Algebraic Immunity of Boolean Functions – Analysis and Construction*

Inmunidad Algebraica de Funciones Booleanas – Análisis y Construcción

Deepak Kumar Dalai[1] **and Subhamoy Maitra**[2]

[1] Department of Mathematics
National Institute of Science Education and Research,
Sachivalay Marg, Bhubaneswar 751005 INDIA
deepak@iopb.res.in
[2] Applied Statistics Unit, Indian Statistical Institute
203 B T Road, Calcutta 700108, INDIA
subho@isical.ac.in

Article received on March 1, 2008, accepted on June 14, 2008

Abstract
In this paper, we first analyse the method of finding algebraic immunity of a Boolean function. Given a Boolean function f on n-variables, we identify a reduced set of homogeneous linear equations by solving which one can decide whether there exist annihilators of f at a specific degree. Moreover, we analyse how an affine transformation on the input variables of f can be exploited to achieve further reduction in the set of homogeneous linear equations. Next, from the design point of view, we construct balanced Boolean functions with maximum possible AI with an additional property which is necessary to resist the fast algebraic attack.
Keywords: Algebraic Attacks, Algebraic Normal Form, Annihilators, Boolean Functions, Fast Algebraic Attacks, Homogeneous Linear Equations.

Resumen
En este artículo, analizamos primero el método que permite encontrar la inmunidad algebraica de una función Booleana. Dada una función Booleana f de n variables, identificamos un conjunto reducido de ecuaciones lineales homogéneas resolviendo cuál de ellas puede ser usada para determinar si existen nulificadores de f de un grado específico. Además analizamos cómo una transformación afín de las variables de entrada de f puede ser aplicada para alcanzar una mayor reducción en el conjunto de ecuaciones lineales homogéneas. En seguida, y analizando desde el punto de vista de diseño, construimos funciones Booleanas balanceadas con inmunidad algebraica máxima y una propiedad adicional necesaria para resistir versiones rápidas de ataques algebraicos.
Palabras Claves: Ataques algebraicos, froma normal algebraica, nulificadores, funciones Booleanas, ataques algebracios rápidos, ecuaciones lineales homogéneas.

1 Introduction

Results on algebraic and fast algebraic attacks have received a lot of attention recently in studying the security of cryptosystems (Armknecht 2004; Batten 2004; Canteaut 2005; Cheon and Lee 2004; Cho and Pieprizyk 2004; Courtois

*This is a substantially revised and merged version of two conference papers. (i) "Reducing the Number of Homogeneous Linear Equations in Finding Annihilators", in *Sequences and Their Applications, SETA '06*, pages 376–390, volume 4086, Lecture Notes in Computer Science, Springer Verlag, 2006. Section 3.1 and Appendix A are added over the conference version. (ii) "Balanced Boolean Functions with (more than) Maximum Algebraic Immunity", in *International Workshop on Coding and Cryptography, WCC '07*, pages 99–108, INRIA, Rocquencourt, France in April 16–20, 2007. The proceedings of WCC '07 is only a workshop record and it is not printed by any publisher.

and Pieprzyk 2002; Courtois and Meier 2003; Courtois 2003; Lee, Kim, Hong, Han, and Moon 2004; Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006; Didier and Tillich 2006; Courtois, Debraize, and Garrido 2005). Boolean functions are important primitives to be used in the cryptosystems and in view of the algebraic attacks, one should concentrate on the annihilators (Braeken and Praneel 2005; Dalai, Gupta, and Maitra 2004; Dalai, Gupta, and Maitra 2005; Dalai, Maitra, and Sarkar 2006; Meier, Pasalic, and Carlet 2004; Nawaz, Gong, and Gupta 2006).

Let B_n be the set of all Boolean functions $\{0,1\}^n \rightarrow \{0,1\}$ on n input variables. One may refer to (Dalai, Gupta, and Maitra 2004) for detailed definitions related to Boolean functions, e.g., truth table, algebraic normal form (ANF), weight (wt), support ($supp$), nonlinearity (nl) and Walsh spectrum of a Boolean function. Any Boolean function can be represented as a multivariate polynomial over $GF(2)$, called the algebraic normal form (ANF), as

$$f(x_1, \ldots, x_n) = a_0 + \sum_{1 \leq i \leq n} a_i x_i + \sum_{1 \leq i < j \leq n} a_{i,j} x_i x_j + \ldots + a_{1,2,\ldots,n} x_1 x_2 \ldots x_n,$$

where the coefficients $a_0, a_i, a_{i,j}, \ldots, a_{1,2,\ldots,n} \in \{0,1\}$. The algebraic degree, $\deg(f)$, is the number of variables in the highest order term with nonzero coefficient.

Given $f \in B_n$, a nonzero function $g \in B_n$ is called an annihilator of f if $fg = 0$. A function f should not be used if f or $1 + f$ has a low degree annihilator. In this regard, an important property of Boolean function is defined as algebraic immunity (in short, AI) (Meier, Pasalic, and Carlet 2004) (annihilator immunity (Dalai, Maitra, and Sarkar 2006)) as follows.

Definition 1 *Given $f \in B_n$, its algebraic immunity is defined as (Meier, Pasalic, and Carlet 2004) the minimum degree of all annihilators of f or $1 + f$, and it is denoted by $\mathcal{AI}(f)$.*

It is also known (Courtois and Meier 2003; Meier, Pasalic, and Carlet 2004) that for any function f or $1 + f$ must have an annihilator at the degree $\lceil \frac{n}{2} \rceil$ i.e., $\mathcal{AI}(f) \leq \lceil \frac{n}{2} \rceil$.

The target of a good design is to use a function f such that neither f nor $1+f$ has an annihilator at a degree less than $\lceil \frac{n}{2} \rceil$. The first construction in this direction appeared in (Dalai, Gupta, and Maitra 2005). Later symmetric functions with this property has been presented in (Dalai, Maitra, and Sarkar 2006; Braeken and Praneel 2005). However, all these constructions are not good in terms of other cryptographic properties.

In this situation, one needs to consider Boolean functions which are rich in terms of other cryptographic properties, and then the AI of the functions has to be checked. One has to find out the annihilators of a given Boolean function for this. Initially a basic algorithm in finding the annihilators has been proposed in (Meier, Pasalic, and Carlet 2004, Algorithm 2). A modification of (Meier, Pasalic, and Carlet 2004, Algorithm 2) has been presented in (Braeken, Lano, and Praneel 2006) to find out relationships for algebraic and fast algebraic attacks. In (Braeken and Praneel 2005), there is an efficient algorithm to find the annihilators of symmetric Boolean functions. Algorithms using Gröbner bases are also interesting in this area (Ars and Faugére 2005), but they are not considerably consistent. Recently more efficient algorithms have been designed in this direction (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006; Didier and Tillich 2006; Didier 2006). The algorithm presented in (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006) can be used efficiently to find out relationships for algebraic and fast algebraic attacks. In (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006), polynomial interpolation has been proposed to solve the annihilator finding problem (of degree d for an n-variable function) in $O(wt(f)\binom{n}{d})$ time complexity. In (Didier and Tillich 2006) a probabilistic algorithm having time complexity $O(n^d)$ has been proposed where the function is divided to its sub functions recursively and the annihilators of the sub functions are checked to study the annihilators of the original function. Using Weidmann's algorithm, a space efficient probabilistic algorithm having $O(n2^n\binom{n}{d})$ time complexity and $O(n2^n)$ space complexity has been proposed in (Didier 2006).

The main idea in our effort is to reduce the size of the matrix (used to solve the system of homogeneous linear equations) as far as possible, which has not yet been studied in a disciplined manner to the best of our knowledge. In the process, some nice structures of the associated matrices could be discovered in this paper. efficiently to find out relationships for algebraic and fast algebraic attacks.

Given a Boolean function f on n-variables, we find a reduced set of homogeneous linear equations by solving which one can decide whether there exist annihilators at degree d or not. Using our method the size of the associated matrix becomes $\nu_f^d \times (\sum_{i=0}^{d} \binom{n}{i} - \mu_f^d)$, where, $\nu_f^d = |\{x|wt(x) > d, f(x) = 1\}|$ and $\mu_f^d = |\{x|wt(x) \leq d, f(x) = 1\}|$ and the required time to construct the matrix is same as the size of the matrix. This is a preprocessing step before the solution to decide on the existence of the annihilators that requires to solve the set of homogeneous linear equations.

We start with an involutary matrix $M_{n,d}(g)$ (see Theorem 1) and we discover certain structures that allow to compute the new equations efficiently by considering the matrix UA^r (see Theorem 3, Section 3). Moreover, each equation associated with a low weight input point directly provides the value of an unknown coefficient of the annihilator, which is the key point that allows to lower the number of unknowns.

Further reduction in the size of the matrix is dependent on getting a proper linear transformation on the input variables of the Boolean function, which is discussed in Section 4. As the affine transformation on the input variables of the Boolean function keeps the degree of the annihilators invariant, our preprocessing step can be more efficiently applied if one can find an affine transformation over $f(x)$ to get $h(x) = f(Bx + b)$ such that μ_h^d is maximized (and in turn ν_h^d is minimized too). We present an efficient heuristic towards this. Our study identifies for what kind of Boolean functions the asymptotic reduction in the size of the matrix is possible.

Our contribution here is two-fold.

1. We prove new algebraic and combinatorial results related to the matrix structure in finding annihilators.
2. For certain class of functions, our technique finds the annihilators more efficiently than (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006), the currently best known general algorithm.

We do not claim that our algorithm is better than (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006; Didier 2006; Didier and Tillich 2006) which work for any Boolean function in general. Our observation identifies a subclass of Boolean functions for which our technique presents the currently best known results.

One should note that a Boolean function, to be used in a cryptosystem, should not have low AI. Good AI provides certain kind of resistance against algebraic attacks done in a particular way, i.e., using linearization. Further, based on some recent works related to fast algebraic attacks (Armknecht and Krause 2003; Courtois 2003; Braeken, Lano, and Praneel 2005; Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006), one should concentrate more carefully on the design parameters of Boolean functions for proper resistance. The weakness of AI against fast algebraic attack has been demonstrated in (Courtois 2005) by mounting an attack on SFINKS (Braeken, Lano, Mentens, Praneel, and Verbauwhede 2005).

Let us now discuss the situation with respect to fast algebraic attack. It has been shown in (Courtois 2003) that given any n-variable Boolean function f, it is always possible to get a Boolean function g with degree at most $\lceil \frac{n}{2} \rceil$ such that $\deg(g) + \deg(h) \leq n$. Thus, while choosing a function f, the cryptosystem designer should be careful that it should not happen that $\deg(g) + \deg(h)$ falls much below n with a nonzero function g whose degree is also much below $\lceil \frac{n}{2} \rceil$. In that case the lower degree of g can be exploited to a faster attack (known as fast algebraic attack).

Take $f \in B_n$ with maximum possible AI $\lceil \frac{n}{2} \rceil$. It may very well happen that $fg = h$, where $\deg(h) = \lceil \frac{n}{2} \rceil$, but $\deg(g) < \lceil \frac{n}{2} \rceil$. In that case the lower degree of g may be exploited to mount the fast algebraic attack even if the AI of f is the maximum possible. In fact, there are examples, where one can get a linear g too. Initial study of Boolean functions in this area has been started in (Braeken, Lano, and Praneel 2005; Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006). Since AI is now understood as a necessary (but not sufficient) condition against resisting algebraic attacks, we feel there is a need to consider the functions with full AI for their performance in terms of $fg = h$ relationship. That is for the functions f with full AI we consider $\deg(h) \geq \lceil \frac{n}{2} \rceil$, and then after fixing the degree of h, we try to get the minimum degree g. Even after this concept, the necessary condition of using functions with maximum possible AI stays, but one needs to check the profile of the functions for other $fg = h$ relations before using that in a cryptosystem. One should be aware that only checking these $fg = h$ relationships are not sufficient in terms of resistance to (fast) algebraic attacks as there are number of scenarios to mount algebraic and fast algebraic attacks (Courtois and Meier 2003; Courtois 2003).

It is always meaningful to consider $fg = h$ only when $\deg(g) \leq \deg(h)$ as otherwise $fg = h$ implies $(1+f)h = 0$. So for all the discussion we will consider $\deg(g) \leq \deg(h)$ for a relation $fg = h$ unless mentioned otherwise.

In this paper, we present a specific class of balanced functions f for even number of input variables n having AI $\frac{n}{2}$ such that for any $fg = h$ relation if $\deg(h) = \frac{n}{2}$ then $\deg(g)$ cannot be less than $\frac{n}{2}$. This class of functions was not known earlier. Further we show that existence of these functions has direct implication towards existence of resilient functions with maximum possible algebraic immunity. A few important open questions are also raised based on our work. The main contribution regarding the construction is presented in Subsection 5.3.

Since the functions we present are modifications of symmetric functions, we do not refer them directly to be used in a cryptosystem. Our main motivation in the construction part is to present new theoretical results related to construction of Boolean functions that were not known earlier. Our ideas may be exploited towards further effort in construction of cryptographically significant Boolean functions resistant against algebraic and fast algebraic attacks.

2 Preliminaries

Consider all the n-variable Boolean functions of degree at most d, i.e., $\mathcal{R}(n, d)$, the Reed-Muller code of order d and length 2^n. Note that $\mathcal{R}(n, d)$ is a vector subspace of the vector space B_n, the set of all n-variable Boolean functions. Any Boolean function can be seen as a multivariate polynomial over $GF(2)$. Now if we consider the elements of $\mathcal{R}(n, d)$ as the multivariate polynomials over $GF(2)$, then the standard basis is the set of all nonzero monomials of degree $\leq d$. That is, the standard basis is

$$S_{n,d} = \{x_{i_1} \dots x_{i_k} : 1 \leq k \leq d \text{ and } 1 \leq i_1 < i_2 < \dots < i_k \leq n\} \cup \{1\},$$

where the input variables of the Boolean functions are $x_1, \dots, x_n$.

The ordering among the monomials is considered in lexicographic ordering ($<_l$) as usual, i.e., $x_{i_1} x_{i_2} \dots x_{i_p} <_l x_{j_1} x_{j_2} \dots x_{j_q}$ if either $p < q$ or $p = q$ and there is $1 \leq k \leq p$ such that $i_p = j_p$, $i_{p-1} = j_{p-1}, \dots, i_{k+1} = j_{k+1}$ and $i_k < j_k$. So, the set $S_{n,d}$ is a totally ordered set with respect to this lexicographical ordering ($<_l$). Using this ordering we refer the monomials according their order, i.e., the p-th monomial as m_p, $1 \leq p \leq \sum_{i=0}^{d} \binom{n}{i}$ following the convention $m_p <_l m_q$ if $p < q$.

Definition 2 *Given $n > 0$, $0 \leq d \leq n$, we define a mapping $v_{n,d} : \{0,1\}^n \mapsto \{0,1\}^{\sum_{i=0}^{d} \binom{n}{i}}$, such that $v_{n,d}(x) = (m_1(x), m_2(x), \dots, m_{\sum_{i=0}^{d} \binom{n}{i}}(x))$. Here $m_i(x)$ is the ith monomial as in the lexicographical ordering ($<_l$) evaluated at the point $x = (x_1, x_2, \dots, x_n)$.*

To evaluate the value of the t-th coordinate of $v_{n,d}(x_1, x_2, \dots, x_n)$ for $1 \leq t \leq \sum_{i=0}^{d} \binom{n}{i}$, i.e., $[v_{n,d}(x_1, \dots, x_n)]_t$, one requires to calculate the value of the monomial m_t (either 0 or 1) at $(x_1, x_2, \dots, x_n)$. Now we define a matrix $M_{n,d}$ with respect to a n-variable function f. To define this we need another similar ordering ($<^l$) over the elements of vector space $\{0,1\}^n$. We say for $u, v \in \{0,1\}^n$, $u <^l v$ if either $wt(u) < wt(v)$ or $wt(u) = wt(v)$ and there is a $1 \leq k \leq n$ such that $u_n = v_n, u_{n-1} = v_{n-1}, \dots, u_{k+1} = v_{k+1}$ and $u_k = 0, v_k = 1$.

Definition 3 *Given $n > 0$, $0 \leq d \leq n$ and an n-variable Boolean function f, we define a $wt(f) \times \sum_{i=0}^{d} \binom{n}{i}$ matrix*

$$M_{n,d}(f) = \begin{bmatrix} v_{n,d}(x_1) \\ v_{n,d}(x_2) \\ \vdots \\ v_{n,d}(x_{wt(f)}) \end{bmatrix}$$

where $supp(f) = \{x_1, x_2, \dots, x_{wt(f)}\}$ and $x_1 <^l x_2 <^l \dots <^l x_{wt(f)}$; $supp(f)$ is the set of input vectors for which f outputs 1.

Note that the matrix $M_{n,d}(f)$ is the transpose of the restricted generator matrix for Reed-Muller code of length 2^n and order d, $\mathcal{R}(d,n)$, to the support of f (see also (Canteaut 2005, Page 7)). Any row of the matrix $M_{n,d}(f)$ corresponding to an input vector $(x_1,\ldots,x_n)$ is

$$\overbrace{1}^{0\deg} \quad \overbrace{x_1,\ \ldots,\ x_i,\ \ldots,\ x_n}^{1\deg} \quad \ldots \quad \overbrace{x_1\ldots x_d,\ \ldots,\ x_{i_1}\ldots x_{i_d},\ \ldots,\ x_{n-d+1}\ldots x_n}^{d\deg}.$$

Each column of the matrix is represented by a specific monomial and each entry of the column tells whether that monomial is satisfied by the input vector which identifies the row, i.e., the rows of this matrix correspond to the evaluations of the monomials having degree at most d on support of f. As already discussed, here we have one-to-one correspondence from the input vectors $x=(x_1,\ldots,x_n)$ to the row vectors $v_{n,d}(x)$ of length $\sum_{i=0}^{d}\binom{n}{i}$. So, each row is fixed by an input vector.

2.1 Annihilator of f and rank of the matrix $M_{n,d}(f)$

We are interested to find out the lowest degree annihilators of $f\in B_n$. Let $g\in B_n$ be an annihilator of f, i.e., $f(x)\,g(x)=0$ for all $x\in\{0,1\}^n$. That means, for each $x=(x_1,\ldots,x_n)\in\{0,1\}^n$,

$$g(x_1,\ldots,x_n)=0 \text{ if } f(x_1,\ldots,x_n)=1. \tag{1}$$

Suppose degree of the function g is $\leq d$, then the ANF of g is of the form $g(x_1,\ldots,x_n)=a_0+\sum_{i=0}^{n}a_ix_i+\cdots+\sum_{1\leq i_1<i_2\cdots<i_d\leq n}a_{i_1,\ldots,i_d}x_{i_1}\cdots x_{i_d}$ where the subscripted a's are from $\{0,1\}$ and not all of them are zero. Following Equation 1, we get the following $wt(f)$ many homogeneous linear equations

$$a_0+\sum_{i=0}^{n}a_ix_i+\cdots+\sum_{1\leq i_1<i_2\cdots<i_d\leq n}a_{i_1,\ldots,i_d}x_{i_1}\cdots x_{i_d}=0, \tag{2}$$

considering the vectors $(x_1,\ldots,x_n)\in supp(f)$. This is a system of homogeneous linear equations on a's with $\sum_{i=0}^{d}\binom{n}{i}$ many a's as variables. The matrix form of this system of equations is $M_{n,d}(f)\;A^{tr}=O$, where $A=(a_0,a_1,a_2,\ldots,a_{n-d+1,\ldots,n})$, the row vector of coefficients of the monomials which are ordered according to the order $<_l$. Each nonzero solution of the system of equations formed by Equation 2 gives an annihilator g of degree $\leq d$. This is basically the Algorithm 1 presented in (Meier, Pasalic, and Carlet 2004). Since the number of solutions of this system of equations are connected to the rank of the matrix $M_{n,d}(f)$, it is worth to study the rank and the set of linear independent rows/columns of matrix $M_{n,d}(f)$. If the rank of matrix $M_{n,d}(f)$ is equal to $\sum_{i=0}^{d}\binom{n}{i}$ (i.e., number of columns) then the only solution is the zero solution. So, for this case f has no annihilator of degree $\leq d$. This implies that the number of rows $\geq$ number of columns, i.e., $wt(f)\geq\sum_{i=0}^{d}\binom{n}{i}$ which is the Theorem 1 in (Dalai, Gupta, and Maitra 2004). If the rank of matrix is equal to $\sum_{i=0}^{d}\binom{n}{i}-k$ for $k>0$ then the number of linearly independent solutions of the system of equations is k which gives k many linearly independent annihilators of degree $\leq d$ and 2^k-1 many number of annihilators of degree $\leq d$. However, to implement algebraic attack one needs only linearly independent annihilators. Hence, finding $\mathcal{AI}(f)$, one can use the following simplest algorithm.

Algorithm 1
Input: $f\in B_n$ *and* n.
Output: $AI(f)$.
for($i=1$ *to* $\lceil\frac{n}{2}\rceil-1$*)* {
    find the rank r_1 *of the matrix* $M_{n,i}(f)$*;*
    find the rank r_2 *of the matrix* $M_{n,i}(1+f)$*;*
    if $\min\{r_1,r_2\}<\sum_{j=0}^{i}\binom{n}{j}$ *then output* i*;*
}
output $\lceil\frac{n}{2}\rceil$*;*

Since either f or $1+f$ has an annihilator of degree $\leq \lceil \frac{n}{2} \rceil$, one needs to check till $i = \lceil \frac{n}{2} \rceil$. This algorithm is equivalent to Algorithm 1 in (Meier, Pasalic, and Carlet 2004).

The simplest and immediate way to find out the rank of $M_{n,d}(f)$ is the Gaussian elimination process. To check the existence or to enumerate the annihilators of degree $\leq \lceil \frac{n}{2} \rceil$ for a balanced function, the complexity is approximately $(2^{n-2})^3$. Considering this time complexity, it is not encouraging to check annihilators of a function of 20 variables or more using the presently available computing power. However, given n and d, the matrix $M_{n,d}(f)$ has pretty good structure, which we explore in this paper towards a better algorithm (that is solving the set of homogeneous linear equations in an efficient way by decreasing the size of the matrix involved).

3 Faster strategy to construct the set of homogeneous linear equations

In this section we present an efficient strategy to reduce the set of homogeneous linear equations. First we present a technical result.

Theorem 1 *Let $g \in B_n$ defined as $g(x) = 1$ iff $wt(x) \leq d$ for $0 \leq d \leq n$. Then $M_{n,d}(g)^{-1} = M_{n,d}(g)$, i.e., $M_{n,d}(g)$ is an involution.*

Proof : Suppose $\mathcal{F} = M_{n,d}(g)M_{n,d}(g)$. Then the i-th row and j-th column entry of $\mathcal{F}$ (denoted by $\mathcal{F}_{i,j}$) is the scalar product of i-th row and j-th column of $M_{n,d}(g)$. Suppose the i-th row is $v_{n,d}(x_1, \ldots, x_n)$ for $(x_1, \ldots, x_n) \in \{0,1\}^n$ having $x_{q_1}, \ldots, x_{q_l}$ as 1 and others are 0. Further consider that the j-th column is the evaluation of the monomial $x_{r_1} \ldots x_{r_k}$ at the vectors belonging to the support of g. If $\{r_1, \ldots, r_k\} \not\subseteq \{q_1, \ldots, q_l\}$ then $\mathcal{F}_{ij} = 0$. Otherwise, $\mathcal{F}_{i,j} = \binom{l-k}{0} + \binom{l-k}{1} + \ldots + \binom{l-k}{l-k} \bmod 2 = 2^{l-k} \bmod 2$. So, $\mathcal{F}_{i,j} = 1$ iff $\{x_{r_1}, \ldots, x_{r_k}\} = \{x_{q_1}, \ldots, x_{q_l}\}$. That implies, $\mathcal{F}_{i,j} = 1$ iff $i = j$ i.e., $\mathcal{F}$ is identity matrix. Hence, $M_{n,d}(g)$ is its own inverse.

See the following example for the structure of $M_{n,d}(g)$ when $n = 4$ and $d = 2$.

Example 1 *Let us present an example of $M_{n,d}(g)$ for $n = 4$ and $d = 2$. We have $\{1, x_1, x_2, x_3, x_4, x_1x_2, x_1x_3, x_2x_3, x_1x_4, x_2x_4, x_3x_4\}$, the list of 4-variable monomials of degree ≤ 2 in ascending order ($<_l$).*

Similarly, $\{(0,0,0,0), (1,0,0,0), (0,1,0,0), (0,0,1,0), (0,0,0,1), (1,1,0,0), (1,0,1,0), (0,1,1,0), (1,0,0,1), (0,1,0,1), (0,0,1,1)\}$ present the 4 dimensional vectors of weight ≤ 2 in ascending order ($<^l$). So the matrix

$$M_{4,2}(g) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

One may check that $M_{4,2}(g)$ is involution.

Lemma 1 *Let A be a nonsingular $m \times m$ binary matrix where the m-dimensional row vectors are $v_1, v_2, \ldots, v_m$. Let U be a $k \times m$ binary matrix, $k \leq m$, where the rows are $u_1, u_2, \ldots, u_k$. Let $W = UA^{-1}$, a $k \times m$ binary matrix. Consider that a matrix A' is formed from A by replacing the rows $v_{i_1}, v_{i_2}, \ldots, v_{i_k}$ of A by the vectors $u_1, u_2, \ldots, u_k$. Further consider that a $k \times k$ matrix W' is formed by the i_1-th, i_2-th, $\ldots, i_k$-th columns of W (out of m columns). Then A' is nonsingular iff W' is nonsingular.*

Proof : Without loss of generality, we can take $i_1 = 1, i_2 = 2, \ldots, i_k = k$. So, the row vectors of A' are $u_1, \ldots, u_k, v_{k+1}, \ldots, v_m$.

We first prove that if the row vectors of A' are not linearly independent then the row vectors of W' are not linearly independent. As the row vectors of A' are not linearly independent, we have $\alpha_1, \alpha_2, \ldots, \alpha_m \in \{0, 1\}$ (not all zero) such that $\sum_{i=1}^{k} \alpha_i u_i + \sum_{i=k+1}^{m} \alpha_i v_i = 0$. If $\alpha_i = 0$ for all i, $1 \leq i \leq k$ then $\sum_{i=k+1}^{m} \alpha_i v_i = 0$ which implies $\alpha_i = 0$ for all i, $k+1 \leq i \leq m$ as $v_{k+1}, v_{k+2}, \ldots, v_m$ are linearly independent. So, all α_i's for $1 \leq i \leq k$ can not be zero.

Further, we have $UA^{-1} = W$, i.e., $U = WA$, i.e.,

$$\begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_k \end{pmatrix} = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_k \end{pmatrix} \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{pmatrix}, \text{ i.e., } u_i = w_i \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{pmatrix}.$$

$$\text{Hence, } \sum_{i=1}^{k} \alpha_i u_i = \sum_{i=1}^{k} \alpha_i w_i \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{pmatrix} = r \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{pmatrix}$$

where $r = (r_1, r_2, \ldots, r_m) = \sum_{i=1}^{k} \alpha_i w_i$.

If the restricted matrix W' were nonsingular, the vector $r' = (r_1, r_2, \ldots, r_k)$ is nonzero as $(\alpha_1, \alpha_2, \ldots, \alpha_k)$ is not all zero. Hence, $\sum_{i=1}^{k} \alpha_i u_i + \sum_{i=k+1}^{m} \alpha_i v_i = 0$, *i.e.*, $\sum_{i=1}^{k} r_i v_i + \sum_{i=k+1}^{m} (r_i + \alpha_i) v_i = 0$. This contradicts that $v_1, v_2, \ldots, v_m$ are linearly independent as $r' = (r_1, r_2, \ldots, r_k)$ is nonzero. Hence W' must be singular. This proves one direction.

On the other direction if the restricted matrix W' is singular then there are $\beta_1, \beta_2, \ldots, \beta_k$ not all zero such that $\sum_{i=0}^{k} \beta_i w_i = (0, \ldots, 0, s_{k+1}, \ldots, s_m)$.

$$\text{Hence, } \sum_{i=0}^{k} \beta_i u_i = \sum_{i=1}^{k} \beta_i w_i \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{pmatrix} = s_{k+1} v_{k+1} + \ldots + s_m v_m, \text{ i.e., } \sum_{i=0}^{k} \beta_i u_i + \sum_{i=k+1}^{m} s_i v_i = 0 \text{ which}$$

implies matrix A' is singular.

Following Lemma 1, one can check the nonsingularity of the larger matrix A' by checking the nonsingularity of the reduced matrix W'. Thus checking the nonsingularity of the larger matrix A' will be more efficient if the computation of matrix product $W = UA^{-1}$ can be done efficiently. The involutive nature of the matrix $M_{n,d}(g)$ presented in Theorem 1 helps to achieve this efficiency. In the following result we present the Lemma 1 in more general form.

Theorem 2 *Let A be a nonsingular $m \times m$ binary matrix with m-dimensional row vectors $v_1, v_2, \ldots, v_m$ and U be a $k \times m$ binary matrix with m-dimensional row vectors $u_1, \ldots, u_k$. Consider $W = UA^{-1}$, a $k \times m$ matrix. The matrix A', formed from A by removing the rows $v_{i_1}, v_{i_2}, \ldots, v_{i_l}$ $(l \leq m)$ from A and adding the rows $u_1, u_2, \ldots, u_k$ $(k \geq l)$, is of rank m iff the rank of restricted $k \times l$ matrix W' including only the i_1-th, i_2-th, $\ldots$, i_l-th columns of W is l.*

Proof : Here, the rank of matrix W' is l. So, there are l many rows of W', say $w'_{p_1}, \ldots, w'_{p_l}$ which are linearly independent. So, following the Lemma 1 we have the matrix A'' formed by replacing the rows $v_{i_1}, \ldots, v_{i_l}$ of A by $u_{p_1}, \ldots, u_{p_l}$ is nonsingular, i.e., rank is m. Hence the matrix A' where some more rows are added to A'' has rank m. The other direction can also be shown similar to the proof of the other direction in Lemma 1.

Now using Theorem 1 and Theorem 2, we describe a faster algorithm to generate smaller system of equations of certain degree d of a Boolean function f. Suppose g be the Boolean function described in Theorem 1, i.e., $supp(g) = \{x | 0 \leq wt(x) \leq d\}$. In Theorem 1, we have already shown that $M_{n,d}(g)$ is nonsingular matrix (in fact it is involution). Let $\{x | wt(x) \leq d \text{ and } f(x) = 0\} = \{X_1, X_2, \ldots, X_l\}$ and $\{x | wt(x) > d \text{ and } f(x) = 1\} = \{Y_1, Y_2, \ldots, Y_k\}$. Then

we consider $M_{n,d}(f)$ as A, $v_{n,d}(X_1), \ldots, v_{n,d}(X_l)$ as $v_{i_1}, \ldots, v_{i_l}$ and $v_{n,d}(Y_1)$, ..., $v_{n,d}(Y_k)$ as $u_1, \ldots, u_k$. Then following Theorem 2 we can ensure whether $M_{n,d}(f)$ is nonsingular. If it is nonsingular, then there is no annihilator of degree $\leq d$, else there are annihilator(s). We may write this in a more concrete form as the following corollary.

Corollary 1 *Let $f \in B_n$. Let A^r be the restricted matrix of $A = M_{n,d}(g)$, by using the columns corresponding to the monomials $x_{i_1}x_{i_2}\ldots x_{i_l}$ such that $l \leq d$ and $f(x_1, \ldots, x_n) = 0$ when $x_{i_1} = 1, x_{i_2} = 1, \ldots, x_{i_l} = 1$ and rest are 0. Further $U = \begin{pmatrix} v_{n,d}(Y_1) \\ v_{n,d}(Y_2) \\ \vdots \\ v_{n,d}(Y_k) \end{pmatrix}$, where $\{Y_1, \ldots, Y_k\} = \{x | wt(x) > d$ and $f(x) = 1\}$. If rank of UA^r is l then f has no annihilator of degree $\leq d$, else f has annihilator(s) of degree $\leq d$.*

Proof : As per Theorem 2, here $W = UA^{-1} = UA$, since A is involution following Theorem 1 and hence W' is basically UA^r. Thus the proof follows.

Now we can use the following technique for fast computation of the matrix multiplication UA^r. For this we first present a technical result and its proof is similar in the line of the proof of Theorem 1.

Proposition 1 *Consider g as in Theorem 1. Let $y \in \{0,1\}^n$ such that $i_1, i_2, \ldots, i_p$-th places are 1 and other places are 0. Consider the j-th monomial $m_j = x_{j_1}x_{j_2}\ldots x_{j_q}$ according the ordering $<_l$. Then the j-th entry of $v_{n,d}(y)M_{n,d}(g)$ is 0 if $\{j_1, \ldots, j_q\} \not\subseteq \{i_1, \ldots, i_p\}$ else the value is $\sum_{i=0}^{d-q} \binom{p-q}{i} \bmod 2$.*

One can precompute the sums $\sum_{i=0}^{d-q} \binom{p-q}{i} \bmod 2$ for $d+1 \leq p \leq n$ and $0 \leq q \leq d$, and store them and the total complexity for calculating them is $O(d^2(n-d))$. These sums will be used to fill up the matrix UA^r which is an $l \times k$ matrix according to Corollary 1. Let us denote $\mu_f^d = |\{x | wt(x) \leq d, f(x) = 1\}|$ and $\nu_f^d = |\{x | wt(x) > d, f(x) = 1\}|$. Then $wt(f) = \mu_f^d + \nu_f^d$ and the matrix UA^r is of dimension $\nu_f^d \times (\sum_{i=0}^{d} \binom{n}{i} - \mu_f^d)$. Clearly $O(d^2(n-d))$ can be neglected with respect to $\nu_f^d \times (\sum_{i=0}^{d} \binom{n}{i} - \mu_f^d)$. Thus we have the following result.

Theorem 3 *Consider U and A^r as in Corollary 1. The time (and also space) complexity to construct the matrix UA^r is of the order of $\nu_f^d \times (\sum_{i=0}^{d} \binom{n}{i} - \mu_f^d)$. Further checking the rank of UA^r (as given in Corollary 1) one can decide whether f has an annihilator at degree d or not.*

In fact, to check the rank of the matrix UA^r using Gaussian elimination process, we need not store the ν_f^d many rows at the same time. One can add one row (following the calculation to compute a row of the matrix given in Proposition 1) at a time incrementally to the previously stored linearly independent rows by checking whether the present row is linearly independent with respect to the already stored rows. If the current row is linearly independent with the existing ones, then we do row operations and add the new row to the previously stored matrix. Otherwise we reject the new row. Hence, our matrix size never crosses the size $(\sum_{i=0}^{d} \binom{n}{i} - \mu_f^d) \times (\sum_{i=0}^{d} \binom{n}{i} - \mu_f^d)$.

If $\nu_f^d < (\sum_{i=0}^{d} \binom{n}{i} - \mu_f^d)$, then there will be nontrivial solutions and we can directly say that the annihilators exist. Thus we always need to concentrate on the case $\nu_f^d \geq (\sum_{i=0}^{d} \binom{n}{i} - \mu_f^d)$, where the matrix size $(\sum_{i=0}^{d} \binom{n}{i} - \mu_f^d) \times (\sum_{i=0}^{d} \binom{n}{i} - \mu_f^d)$ provides a further reduction than the matrix size $\nu_f^d \times (\sum_{i=0}^{d} \binom{n}{i} - \mu_f^d)$ and one can save more space. This will be very helpful when one tries to check the annihilators of small degree d.

One may refer to Appendix A to get detailed description why our strategy provides asymptotic improvement than (Meier, Pasalic, and Carlet 2004) in terms of constructing this reduced set of homogeneous linear equations. In terms of the overall algorithm to find the annihilators, our algorithm works around eight times faster than (Meier, Pasalic, and Carlet 2004) in general. Using our strategy to find the reduced matrix first and then using the standard Gaussian elimination technique, we could find the annihilators of any random balanced Boolean functions on 16 variables in around 2 hours in a Pentium 4 personal computer with 1 GB RAM. Note that, the very recently known

efficient algorithms (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006; Didier and Tillich 2006) can work till 20 variables. Comparison of our algorithm with that of (Meier, Pasalic, and Carlet 2004) is not to demonstrate the efficiency of our algorithm as there are more efficient general algorithms known (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006; Didier and Tillich 2006), but to explain how our strategy works efficiently to reduce the matrix size. In fact, when the matrix size is reduced significantly, then our algorithm is currently the best known and for such class of functions, it is better than the algorithm of (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006). Below we describe that clearly.

3.1 Efficiency of our strategy

Our algorithm works efficiently for the functions having higher value of μ_f^d.

Proposition 2 *Let $f \in B_n$ with $\mu_f^d = \sum_{i=0}^{d} \binom{n}{i} - c(\sum_{i=0}^{d} \binom{n}{i})^{\delta}$, for some constant $c \geq 0$, and some $\delta \geq 0$. Then the time and space complexity to check the existence of d degree annihilators are $O(\nu_f^d(\sum_{i=0}^{d} \binom{n}{i})^{2\delta})$ and $O((\sum_{i=0}^{d} \binom{n}{i})^{2\delta})$ respectively.*

The proof follows from Theorem 3 and the paragraph next to Theorem 3. Hence low value of δ increases efficiency in both the time and space complexities. In the following theorem we present when our strategy for checking optimal AI will be faster than the currently best known algorithm (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006).

Theorem 4 *Let $f \in B_n$ is balanced and*

1. *n is odd with $\mu_f^{\frac{n-1}{2}} = 2^{n-1} - c(2^{n-1})^{\frac{2}{\omega}-\epsilon}$, for some constant $c \geq 0$, $\epsilon > 0$ and ω is the time complexity order to solve a system of linear equations. Then the time complexity to check the existence of $\frac{n-1}{2}$-degree annihilator of f is $O((2^{n-1})^{2-\omega\epsilon})$.*
2. *n is even with $\mu_f^{\frac{n}{2}-1} = \sum_{i=0}^{\frac{n}{2}-1} \binom{n}{i} - c(\sum_{i=0}^{\frac{n}{2}-1} \binom{n}{i})^{0.5-\epsilon}$, for some constant $c \geq 0$, $\epsilon > 0$. Then the time complexity to check the existence of $(\frac{n}{2}-1)$-degree annihilator of f is $O(2^{n-2}(\sum_{i=0}^{\frac{n}{2}-1} \binom{n}{i})^{1-2\epsilon})$.*

Proof : For odd n, $\sum_{i=0}^{\frac{n-1}{2}} \binom{n}{i} = 2^{n-1} = wt(f)$. Hence, UA^r is a square matrix of dimension $c(2^{n-1})^{\frac{2}{\omega}-\epsilon}$. Then the result follows as ω is time complexity order for solving a square matrix.

When n is even, the matrix is $\nu_f^{\frac{n}{2}-1} \times c(\sum_{i=0}^{\frac{n}{2}-1} \binom{n}{i})^{0.5-\epsilon}$. Hence, the time complexity to solve the system is $O(\nu_f^{\frac{n}{2}-1}(\sum_{i=0}^{\frac{n}{2}-1} \binom{n}{i})^{1-2\epsilon})$ i.e., $O(2^{n-2}(\sum_{i=0}^{\frac{n}{2}-1} \binom{n}{i})^{1-2\epsilon})$.

Now we compare the complexity of our strategy with the fastest known algorithm (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006) for the class of functions presented in Theorem 4. The algorithm in (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006) requires $O((2^{n-1})^2)$ and $O(2^{n-1}\sum_{i=0}^{\frac{n}{2}-1} \binom{n}{i})$ time complexities for n odd and even respectively. Therefore, for this class of functions, our strategy gains $O((2^{n-1})^{\omega\epsilon})$ and $O((\sum_{i=0}^{\frac{n}{2}-1} \binom{n}{i})^{2\epsilon})$ over the algorithm in (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006) for n odd and even respectively. Further, for this type of functions, our strategy gains in space complexity also. Our strategy needs $O(4^{n-1})^{\frac{2}{\omega}-\epsilon}$ and $O(\sum_{i=0}^{\frac{n}{2}-1} \binom{n}{i}^{1-2\epsilon})$ memory for n odd and even respectively, where as the algorithm in (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006) needs 4^{n-1} and $O((\sum_{i=0}^{\frac{n}{2}-1} \binom{n}{i})^2)$ memory respectively.

Therefore, our technique to find the existence of annihilators of degree d is more efficient than (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006) for the functions having higher value of μ_f^d. The size of this class is $\binom{2^{n-1}}{c(2^{n-1})^{\frac{2}{\omega}-\epsilon}}^2$ when n is odd and $\binom{\sum_{i=0}^{\frac{n}{2}-1} \binom{n}{i}}{c(\sum_{i=0}^{\frac{n}{2}-1} \binom{n}{i})^{0.5-\epsilon}}^2 \binom{\binom{n}{\frac{n}{2}}}{\frac{1}{2}\binom{n}{\frac{n}{2}}}$ when n is even for each c and ϵ. Though the

relative size of this class is very small compare to 2^{2^n}, the size of B_n, in absolute terms this class is quite big. Further with the results of the next section, this class will be substantially extended.

We now present some examples to show the efficiency of our algorithm. Consider a function f on n variables when n is odd. To check for optimal AI $\frac{n+1}{2}$, we have to check the existence of annihilators at degree $\frac{n-1}{2}$. Here we consider $\omega = 3$ as the order of time complexity for Gaussian elimination technique. Further there are better techniques when $\omega = 2.8$ (Strassen 1969) and $\omega = 2.4$ (Coppersmith and Winograd 1990).

Consider balanced $f \in B_{51}$ such that $\mu_f^{25} = 2^{51-1} - (2^{51-1})^{\frac{2}{3}-0.4} \approx 2^{50} - 2^{13}$, where $c = 1$ and $\epsilon = 0.4$. Then one needs $2^{13 \cdot 2} = 2^{26}$ memory bits and $2^{13 \cdot 3} = 2^{39}$ operations to check the existence of a 25-degree annihilator of f. In this case, the algorithm in (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006) needs $2^{2 \cdot 50} = 2^{100}$ operations and memory bits.

Next, we present an example of 11-variable function of this class. The following is the hexadecimal representation of the truth table of $f \in B_{11}$ balanced Boolean function having nonlinearity 796 and degree 10.

```
FFFFFFFDFFFFFFF7FFFBBFF7FEF7F771FFFBFEF7FFF7F773FFF7F771F7737110
FFDFFFF7FFF7F771FFF7D771F7617150FFF7D770F7717110F771711071101000
FDFFFFF7FFF7E571FFF7FF71F7617110FFF7F751F7617110F771715031101001
FFF7F751FF517110F771711071101200F771711071101000711010841000020
EFFFBFF7FEF7F771FFB5F771F7717110FFF7F775F7717110F761713071101000
FFF7F771E771F110F771511071101200F771311071181084711410801020000
FFF7F771D771711077757110701114800FF7171107110100471101010000000
F773711071101008713010001800000071101000100000001000000000000000
```

Here $\mu_f^5 = 2^{10} - 2^{10}(\frac{2}{3} - \frac{1}{6}) = 2^{10} - 2^5$. To check whether the AI of f is 6, we need to find the rank of a $2^5 \times 2^5$ matrix. Hence we need an order of $2^{5 \cdot 3} = 2^{15}$ operations. However, using the algorithm presented in (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006), one needs an order of $2^{10 \cdot 2} = 2^{20}$ operations.

Now we consider the cases when n is even. In this case the complexity computation is more complex as the matrix is not square. Since the value of $\nu_f^{\frac{n}{2}-1} > \frac{1}{2}\binom{n}{\frac{n}{2}} \approx 2^{n-2}$, the time complexity is not as low as in the case of odd n. Still we get better efficiency than (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006) for the class of functions we concentrate on. Consider balanced $f \in B_{30}$ such that $\mu_f^{14} = \sum_{i=0}^{14}\binom{30}{i} - (\sum_{i=0}^{14}\binom{30}{i})^{0.1}$, where $c = 1$ and $\epsilon = 0.4$. Then one needs $2^{28}(\sum_{i=0}^{14}\binom{30}{i})^{0.2} \approx 2^{34}$ operations and $(\sum_{i=0}^{14}\binom{30}{i})^{0.2} \approx 2^6$ memory bits to check the existence of a 14-degree annihilator of f. In this case the algorithm in (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006) needs $2^{29}\sum_{i=0}^{14}\binom{30}{i} \approx 2^{57}$ operations and $(\sum_{i=0}^{14}\binom{30}{i})^2 \approx 2^{56}$ memory bits.

Next we present a function of 12 variables. The following is the hexadecimal representation of the truth table of a 12-variable balanced Boolean function having nonlinearity 1586 and degree 11.

```
FFFFFEFFFFFFFFF7FFFF7FFFFFF7FF75FFFFFFFFFFFDFB73FFF7FBF7FFF17311
FFFFFFFBFFF7FD73FBF7F7F3FB71F510FFF7F5F1F7F37570F7F5771071503000
FFFFFFFFFFF7FFF3FF7FFF7BFFF57714FF77DFF5FF73F710F773D550F3315010
FFF7F775F7737750FFF7777173711010FFF773517350110073705100313012 80
FFFFFFFFFFF7FF73EFF7F7F5FFF7F310FFFEF7F1FFE1F170F7757710B1317110
FFF7FF73F7F17710F7F1F77173105010F771777071101110F511511058401080
FFFFEF75FFE27170FE717371F5315010FF75713073113100735810003080 8000
FFF5F5317150701071111010311400 20F3101880501010003010000000000020
FFFBFFF7FFFFFFF1FFFFEFF3FF77DD78FFF7FF75F7F67330FF757370F1D01100
FFF6F7F5FF75F570FF77F77175711000F777F55071711110F731111070101000
F7FFF774B7F1F371F7F7751173515100F7757370F57151107511100038101000
F777F311E7113010F131511010001000F3517110301030005110000010000400
FFF7FFF3F7F1F510FF75F370F1315110B777F711F7311100F118311450101000
```

```
FFF77F11773050007770511070000000715171105010000031100000000000000
F777733175121011F71130007000100OF37011103010000010103000100200000
71715110711011001100100000000000551210000000000000000000000000000
```

Here $\mu_f^5 = 1554 = 1586 - 32 = \sum_{i=0}^{5} \binom{12}{i} - 2^5$. To verify the optimal AI, i.e., $AI = 6$ we have to find the rank of $2^{11} - 1554 \times 2^5$ i.e., 494×2^5 matrix. Hence, we need an order of $494 \times 2^{2 \cdot 5} \approx 2^{19}$ operations and $2^5 \times 2^5 = 2^{10}$ memory bits. But using the algorithm presented in (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006), one needs an order of $2^{11} \times 1586 \approx 2^{21.5}$ operations and $1586 \times 1586 \approx 2^{21}$ memory bits.

4 Further reduction in matrix size applying linear transformation over the input variables of the function

In this section we will demonstrate a more general class of Boolean functions than what presented in Section 3.1 for which our strategy works efficiently.

To check for the annihilators, we need to compute the rank of the matrix UA^r. Following Theorem 3, it is clear that the size of the matrix UA^r will decrease if μ_f^d increases and ν_f^d decreases. Let B be an $n \times n$ nonsingular binary matrix and b be an n-bit vector. The function $f(x)$ has an annihilator at degree d iff $f(Bx + b)$ has an annihilator at degree d. Thus one will try to get the affine transformation on the input variables of $f(x)$ to get $h(x) = f(Bx + b)$ such that $|\{x|h(x) = 1, wt(x) \leq d\}|$ is maximized. This is because in this case μ_h^d will be maximized and ν_h^d will be minimized and hence the dimension of the matrix UA^r, i.e., $\nu_f^d \times (\sum_{i=0}^{d} \binom{n}{i} - \mu_f^d)$ will be minimized. This will indeed decrease the complexity at the construction step (discussed in the previous section). More importantly, it will decrease the complexity to solve the system of homogeneous linear equations.

See the following example that explains the efficiency for a 5-variable function.

Example 2 *Consider the* 5*-variable Boolean function* f *constructed using the method presented in (Dalai, Gupta, and Maitra 2005) such that* $\mathcal{AI}(f) = 3$*. The standard truth table representation of the function is* 01010110010101100101011001101001*, i.e., the outputs are corresponding to the inputs which are of increasing value. One can check that* $|\{x \in \{0,1\}^5 \mid f(x) = 1 \ \& \ wt(x) < 3\}| = 6$*. Now if we consider the function* $h(x) = f(Bx + b)$ *such that* $B = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$*, and* $b = \{1, 1, 0, 0, 1\}$*, then* $|\{x \in \{0,1\}^5 \mid h(x) = 1 \ \& \ wt(x) < 3\}| = 16$ *and one can immediately conclude (from the results in (Dalai, Maitra, and Sarkar 2006)) that* $\mathcal{AI}(h) = 3$*. This is an example where after finding the affine transformation there is even no need for the solution step at all. For the function* f*, here* $h(x) = f(Bx + b)$ *such that* $|\{x|h(x) = 1, wt(x) \leq d\}|$ *is maximized.*

We also present an example for a sub optimal case. In this case we consider $B = \begin{bmatrix} 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix}$*, and* b *an all zero vector, then* $|\{x \in \{0,1\}^5 \mid h(x) = 1 \ \& \ wt(x) < 3\}| = 14$*. Thus the dimension of the matrix* UA^r *becomes* 2×2 *as* $\nu_f^d = 2$ *and* $\sum_{i=0}^{d} \binom{n}{i} - \mu_f^d = 2$*. Thus one needs to check the rank of a* 2×2 *matrix only.*

Now the question is how to find such an affine transformation (for the optimal or even for sub optimal cases) efficiently.

For exhaustive search to get the optimal affine transform one needs to check $f(Bx + b)$ for all $n \times n$ nonsingular binary matrices B and n bit vectors b. Since there are $\prod_{i=0}^{n-1}(2^n - 2^i)$ many nonsingular binary matrices and 2^n many

n bit vectors, one needs to check $2^n \prod_{i=0}^{n-1}(2^n - 2^i)$ many cases for an exhaustive search. As weight of the input vectors are invariant under permutation of the arguments, checking for only one nonsingular matrix from the set of all nonsingular matrices whose rows are equivalent under certain permutation will suffice. Hence the exact number of search options is $\frac{1}{n!}2^n \prod_{i=0}^{n-1}(2^n - 2^i)$. One can check for $n \times n$ nonsingular binary matrices B where $row_i < row_j$ for $i < j$ (row_i is the decimal value of binary pattern of ith row). It is clear that the search is infeasible for $n \geq 8$.

Now we present a heuristic towards this. Our aim is to find out an affine transformation $h(x)$ of $f(x)$, i.e., $h(x) = f(Bx + b)$, which maximizes the value of μ_h^d. This means the weight of the most of the input vectors having weight $\leq d$ should be in $supp(h)$. So we attempt to get an affine transformation for a Boolean function f such that the transformation increases the probability that an input vector, having output 1, will be translated to a low weight input vector.

Consider $h(Vx + v) = f(x)$, where V is an $n \times n$ binary matrix and $v = (v_1, v_2, \ldots, v_n) \in \{0,1\}^n$. Suppose $r_1, r_2, \ldots, r_n \in \{0,1\}^n$ are the row vectors of the transformation V. By $Vx + v = y$ we mean $Vx^{tr} + v = y^{tr}$, where $x = (x_1, x_2, \ldots, x_n), y = (y_1, y_2, \ldots, y_n) \in \{0,1\}^n$. Given an x, we find a y by this transformation and then $h(y)$ is assigned to the value of $f(x)$. If $f(x) = 1$, we like that the corresponding $y = Vx + v$ should be of low weight. The chance of $(y_1, y_2, \ldots, y_n)$ getting low weight increases if the probability of $y_i = 0, 1 \leq i \leq n$ is increased. That means the probability of $r_i \cdot (x_1, x_2, \ldots, x_n) + v_i = 0$ for $1 \leq i \leq n$ needs to be increased. Hence we will like to choose a linearly independent set $r_i \in \{0,1\}^n$, $1 \leq i \leq n$ and $v \in \{0,1\}^n$ such that the probability $r_i \cdot (x_1, x_2, \ldots, x_n) + b_i = 0$, $1 \leq i \leq n$ is high when $(x_1, x_2, \ldots, x_n) \in supp(f)$. Since we use the relations $h(Vx + v) = f(x)$, and $h(x) = f(Bx + b)$, that means $B = V^{-1}$ and $b = V^{-1}v$.

The heuristic is presented below. By $bin[i]$ we denote the n-bit binary representation of the integer i.

Heuristic 1

1. $loop = 0$; $max = |\{x | f(x) = 1, wt(x) \leq d\}|$;
2. *For* $(i = 1; i < 2^n; i++)$ {
   (a) $t = |\{x = (x_1, x_2, \ldots, x_n) \in supp(f) | bin[i] \cdot x = 0\}|$
   (b) *if* $t \geq \frac{wt(f)}{2}$, $val[i] = t$ *and* $a_i = 0$ *else* $val[i] = wt(f) - t$ *and* $a_i = 1$.
   }
3. *Arrange the triplets* $(bin[i], a_i, val[i])$ *in descending order of* $val[i]$.
4. *Choose suitable* n *many triplets* (r_j, v_j, k_j) *for* $1 \leq j \leq n$ *such that* r_j*s are linearly independent and* k_j*'s are high.*
5. *Construct the nonsingular matrix* V *taking* r_j, $1 \leq j \leq n$ *as* j*-th row and* $v = (v_1, v_2, \ldots, v_n)$.
6. *Increment* $loop$ *by* 1*; while* $(loop < maxval)$
   (a) $B = V^{-1}$, $b = V^{-1}v$.
   (b) *if* $max < |\{x | f(Bx + b) = 1, wt(x) \leq d\}|$ *replace* $f(x)$ *by* $f(Bx + b)$ *and update* max *by* $|\{x | f(Bx + b) = 1, wt(x) \leq d\}|$.
   (c) *Go to step 2.*

The time complexity of this heuristic is $(maxval \times n2^{2n})$. See the following example, where we trace Heuristic 1 for the 5-variable function f given in Example 2.

Example 3 *We have* $f = 01010110010101100101011001101001$ *and check that* $|\{x \in \{0,1\}^5 \mid f(x) = 1 \ \& \ wt(x) \leq 2\}| = 6$. *In step 2, we get* $(val[i], a_i)$ *for* $1 \leq i \leq 31$ *as* $1 : (11,1)$, $2 : (8,1)$, $3 : (11,1)$, $4 : (8,1)$, $5 : (11,1)$, $6 : (8,1)$, $7 : (9,0)$, $8 : (8,1)$, $9 : (9,1)$, $10 : (8,1)$, $11 : (9,1)$, $12 : (8,1)$, $13 : (9,1)$, $14 : (8,1)$, $15 : (11,0)$, $16 : (8,1)$, $17 : (9,1)$, $18 : (8,1)$, $19 : (9,1)$, $20 : (8,1)$, $21 : (9,1)$, $22 : (8,1)$, $23 : (11,0)$, $24 : (8,1)$, $25 : (9,0)$, $26 : (8,1)$, $27 : (9,0)$, $28 : (8,1)$, $29 : (9,0)$, $30 : (8,1)$, $31 : (11,1)$. *Then after ordering according the value of* $val[i]$, *we choose the row of matrix* V *as the* 5*-bit binary expansion of* $1, 3, 5, 15$ *and* 7 *with frequency values of* 0*'s as* $11, 11, 11, 11, 9$ *respectively and* $v = (a_1, a_3, a_5, a_{15}, a_7) = (1, 1, 1, 1, 0)$. *Here the matrix* V *is a nonsingular matrix. The new function is* $g = f(Bx + b)$, *where* $B = V^{-1}$, $b = V^{-1}v$ *and one can check that* $|\{x \in \{0,1\}^5 \mid g(x) = 1 \ \& \ wt(x) \leq 2\}| = 16$.

Experiments with this heuristic on different Boolean functions provide encouraging results. First of all we have considered the functions which are random affine transformations $g(x)$ of the function (Dalai, Maitra, and Sarkar 2006), $f_s(x) = 1$ for $wt(x) \leq \lfloor \frac{n-1}{2} \rfloor$ and $f_s(x) = 0$ for $wt(x) \geq \lfloor \frac{n+1}{2} \rfloor$, which has no annihilator having degree $\leq \lfloor \frac{n-1}{2} \rfloor$. This experimentation has been done for $n = 5$ to 16. For all the cases running Heuristic 1 on $g(x)$ we could go back to $f_s(x)$. Then we have randomly changed $2^{\zeta n}$ bits on the upper half of $f_s(x)$ ($0.5 \leq \zeta \leq 0.8$ at steps of 0.1) to get $f'_s(x)$ and then put random transformations on $f'_s(x)$ to get $g(x)$. Running Heuristic 1, we could also go back to $f'_s(x)$ easily. For experiments we have taken $maxval = 20$.

The important issue is exactly when this matrix size is asymptotically reduced than the trivial matrix size $wt(f) \times \sum_{i=0}^{d} \binom{n}{i}$ if one writes down the equations by looking at the truth table of the function only. This happens only when μ_f^d is very close to $\sum_{i=0}^{d} \binom{n}{i}$ (see Proposition 2 and Theorem 4). Let $\sum_{i=0}^{d} \binom{n}{i} - \mu_f^d \leq 2^{\zeta n}$, where ζ is a constant such that $0 < \zeta < 1$. Note that $2^{\zeta n}$ is the approximation of $c(\sum_{i=0}^{d} \binom{n}{i})^{\delta}$ in Proposition 2. In that case the matrix size will be less than or equal to $(wt(f) + 2^{\zeta n} - \sum_{i=0}^{d} \binom{n}{i}) \times 2^{\zeta n}$. When $d = \lfloor \frac{n}{2} \rfloor$ and n odd, $\sum_{i=0}^{d} \binom{n}{i} = 2^{n-1}$. Thus for a balanced function, the size of the matrix becomes as low as $2^{\zeta n} \times 2^{\zeta n}$. We summarize the result as follows.

Theorem 5 *Predetermine a constant* ζ, *such that* $0 < \zeta < 1$. *Consider any Boolean function* $f(x) \in B_n$ *for which there exist a nonsingular binary matrix* B *and an* n*-bit vector* b *such that* $\sum_{i=0}^{d} \binom{n}{i} - |\{x | f(Bx + b) = 1, wt(x) \leq d\}| \leq 2^{\zeta n}$. *If* B *and* b *are known, then the size of the matrix* UA^r *will be less than or equal to* $(wt(f) + 2^{\zeta n} - \sum_{i=0}^{d} \binom{n}{i}) \times 2^{\zeta n}$ *which is asymptotically reduced in size than* $wt(f) \times \sum_{i=0}^{d} \binom{n}{i}$.

That matrices B, b may be available as output of Heuristic 1.

Next we have run our heuristics on randomly chosen balanced functions. The number of inputs up to weight d for a Boolean function is $\sum_{i=0}^{d} \binom{n}{i}$. Thus for a randomly chosen balanced function, it is expected that there will be $\frac{1}{2} \sum_{i=0}^{d} \binom{n}{i}$ many inputs up to weight d for which the outputs are 1. Below we present the improvement (on an average of 100 experiments in each case) we got after running Heuristic 1 with $maxval = 20$ for $n = 12$ to 16.

Table 1. Efficiency of Heuristic 1 on random balanced functions

n	12			13			14			15			16		
d	3	4	5	4	5	6	4	5	6	5	6	7	5	6	7
$\sum_{i=0}^{d} \binom{n}{i}$	299	794	1586	1093	2380	4096	1471	3473	6476	4944	9949	16384	6885	14893	26333
$\lceil \frac{1}{2} \sum_{i=0}^{d} \binom{n}{i} \rceil$	149	397	793	541	1190	2048	735	1736	3238	2472	4974	8192	3442	7446	13166
Heuristic Value	228	535	964	717	1438	2322	957	2051	3648	2917	5525	8811	3995	8194	14114

It should be noted that after running our heuristic on random balanced functions, the improvement is not significant. There are improvements as we find that the the values are significantly more than $\frac{1}{2} \sum_{i=0}^{d} \binom{n}{i}$ (making our algorithm efficient), but the value is not very close to $\sum_{i=0}^{d} \binom{n}{i}$. This is not a problem with the efficiency of the heuristic, but with the inherent property of a random Boolean function that there may not be an affine transformation at all on $f(x)$ such that $|\{x | f(Bx + b) = 1, wt(x) \leq d\}|$ is very high. In fact we can show that for highly nonlinear functions $f(x)$,

the increment from $|\{x|f(x) = 1, wt(x) \leq d\}|$ to $|\{x|f(Bx + b) = 1, wt(x) \leq d\}|$ may not be high for any choice of B, b. The reason for this is as follows.

Proposition 3 *Let $f \in B_n$ be a balanced function (n odd) having nonlinearity $nl(f) = 2^{n-1} - 2^{\frac{n-1}{2}}$. Then for any nonsingular $n \times n$ matrix B and any n-bit vector b, $2^{n-1} - |\{x|f(Bx+b) = 1, wt(x) \leq \frac{n-1}{2}\}| \geq \frac{1}{2}\binom{n-1}{\frac{n-1}{2}} - 2^{\frac{n-1}{2}-1}$.*

Proof : Let $f \in B_n$ be a balanced function (n odd) having nonlinearity $nl(f) = 2^{n-1} - 2^{\frac{n-1}{2}}$. Let $g \in B_n$ be the function such that $g(x) = 1$ for $wt(x) \leq \frac{n-1}{2}$. By (Dalai, Maitra, and Sarkar 2006, Theorem 3), $nl(g) = 2^{n-1} - \binom{n-1}{\frac{n-1}{2}}$. Now we like to find out a function $h(x) = f(Bx + b)$ such that $|\{x|h(x) = 1, wt(x) \leq \frac{n-1}{2}\}|$ is high. Consider the value $T = |supp(g) \cap supp(h)|$, i.e., $T = |\{x : h(x) = 1 \ \& \ wt(x) \leq \frac{n-1}{2}\}|$. Without loss of generality consider $T \geq 2^{n-2}$. Hence, $d(h, g) = 2(2^{n-1} - T) = 2^n - 2T$. Now, $nl(f) = nl(h) \leq nl(g) + d(h, g) = (2^{n-1} - \binom{n-1}{\frac{n-1}{2}}) + 2^n - 2T$. Thus, $2^{n-1} - 2^{\frac{n-1}{2}} \leq (2^{n-1} - \binom{n-1}{\frac{n-1}{2}}) + 2^n - 2T$, i.e., $2^{n-1} - T \geq \frac{1}{2}\binom{n-1}{\frac{n-1}{2}} - 2^{\frac{n-1}{2}-1}$.

Thus if one predetermines a ζ, then for a large n we may not satisfy the condition that $\sum_{i=0}^{\frac{n-1}{2}} \binom{n}{i} - |\{x|f(Bx+b) = 1, wt(x) \leq d\}| \leq 2^{\zeta n}$.

In this direction we present the following general result where the constraint of nonlinearity is removed.

Theorem 6 *Suppose $f \in B_n$ be a randomly chosen balanced function. Then the probability to get an affine transformation such that*

$$|\{x|f(Bx + b) = 1, wt(x) \leq \lfloor \frac{n-1}{2} \rfloor\}| > \sum_{i=0}^{\lfloor \frac{n-1}{2} \rfloor} \binom{n}{i} - k \text{ is}$$

1. *less than* $\frac{(n+1)2^n \sum_{i=0}^{k-1} \binom{2^{n-1}}{i}^2}{\binom{2^n}{2^{n-1}}}$ *for n odd.*

2. *less than* $\frac{(n+1)2^n \sum_{i=0}^{k-1} \binom{\sum_{j=0}^{\frac{n}{2}-1} \binom{n}{j}}{i} \binom{2^n - \sum_{j=0}^{\frac{n}{2}-1} \binom{n}{j}}{i + \frac{1}{2}\binom{n}{\frac{n}{2}}}}{\binom{2^n}{2^{n-1}}}$ *for n even.*

Proof : First we prove it for n odd. The number of balanced functions $h \in B_n$ such that $|\{x|h(x) = 1, wt(x) \leq \frac{n-1}{2}\}| > 2^{n-1} - k$ is $\sum_{i=0}^{k-1} \binom{2^{n-1}}{i}^2$ (consider the upper and lower half in the truth table of the function). So, there will be at most $\sum_{i=0}^{k-1} \binom{2^{n-1}}{i}^2$ many affinely invariant classes of such functions. Further the total number of balanced function is $\binom{2^n}{2^{n-1}}$. Hence the total number of affinely invariant classes of balanced function is $\geq \frac{\binom{2^n}{2^{n-1}}}{2^n(2^n-1)(2^n-2^1)\ldots(2^n-2^{n-1})} > \frac{\binom{2^n}{2^{n-1}}}{(n+1)2^n}$. Hence the probability of a randomly chosen balanced function will be function type h is bounded by $\frac{(n+1)2^n \sum_{i=0}^{k-1} \binom{2^{n-1}}{i}^2}{\binom{2^n}{2^{n-1}}}$. Similarly, the case for n even can be proved.

If one takes $k \leq 2^{\frac{3}{4}n}$, then it can be checked easily that the probability decreases fast towards zero as n increases. Thus for a random balanced function f, the probability of getting an affine transformation (which generates the function h from f) such that $|\{x|f(Bx + b) = 1, wt(x) \leq \lfloor \frac{n-1}{2} \rfloor\}| > \sum_{i=0}^{\lfloor \frac{n-1}{2} \rfloor} \binom{n}{i} - 2^{\frac{3}{4}n}$ is almost improbable.

Thus when one randomly chosen balanced function is considered, using the strategy of considering the function after affine transformation, one can indeed reduce the matrix size by constant factor, but the reduction may not be significant in asymptotic terms when the annihilators at the degree of $\lfloor \frac{n-1}{2} \rfloor$ are considered for large n.

5 Additional constraint over maximum AI

In this section we consider the functions $f \in B_n$ with maximum possible AI value $\lceil \frac{n}{2} \rceil$ with the following additional constraint: given $fg = h$, when $\deg(h) = \lceil \frac{n}{2} \rceil$ then $\deg(g)$ must be greater than or equal to $\lfloor \frac{n}{2} \rfloor$.

These functions are indeed better than any functions with only maximum AI with respect to fast algebraic attacks since one can not get a g having $\deg(g) < \lfloor \frac{n}{2} \rfloor$ when $\deg(h)$ is fixed at $\lceil \frac{n}{2} \rceil$. This is the best possible case when $\deg(h)$ is fixed at $\lceil \frac{n}{2} \rceil$ as from (Courtois 2003, Theorem 7.2.1), there always exist g, h, such that $fg = h$, with $\deg(g) + \deg(h) \leq n$.

5.1 Some Basic results

First concentrate on functions having full AI as presented in (Dalai, Gupta, and Maitra 2005; Dalai, Maitra, and Sarkar 2006). For such a function $f \in B_{2k}$, the lowest degree annihilators are at degree k and for its complement $1 + f$, the lowest degree annihilators are at degree $k + 1$ and hence it can be shown that these functions cannot have $fg = h$ relation such that $\deg(h) = k$ and $\deg(g) < k$. Now one can also check that the $(2k + 1)$-variable function $F = x_{2k+1} + f$ is of AI $k + 1$; further F is balanced. One can also check that the function $x_{2k+2} + x_{2k+1} + f$ has algebraic immunity $k + 1$ and it is also an 1-resilient function. We summarize these results below.

Theorem 7

1. *For any even n, it is possible to get unbalanced $f \in B_n$ with maximum possible AI $\frac{n}{2}$ such that given any $fg = h$ relation having $\deg(h) = \frac{n}{2}$, $\deg(g) \neq \frac{n}{2}$.*
2. *For any even n it is possible to get 1-resilient function having full AI.*

With respect to Theorem 7(1), it is open to get such balanced functions f_b when n is even. We solve this problem in Subsection 5.2 for all even n except when n is an exact power of 2 and then considering $x_{n+1} + f_b$ the corresponding case for Theorem 7(2) will be solved for $n + 1$ (odd) variable functions. Note that experimental evidences of resilient functions with full AI are available in (Dalai, Gupta, and Maitra 2004), but no theoretical result is available in the literature.

Note that the results in Theorem 7 are proved using the functions available in (Dalai, Gupta, and Maitra 2005; Dalai, Maitra, and Sarkar 2006) which are of the property that only one of f and $1+f$ has minimum degree annihilators at $\mathcal{AI}(f)$ and the other one has minimum degree annihilators at degree $1 + \mathcal{AI}(f)$. For such functions (Dalai, Gupta, and Maitra 2006, Proposition 5), $wt(f) = 2^{2k-1} - \binom{2k-1}{k}$ (i.e., these functions are not balanced) and $nl(f) \leq 2^{2k-1} - \binom{2k-1}{k}$.

5.2 Annihilators of f and $1 + f$ at the same degree

Now we will concentrate on the functions such that the minimum degree annihilators of the function and its complement are at the same degree but they never cancel out when added. We formally define this as below.

Definition 4 *Suppose $f \in B_{2k}$ be such that $\mathcal{AI}(f) = k$, the maximum possible; the lowest degree annihilators of both f and $1 + f$ are at degree k. Further there is no two nonzero k-degree annihilators g and h of f and $1 + f$ respectively, such that $\deg(g + h) < k$. We denote such functions by P_{2k} functions.*

Theorem 8 *Suppose f be a P_{2k} function. Then*

1. $\mathcal{AI}(x_{2k+1} + f) = k + 1$, *which is the maximum possible;*
2. *if for* $f_1, f_2 \in B_{2k}$, $ff_1 = f_2$ *where* $\deg(f_1) \leq \deg(f_2) = k$ *then* $\deg(f_1) = k$*;*
3. $nl(f) \geq 2^{2k-1} - \binom{2k-1}{k-1}$.

Proof : Let us denote $F = x_{2k+1} + f$. Any annihilator of F is of the form $g_1 + x_{2k+1}(g_1 + g_2)$, where $g_1 \in AN(f)$ and $g_2 \in AN(1 + f)$ and both g_1, g_2 are not 0 at the same time. Similarly any annihilator of $1 + F$ is of the form $g_2 + x_{2k+1}(g_1 + g_2)$. As $g_1 \neq g_2$ and their highest degree terms can not cancel out in $g_1 + g_2$, their degree of the annihilators can not be $\leq k$. Thus $\mathcal{AI}(F) = k + 1$.

Now we prove item 2. Consider we have some f_1, f_2 such that $ff_1 = f_2$ with $\deg(f_1) \leq k$, $\deg(f_2) = k$. Note that $ff_1 = f_2$ iff $f(f_1 + f_2) = 0$ and $(1 + f)f_2 = 0$ (Braeken, Lano, and Praneel 2005). So, $f_1 = (f_1 + f_2) + f_2$ is the sum of the two k degree annihilators $f_1 + f_2$ and f_2 of f and $1 + f$ respectively. As their highest degree terms never cancel out we have $\deg(f_1) = k$.

Next we prove the last item. Since $x_{2k+1} + f$ is of full AI $k + 1$, following (Lobanov 2005, Corollary 1), one gets $nl(x_{2k+1} + f) \geq 2^{2k} - \binom{2k}{k}$. As for every $2k$-variable function f, we have $nl(x_{2k+1} + f) = 2nl(f)$, we get $nl(f) \geq 2^{2k-1} - \binom{2k-1}{k-1}$.

This kind of function provides the best possible relationship when we use functions $f \in B_n$ and consider $fg = h$ relationship with $\deg(h) = \frac{n}{2}$ as in that case $\deg(g)$ can not be less than $\frac{n}{2}$. This is the optimum situation when $\deg(h) = \frac{n}{2}$.

Now consider the following construction from (Dalai, Maitra, and Sarkar 2006; Dalai, Gupta, and Maitra 2006).

Construction 1 *Consider $\zeta_{2k} \in B_{2k}$, $k \geq 0$, as follows:*

$$\zeta_{2k}(x) = \begin{cases} 0 \text{ for } wt(x) < k, \\ a_x \text{ for } wt(x) = k,\ a_x \in \{0, 1\}, \\ 1 \text{ for } wt(x) > k. \end{cases}$$

We will specifically consider the case where the outputs a_x corresponding to weight k inputs take both the distinct values $0, 1$ i.e., the function is nonsymmetric. One can get a balanced $\zeta_{2k}(x)$ if the outputs corresponding to half of the weight k inputs are 0 and the outputs corresponding to half of the weight k inputs are 1.

Note that there are $\begin{pmatrix} \binom{2k}{k} \\ \frac{1}{2}\binom{2k}{k} \end{pmatrix}$ many balanced functions of the form ζ_{2k} in Construction 1. From $\zeta_{2k}(x)$, the following construction is attempted (Dalai, Maitra, and Sarkar 2006; Dalai, Gupta, and Maitra 2006) to get balanced functions.

Construction 2

$$\begin{aligned} G(x_1, \ldots, x_{2k}) &= 0 \text{ for } wt(x_1, \ldots, x_{2k}) < k, \\ &= 1 \text{ for } wt(x_1, \ldots, x_{2k}) > k, \\ &= b(x_1, \ldots, x_{2k}) \text{ for } wt(x_1, \ldots, x_{2k}) = k, \end{aligned}$$

where $b(x_1, \ldots, x_{2k})$ is a Maiorana-McFarland type bent function.

1. *If $wt(G) < 2^{2k-1}$, then we choose $2^{2k-1} - wt(G)$ points randomly from the inputs having weight k and output 0 of G and toggle those outputs to 1 to get ζ_{2k}.*
2. *If $wt(G) > 2^{2k-1}$, then we choose $wt(G) - 2^{2k-1}$ points randomly from the inputs having weight k and output 1 of G and toggle those outputs to 0 to get ζ_{2k}.*

Thus one gets balanced ζ_{2k}.

Now we like to point out the problems with the Constructions 1, 2 where the annihilators of f and $1+f$ are at the same degree.

1. The constructions are randomized and hence the exact nonlinearity of the functions cannot be calculated. In fact, the experimental results show that the nonlinearity of the functions are slightly less than $2^{2k-1} - \binom{2k-1}{k-1}$.
2. Experimental results (Dalai, Gupta, and Maitra 2006, Table 3) show that there exists g having $\deg(g) < k$ such that $\zeta_{2k} g = h$, where $\deg(h) = k$.

We solve these problems in the construction presented in the following subsection where the functions will have nonlinearity not less than $2^{2k-1} - \binom{2k-1}{k-1}$ (see Corollary 2 later) and there cannot be any $\deg(g) < k$ (see Theorem 9 later).

5.3 The exact construction

We present the following construction that has been considered in (Armknecht and Krause 2006, Theorem 1) in terms of algebraic immunity. However, the additional property over algebraic immunity that we are considering here was not studied in (Armknecht and Krause 2006).

Construction 3 *Consider $\eta_{2k} \in B_{2k}$, as follows:*

$\eta_{2k}(x) = \{\, 1$ *for* $wt(x) < k$,
a_x *for* $wt(x) = k,\ a_x \in \{0,1\},$ *with the constraint* $a_x = a_{\overline{x}}$,
0 *for* $wt(x) > k$,

where $\overline{x}$ is the bitwise complement of the vector x. Further all the a_x's are not same, i.e., η_{2k} is non-symmetric.

Theorem 9 *The functions $\eta_{2k}(x)$ as in Construction 3 are P_{2k} functions.*

Proof : Using the similar proof technique used in (Dalai, Maitra, and Sarkar 2006, Theorem 1), one gets that both η_{2k} and $1+\eta_{2k}$ has no annihilators at degree less than k. Further, $\sum_{i=0}^{k} \binom{n}{i}$ is greater than both $wt(\eta_{2k})$ and $wt(1+\eta_{2k})$ and hence from (Dalai, Gupta, and Maitra 2004, Theorem 1), both η_{2k} and $1+\eta_{2k}$ must have annihilator at degree less than or equal to k. Hence both $\eta_{2k}(x)$ and $1+\eta_{2k}(x)$ have minimum degree annihilators exactly at degree k.

Any k degree function $g \in B_{2k}$ can be written as

$$a_0 + \sum_{i=0}^{n} a_i x_i + \ldots + \sum_{1 \le < i_1 < \ldots < i_k \le n} a_{i_1,\ldots,i_k} x_{i_1} \ldots x_{i_k},$$

where the coefficients a's are either 0 or 1. If g is an annihilator of η_{2k} then $g(x) = 0$ when $\eta_{2k}(x) = 1$. Since $\eta_{2k}(x) = 1$ for $wt(x) < k$, we can eliminate all the coefficients (a's) associated to monomials of degree $\le k-1$ of g. Then we have $\eta_{2k}(x) = 1$ for some input vectors x of weight k. For such an $x = (b_1, \ldots, b_n)$, where $b_{i_1} = \ldots = b_{i_k} = 1$ and rest 0, one can eliminate the coefficient $a_{i_1,\ldots,i_k}$. Thus the k degree independent annihilators of η_{2k} form the set $S_1 = \{x_{j_1} \ldots x_{j_k} : \eta_{2k}(b_1, \ldots, b_n) = 0$ and $b_{j_1} = \ldots = b_{j_k} = 1,$ rest are $0\}$. Here any k-degree annihilator of η_{2k} does not contain any monomial of degree $< k$.

Define $f'(x) = 1 + \eta_{2k}(\overline{x})$. Following the similar proof for $\eta_{2k}(x)$, one can prove that the space of k degree annihilators of f' is generated by the basis set $\{x_{j_1} \ldots x_{j_k} : f'(b_1, \ldots, b_n) = 0$ and $b_{j_1} = \ldots = b_{j_k} = 1,$ rest are $0\}$. Hence, the k degree annihilator space of $f'(\overline{x}) = 1 + \eta_{2k}(x)$ is generated by the basis set $\{(1+x_{j_1}) \ldots (1+x_{j_k}) : f'(1+b_1, \ldots, 1+b_n) = 1 + \eta_{2k}(b_1, \ldots, b_n) = 0$ and $b_{j_1} = \ldots = b_{j_k} = 0,$ rest are $1\}$. So, the subspace of k degree monomials of k degree annihilators of $1+\eta_{2k}$ is generated by the basis set $S_2 = \{x_{j_1} \ldots x_{j_k} : \eta_{2k}(b_1, \ldots, b_n) =$

1 and $b_{j_1} = \ldots = b_{j_k} = 0,$ rest are $1\}$. One can check that these two sets S_1 and S_2 are disjoint iff $\eta_{2k}(x) = \eta_{2k}(\overline{x})$ for $wt(x) = k$.

Since the basis sets S_1, S_2 are disjoint, the k degree terms of any annihilator of η_{2k} and the k degree terms of any annihilator of $1 + \eta_{2k}$ cannot be the same. Thus the proof.

Corollary 2 $nl(\eta_{2k}) \geq 2^{2k-1} - \binom{2k-1}{k-1}$.

Proof : Consider the $(2k+1)$-variable function $x_{2k+1} + \eta_{2k}(x_1, \ldots, x_{2k})$. As $\eta_{2k}(x)$ is a P_{2k} function, from item 1 of Theorem 8, $x_{2k+1} + \eta_{2k}$ is of full AI $k+1$ and hence following (Lobanov 2005, Corollary 1), one gets $nl(x_{2k+1} + \eta_{2k}) \geq 2^{2k} - \binom{2k}{k}$. As for every $2k$-variable function f, we have $nl(x_{2k+1} + f) = 2nl(f)$, we get the proof.

Note that the above proof is similar to the proof of (Carlet, Dalai, Gupta, and Maitra 2006, Theorem 6). Now we concentrate on balanced functions.

Corollary 3 *One can get a balanced η_{2k} iff $2k$ is not a power of 2 and the count of such balanced functions is* $\begin{pmatrix} \frac{1}{2}\binom{2k}{k} \\ \frac{1}{4}\binom{2k}{k} \end{pmatrix}$.

Proof : For a $2k$-variable function, there are $\binom{2k}{k}$ many input vectors of weight k and there are $\frac{1}{2}\binom{2k}{k}$ many $(x, \overline{x})$ distinct pairs of weight k. One can construct a balanced η_{2k} if and only if $\frac{1}{2}\binom{2k}{k}$ is even, i.e., $\binom{2k}{k}$ is divisible by 4. Since $\binom{2k}{k} = 2\binom{2k-1}{k-1}$, we need to test whether $\binom{2k-1}{k-1}$ is even.

Suppose the $t = \lfloor \log_2 2k \rfloor + 1$ bit binary representations of $2k, k, 2k-1$ and $k-1$ are as follows (most significant bit at the left most position):

$$\begin{array}{lcccccccccc} 2k & = & b_t & b_{t-1} & \ldots & b_{l+1} & b_l = 1 & 0 & 0 & \ldots & 0, \\ k & = & 0 & b_t & \ldots & b_{l+2} & b_{l+1} & b_l = 1 & 0 & \ldots & 0, \\ 2k-1 & = & b_t & b_{t-1} & \ldots & b_{l+1} & 1 + b_l = 0 & 1 & 1 & \ldots & 1, \\ k-1 & = & 0 & b_t & \ldots & b_{l+2} & b_{l+1} & 1 + b_l = 0 & 1 & \ldots & 1, \end{array}$$

where $1 < l \leq t$, $b_i \in \{0, 1\}$ and $b_t = b_l = 1$. Now following Lucas' theorem (Comtet 1974, Page 79) with the prime 2, we have $\binom{2k-1}{k-1} \equiv \binom{b_t}{0}\binom{b_{t-1}}{b_t} \ldots \binom{0}{b_{l+1}}\binom{1}{0}\binom{1}{1} \ldots \binom{1}{1} \bmod 2$. If $2k$ is a power of 2, then $t = l$. So, $\binom{2k-1}{k-1} \equiv \binom{1}{0}\binom{1}{1} \ldots \binom{1}{1} \bmod 2$, i.e., $\binom{2k-1}{k-1} \equiv 1 \bmod 2$. Hence $\binom{2k-1}{k-1}$ is odd.

If $2k$ is not a power of 2, then $\binom{2k-1}{k-1} \equiv \binom{b_{t-1}}{b_t} \ldots \binom{b_{l+1}}{b_{l+2}}\binom{0}{b_{l+1}} \bmod 2$. At some place we will get $b_s = 0$ and $b_{s+1} = 1$ for $l \leq s < t$ because $b_t = 1$. Hence $\binom{2k-1}{k-1}$ is even if $2k$ is not a power of 2.

Thus $\binom{2k}{k}$ is divisible by 4, when $2k$ is not exactly a power of 2. In such a case, there will be $\frac{1}{2}\binom{2k}{k}$ many distinct pairs of $(x, \overline{x})$, where x is a $2k$ bit binary pattern of weight k. One can choose $\frac{1}{4}\binom{2k}{k}$ many distinct pairs and in such inputs of η_{2k}, output 1 is assigned and for the rest of $\frac{1}{4}\binom{2k}{k}$ many distinct pairs of inputs, output 0 is assigned. This provides a balanced η_{2k}. Note that the number of such distinct balanced η_{2k} is $\begin{pmatrix} \frac{1}{2}\binom{2k}{k} \\ \frac{1}{4}\binom{2k}{k} \end{pmatrix}$.

Now an important question is whether there exist balanced P_{2k} functions when $2k$ is a power of 2. We have checked that for $2k = 4 = 2^2$, there is no balanced P_4 function by running exhaustive computer program.

For $2k = 6$, we have exhaustively checked all the η_6 functions. There are $2^{10} - 2$ such functions (including $\binom{10}{5}$ many balanced functions). All of them are of nonlinearity 22 and algebraic degree either 4 or 5. The algebraic degree of all 252 balanced functions is 5. If we consider the $\eta_6 g = h$ kind of relations, then we find $\deg(g) = 1$, when $\deg(h) = 4$ for each of the η_6 functions.

For $2k = 10$, it is not possible to experimentally study all the η_{10} functions. We have checked randomly chosen 100 many balanced η_{10} functions. Always we achieved the nonlinearity 386 and algebraic degree 8. If we consider the $\eta_{10} g = h$ kind of relations, then we find $\deg(g) \geq 2$, when $\deg(h) = 6$ for each of the 100 many balanced η_{10} functions we have randomly chosen.

Similarly, for $2k = 12$, we have checked randomly chosen 100 many balanced η_{12} functions. Always we achieved the nonlinearity 1586 and algebraic degree 8. If we consider the $\eta_{12} g = h$ kind of relations, then we find $\deg(g) \geq 3$, when $\deg(h) = 7$ for each of the 100 many balanced η_{12} functions we have randomly chosen.

Theoretically proving the algebraic degree and nonlinearity of balanced η_{2k} functions and finding the degrees of g, when $\eta_{2k} g = h$ and $\deg(h) > k$ are interesting open questions.

5.4 Functions on odd number of input variables

Now let us study the functions f on odd number of input variables $2k + 1$ having maximum possible AI $k + 1$. That is the functions must be balanced (Dalai, Gupta, and Maitra 2004). Consider the following balanced symmetric functions (Dalai, Maitra, and Sarkar 2006; Braeken and Praneel 2005; Braeken, Lano, and Praneel 2005) on $2k + 1$ variables having full algebraic immunity $k + 1$.

Construction 4 *Consider* $\tau_{2k+1} \in B_{2k+1}$, *as follows:*

$$\tau_{2k+1}(x) = \begin{cases} 1 \text{ for } wt(x) \leq k, \\ 0 \text{ for } wt(x) \geq k+1, \end{cases}$$

We list a few experimental values of minimum degree of g when $\tau_{2k+1} g = h$ and $\deg(h) = k + 1$. In the format $< 2k + 1, \deg(g), \deg(h) >$ these values are $< 5, 1, 3 >$, $< 7, 1, 4 >$, $< 9, 1, 5 >$, $< 11, 2, 6 >$. Note that the minimum degree of g is substantially less than k and hence the functions τ_{2k+1} are not interesting in resistance against fast algebraic attacks.

To get a better resistance against fast algebraic attack, we are interested about the balanced functions with the following additional property. Given any $fg = h$ relation having $\deg(h) = k + 1$, we require that $\deg(g) \geq k$.

We run exhaustive search for $2k + 1 = 5$ variable functions and found such functions. One example is the truth table 00000001000101110001101111011111 which is of nonlinearity 10 and algebraic degree 4. Note that there is no nonlinearity 12 function on 5 variables with such property. Existence of such functions for 7 variables onwards is an open question.

6 Conclusion

In this paper, first we study how to reduce the matrix size which is involved in finding the annihilators of a Boolean function. Our results show that considerable reduction in the size of the matrix is achievable. We identify the classes where it provides asymptotic improvement. We also note that for randomly chosen balanced functions, the improvement is rather constant than asymptotic. The reduction in matrix size helps in running the actual annihilator finding steps by Gaussian elimination method. Though our method is less efficient in general than the recently known efficient algorithms (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006; Didier and Tillich 2006) to find the annihilators, our main motivation is to theoretically understand the structure of the matrix involved. Further, for certain classes of functions, our technique provides better effciency than the currently best known algorithm (Armknecht, Carlet, Gaborit, Kuenzli, Meier, and Ruatta 2006).

Next we present theoretical results on balanced Boolean functions having some additional properties over maximum possible AI. Our construction provides n-variable (n even) balanced functions f with maximum possible AI

$\frac{n}{2}$ such that given two n-variable Boolean functions g, h with $fg = h$, if $\deg(h) = \frac{n}{2}$, then $\deg(g)$ will be greater than or equal to $\frac{n}{2}$. This is the first time such a result is demonstrated. Following this result, one can get theoretical construction of resilient Boolean functions having maximum possible AI. Though the nonlinearity of the functions we construct are not encouraging to use them as building blocks in cryptosystem, our results provide theoretical insights in the area of constructing Boolean functions that are resistant to certain kinds of algebraic attacks.

References

Armknecht, F. (2004). Improving fast algebraic attacks. In *Fast Software Encryptions 2004, Proceedings*, Volume 3017 of *Lecture Notes in Computer Science*, pp. 65–82. Springer.

Armknecht, F., C. Carlet, P. Gaborit, S. Kuenzli, W. Meier, and O. Ruatta (2006). Efficient computation of algebraic immunity for algebraic and fast algebraic attacks. In *Advances in Cryptology - EUROCRYPT 2006, Proceedings*, Volume 4004 of *Lecture Notes in Computer Science*, pp. 147–164. Springer.

Armknecht, F. and M. Krause (2003). Algebraic attacks on combiners with memory. In *Advances in Cryptology - CRYPTO 2003, Proceedings*, Volume 2729 of *Lecture Notes in Computer Science*, pp. 162–175. Springer.

Armknecht, F. and M. Krause (2006). Constructing single- and multi-output boolean functions with maximal immunity. In *33rd International Colloquium on Automata, Languages and Programming(ICALP) 2006, Proceedings*, Volume 4052 of *Lecture Notes in Computer Science*, pp. 180–191. Springer.

Ars, G. and J. Faugére (2005). Algebraic immunities of functions over finite fields. INRIA Techno report.

Batten, L. M. (2004). Algebraic attacks over gf(q). In *Progress in Cryptology - INDOCRYPT 2004, Proceedings*, Volume 3348 of *Lecture Notes in Computer Science*, pp. 84–91. Springer.

Braeken, A., J. Lano, N. Mentens, B. Praneel, and I. Verbauwhede (2005). Sfinks: A synchronous stream cipher for restricted hardware environments. In *SKEW - Symmetric Key Encryption Workshop, 2005, Proceedings*.

Braeken, A., J. Lano, and B. Praneel (2005). Evaluating the resistance of filters and combiners against fast algebraic attacks. Eprint on ECRYPT. http://eprint.iacr.org/.

Braeken, A., J. Lano, and B. Praneel (2006). Evaluating the resistance of stream ciphers with linear feedback against fast algebraic attacks. In *11th Australasian Conference on Information Security and Privacy(ACISP) 2006, Proceedings*, Volume 4058 of *Lecture Notes in Computer Science*, pp. 40–51. Springer.

Braeken, A. and B. Praneel (2005). On the algebraic immunity of symmetric boolean functions. In *Progress in Cryptology - INDOCRYPT 2005, Proceedings*, Volume 3797 of *Lecture Notes in Computer Science*, pp. 35–48. Springer. Also available at Cryptology ePrint Archive, http://eprint.iacr.org/, No. 2005/245, 26 July, 2005.

Canteaut, A. (2005). Open problems related to algebraic attacks on stream ciphers. In *International Workshop on Coding and Cryptography(WCC) 2005. Proceedings*, pp. 1–10.

Carlet, C., D. K. Dalai, K. C. Gupta, and S. Maitra (2006). Algebraic immunity for cryptographically significant boolean functions: Analysis and construction. *IEEE Transactions on Information Theory 52*(7), 3105–3121.

Cheon, J. H. and D. H. Lee (2004). Resistance of s-boxes against algebraic attacks. In *Fast Software Encryptions 2004, Proceedings*, Volume 3017 of *Lecture Notes in Computer Science*, pp. 83–94. Springer.

Cho, J. Y. and J. Pieprizyk (2004). Algebraic attacks on sober-t32 and sober-128. In *Fast Software Encryptions 2004, Proceedings*, Volume 3017 of *Lecture Notes in Computer Science*, pp. 49–64. Springer.

Comtet, L. (1974). *Advanced combinatorics*. Reidel Publication.

Coppersmith, D. and S. Winograd (1990). Matrix multiplication via arithmatic programming. *Journal of Symbolic Computation 9*(3), 251–280.

Courtois, N. (2003). Fast algebraic attacks on stream ciphers with linear feedback. In *Advances in Cryptology - CRYPTO 2003, Proceedings*, Volume 2729 of *Lecture Notes in Computer Science*, pp. 176–194. Springer.

Courtois, N. (2005). Cryptanalysis of sfinks. In *8th International Conference on Information Security and Cryptology(ICISC) 2005, Proceedings*, Volume 3935 of *Lecture Notes in Computer Science*, pp. 261–269. Springer.Also available at Cryptology ePrint Archive, http://eprint.iacr.org/, Report 2005/243, 2005.

Courtois, N., B. Debraize, and E. Garrido (2005). On exact algebraic [non-]immunity of s-boxes based on power functions. In *11th Australasian Conference on Information Security and Privacy(ACISP) 2006, Proceedings*, Volume 4058 of *Lecture Notes in Computer Science*, pp. 76–86. Springer.

Courtois, N. and W. Meier (2003). Algebraic attacks on stream ciphers with linear feedback. In *Advances in Cryptology - EUROCRYPT 2003, Proceedings*, Volume 2656 of *Lecture Notes in Computer Science*, pp. 345–359. Springer.

Courtois, N. and J. Pieprzyk (2002). Cryptanalysis of block ciphers with overdefined systems of equations. In *Advances in Cryptology - ASIACRYPT 2002, Proceedings*, Volume 2501 of *Lecture Notes in Computer Science*, pp. 267–287. Springer.

Dalai, D. K., K. C. Gupta, and S. Maitra (2004). Results on algebraic immunity for cryptographically significant boolean functions. In *Progress in Cryptology - INDOCRYPT 2004, Proceedings*, Volume 3348 of *Lecture Notes in Computer Science*, pp. 92–106. Springer.

Dalai, D. K., K. C. Gupta, and S. Maitra (2005). Cryptographically significant boolean functions: Construction and analysis in terms of algebraic immunity. In *Fast Software Encryptions 2005, Proceedings*, Volume 3557 of *Lecture Notes in Computer Science*, pp. 98–111. Springer.

Dalai, D. K., K. C. Gupta, and S. Maitra (2006). Notion of algebraic immunity and its evaluation related to fast algebraic attacks. In *Second Workshop on Boolean Functions: Cryptography and Applications (BFCA 2006). Proceedings*. .Also available at Cryptology ePrint Archive, http://eprint.iacr.org/, No. 2006/018.

Dalai, D. K., S. Maitra, and S. Sarkar (2006). Basic theory in construction of boolean functions with maximum possible annihilator immunity. *Design, Codes and Cryptography 40*(1), 41–58.

Didier, F. (2006). Using wiedemann's algorithm to compute the immunity against algebraic and fast algebraic attacks. In *Progress in Cryptology - INDOCRYPT 2006, Proceedings*, Volume 4329 of *Lecture Notes in Computer Science*, pp. 236–250. Springer.

Didier, F. and J. Tillich (2006). Computing the algebraic immunity efficiently. In *Fast Software Encryptions 2006, Proceedings*, Volume 4047 of *Lecture Notes in Computer Science*, pp. 359–374. Springer.

Lee, D. H., J. Kim, J. Hong, J. W. Han, and D. Moon (2004). Algebraic attacks on summation generators. In *Fast Software Encryptions 2004, Proceedings*, Volume 3017 of *Lecture Notes in Computer Science*, pp. 34–48. Springer.

Lobanov, M. (2005). Tight bound between nonlinearity and algebraic immunity. Cryptology ePrint Archive, Report 2005/441. http://eprint.iacr.org/.

Meier, W., E. Pasalic, and C. Carlet (2004). Algebraic attacks and decomposition of boolean functions. In *Advances in Cryptology - EUROCRYPT 2004, Proceedings*, Volume 3027 of *Lecture Notes in Computer Science*, pp. 474–491. Springer.

Nawaz, Y., G. Gong, and K. C. Gupta (2006). Upper bounds on algebraic immunity of power functions. In *Fast Software Encryptions 2006, Proceedings*, Volume 4047 of *Lecture Notes in Computer Science*, pp. 375–389. Springer.

Strassen, V. (1969). Guassian elimination is not optimal. *Numerische Mathematik 13*, 354–356.

Appendix A: Comparison with Meier et. al. Algorithm

Here we compare the time and space complexity of our strategy with (Meier, Pasalic, and Carlet 2004, Algorithm 2). In paper (Meier, Pasalic, and Carlet 2004), Algorithm 2 is probabilistic. In this section we study the time and space complexity of the algorithm along with it's deterministic version. Using these algorithms we check whether there exist annihilators of degree less than or equal to d of an n-variable function f. As we have already described, ANF of any n-variable function g of degree d is of the form

$$g(x_1, \ldots, x_n) = a_0 + \sum_{i=0}^{n} a_i x_i + \cdots + \sum_{1 \leq i_1 < i_2 \cdots < i_d \leq n} a_{i_1,\ldots,i_d} x_{i_1} \cdots x_{i_d}$$

where subscripted a's are from $\{0, 1\}$. First we present the exact probabilistic algorithm (Meier, Pasalic, and Carlet 2004, Algorithm 2).

Algorithm 2
Input: $f \in B_n$ *and* n.
Output: $AI(f)$.

1. *Initialize weight* $w = 0$.
2. *For all* x*'s of weight* w *with* $f(x) = 1$*, substitute each* x *in* $g(x) = 0$ *to derive a linear equation on the coefficients of* g*, with a single coefficient of* w *degree monomial. Use this equation to express this coefficient iteratively by coefficients of lower degree monomials.*
3. *If* $w < d$*, increment* w *by* 1 *and go to step 2.*
4. *Choose random arguments* x *of arbitrary weight such that* $f(x) = 1$ *and substitute in* $g(x) = 0$*, until there are same number of equations as unknowns.*
5. *Solve the linear system. If there is no solution, output no annihilator of degree* d*, but if there is a solution then it is not clear whether there is an annihilator of degree* d *or not.*

Next we present the deterministic version of the original probabilistic algorithm (Meier, Pasalic, and Carlet 2004, Algorithm 2).

Algorithm 3
Input: $f \in B_n$ *and* n.
Output: $AI(f)$.

1. *Initialize weight* $w = 0$.
2. *For all* x*'s of weight* w *with* $f(x) = 1$*, substitute each* x *in* $g(x) = 0$ *to derive a linear equation in the coefficients of* g*, with a single coefficient of* w *degree monomials. Use this equation to express this coefficient iteratively by coefficients of lower degree monomials.*
3. *If* $w < d$*, increment* w *by* 1 *and go to step 2.*
4. *Substitute* x *such that* $wt(x) > d$ *and* $f(x) = 1$ *in* $g(x) = 0$ *to get linear equation in the coefficient of* g.
5. *Solve the linear system. Output no annihilator of degree* d *iff there is no nonzero solution.*

Since first three steps of both algorithms are same, we initially study the time and space complexity of both the algorithms for first three steps for a randomly chosen balanced function f. In step 2, we apply x, such that $wt(x) \leq d$ and $f(x) = 1$, in $g(x)$ and hence we get a linear equation in the coefficient of g such that a single coefficient of that weight is expressed as linear combination of its lower weight coefficients. Here we consider a particular w for each iteration. As f is random and balanced, one can expect that there are $\frac{1}{2}\binom{n}{w}$ many input vectors of weight w in set $supp(f)$. For each $x = (x_1, \ldots, x_n) \in supp(f)$ where $x_{i_1}, \ldots, x_{i_w}$ are 1 and others are 0 of weight w, we will get linear equation of the form

$$a_{i_1,\ldots,i_w} = a_0 + \sum_{j=1}^{w} a_{i_j} + \ldots + \sum_{\{k_1,\ldots,k_{w-1}\} \subset \{i_1,\ldots i_w\}} a_{k_1,\ldots,k_{w-1}}. \tag{3}$$

To store one equation we need $\sum_{i=0}^{w}\binom{n}{i}$ many memory bits (some places will be 0, some will be 1). There are $\sum_{i=0}^{w-1}\binom{w}{i}$ many coefficients in the right hand side of the Equation 3. As f is random, one can expect that half of them can be eliminated using the equations obtained by lower weight input support vectors. So, $\sum_{i=0}^{w}\binom{n}{i} + \frac{1}{2}\sum_{i=0}^{w-1}(\binom{w}{i}\sum_{j=0}^{i-1}\binom{n}{j})$ order of computation is required to establish an equation. Here w varies from 0 to d and there are approximately $\frac{1}{2}\sum_{w=0}^{d}\binom{n}{w}$ many support vectors of weight less than or equal to d. Hence at the starting of step 4 the space complexity is

$$S1 = \frac{1}{2}\sum_{w=0}^{d}(\binom{n}{w}\sum_{i=0}^{w}\binom{n}{i})$$

and time complexity is

$$T1 = \frac{1}{2}\sum_{w=0}^{d}(\binom{n}{w}(\sum_{i=0}^{w}\binom{n}{i} + \frac{1}{2}\sum_{i=0}^{w-1}\binom{w}{i}\sum_{j=0}^{i-1}\binom{n}{j})).$$

Now we study the time and space complexity for steps 4 and 5 in both probabilistic and deterministic version. To represent each equation for the system of equation one needs $\sum_{w=0}^{d}\binom{n}{w}$ memory bits.

First we consider the probabilistic one. For probabilistic case one has to choose approximately $\frac{1}{2}\sum_{w=0}^{d}\binom{n}{w}$ many support input vectors of weight greater than d. Hence each linear equation obtained from these vectors has at least $\sum_{i=0}^{d}\binom{d+1}{i}$ many coefficients of g and half of them can be eliminated using the equations obtained in previous steps. So, to get each equation one needs at least $\sum_{w=0}^{d}\binom{n}{w} + \frac{1}{2}\sum_{i=0}^{d}(\binom{d+1}{i}\sum_{j=0}^{i-1}\binom{n}{j})$ computations. Hence the space complexity during 4th step is $SP2 \geq \frac{1}{2}(\sum_{w=0}^{d}\binom{n}{w})^2$ and time complexity is $TP2 \geq \frac{1}{2}\sum_{w=0}^{d}\binom{n}{w}(\sum_{w=0}^{d}\binom{n}{w} + \frac{1}{2}\sum_{i=0}^{d}(\binom{d+1}{i}\sum_{j=0}^{i-1}\binom{n}{j}))$. Finally, to generate system of homogeneous linear equations one requires

$SP = S1 + SP2 \geq \frac{1}{2}\sum_{w=0}^{d}(\binom{n}{w}\sum_{i=0}^{w}\binom{n}{i}) + \frac{1}{2}(\sum_{w=0}^{d}\binom{n}{w})^2$ memory bits and

$TP = T1 + TP2 \geq \frac{1}{2}\sum_{w=0}^{d}(\binom{n}{w}(\sum_{i=0}^{w}\binom{n}{i} + \frac{1}{2}\sum_{i=0}^{w-1}\binom{w}{i}\sum_{j=0}^{i-1}\binom{n}{j})) + \frac{1}{2}\sum_{w=0}^{d}\binom{n}{w}(\sum_{w=0}^{d}\binom{n}{w} + \frac{1}{2}\sum_{i=0}^{d}(\binom{d+1}{i}\sum_{j=0}^{i-1}\binom{n}{j}))$

computations. In step 5, we have to solve $\frac{1}{2}\sum_{w=0}^{d}\binom{n}{w}$ many linear equations with same number of variables. To solve this system one needs $TP3 = (\frac{1}{2}\sum_{w=0}^{d}\binom{n}{w})^3$ computations using the Gaussian elimination technique.

Now we study space and time complexity for deterministic one. Since f is balanced, there are approximately $2^{n-1} - \frac{1}{2}\sum_{w=0}^{d}\binom{n}{w} = \frac{1}{2}\sum_{w=d+1}^{n}\binom{n}{w}$ many support vectors having weight greater than d and these many are considered to find out equations. Hence each linear equation obtained from these vectors of weight $w > d$ contains $\sum_{i=0}^{d}\binom{w}{i}$ many coefficients of g and half of them can be eliminated using the equations obtained in steps 1, 2 and 3.

To get this equation one needs $\sum_{i=0}^{d}\binom{n}{i}+\frac{1}{4}\sum_{i=0}^{d}((\binom{w}{i}\sum_{j=0}^{i-1}\binom{n}{j}))$ computations. Hence the total space complexity during 4th step is $SD2=\frac{1}{4}\sum_{w=d+1}^{n}\binom{n}{w}\sum_{w=0}^{d}\binom{n}{d})$ and time complexity is $TD2=\frac{1}{2}\sum_{w=d+1}^{n}\binom{n}{w}(\sum_{i=0}^{d}\binom{n}{i}+\frac{1}{4}\sum_{i=0}^{d}((\binom{w}{i}\sum_{j=0}^{i-1}\binom{n}{j}))$. Finally, to generate homogeneous linear equations one needs

$SD=S1+SD2=\frac{1}{2}\sum_{w=0}^{d}((\binom{n}{w}\sum_{i=0}^{w}\binom{n}{i}))+\frac{1}{4}\sum_{w=d+1}^{n}\binom{n}{w}\sum_{w=0}^{d}\binom{n}{d}))$ memory bits and

$TD=T1+TD2=\frac{1}{2}\sum_{w=0}^{d}((\binom{n}{w}(\sum_{i=0}^{w}\binom{n}{i}+\frac{1}{2}\sum_{i=0}^{w-1}\binom{w}{i}\sum_{j=0}^{i-1}\binom{n}{j}))+\frac{1}{2}\sum_{w=d+1}^{n}\binom{n}{w}(\sum_{i=0}^{d}\binom{n}{i}+\frac{1}{4}\sum_{i=0}^{d}((\binom{w}{i}\sum_{j=0}^{i-1}\binom{n}{j}))$

computations. Further, in step 5, we have to solve $\frac{1}{2}\sum_{w=d+1}^{n}\binom{n}{w}$ many linear equations with $\frac{1}{2}\sum_{w=0}^{d}\binom{n}{w}$ number of variables. To solve this system one needs $TD3=(\frac{1}{2}\sum_{w=d+1}^{n}\binom{n}{w})^3$ computations.

The system of equations generated by our strategy as well as Meier et al (Meier, Pasalic, and Carlet 2004) algorithms are same. So, it takes same complexities to solve them. Only difference is during generation of the system of equations. In the following table we show the complexities for both algorithms for generating the system of equations.

Table 2. Time and Space complexity comparison of Probabilistic algorithms to generate equations.

	Space	Time
Meier's algorithm	$\frac{1}{2}\sum_{w=0}^{d}((\binom{n}{w}\sum_{i=0}^{w}\binom{n}{i}))$ $+\frac{1}{2}(\sum_{w=0}^{d}\binom{n}{w})^2$	$\frac{1}{2}\sum_{w=0}^{d}((\binom{n}{w}(\sum_{i=0}^{w}\binom{n}{i}+\frac{1}{2}\sum_{i=0}^{w-1}\binom{w}{i}\sum_{j=0}^{i-1}\binom{n}{j}))+$ $\frac{1}{2}\sum_{w=0}^{d}\binom{n}{w}(\sum_{w=0}^{d}\binom{n}{w}+\frac{1}{2}\sum_{i=0}^{d}(\binom{d+1}{i}\sum_{j=0}^{i-1}\binom{n}{j}))$
Our algorithm	$\frac{1}{4}(\sum_{w=0}^{d}\binom{n}{w})^2$	$\frac{1}{4}(\sum_{w=0}^{d}\binom{n}{w})^2$

Table 3. Time and Space complexity comparison of Deterministic algorithms to generate equations.

	Space	Time
Meier's algorithm	$\frac{1}{2}\sum_{w=0}^{d}((\binom{n}{w}\sum_{i=0}^{w}\binom{n}{i}))+$ $\frac{1}{4}\sum_{w=d+1}^{n}\binom{n}{w}\sum_{w=0}^{d}\binom{n}{d}))$	$\frac{1}{2}\sum_{w=0}^{d}((\binom{n}{w}(\sum_{i=0}^{w}\binom{n}{i}+\frac{1}{2}\sum_{i=0}^{w-1}\binom{w}{i}\sum_{j=0}^{i-1}\binom{n}{j}))+$ $\frac{1}{2}\sum_{w=d+1}^{n}\binom{n}{w}(\sum_{i=0}^{d}\binom{n}{i}+\frac{1}{4}\sum_{i=0}^{d}((\binom{w}{i}\sum_{j=0}^{i-1}\binom{n}{j}))$
Our algorithm	$\frac{1}{4}\sum_{w=d+1}^{n}\binom{n}{w}\sum_{w=0}^{d}\binom{n}{w}$	$\frac{1}{4}\sum_{w=d+1}^{n}\binom{n}{w}\sum_{w=0}^{d}\binom{n}{w}$

Deepak Kumar Dalai *received his Master of Science in Mathematics degree in the year 2001 from Utkal University, Bhubaneswar, India and Master of Technology in Computer Science in the year 2003 from Indian Statistical Institute, Kolkata. He obtained Ph.D. degree in Computer Science from Indian Statistical Institute, Kolkata in 2006. Currently he is with the Mathematics Department of National Institute of Science Education and Research, Bhubaneswar, India.*

Subhamoy Maitra *received his Bachelor of Electronics and Telecommunication Engineering degree in the year 1992 from Jadavpur University, Kolkata and Master of Technology in Computer Science in the year 1996 from Indian Statistical Institute, Kolkata. He has completed Ph.D. from Indian Statistical Institute in 2001. Currently he is an Associate Professor at Indian Statistical Institute. His research interest is in Cryptology.*