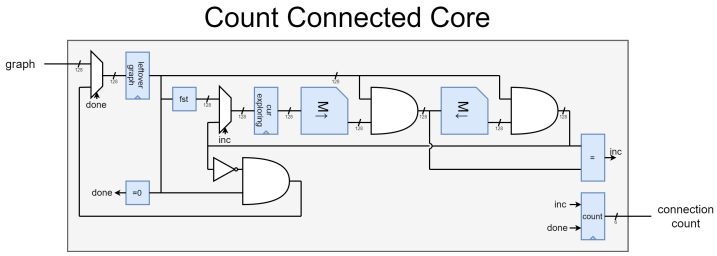

Figure 2: Register Transfer Level Design of the CountConnected Core

4 Computation on Noctua 2

We implemented this hardware accelerator on the Intel Stratix 10 GX 2800 cards found in Paderborn University's Noctua 2 supercomputer. We were able to fit 300 of these CountConnected Cores on a single field-programmable gate array (FPGA) die. These CountConnected Cores run at 450MHz. This gives us a throughput of about 33 Billion CountConnected operations per second. At this rate, a single FPGA processes about 5.2 α values per second, taking 47'000 FPGA hours to compute D(9) on Noctua 2, or about 3 months real-time.
The computation is split across the system along the lines of Equation 10. α values (also named tops) are divided on the job level. There are 490M tops to be processed for D(9). We split these into 15000 jobs of 30000 tops each. The β values per top (also named bottoms) are placed in large buffers of 46M bots on average, and sent over PCIe (Peripheral Component Interconnect Express) to the FPGA. The FPGA then computes all 5040 permutations (γ) of each bottom, computes and adds up their P-Coefficients. This result is stored in an output buffer of the same size.
The artifact of this computation is a dataset with an intermediary result for each of the 490M α values. Each of these can be checked separately[2], and the whole file sums to 28638657766829841112846915166759849881236 6.

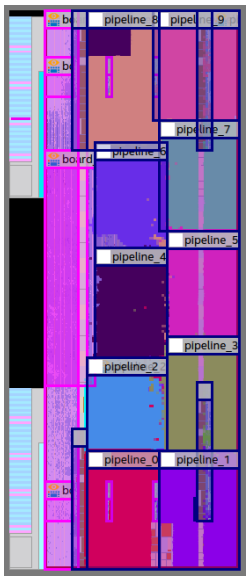

Figure 3: The FPGA Accelerator Die

[2]It takes about 10-200s to compute a single α result on 128 AMD Epyc CPU cores

5 Correctness

As much of the code as possible is written generically. This means the same system is used for computing D(3) - D(8). All of these yield the correct results. Of course, the FPGA kernel is written specifically for D(9) computation, so its correctness was verified by comparing its results with the CPU results for a small sample. In effect, both methods verified each other's correctness.
We did apply a number of additional checks to increase our confidence in the result:

- The most direct is the $D(9) \equiv 6 \ mod\ 210$ check provided by Pawelski & Szepietowski [10]. Our result passes this check. Sadly, due to the structure of our computation, nearly all terms are divisible by 210, which strongly hampers the usefulness of this check. One thing that this check does give us is that no integer overflow has occurred, which was an important concern given we were working with integers of 128 and 192 bits wide.
- Our computation was plagued by one issue in particular. Namely that there is a bug in the vendor library for communication over PCIe, wherein, occasionally and at a low incidence rate, full 4K pages of FPGA data are not copied properly from FPGA memory to host memory. This results in large blocks of incorrect bottoms for some tops. We encountered this issue in about 2300 tops. We were able to mitigate this issue by including extra data from the FPGA to host memory, namely the 'valid permutation count'. By checking these values, we could determine if a bottom buffer had been corrupted. Additionally, adding all of these counts yields the value for D(8), which shows that the correct number of terms have been added.
- Finally, there is an estimation formula, which gives us an estimation which is relatively close to our result. The Korshunov estimation formula estimates $D(9) = 1.15 * 10^{41}$ which is off by about a factor 2. [3]

6 The danger of SEU events

The one way our result could have still been wrong was due to a Single Event Upset (SEU), such as a bitflip in the FPGA fabric during processing, or a bitflip during data transfer from FPGA DDR memory to Main Memory.
It is difficult to characterise the odds of these SEU events. The expected number of occurrences for the FPGAs we used are not available to the best of our knowledge. But example values shown on Intel's website pin the error rate at around 5000 SEU events per billion FPGA hours. In that case, given our 47000 FPGA hours, we expected to see 0.235 errors Poisson distributed, giving us a

[3] This isn't too unusual though, as the results for odd values are off by quite a lot. Estimation for D(3) overestimates by a factor 2, D(5) also overestimates by a factor 2, and D(7) overestimates roughly 10%

chance of 20% of a hit. Of course, this is just an example and the real odds might be have been higher than that.
But, given that we have Jäkel reaching an identical result [2], the odds of a stochastic error affecting both implementations in exactly the same way are so astronomically small, that we can rule them out.

7 Conclusion

In conclusion, our method for computing D(9) works, our implementation should theoretically give the correct result. All that remains is: Have any bit errors occurred during this first computation? Our plan was to start up a second run. Each subresult would have been computed a second time, and any values that differ could be recomputed a third time as a tiebreaker. On April 4^{th} however, a preprint claiming D(9) was published, right before the present publication by Christian Jäkel [2]. This paper confirmed our result as we obtained it on the 8^{th} of March. So, the 9^{th} Dedekind Number was found on the 8^{th} of March, 2023 using the Noctua 2 supercluster at Paderborn University. This value was registered in the corresponding github commit: https://github.com/VonTum/Dedekind/commit/1cf7b019afca655586e8210f97fbb5399d61e842 All code is available at https://github.com/VonTum/Dedekind.

References

[1] R. Dedekind. Über Zerlegungen von Zahlen Durch Ihre Grössten Gemeinsamen Theiler, pages 1–40. Vieweg+Teubner Verlag, Wiesbaden, 1897.

[2] Christian Jäkel. A computation of the ninth dedekind number, 2023.

[3] Doug Wiedemann. A computation of the eighth dedekind number. https://link.springer.com/article/10.1007%2FBF00385808, 1991.

[4] Tamon Stephen and Timothy Yusun. Counting inequivalent monotone boolean functions. Discrete Applied Mathematics, 167:15–24, 2014.

[5] Bartłomiej Pawelski. On the number of inequivalent monotone boolean functions of 8 variables, 2021.

[6] Bartłomiej Pawelski. On the number of inequivalent monotone boolean functions of 9 variables, 2023.

[7] Patrick De Causmaecker and Stefan De Wannemacker. On the number of antichains of sets in a finite universe, 2014.

[8] Patrick De Causmaecker, Stefan De Wannemacker, and Jay Yellen. Intervals of antichains and their decompositions, 2016.

[9] Lennart Van Hirtum. A path to compute the 9th dedekind number using fpga supercomputing. https://hirtum.com/thesis.pdf, 2021. KU Leuven, Masters Thesis.

[10] Bartlomiej Pawelski and Andrzej Szepietowski. Divisibility properties of dedekind numbers, 2023.

A family of optimal linear codes from simplicial complexes

Zhao Hu, Zhexin Wang, Nian Li, Xiangyong Zeng, and Xiaohu Tang

Abstract

In this paper, we construct a large family of projective linear codes over $\mathbb{F}_q$ from the general simplicial complexes of $\mathbb{F}_q^m$ via the defining-set construction, which generalizes the results of [IEEE Trans. Inf. Theory 66(11):6762-6773, 2020]. The parameters and weight distribution of this class of codes are completely determined. By using the Griesmer bound, we give a necessary and sufficient condition such that the codes are Griesmer codes and a sufficient condition such that the codes are distance-optimal. For a special case, we also present a necessary and sufficient condition for the codes to be near Griesmer codes. Moreover, by discussing the cases of simplicial complexes with one, two and three maximal elements respectively, many infinite families of optimal linear codes with few weights over $\mathbb{F}_q$ are obtained, including Griesmer codes, near Griesmer codes and distance-optimal codes.

Index Terms

Optimal linear code, Simplicial complex, Griesmer code, Near Griesmer code, Weight distribution

I. Introduction

Let $\mathbb{F}_{q^m}$ be the finite field with q^m elements and $\mathbb{F}_{q^m}^* = \mathbb{F}_{q^m} \backslash \{0\}$, where q is a power of a prime p and m is a positive integer. An $[n,k,d]$ linear code $\mathcal{C}$ over $\mathbb{F}_q$ is a k-dimensional subspace of $\mathbb{F}_q^n$ with minimum (Hamming) distance d. An $[n,k,d]$ linear code $\mathcal{C}$ over $\mathbb{F}_q$ is called distance-optimal if no $[n,k,d+1]$ code exists (i.e., $\mathcal{C}$ has the largest minimum distance for given n and k) and it is called almost distance-optimal if there exists an $[n,k,d+1]$ distance-optimal code. An $[n,k,d]$ linear code $\mathcal{C}$ is called optimal (resp. almost optimal) if its parameters n, k and d (resp. $d+1$) meet any bound on linear codes with equality [8]. The Griesmer bound [7], [14] for an $[n,k,d]$ linear code $\mathcal{C}$ over $\mathbb{F}_q$ is given by

$$n \geq g(k,d) := \sum_{i=0}^{k-1} \lceil \frac{d}{q^i} \rceil,$$

where $\lceil \cdot \rceil$ denotes the ceiling function. An $[n,k,d]$ linear code $\mathcal{C}$ is called a Griesmer code (resp. near Griesmer code) if its parameters n (resp. $n-1$), k and d achieve the Griesmer bound. Griesmer codes have been an interesting topic of study for many years due to not only their optimality but also their geometric applications [4], [5]. In coding theory, it's a fundamental problem to construct (distance-)optimal codes.

Recently, constructing optimal or good linear codes from mathematical objects attracts much attention and many attempts have been made in this direction. In the various kinds of mathematical objects, simplicial complexes (which are certain subsets of $\mathbb{F}_q^m$ with good algebraic structure) are really useful to construct optimal or good linear codes. The investigation of constructing linear codes from simplicial complexes, to the best of our knowledge, first appeared in [3] (in 2018), in which Chang and Hyun constructed the first infinite family of binary minimal linear codes violating the Ashikhmin-Barg condition [2] by employing simplicial complexes of $\mathbb{F}_2^m$ with two maximal elements. In 2020, Hyun et al. [10]

Z. Hu, X. Zeng and X. Tang are with the Hubei Key Laboratory of Applied Mathematics, Faculty of Mathematics and Statistics, Hubei University, Wuhan, 430062, China. Z. Wang and N. Li are with the Hubei Key Laboratory of Applied Mathematics, School of Cyber Science and Technology, Hubei University, Wuhan, 430062, China. X. Tang is also with the Information Coding & Transmission Key Lab of Sichuan Province, CSNMT Int. Coop. Res. Centre (MoST), Southwest Jiaotong University, Chengdu, 610031, China. Email: zhao.hu@aliyun.com, zhexin.wang@aliyun.com, nian.li@hubu.edu.cn, xiangyongzeng@aliyun.com, xhutang@swjtu.edu.cn

*The full version of this paper has been submitted to the journal IEEE Transactions on Information Theory

constructed infinite families of optimal binary linear codes from the general simplicial complexes of $\mathbb{F}_2^m$ via the defining-set construction. Later, by using simplicial complexes of $\mathbb{F}_2^m$ with one maximal element, several classes of optimal or good binary linear codes with few weights were derived in [11], [16], [18] via different construction approaches. Shortly after, simplicial complexes of $\mathbb{F}_2^m$ with one and two maximal elements were utilized to construct quaternary optimal linear codes in [17], [19] by studying new defining sets of $\mathbb{F}_4^m$. Recently, some researchers also concentrated on linear codes constructed from simplicial complexes of $\mathbb{F}_q^m$ with $q > 2$. Hyun et al. [9] first defined the simplicial complexes of $\mathbb{F}_p^m$ for an odd prime p in 2019, and after that several classes of optimal p-ary few-weight linear codes were constructed in [9], [13], [15] by using different simplicial complexes of $\mathbb{F}_p^m$ with one maximal element. Later, Pan and Liu [12] defined the simplicial complexes of $\mathbb{F}_3^m$ in another way and presented three classes of few-weight ternary codes with good parameters from their defined simplicial complexes of $\mathbb{F}_3^m$ with one and two maximal elements.

In this paper, we first define the simplicial complexes of $\mathbb{F}_q^m$ for any prime power q (see the details in next section) in a different way from the definitions given by [9], [12] for a prime p and $q = 3$ resepctively. Then we employ the general simplicial complexes of $\mathbb{F}_q^m$ to construct projective linear codes $\mathcal{C}$ over $\mathbb{F}_q$ via the defining-set construction. We completely determine the parameters and weight distribution of $\mathcal{C}$. Moreover, we characterize the optimality of this family of linear codes, which shows that many (distance-)optimal codes can be produced from this construction. In addition, by studying the three cases of simplicial complexes of $\mathbb{F}_q^m$ with one, two and three maximal elements respectively, it shows that infinite families of optimal linear codes with few weights are produced from our construction, including Griesmer codes, near Griesmer codes and distance-optimal codes.

II. Preliminaries

In this section, we present some preliminaries which will be used for the subsequent sections.

Here we introduce the concept of simplicial complexes of $\mathbb{F}_q^m$, where q can be any prime power. For two vectors $u = (u_1, u_2, \ldots, u_m)$ and $v = (v_1, v_2, \ldots, v_m)$ in $\mathbb{F}_q^m$, we say that u covers v, denoted $v \preceq u$, if $\mathrm{Supp}(v) \subseteq \mathrm{Supp}(u)$, where $\mathrm{Supp}(u) = \{1 \le i \le m : u_i \neq 0\}$ is the support of u. A subset Δ of $\mathbb{F}_q^m$ is called a simplicial complex if $u \in \Delta$ and $v \preceq u$ imply $v \in \Delta$. An element u in Δ with entries 0 or 1 is said to be maximal if there is no element $v \in \Delta$ such that $\mathrm{Supp}(u)$ is a proper subset of $\mathrm{Supp}(v)$. For a simplicial complex $\Delta \subseteq \mathbb{F}_q^m$, let $\mathcal{F} = \{F_1, F_2, \ldots, F_h\}$ be the set of maximal elements of Δ, where h is the number of maximal elements in Δ and F_i's are maximal elements of Δ. Let $A_i = \mathrm{Supp}(F_i)$ for $1 \le i \le h$, which implies $A_i \subseteq [m] := \{1, 2, \ldots, m\}$. Note that $A_i \setminus A_j \neq \emptyset$ for any $1 \le i \neq j \le h$ by the definition. Let $\mathcal{A} = \{A_1, A_2, \ldots, A_h\}$ be the set of supports of maximal elements of Δ, and $\mathcal{A}$ be called the support of Δ, denoted $\mathrm{Supp}(\Delta) = \mathcal{A}$. Then one can see that a simplicial complex Δ is uniquely generated by $\mathcal{A}$, denoted $\Delta = \langle \mathcal{A} \rangle$. Notice that both the set of maximal elements $\mathcal{F}$ and the support $\mathcal{A}$ of Δ are unique for a fixed simplicial complex Δ. For any set $\mathcal{B}$ consisting of some subsets of $[m]$, we say that a simplicial complex Δ of $\mathbb{F}_q^m$ is generated by $\mathcal{B}$, denoted $\Delta = \langle \mathcal{B} \rangle$, if Δ is the smallest simplicial complex of $\mathbb{F}_q^m$ containing every element in $\mathbb{F}_q^m$ with the support $B \in \mathcal{B}$.

Notice that the above definition of simplicial complexes of $\mathbb{F}_q^m$ is a generalization of the original definition of simplicial complexes of $\mathbb{F}_2^m$ [1], [10], and it is different from the two definitions presented in [9] for $\mathbb{F}_p^m$ and in [12] for $\mathbb{F}_3^m$.

We will construct a large family of linear code from simplicial complexes of $\mathbb{F}_q^m$ via the defining-set construction in this paper. In 2007, Ding and Niederreiter [6] introduced a nice and generic way to construct linear codes via trace functions. Let $D \subset \mathbb{F}_{q^m}$ and define

$$\mathcal{C}_D = \{c_a = (\mathrm{Tr}_q^{q^m}(ax))_{x \in D} : a \in \mathbb{F}_{q^m}\}. \tag{1}$$

Then $\mathcal{C}_D$ is a linear code over $\mathbb{F}_q$ of length $n := |D|$. The set D is called the defining set of $\mathcal{C}_D$ and the above construction is accordingly called the defining-set construction.

The following notation will be used frequently in this paper. Let $0 \le T < q^{m-1}$ be an integer. Then T can be uniquely written as $T = \sum_{j=0}^{m-2} t_j q^j$, where $0 \le t_j \le q-1$ is an integer for $0 \le j \le m-2$. Let $v(T)$ (resp. $u(T)$) denote the smallest (resp. largest) integer in the set $\{0 \le j \le m-2 : t_j \neq 0\}$ and $\ell(T) = \sum_{j=0}^{m-1} t_j$.

III. The projective linear codes over $\mathbb{F}_q$ from the general simplicial complexes

Let Δ be a simplicial complex of $\mathbb{F}_{q^m}$, and Δ^c be the complement of Δ, namely, $\Delta^c = \mathbb{F}_q^m \backslash \Delta$. Notice that if $x \in \Delta$, then $yx \in \Delta$ for any $y \in \mathbb{F}_q^*$ due to the definition of simplicial complexes. Hence for any simplicial complex Δ, Δ^c can be expressed as

$$\Delta^c = \mathbb{F}_q^* \overline{\Delta}^c = \{yz : y \in \mathbb{F}_q^* \text{ and } z \in \overline{\Delta}^c\}$$

where $z_i/z_j \notin \mathbb{F}_q^*$ for distinct elements z_i and z_j in $\overline{\Delta}^c$, and clearly $|\overline{\Delta}^c| = |\Delta^c|/(q-1)$.

In this section, we investigate the projective codes $\mathcal{C}_{\overline{\Delta}^c}$ defined as in (1).

Theorem 1. *Let Δ be a simplicial complex of $\mathbb{F}_{q^m}$ with the support $\mathcal{A} = \{A_1, A_2, \ldots, A_h\}$, where $1 \le |A_1| \le |A_2| \le \cdots \le |A_h| < m$. Assume that $A_i \backslash (\cup_{1 \le j \le h, j \neq i} A_j) \neq \emptyset$ for any $1 \le i \le h$ and $q^m > \sum_{1 \le i \le h} q^{|A_i|}$. Denote $T = \sum_{1 \le i \le h} q^{|A_i|-1}$. Let $\mathcal{C}_{\overline{\Delta}^c}$ be defined as in* (1)*. Then*

1) *$\mathcal{C}_{\overline{\Delta}^c}$ has parameters $[(q^m - |\Delta|)/(q-1), m, q^{m-1} - T]$, where $|\Delta| = \sum_{\emptyset \neq S \subseteq \mathcal{A}} (-1)^{|S|-1} q^{|\cap S|}$ and $\cap S$ is defined as $\cap S = \cap_{A \in S} A$.*
2) *$\mathcal{C}_{\overline{\Delta}^c}$ is a Griesmer code if and only if $|A_i \cap A_j| = 0$ for any $1 \le i < j \le h$ and at most $q-1$ of $|A_i|$'s are the same.*
3) *$\mathcal{C}_{\overline{\Delta}^c}$ is distance-optimal if $|\Delta| - 1 + (q-1)(v(T)+1) > qT - \ell(T)$.*
4) *$\mathcal{C}_{\overline{\Delta}^c}$ has the following weight enumerator*

$$\sum_{\emptyset \neq R \subseteq \Omega} |\Psi_R| z^{q^{m-1} - \sum_{S \in R} (-1)^{|S|-1} q^{|\cap S|-1}} + (q^{m - |\cup_{i=1}^h A_i|} - 1) z^{q^{m-1}} + 1$$

where $\Omega = \{S : S \subseteq \mathcal{A}, S \neq \emptyset\}$ and

$$|\Psi_R| = q^{m - |\cup_{S \in \Omega \backslash R} (\cap S)|} - \sum_{\emptyset \neq E \subseteq R} (-1)^{|E|-1} q^{m - |(\cup_{L \in E} (\cap L)) \cup (\cup_{S \in \Omega \backslash R} (\cap S))|}.$$

Remark 1. *Note that $qT - |\Delta| = \sum_{S \subseteq \mathcal{A}, |S| \ge 2} (-1)^{|S|-1} q^{|\cap S|}$ whose value heavily relies on those of $|A_i \cap A_j|$ for $1 \le i < j \le h$. By the definition, $v(T) \ge |A_1|$ and $\ell(T) \le h$. Thus the condition in 3) of Theorem 1 can be easily satisfied if $|A_1|$ is large enough and $|A_i \cap A_j|$'s are small enough.*

Remark 2. *The given formula in 4) of Theorem 1 to compute the weight distribution of $\mathcal{C}_{\overline{\Delta}^c}$ is completely computable for a given Δ with support $\mathcal{A} = \{A_1, A_2, \ldots, A_h\}$ although the expression seems not so simple. Thus we say that the weight distribution of $\mathcal{C}_{\overline{\Delta}^c}$ is completely determined in Theorem 1.*

In the following corollary, we take a more in-depth discussion on the case that $|A_i \cap A_j| = 0$ for all $1 \le i < j \le h$ for the code $\mathcal{C}_{\overline{\Delta}^c}$ in Theorem 1.

Corollary 1. *Let Δ be a simplicial complex of $\mathbb{F}_{q^m}$ with the support $\mathcal{A} = \{A_1, A_2, \ldots, A_h\}$, where $1 \le |A_1| \le |A_2| \le \cdots \le |A_h| < m$. Assume that $|A_i \cap A_j| = 0$ for $1 \le i < j \le h$. Denote $T = \sum_{1 \le i \le h} q^{|A_i|-1}$. Let $\mathcal{C}_{\overline{\Delta}^c}$ be defined as in* (1)*. Then $\mathcal{C}_{\overline{\Delta}^c}$ is an at most 2^h-weight $[(q^m - \sum_{i=1}^h q^{|A_i|} + h - 1)/(q-1), m, q^{m-1} - T]$ linear code with weight enumerator*

$$\sum_{\emptyset \neq R \subseteq [h]} (q^{m - \sum_{i \in [h] \backslash R} |A_i|} - \sum_{\emptyset \neq E \subseteq R} (-1)^{|E|-1} q^{m - \sum_{i \in E} |A_i| - \sum_{i \in [h] \backslash R} |A_i|}) z^{q^{m-1} - \sum_{i \in R} q^{|A_i|-1}} + (q^{m - \sum_{i=1}^h |A_i|} - 1) z^{q^{m-1}} + 1.$$

Moreover, we have the followings:

1) *$\mathcal{C}_{\overline{\Delta}^c}$ is a Griesmer code if and only if at most $q-1$ of $|A_i|$'s are the same;*
2) *$\mathcal{C}_{\overline{\Delta}^c}$ is a near Griesmer code if and only if $\ell(T) = h - (q-1)$; and*

3) $\mathcal{C}_{\overline{\Delta}^c}$ *is distance-optimal if* $\ell(T)+(q-1)(v(T)+1)>h$. *Specially, when* $|A_i|=\varepsilon$ *for* $1\le i\le h$, *where* ε *is a positive integer, it is distance-optimal if* $\ell(h)+(q-1)(v(h)+\varepsilon)>h$.

Remark 3. *The Griesmer codes in Corollary 1 (or Theorem 1) are indeed the Solomon-Stiffler codes. Definitely, for the other cases (not the Griesmer codes), our codes* $\mathcal{C}_{\overline{\Delta}^c}$ *in Corollary 1 and Theorem 1 are different from the Solomon-Stiffler codes.*

Remark 4. *Notice that the condition in 2) of Corollary 1 can be easily satisfied by selecting proper* A_i*'s. Moreover, the condition* $\ell(T)+(q-1)(v(T)+1)>h$ *for* $\mathcal{C}_{\overline{\Delta}^c}$ *to be distance-optimal can be easily satisfied if* $|A_1|$ *is large enough since* $1\le\ell(T)\le h$ *and* $v(T)\ge|A_1|$, *and consequently many distance-optimal linear codes can be produced in Corollary 1 besides (near) Griesmer codes.*

Next, we give more explicit results on the cases $h=1,2,3$ of Theorem 1.

Corollary 2. *Let* Δ *be a simplicial complex of* $\mathbb{F}_{q^m}$ *with exactly one maximal element and its support is* $\{A\}$ *with* $A\subseteq[m]$ *and* $1\le|A|<m$. *Then* $\mathcal{C}_{\overline{\Delta}^c}$ *defined by* (1) *is a* 2*-weight* $[(q^m-q^{|A|})/(q-1),m,q^{m-1}-q^{|A|-1}]$ *linear code with weight distribution*

Weight w	*Multiplicity* A_w
0	1
q^{m-1}	$q^{m-\|A\|}-1$
$q^{m-1}-q^{\|A\|-1}$	$q^m-q^{m-\|A\|}$

and it is a Griesmer code.

Corollary 3. *Let* Δ *be a simplicial complex of* $\mathbb{F}_{q^m}$ *with the support* $\mathcal{A}=\{A_1,A_2\}$, *where* $1\le|A_1|\le|A_2|<m$. *Assume that* $q^m>q^{|A_1|}+q^{|A_2|}$. *Let* $T=q^{|A_1|-1}+q^{|A_2|-1}$. *Then* $\mathcal{C}_{\overline{\Delta}^c}$ *defined by* (1) *is an at most* 5*-weight* $[(q^m-q^{|A_1|}-q^{|A_2|}+q^{|A_1\cap A_2|})/(q-1),m,q^{m-1}-q^{|A_1|-1}-q^{|A_2|-1}]$ *linear code and its weight distribution is given by*

Weight w	*Multiplicity* A_w
0	1
q^{m-1}	$q^{m-\|A_1\cup A_2\|}-1$
$q^{m-1}-q^{\|A_2\|-1}$	$q^{m-\|A_1\|}-q^{m-\|A_1\cup A_2\|}$
$q^{m-1}-q^{\|A_1\|-1}$	$q^{m-\|A_2\|}-q^{m-\|A_1\cup A_2\|}$
$q^{m-1}-q^{\|A_1\|-1}-q^{\|A_2\|-1}$	$q^{m-\|A_1\cap A_2\|}-q^{m-\|A_1\|}-q^{m-\|A_2\|}+q^{m-\|A_1\cup A_2\|}$
$q^{m-1}-q^{\|A_1\|-1}-q^{\|A_2\|-1}+q^{\|A_1\cap A_2\|-1}$	$q^m-q^{m-\|A_1\cap A_2\|}$

Moreover, we have the followings:

1) *When* $|A_1\cap A_2|=0$ *and* $|A_1|=|A_2|$, $\mathcal{C}_{\overline{\Delta}^c}$ *is a near Griesmer code (also distance-optimal) if* $q=2$ *and it is a Griesmer code if* $q>2$. *It reduces to a* 3*-weight code in this case.*
2) *When* $|A_1\cap A_2|=0$ *and* $|A_1|<|A_2|$, $\mathcal{C}_{\overline{\Delta}^c}$ *is a Griesmer code and it reduces to a* 4*-weight code.*
3) *When* $|A_1\cap A_2|>0$ *and* $|A_1|=|A_2|$, $\mathcal{C}_{\overline{\Delta}^c}$ *is distance-optimal if* $\ell(T)+(q-1)(v(T)+1)>q^{|A_1\cap A_2|}+1$ *and it reduces to a* 4*-weight code. Specially,* $\mathcal{C}_{\overline{\Delta}^c}$ *is a near Griesmer code if* $q>2$ *and* $|A_1\cap A_2|=1$.
4) *When* $|A_1\cap A_2|>0$ *and* $|A_1|<|A_2|$, $\mathcal{C}_{\overline{\Delta}^c}$ *is distance-optimal if* $(q-1)|A_1|+1>q^{|A_1\cap A_2|}$. *Specially,* $\mathcal{C}_{\overline{\Delta}^c}$ *is a near Griesmer code if* $|A_1\cap A_2|=1$.

Corollary 4. *Let* Δ *be a simplicial complex of* $\mathbb{F}_{q^m}$ *with the support* $\mathcal{A}=\{A_1,A_2,A_3\}$, *where* $1\le|A_1|\le|A_2|\le|A_3|<m$. *Assume that* $A_i\backslash(\cup_{1\le j\le3,j\ne i}A_j)\ne\emptyset$ *for any* $1\le i\le3$, *and* $q^m>\sum_{1\le i\le3}q^{|A_i|}$. *Let* $T=\sum_{1\le i\le3}q^{|A_i|-1}$. *Then* $\mathcal{C}_{\overline{\Delta}^c}$ *defined by* (1) *is a* $[(q^m-|\Delta|)/(q-1),m,q^{m-1}-T]$ *linear code, where* $|\Delta|=\sum_{i=1}^3q^{|A_i|}-\sum_{1\le i<j\le3}q^{|A_i\cap A_j|}+q^{|A_1\cap A_2\cap A_3|}$. *Moreover, we have the followings:*

1) $\mathcal{C}_{\overline{\Delta}^c}$ *is a Griesmer code if and only if* $|A_i\cap A_j|=0$ *for* $1\le i<j\le3$ *and at most* $q-1$ *of* $|A_i|$*'s are the same (which always holds for* $q>3$*).*

2) $\mathcal{C}_{\overline{\Delta}^c}$ *is a near Griesmer code if one of the followings holds: i)* $|A_i \cap A_j| = 1$ *for only one element* (i,j) *in the set* $\{(i,j) : 1 \le i < j \le 3\}$ *and* $|A_i \cap A_j| = 0$ *for the other two* (i,j)*'s, and at most* $q-1$ *of* $|A_i|$*'s are the same; ii)* $q = 3$, $|A_i \cap A_j| = 0$ *for* $1 \le i < j \le 3$, *and* $|A_1| = |A_2| = |A_3|$*; and iii)* $q = 2$, $|A_i \cap A_j| = 0$ *for* $1 \le i < j \le 3$, *and* $|A_1| = |A_2| < |A_3| - 1$ *or* $|A_1| \le |A_2| = |A_3|$.

3) $\mathcal{C}_{\overline{\Delta}^c}$ *is distance-optimal if* $(q-1)(v(T)+1) + \ell(T) - 1 > \sum_{1 \le i < j \le 3} q^{|A_i \cap A_j|} - q^{|A_1 \cap A_2 \cap A_3|}$.

Remark 5. *The weight distribution of* $\mathcal{C}_{\overline{\Delta}^c}$ *in Corollary 4 also can be determined by the formula in 4) of Theorem 1, which is at most* 19*-weight.*

IV. Concluding remarks

The main contributions of this paper are summarized as follows:

- We constructed a large family of projective linear codes $\mathcal{C}_{\overline{\Delta}^c}$ over $\mathbb{F}_q$ from the general simplicial complexes Δ of $\mathbb{F}_q^m$ via the defining-set construction. This totally extends the results of [10] from $\mathbb{F}_2$ to $\mathbb{F}_q$. To the best of our knowledge, this paper is the first to study linear codes over $\mathbb{F}_q$ constructed from the general simplicial complexes of $\mathbb{F}_q^m$ for a prime power $q > 2$.
- The parameters and weight distribution of $\mathcal{C}_{\overline{\Delta}^c}$ were completely determined (see Theorem 1) in this paper. Thus this paper also determines the weight distribution of the binary codes constructed from the general simplicial complexes of $\mathbb{F}_2^m$ in [10, Theorem IV.6], in which the weight distribution of the binary codes were studied only for the case of simplicial complexes of $\mathbb{F}_2^m$ with two maximal elements. Moreover, as a byproduct, the weight distributions of the Solomon-Stiffler codes are determined in Corollary 1 for the case that the corresponding subspaces in $\mathbb{F}_q^m$ of the projective subspaces U_i are spanned by some subsets of a certain basis of $\mathbb{F}_q^m$.
- By using the Griesmer bound, we gave a necessary and sufficient condition such that $\mathcal{C}_{\overline{\Delta}^c}$ is a Griesmer code and a sufficient condition such that $\mathcal{C}_{\overline{\Delta}^c}$ is distance-optimal. In addition, we also presented a necessary and sufficient condition for $\mathcal{C}_{\overline{\Delta}^c}$ to be a near Griesmer code in a special case. This shows that many infinite families of (distance-)optimal linear codes can be produced from our construction.
- By studying the cases of the simplicial complexes Δ with one, two and three maximal elements respectively, we derived infinite families of optimal linear codes with few weights over $\mathbb{F}_q$ including Griesmer codes, near Griesmer codes and distance-optimal codes.

References

[1] M. Adamaszek, Face numbers of down-sets, Amer. Math. Monthly 122(4) (2015), pp. 367-370.
[2] A. Ashikhmin, A. Barg, Minimal vectors in linear codes, IEEE Trans. Inf. Theory 44(5) (1998), pp. 2010-2017.
[3] S. Chang, J.Y. Hyun, Linear codes from simplicial complexes, Des. Codes Cryptogr. 86 (2018), pp. 2167-2181.
[4] C. Ding, Codes from Difference Sets, World Scientific, Singapore (2015).
[5] C. Ding, Designs from Linear Codes, World Scientific, Singapore (2018).
[6] C. Ding, H. Niederreiter, Cyclotomic linear codes of order 3, IEEE Trans. Inf. Theory 53(6) (2007), pp. 2274-2277.
[7] J.H. Griesmer, A bound for error correcting codes, IBM J. Res. Dev. 4 (1960), pp. 532-542.
[8] W. Huffman, V. Pless, Fundamentals of error-correcting codes, Cambridge University Press (1997).
[9] J.Y. Hyun, H.K. Kim, M. Na, Optimal non-projective linear codes constructed from down-sets, Discrete Appl. Math. 254 (2019), pp. 135-145.
[10] J.Y. Hyun, J. Lee, Y. Lee, Infinite families of optimal linear codes constructed from simplicial complexes, IEEE Trans. Inf. Theory 66(11) (2020), pp. 6762-6773.
[11] X. Li, M. Shi, A new family of optimal binary few-weight codes from simplicial complexes, IEEE Communications Letters 25(4) (2021), pp. 1048-1051.
[12] Y. Pan, Y. Liu, New classes of few-weight ternary codes from simplicial complexes, AIMS Math. 7(3) (2022), pp. 4315-4325.
[13] M. Shi, X. Li, Two classes of optimal p-ary few-weight codes from down-sets, Discret. Appl. Math. 290 (2021), pp. 60-67.
[14] G. Solomon, J.J. Stiffler, Algebraically punctured cyclic codes, Inform. and Control 8 (1965), pp. 170-179.
[15] Y. Wu, J.Y. Hyun, Few-weight codes over $\mathbb{F}_p + u\mathbb{F}_p$ associated with down sets and their distance optimal Gray image, Discret. Appl. Math. 283 (2020), pp. 315-322.
[16] Y. Wu, Y. Lee, Binary LCD codes and self-orthogonal codes via simplicial complexes, IEEE Communications Letters 24(6) (2020), pp. 1159-1162.
[17] Y. Wu, C. Li, F. Xiao, Quaternary linear codes and related binary subfield codes, IEEE Trans. Inf. Theory 68(5) (2022), pp. 3070-3080.
[18] Y. Wu, X. Zhu, Q. Yue, Optimal few-weight codes from simplicial complexes, IEEE Trans. Inf. Theory 66(6) (2020), pp. 3657-3663.
[19] X. Zhu, Y. Wei, Few-weight quaternary codes via simplicial complexes, AIMS Math. 6(5) (2021), pp. 5124-5132.

Stability of $x^3 + x^2 + 1$ from the perspective of periodic sequences

Tong Lin[1] Qiang Wang[1]

Abstract

We have recently proved [10] the conjecture by Ahmadi and Monsef-Shokri [2] that $f(x) = x^3 + x^2 + 1$ is stable over $\mathbb{F}_2$. In this paper, we introduce a periodic sequence $(S_{k,n,i})_{i\geq -1}$ for each $k \in \mathbb{N}, n \in \mathbb{N}_0$ satisfying a non-linear recurrence relation, and establish connections between the stability of f over $\mathbb{F}_{2^k}$ and properties of $(S_{k,n,i})_{i\geq -1}$ (namely, its recurrence relations, least period and distribution of zero terms). We also give equivalent characterizations of the roots of $(f_{k,n})_{n\geq 0}$ as well as closed-form formulas for $(S_{k,n,i})_{i\geq -1}$ in terms of the Fibonacci sequence.

1 Introduction and main results

We say a polynomial $t(x) \in \mathbb{K}[x]$, where $\mathbb{K}$ is a field, is stable over $\mathbb{K}$ if for each $n \in \mathbb{N}$, the n-th iterate $t^{(n)}(x) = t(t(\ldots t(t(x))))$ of t is irreducible over $\mathbb{K}$. Problems concerning stability of polynomials over fields date back to the 1980s, when Odoni came up with one of the first examples [11, Proposition 4.1] and one of the first counter-examples [12, Corollary 1.6], respectively, of stable polynomials over a field. Stability of polynomials, especially those of low degrees, over various fields have been extensively studied ever since.

In 2012, Jones and Boston [8, Proposition 2.3] gave necessary and sufficient conditions for a quadratic polynomial to be stable over a finite field of odd characteristic in terms of the so-called adjusted critical orbits (using which Ostafe and Shparlinski [13, Corollary 2] estimated the complexity of testing stability of quadratic polynomials over a finite field of odd characteristic.) Then Ahmadi et al. [1, Theorem 4, Corollary 11] showed that *almost all* monic quadratic polynomials in $\mathbb{Z}[x]$ are stable over $\mathbb{Q}$ and that no quadratic polynomial is stable over a finite field of characteristic 2. In 2014, Goméz-Pérez and Nicolás, in collaboration with Ostafe and Sardonil [6, Theorem 5.5], estimated the number of stable polynomials of any degree $d \in \mathbb{N}$ over a finite field of odd characteristic.

When it comes to polynomials of degree greater than 2, determining whether they are stable over a field is more sophisticated than in the quadratic case. It is conjectured in [2, Conjecture 14] that $f(x) = x^3 + x^2 + 1$ is stable over $\mathbb{F}_2$, and a stability test based on *Capelli's Lemma* is proposed.

[1]School of Mathematics and Statistics, Carleton University, 1125 Colonel By Drive, Ottawa ON K1S 5B6, Canada.
The authors were supported by the Natural Sciences and Engineering Research Council of Canada (RGPIN-2017-06410).
E-mail addresses: tonglin4@cmail.carleton.ca (T. Lin), wang@math.carleton.ca (Q. Wang).

Lemma 1.1 ([2, **Lemma 13**]). *Let $q > 1$ be a prime power, and let $F(x) \in \mathbb{F}_q[x]$ be an irreducible polynomial of degree $d \in \mathbb{N}$. If $G(x) \in \mathbb{F}_q[x]$, then $F(G(x))$ is irreducible over $\mathbb{F}_q$ iff $G(x) - \alpha$ is irreducible over $\mathbb{F}_{q^d} \cong \mathbb{F}_q[x]/\langle F(x)\rangle$, where α is a root of $F(x)$ in $\mathbb{F}_{q^d}$.*

Let $k \in \mathbb{N}$. Using the above result, we construct a sequence $(\alpha_{k,n})_{n\geq 0}$ such that for each $n \in \mathbb{N}_0$, $\alpha_{k,n}$ is a root of $f^{(n)}$ in $\mathbb{F}_{2^{3^n k}}$ and that $f(\alpha_{k,n+1}) = \alpha_{k,n}$. Two new sequences $(\beta_{k,n})_{n\geq 0}$ and $(f_{k,n})_{n\geq 0}$ arise from $(\alpha_{k,n})_{n\geq 0}$. More precisely,

$$\beta_{k,n} = 1 + \alpha_{k,n} \in \mathbb{F}_{2^{3^n k}} \tag{1}$$

$$f_{k,n}(x) = x^3 + x + \beta_{k,n} \tag{2}$$

In [10], with the help of the above-mentioned sequences, we proved the following result having [2, Conjecture 14] as a special case.

Theorem 1.2. *Let $k \in \mathbb{N}$.*

(1) If $3 \nmid k$, then $f_{k,n}$ is irreducible over $\mathbb{F}_{2^{3^n k}}$ for each $n \in \mathbb{N}_0$. In particular, $f(x) = x^3 + x^2 + 1$ is stable over $\mathbb{F}_{2^k}$.

(2) If $3 \mid k$, then $f_{k,n}$ splits completely into linear factors over $\mathbb{F}_{2^{3^n k}}$ for each $n \in \mathbb{N}_0$.

We note that for each $k \in \mathbb{N}, n \in \mathbb{N}_0$, $xf_{k,n}(x) = x^4 + x^2 + \beta_{k,n}x$ is a linearized polynomial over $\mathbb{F}_{2^{3^n k}}$. From works in [7] and [14, Corollary 4] on inverses of linearized polynomials, we construct a sequence $(S_{k,n,i})_{i\geq -1}$, where

(1) $S_{k,n,-1} = 0$ and $S_{k,n,0} = 1$;

(2) $S_{k,n,i} = S_{k,n,i-1} + \beta_{k,n}^{2^{i-1}} S_{k,n,i-2}$.

Remark 1.3. *We note that every three consecutive terms in $(S_{k,n,i})_{i\geq -1}$ satisfy a different non-linear relation. However, $(S_{k,n,i})_{i\geq -1}$ can be defined by means of a single non-linear recurrence relation, namely, for each $i \in \mathbb{N}$,*

$$S_{k,n,i} = S_{k,n,i-1}^2 + \beta_{k,n}^2 S_{k,n,i-2}^4 \tag{3}$$

To view stability of f over $\mathbb{F}_{2^k}$ (or equivalently, irreducibility of $(f_{k,n})_{n\geq 0}$) from the perspective of $(S_{k,n,i})_{i\geq -1}$, we present our main results.

Theorem 1.4. *Let $k \in \mathbb{N}$ be odd. For each $n \in \mathbb{N}_0$, $(S_{k,n,i})_{i\geq -1}$ is periodic, and if $t_{k,n}$ is its least period, then the following are equivalent.*

(1) $f_{k,n}$ is irreducible over $\mathbb{F}_{2^{3^n k}}$;

(2) $xf_{k,n}(x)$ is a permutation polynomial over $\mathbb{F}_{2^{3^n k}}$;

(3) $S_{k,n,3^n k} + \beta_{k,n} S_{k,n,3^n k-2}^2 = 1$;

(4) $S_{k,n,3^n k-1} \neq 0$;

(5) $t_{k,n} = 3^{n+1}k$;

(6) $3 \nmid k$.

Moreover, f is stable over $\mathbb{F}_{2^k}$ iff for each $n \in \mathbb{N}_0$, any of the above conditions holds.

We remark that for general $k \in \mathbb{N}$, (1), (2), (3), (4), (6) are still equivalent and (5) implies all of them.

2 Properties of $(S_{k,n,i})_{i\geq -1}$

In order to structurally understand the solutions to the equation $x^{2^\ell+1}+x+a=0$ in $\mathbb{F}_{2^m}$, where $\ell<m$ are positive integers and $a\in\mathbb{F}_{2^m}^*$, a sequence of polynomials $(C_i(x))_{i=1}^{r+1}$, where $m=rd$ and $d=\gcd(\ell,m)$, defined over $\overline{\mathbb{F}_2}$ is introduced in [7, Equation (5)]. (We also note that a more general sequence is studied in [9].)

(1) $C_1(x)=C_2(x)=1$;

(2) $C_{i+2}(x)=C_{i+1}(x)+x^{2^{i\ell}}C_i(x)$ $(1\leq i\leq r-1)$.

Clearly, $(C_i(x))_{i=1}^{r+1}$ can be extended to an infinite sequence satisfying the above relations. Let $C_0(x)=0$. Let $k\in\mathbb{N}, n\in\mathbb{N}_0$. When $\ell=d=1$ and $m=r=3^nk$, induction yields that $S_{k,n,i}=C_{i+1}(\beta_{k,n})$. Moreover, the following results follow immediately from properties of $(C_i(x))_{i\geq 0}$.

Proposition 2.1. *For each $i\in\mathbb{N}$,*

(1) $S_{k,n,i}=S_{k,n,i-1}^2+\beta_{k,n}^2S_{k,n,i-2}^4$;

(2) $\beta_{k,n+1}^{2^i}=S_{k,n,i-1}\beta_{k,n+1}^2+\left(S_{k,n,i-2}^2\beta_{k,n}\right)\beta_{k,n+1}$;

(3) $S_{k,n,m}+\beta_{k,n}S_{k,n,m-2}^2\in\mathbb{F}_2$.

As a consequence of the above results, one can show that $(S_{k,n,i})_{i\geq -1}$ is periodic. For each $n\in\mathbb{N}_0$, let $\mathbb{F}_{2^{r_{k,n}}}$ be the smallest subfield of $\mathbb{F}_{2^{3^nk}}$ containing $\beta_{k,n}$.

Proposition 2.2. *For each $n\in\mathbb{N}_0$,*

(1) $r_{k,n+1}=r_{k,n}$ *or* $3r_{k,n}$;

(2) if $r_{k,n}<r_{k,n+1}$, *then* $(S_{k,n,i})_{i\geq -1}$ *is of least period* $r_{k,n+1}$;

(3) if $r_{k,n}=r_{k,n+1}$, *then* $S_{k,n,r_{k,n}}=1$ *or* $\beta_{k,n}^{-1}\beta_{k,n+1}$;

(4) if $r_{k,n}=r_{k,n+1}, S_{k,n,r_{k,n}}=1$, *then* $(S_{k,n,i})_{i\geq -1}$ *is of least period* $r_{k,n}$;

(5) if $r_{k,n}=r_{k,n+1}, S_{k,n,r_{k,n}}=\beta_{k,n}^{-1}\beta_{k,n+1}$, *then* $(S_{k,n,i})_{i\geq -1}$ *is of least period* $2r_{k,n}$.

While studying solutions to $x^3+x+a=0$, where $a\in\mathbb{F}_{2^m}^*$ for some $m\in\mathbb{N}$, Berlekamp et al. constructed the following polynomial sequence $(P_i(x))_{i\geq 1}$, which turns out to be also closely related to $(S_{k,n,i})_{i\geq -1}$.

Theorem 2.3. *[4, Theorem 4] Let $m\in\mathbb{N}$ and $a\in\mathbb{F}_{2^m}^*$. The polynomial x^3+x+a splits completely into linear factors over $\mathbb{F}_{2^m}$ iff $P_m(a)=0$, where*

(1) $P_1(x)=P_2(x)=x$;

(2) $P_i(x)=P_{i-1}(x)+x^{2^{i-3}}P_{i-2}(x)$ *for each* $i\geq 3$.

In fact, if we add an initial term $P_0(x)=0$ to $(P_i(x))_{i\geq 1}$, then it is easy to see that the extended sequence $(P_i(x))_{i\geq 0}$ satisfies the above relations. By induction, the following holds.

Proposition 2.4. *For each $k\in\mathbb{N}, n,t\in\mathbb{N}_0$ and each $i\in\mathbb{N}_0\cup\{-1\}$,*

$$S_{k,n,i}^{2^{t-1}}=\beta_{k,n}^{-2^t}P_{i+1}\left(\beta_{k,n}^{2^t}\right) \tag{4}$$

Together, these propositions lead to Theorem 1.4.

3 Formulas for $(S_{k,n,i})_{i\geq -1}$

Let $k \in \mathbb{N}, n \in \mathbb{N}_0$. We give three closed-form formulas for $(S_{k,n,i})_{i\geq -1}$.

Proposition 3.1. *For each $i \in \mathbb{N}_0$, if $m = \left\lfloor \frac{i}{2} \right\rfloor$, then*

$$S_{k,n,i} = 1 + \sum_{j_1=1}^{i-1} \beta_{k,n}^{2^{j_1}} + \sum_{j_2=3}^{i-1} \sum_{j_1=1}^{j_2-2} \beta_{k,n}^{2^{j_1}+2^{j_2}} + \cdots + \sum_{j_m=2m-1}^{i-1} \cdots \sum_{j_1=1}^{j_2-2} \beta_{k,n}^{2^{j_1}+\cdots+2^{j_m}} \tag{5}$$

In fact, this result follows from a property of $(P_i(x))_{i\geq 1}$. Let $(B_i)_{i\geq 0}$ be such that $B_0 = 0$ and that the subsequence $(B_i)_{i\geq 1}$ is the ascending sequence of positive integers whose binary representations start with 1 and contain no consecutive 1's. Let $(F_i)_{i\geq 0}$ be the Fibonacci sequence. Then Eq. (5) is equivalent to the following.

Proposition 3.2. *For each $i \in \mathbb{N}_0 \cup \{-1\}$,*

$$S_{k,n,i} = \sum_{j=0}^{F_{i+1}-1} \beta_{k,n}^{2B_j} \tag{6}$$

A third formula of $(S_{k,n,i})_{i\geq -1}$ as a polynomial in $\beta_{k,n}^{-1}$ can also be derived to reduce computational complexity that comes with the usage of Eq. (6). Let $C_0 = 0$ and $(C_j)_{j\geq 1} = (1, 3, 5, 7, 11, \dots)$ be the ascending sequence of positive integers whose binary representations begin and end with 1 and contain no consecutive 0's.

Proposition 3.3. *If $T \in \mathbb{N}$ is a period of $(S_{k,n,i})_{i\geq -1}$, then*

$$S_{k,n,T-i} = \sum_{j=F_i}^{F_{i+1}-1} \beta_{k,n}^{-C_j 2^{T-(i-1)}} \qquad (0 \leq i \leq T) \tag{7}$$

4 Characterization of roots of $(f_{k,n})_{n\geq 0}$

Let $k \in \mathbb{N}$. In view of Theorem 1.2, studying stability of f over $\mathbb{F}_{2^k}$ is equivalent to determining whether $f_{k,n}$ is irreducible over $\mathbb{F}_{2^{3^n k}}$ for each $n \in \mathbb{N}_0$. When $f_{k,n}$ is reducible over $\mathbb{F}_{2^{3^n k}}$, it is natural to ask what its roots are in $\mathbb{F}_{2^{3^n k}}$. Using the fact that $\beta_{k,n+1}^3 + \beta_{k,n+1} = \beta_{k,n}$ and that $\mathrm{Tr}_{3^n k}\left(\beta_{k,n}^{-1}\right) = \mathrm{Tr}_{3^n k}(1)$, one can show that if $f_{k,n}$ has a root in $\mathbb{F}_{2^{3^n k}}$, then it splits completely into linear factors over $\mathbb{F}_{2^{3^n k}}$. [10]

Remark 4.1. *According to [3, Equations 8, 9], we note that $f_{k,n}$ splits completely into linear factors over $\mathbb{F}_{2^{3^n k}}$ iff there exists some $v \in \mathbb{F}_{2^{3^n k}} \setminus \mathbb{F}_{2^2}$ such that*

$$\beta_{k,n} = \frac{v + v^{-1}}{(1 + v + v^{-1})^3} \tag{8}$$

If Eq. (8) is satisfied, then the roots of $f_{k,n}$ in $\mathbb{F}_{2^{3^n k}}$ are

$$x_0 = \frac{v + v^{-1}}{1 + v + v^{-1}}, \qquad x_1 = \frac{v}{1 + v + v^{-1}}, \qquad x_2 = \frac{v^{-1}}{1 + v + v^{-1}} \tag{9}$$

Alternatively, if x_0 is a root of $f_{k,n}$ in $\mathbb{F}_{2^{3^n k}}$, then

$$f_{k,n}(x) = (x + x_0)\left(x^2 + x_0 x + \left(x_0^2 + 1\right)\right) \tag{10}$$

where the quadratic factor have two roots in $\mathbb{F}_{2^{3^n k}}$. Then by Vieta's formulas, the three roots of $f_{k,n}$ in $\mathbb{F}_{2^{3^n k}}$ are $x_0, u^2 x_0$ and $\left(1 + u^2\right) x_0$. The two characterizations are equivalent, and the latter in fact follows from [5, Theorem 2.5], [9, Theorem 8].

References

[1] O. Ahmadi, F. Luca, A. Ostafe, I.E. Shparlinski, On stable quadratic polynomials, *Glasgow Mathematical Journal.* **54**(2): 359–369, 2012.

[2] O. Ahmadi, K. Monsef-Shokri, A note on the stability of trinomials over finite fields, *Finite fields and Their Applications.* **63**: 101649, 2020.

[3] E.R. Berlekamp, H. Rumsey, G. Solomon, Solutions of algebraic equations over fields of characteristic 2, *JPL Space Program Summary.* **IV**(37–39), 1966.

[4] E.R. Berlekamp, H. Rumsey, G. Solomon, On the solution of algebraic equations over finite fields, *Information and Control.* **10**(6): 553–564, 1967.

[5] A.W. Bluher, On $x^{q+1} + ax + b$, *Finite fields and Their Applications.* **10**(3): 285–305, 2004.

[6] D. Goméz-Pérez, A.P. Nicolás, A. Ostafe, D. Sardonil, Stable polynomials over finite fields, *Revista Matemática Iberoamericana.* **30**(2), 523–535, 2014.

[7] T. Helleseth, A. Kholosha, $x^{2^\ell+1}+x+a = 0$ and related affine polynomials over GF $\left(2^k\right)$, *Cryptography and Communications.* **2**: 85–109, 2010.

[8] R. Jones, N. Boston, Settled polynomials over finite fields, *Proceedings of the American Mathematical Society.* **140**(6): 1849–1863, 2012.

[9] K.H. Kim, J. Choe, S. Mesnager, Solving $X^{q+1} + X + a$ over finite fields. *Finite Fields and Their Applications.* **70**: 101797, 2021.

[10] T. Lin, Q. Wang, On stability of $x^3 + x^2 + 1$, *https://arxiv.org/abs/2304.03992.*

[11] R.W.K. Odoni, On the prime divisors of the sequence $w_{n+1} = 1 + w_1 \dots w_n$, *Journal of the London Mathematical Society. (Ser. 2)* **32**(1), 1–11, 1985.

[12] R.W.K. Odoni, The Galois theory of iterates and composites of polynomials, *Proceedings of the London Mathematical Society. (Ser. 3)* **51**(3) 385–414, 1985.

[13] A. Ostafe, I.E. Shparlinski, On the length of critical orbits of stable quadratic polynomials, *Proceedings of the American Mathematical Society.* **138**(8), 2653—2656, 2010.

[14] Y. Zheng, Q. Wang, W. Wei, On inverses of permutation polynomials of small degree over finite fields, *IEEE Transactions on Information Theory.* 66(2), 914–922, 2020.