Efficient probabilistic algorithm for estimating the algebraic properties of Boolean functions for large n

Yongzhuang Wei [*] Enes Pasalic [†] Fengrong Zhang [‡] Samir Hodžić [§]

Abstract

Although several methods for estimating the resistance of a random Boolean function against (fast) algebraic attacks were proposed, these methods are usually infeasible in practice for relative large input variables n (for instance $n \geq 30$) due to increased computational complexity. An efficient estimation the resistance of Boolean function (with relative large input variables n) against (fast) algebraic attacks appears to be a rather difficult task. In this paper, the concept of partial linear relations decomposition is introduced, which decomposes any given nonlinear Boolean function into many linear (affine) subfunctions by using the disjoint sets of input variables. Based on this result, a general probabilistic decomposition algorithm for nonlinear Boolean functions is presented which gives a new framework for estimating the resistance of Boolean function against (fast) algebraic attacks. It is shown that our new probabilistic method gives very tight estimates (lower and upper bound) and it only requires about $O(n^2 2^n)$ operations for a random Boolean function with n variables, thus having much less time complexity than previously known algorithms.

Keywords : Stream ciphers, fast algebraic attacks, time complexity, algebraic immunity.

[*]Guangxi Key Laboratory of Cryptography and Information Security, Guilin University of Electronic Technology, Guilin 541004, P.R. China, and also with Guangxi Cooperative Innovation Center of Cloud Computing and Big Data, Guilin University of Electronic Technology, Guilin 541004, P.R. China, e-mail: walker_wei@msn.com. Yongzhuang Wei is supported in part by the Natural Science Foundation of China (61572148), in part by the Guangxi Natural Science Found (2015GXNSFGA139007), in part by the project of Outstanding Young Teachers Training in Higher Education Institutions of Guangxi.

[†]University of Primorska, FAMNIT and IAM, Koper, Slovenia, e-mail: enes.pasalic6@gmail.com. Enes Pasalic is partly supported by the Slovenian Research Agency (research program P3-0384 and research project J1-6720).

[‡]School of Computer Science and Technology, China University of Mining and Technology, Xuzhou, Jiangsu 221116, P.R. China, e-mail: zhfl203@cumt.edu.cn. Fengrong Zhang is supported in part by National Science Foundation of China(61303263), and in part by the Fundamental Research Funds for the Central Universities (Grant No. 2015XKMS086), and in part by the China Postdoctoral Science Foundation funded project (Grant No. 2015T80600).

[§]University of Primorska, FAMNIT, Koper, Slovenia, e-mail: samir.hodzic@famnit.upr.si

1 Introduction

Boolean functions play an important role in the design of symmetric encryption algorithms, more precisely in certain designs of stream ciphers. For instance, nonlinear filter generator and combination generator are two typical representative of hardware oriented design schemes, which consist of single or multiple linear feedback shift registers (LFSRs) and a nonlinear Boolean function. The security of these LFSR-based stream ciphers heavily relies on the algebraic properties of the used Boolean function. Over the last decades, Boolean functions satisfying some particular cryptographic properties (such as high nonlinearity, high algebraic immunity (AI) etc.) have been studied [3, 11, 18, 19].

Algebraic attacks (AA) and fast algebraic attacks (FAA) were respectively proposed in [4, 5], which are two famous and powerful attacks that are easily applied to LFSR-based stream ciphers. The core idea behind the two attacks can be summarized as follows. The first step is to set up a low degree algebraic system of multivariate equations in the secret key/state bits, where the degree of these equations is closely related to the algebraic properties of the used nonlinear Boolean function. The second step is to solve the system of equations and recover the secret key/state bits. Whereas the second step is well elaborated and understood, the first step of finding low degree multivariate equations for relatively large number of input variables n is still an open problem due to the complexity issues.

The concept of algebraic immunity for an arbitrary Boolean function f was introduced in [15] and it reflects the resistance of a Boolean function f against AA. More precisely, this criterion measures the minimum algebraic degree of its annihilators, i.e., $AI_f = \min_{\deg(g)}\{A(f), A(f \oplus 1)\}$, where $A(f) = \{g : fg = 0, g \neq 0\}$ and $A(f \oplus 1) = \{g : (f \oplus 1)g = 0, g \neq 0\}$. It was shown that an optimal resistance of a Boolean function f against AA is achieved if $AI_f = \lceil n/2 \rceil$. On the other hand, a Boolean function with an optimal AI still cannot adequately ensure a good resistance against FAA that use the existence of the function pairs (g, h) (with algebraic degree $\deg(g)$ and $\deg(h)$ respectively) such that $fg = h$ and $\deg(g) + \deg(f)$ is not large. The value of $\deg(g) + \deg(h)$ measures the resistance of a Boolean function against FAA. An optimal resistance of Boolean functions (used in LFSR-based stream ciphers) against FAA implies that the minimum values of $\deg(g) + \deg(h)$ is always equal to n for any function pairs (g, h) such that $fg = h$, though such functions are very rare. In addition, it was shown that for balanced Boolean functions $\deg(g) + \deg(h) \geq n$ if and only if either $n = 2^k$ or $n = 2^k + 1$ for some positive integer k [13].

During the past decade, an efficient evaluation of the resistance of nonlinear Boolean functions against AA and FAA has been addressed in many works due to a great significance of these estimates from both the design and cryptanalysis point of view. At EUROCRYPT 2003, the first algorithm for determining the existence of annihilators of degree d of a Boolean function with n variables was proposed in [4]. Its time complexity is about $O(D^3)$ operations, where $D = \sum_{i=0}^{d} \binom{n}{i}$. At FSE 2006, an algorithm for checking the existence of annihilators or multiples of degree less than or equal to d was introduced in [7] with time complexity of about $O(n^d)$ operations for an n-variable Boolean function. At EUROCRYPT 2006, based on the multivariate polynomial interpolation, Armknecht *et al.* [1] proposed an algorithm for computing $AI = d$ of a Boolean function with n variables [1] requiring $O(D^2)$ operations, where $D = \sum_{i=0}^{d} \binom{n}{i}$. Moreover, an algorithm for determining the immunity against FAA was also presented running in time complexity of about $O(D^2E)$ operations for an n-variable Boolean function, where $E = \sum_{i=0}^{e} \binom{n}{i}$ and d is generally much smaller than e, $(\deg(g), \deg(h)) = (d, e)$. At ACISP 2006, an algorithm to evaluate the resistance of Boolean functions against FAA was developed in [2], whose time complexity is about $O(DE^2 + D^2)$ operations for an n-variable Boolean function. At INDOCRYPT 2006, based on the Wiedemann's algorithm, Didier proposed a new algorithms to evaluate the resistance of an n-variable Boolean functions against AA and FAA in [8] with time complexity of about $O(n^{2n}D)$ operations and a memory complexity of about $O(n^{2n})$. Finally, Jiao *et al.* [14] revised the algorithm of [1] to compute the resistance against AA and FAA, reducing the complexity to $O(D^{2\pm\varepsilon})$ operations, where $\varepsilon \approx 0.5$ and D is the same as above.

Despite the development of the above mentioned algorithms, the exact evaluation of the algebraic properties of a Boolean function remains infeasible for relatively large input variables n (for instance $n \geq 30$). For instance, in order to estimate exactly the resistance of a random Boolean function with 30 variables against AA and FAA, the best known algorithm of [14] still requires $\binom{n}{d}^{2.5} = \binom{30}{15}^{2.5} \approx 2^{68}$ operations, for $n = 30$ and $d = 15$. It appears to be a rather difficult task to efficiently estimate the resistance of Boolean function (with relative large input variables n) against AA and FAA. The purpose of this paper is to present an efficient probabilistic algorithm for determining the resistance of a random Boolean function against AA and FAA. A suitable choice of input parameters gives a high success rate of the algorithm so that the estimates are correct with probability very close to one. The algorithm employs partial linear relations, derived form the decomposition of an arbitrary nonlinear Boolean function into many small partial linear subfunctions by using the disjoint sets of input variables. A general probabilistic decomposition algorithm for nonlinear Boolean functions is given along with the sufficient conditions regarding the existence of low degree annihilators (or multipliers). This probabilistic algorithm provides a new framework for estimating the resistance of Boolean function against AA and FAA requiring only about $O(n^2 2^n)$ operations (for an n-variable Boolean function), thus offering much less complexity at the price of being probabilistic. The lower and upper bound on AI and FAA that we derive appears to be very tight for randomly selected Boolean functions thus giving a close estimate of the algebraic properties for large n where due to computational complexity the deterministic algorithms cannot be applied. Several examples are provided justifying the tightness of our bounds when compared to the actual algebraic properties of a given function for relatively small values of n for which the deterministic algorithms could be applied.

The rest of the paper is organized as follows. In Section 2, some basic definitions and notations are recalled. In Section 3, a new concept of partial linear relations decomposition is introduced, and then a general dissection algorithm for nonlinear Boolean functions is proposed. An efficient algorithm for determining the resistance of Boolean functions (with relatively large input variables n) against AA and FAA is descried in Section 4. Finally, some concluding remarks are given in Section 5.

2 Preliminaries

In this section, some basic definitions and notations related to Boolean functions are given. Let $GF(2)$ denote the binary Galois field and $GF(2)^n$ an n-dimensional vector space spanned over $GF(2)$. The operation "$\oplus$" will denote the addition over $GF(2)$. Throughout this article $|\cdot|$ will denote the absolute value of an integer and the cardinality of a set B will be denoted as $||B||$.

Definition 1 *A Boolean function is a mapping $f : GF(2)^n \longrightarrow GF(2)$. The set of all Boolean functions $f(x_1, \ldots, x_n)$ is denoted by $\mathbb{B}_n$.*

Definition 2 *The algebraic normal form (**ANF**) of an n-variable Boolean function is the multivariate polynomial expression*

$$f(x_1, \ldots, x_n) = \sum_{c \in GF(2)^n} \tau_c(\prod_{i=1}^{n} x_i^{c_i}), \tag{1}$$

where $c = (c_1, \ldots, c_n) \in GF(2)^n, \tau_c, x_i \in GF(2), (i = 1, \ldots, m)$. Moreover, the algebraic degree of f, denoted by $deg(f)$, is the maximal value of the Hamming weight of c satisfying the condition $\tau_c \neq 0$. If $\tau_c \neq 0$ for all $c = (c_1, \ldots, c_n) \in GF(2)^n$, the Boolean function $f \in \mathbb{B}_n$ is called the all term function. If $deg(f) \leq 1$, a Boolean function $f \in \mathbb{B}_n$ is called an affine function. Especially, for an affine Boolean function, if its constant term is zero, then the function is linear.

Definition 3 *Let $f \in \mathbb{B}_n$ be a nonlinear Boolean function, and $X = (x_1, \ldots, x_n) \in GF(2)^n$, $X_i' = (x_{j_1}, \ldots, x_{j_i}) \in GF(2)^i$, $X_{n-i}'' = (x_{j_{i+1}}, \ldots, x_{j_n}) \in GF(2)^{n-i}$, where $\{j_1, \ldots, j_i\} \subset \{1, \ldots, n\}$, $\{j_{i+1}, \ldots, j_n\} \subset \{1, \ldots, n\}$ and $\{j_1, \ldots, j_i\} \cap \{j_{i+1}, \ldots, j_n\} = \emptyset$. If by fixing $X_i' = a$, the function $f(a, X_{n-i}'') = f_{X_i'=a}(X_{n-i}'')$ is an $(n-i)$-variable linear subfunction or a constant function, then $f_{X_i'=a}(X_{n-i}'')$ is called a partial linear relation with respect to $a \in GF(2)^i$. The set of all partial linear relations with $n-i$ variables is denoted by $\mathbb{L}_{n-i}$.*

3 A probabilistic decomposition algorithm for nonlinear Boolean functions

In this section, a probabilistic decomposition algorithm for nonlinear Boolean functions which decomposes any Boolean functions into a set of partial linear relations is discussed. This decomposition is generic, deterministic and valid for an arbitrary Boolean functions (fully specifying a given function) but it is not unique. The consequence is that different choices of such a decomposition may yield different estimates of algebraic properties, though since the number of these decompositions is not large the algorithm may exhaustively check for the best decomposition. For brevity, in the result below we use notation introduced in Definition 3.

Theorem 1 *Let $B_i \subseteq GF(2)^i$ and $B'_i = B_i \times GF(2)^{n-i}$ such that $\bigcup_{i=1}^{n-1} B'_i = GF(2)^n$ and $B'_{i_1} \cap B'_{i_2} = \emptyset$, $(1 \leq i \leq n-1,\ 1 \leq i_1 < i_2 \leq n-1)$. Let $X = (x_1, \ldots, x_n) \in GF(2)^n$, and denote by $D_i = \{L(X''_{n-i}) \mid L(X''_{n-i}) = c \cdot X''_{n-i} \oplus b, c \in GF(2)^{n-i}, b \in GF(2)\}$. Then any nonlinear Boolean function $f \in \mathbb{B}_n$ can be decomposed and represented as below:*

$$f(X) = f(X'_i, X''_{n-i}) = \sum_{i=1}^{n-1} \sum_{\sigma=(\sigma_{j_1},\ldots\sigma_{j_i}) \in B_i} \left(\prod_{s=j_1}^{j_i} (x_s \oplus \sigma_s \oplus 1) \right) \cdot \varphi_{i,[\sigma]}(X''_{n-i}), X'_i \in B_i, \quad (2)$$

where $\varphi_{i,[\sigma]}$ are injective mappings from B_i to D_i, and $||B_i|| \neq 0$.

Proof. For any nonlinear Boolean function $f \in \mathbb{B}_n$, for a given $X'_i = (x_{j_1}, \ldots x_{j_i}) = a \in GF(2)^i$, the restriction $f(a, X''_{n-i}) = f_{X'_i=a}(X''_{n-i})$ is either a partial linear relation or a nonlinear function. Let $B_i = \{X'_i \mid f_{X'_i=a}(X''_{n-i}) \in \mathbb{L}_{n-i}, X'_i = a \in GF(2)^i\}$. Moreover, if $X'_i \in GF(2)^i \setminus B_i$, let $X'_{i+1} = (X'_i, x_{j_{i+1}})$, then either $X'_{i+1} \in B_{i+1}$ or not. If $X'_{i+1} \in GF(2)^{i+1} \setminus B_{i+1}$, then we can increase the size of X'_{i+1} to X'_{i+2}. Iteratively, we reach the case $i = n-1$ for which $X'_{n-1} \in B_{n-1}$ always holds. Consequently, we can obtain $B_i \subseteq GF(2)^i$ and $B'_i = B_i \times GF(2)^{n-i}$ such that $\bigcup_{i=1}^{n-1} B'_i = GF(2)^n$ and $B'_{i_1} \cap B'_{i_2} = \emptyset$, $1 \leq i_1 < i_2 \leq n-1$, where $1 \leq i \leq n-1$. Moreover, for $D_i = \{f_{X'_i=a}(X''_{n-i}) \mid a \in B_i\}$ we easily find injective mappings $\varphi_{i,[\sigma]}$, which are mappings from B_i to D_i, and $||B_i|| \neq 0$. □

The following corollary is an easy consequence of the above result.

Corollary 1 *Using the notation of Theorem 1 the sets $B_i, (i = 1, \ldots, n-1)$ satisfies the relations below.*

1. $\sum_{i=1}^{n-1} ||B_i|| \times 2^{n-i} = 2^n$, $||B_i|| \leq 2^i$.
2. $||B_i|| \leq ||B_j||$ *for non-empty sets B_i and B_j employed in decomposition (2), $(i < j)$.*
3. *If $f(x_1, \ldots, x_n)$ is an affine function, then $||B_1|| = 2$ and $||B_i|| = 0, (i = 2, \ldots, n-1)$.*
4. *If $f(x_1, \ldots, x_n) = x_1 \cdot x_2 \ldots x_n$, then $||B_{n-1}|| = 2$ and $||B_i|| = 1, (i = 1, \ldots, n-2)$.*
5. *If $f(x_1, \ldots, x_n)$ is a full term function, then $||B_{n-1}|| = 2^{n-1}$ and $||B_i|| = 0, (i = 1, \ldots, n-2)$.*

Example 1 *Let $f(x_1, \ldots, x_4) = x_1 \oplus x_4 \oplus x_1x_2 \oplus x_1x_2x_3$. To write the function f in the form (2), we need to fix particular coordinates, so that the restrictions of the function f are linear or constant. For instance, by fixing $x_1 = 0$, or $x_2 = 0$, or $x_2 = 1$ and $x_3 = 0$, or $x_2 = 1$ and $x_3 = 1$, $f(x_1, \ldots, x_4)$ will be decomposed into linear functions. More precisely (neglecting the other cases):*

1. *If $x_2 = 0$, then $f(x_1, 0, x_3, x_4) = x_1 \oplus x_4$. The corresponding set of fixed coordinates (a single coordinate in this case) is $B_1^{(x_2)} = \{0\}$, and the corresponding set of linear functions is $D_1 = \{x_1 \oplus x_4\}$.*

2. If $(x_2, x_3) = (1,0)$, then $f(x_1, 1, 0, x_4) = x_4$, and if $(x_2, x_3) = (1,1)$ then $f(x_1, 1, 1, x_4) = x_1 \oplus x_4$. The corresponding set of fixed coordinates (2-tuples) is $B_2^{(x_2,x_3)} = \{(1,0), (1,1)\}$. The corresponding set of linear relations is $D_2 = \{x_4, x_1 \oplus x_4\}$.

We have that $B_3 = \emptyset$. Clearly,

$$||B_1|| \times 2^3 + ||B_2|| \times 2^2 + ||B_3|| \times 0 = 1 \times 2^3 + 2 \times 2^2 + 0 = 2^4,$$

which means that the union of subsets (subspaces) of $GF(2)^4$ with fixed coordinates $x_2 = 0$, $(x_2, x_3) = (1,0)$ and $(x_2, x_3) = (1,1)$ actually give the whole space $GF(2)^4$, i.e.,

$$\{(x_1, 0, x_3, x_4) \mid x_i \in GF(2),\ i = 1, 3, 4.\} \cup \{(x_1, 1, 0, x_4) \mid x_i \in GF(2),\ i = 1, 4.\}$$

$$\cup \{(x_1, 1, 1, x_4) \mid x_1, x_4 \in GF(2)\} = GF(2)^4.$$

Let

$$\varphi_{1,[\sigma=(x_2)=(0)]}(B_1) = x_1 \oplus x_4 \in D_1,$$

$$\varphi_{2,[\sigma=(x_2,x_3)=(1,0)]}(B_2) = x_4 \in D_2,$$

$$\varphi_{2,[\sigma=(x_2,x_3)=(1,1)]}(B_2) = x_1 \oplus x_4 \in D_2.$$

Then the function f can be written as:

$$\begin{aligned} f(x_1, x_2, x_3, x_4) &= \sum_{\sigma=(x_2)\in B_1} \left(\prod_{s=2}^{2} (x_s \oplus \sigma_s \oplus 1) \right) \cdot \varphi_{1,[\sigma]}(B_1) \\ &\oplus \sum_{\sigma=(x_2,x_3)\in B_2} \left(\prod_{s=2}^{3} (x_s \oplus \sigma_s \oplus 1) \right) \cdot \varphi_{2,[\sigma]}(B_2) \\ &= (x_2 \oplus 1)(x_1 \oplus x_4) \oplus x_2(x_3 \oplus 1)x_4 \oplus x_2 x_3 (x_1 \oplus x_4). \end{aligned}$$

The above result immediately leads to the following algorithm which decomposes an arbitrary Boolean function into a set of partial linear relations. We notice that the output of the algorithm heavily depends on the given choice (order) of variables which are to be fixed during its execution, see also Remark 1. In other words, the decomposition into linear subfunctions with respect to the cardinalities of B_i is quite likely not optimal and therefore a more refined search for the best decomposition (out of $n!$ possible ones) is later proposed, namely Algorithm 2.

Algorithm 1 (Partial Linear Relations Decomposition)

Step 1 For a given n-variable Boolean function $f \in \mathbb{B}_n$, let $k = \lceil \log_2 n \rceil$. Set counters $T_{B_i} = 0, (i = 1, \ldots, n-1)$. Without loss of generality, we assume the fixed decomposition order of $(n-1)$ variables to be $(x_1 \to x_2 \to \ldots \to x_{n-1})$.

Step 2 For each $x_1 = a_1 \in GF(2)$, randomly choose different 2^k pairs $(\alpha_{n-1}^j, \beta_{n-1}^j)$,

$\alpha_{n-1}^j, \beta_{n-1}^j \in GF(2)^{n-1}$, for $j = 1, \ldots, 2^k$. Let $g_1(x_2, \ldots, x_n) = f_{x_1=a_1}(x_2, \ldots, x_n)$. For each pair $(\alpha_{n-1}^j, \beta_{n-1}^j)$, perform the linear relation test below :

$$g_1(\alpha_{n-1}^j \oplus \beta_{n-1}^j) = g_1(\alpha_{n-1}^j) \oplus g_1(\beta_{n-1}^j) \oplus g_1(0, \ldots, 0).$$

(2.1) If all 2^k pairs $(\alpha_{n-1}^j, \beta_{n-1}^j)$ pass this linear test (thus satisfy the above equality), then let $T_{B_1} = T_{B_1} + 1$.

(2.2) Otherwise, for each $(x_1, x_2) = a_2 \in GF(2)^2$, randomly choose different 2^k pairs $(\alpha_{n-2}^j, \beta_{n-2}^j)$, $\alpha_{n-2}^j, \beta_{n-2}^j \in GF(2)^{n-2}$, for $j = 1, \ldots, 2^k$. Let $g_2(x_3, \ldots, x_n) = f_{(x_1,x_2)=a_2}(x_3, \ldots, x_n)$ and again for each pair $(\alpha_{n-2}^j, \beta_{n-2}^j)$ perform 2^k linear relation tests:

$$g_2(\alpha_{n-2}^j \oplus \beta_{n-2}^j) = g_2(\alpha_{n-2}^i) \oplus g_2(\beta_{n-2}^i) \oplus g_2(0, \ldots, 0), \quad j = 1, \ldots, 2^k.$$

(2.2.1) If all 2^k pairs $(\alpha_{n-2}^j, \beta_{n-2}^j)$ pass the linear relation test, then let $T_{B_2} = T_{B_2} + 1$.

Otherwise, repeat the above steps by increasing the size of input variables, thus increase $i \to i+1$ and use $(x_1, \ldots, x_{i+1}) = a_{i+1} \in GF(2)^{i+1}$, for $i \leq n-k$. For any such a_{i+1} perform 2^k linear tests for randomly chosen pairs $(\alpha_{n-i-1}^j, \beta_{n-i-1}^j) \in GF(2)^{n-i-1} \times GF(2)^{n-i-1}$, and update the values $T_{B_{i+1}}$.

For $i = n-k+1, \ldots, n-2$, randomly choose different 2^{n-i} pairs $(\alpha_{n-i}^j, \beta_{n-i}^j) \in GF(2)^{n-i} \times GF(2)^{n-i}$, for $j = 1, \ldots, 2^{n-i}$, and check whether all 2^{n-i} pairs can pass the linear relation test or not using $g_i = f_{(x_1,\ldots,x_i)=a_i}(x_{i+1}, \ldots, x_n)$.

Step 3 Return the values of $||B_i|| = T_{B_i}$, for $i = 1, \ldots, n-1$.

To estimate the success rate of this algorithm, we notice that each $g_i = f_{(x_1,\ldots,x_i)=a_i}(x_{i+1}, \ldots, x_n)$ (for different $a_i \in GF(2)^i$) can pass all 2^k linear relation tests only with a probability $\frac{1}{2^{2^k}}$ using 2^k random pairs, for $i = 1, \ldots, n-k$. However, there are $\sum_{i=1}^{n-k} ||B_i||$ subfunctions which need to be checked. It also means that there are about

$$\sum_{i=1}^{n-k} ||B_i|| \times 2^{-2^k} \leq 2^{n-1} \times 2^{-n} = \frac{1}{2} < 1$$

nonlinear subfunction g_i that could pass the linear relation tests.

Moreover, when $i \in [n-k+1, n-2]$, then each $g_i = f_{(x_1,\ldots,x_i)=a_i}(x_{i+1}, \ldots, x_n)$ only has 2^{n-i} input values in total. In this case, if g_i is a nonlinear function, the probability of passing the linear relation tests is only $\frac{1}{2^{2^{n-i}}}$, using 2^{n-i} random pairs. For instance, if $n-i = 3$, the probability is only about $\frac{1}{2^8} \approx 0.0039$. But for $n-i = 2$, to further improve the accuracy of the linear relation tests in practice, we can slightly increase the numbers of testing pairs to 6. In fact, if $n-i = 2$, there are $2^2 = 4$ different input values for each g_{n-2} in total, which gives $\binom{4}{2} = 6$ different pairs, i.e., $\{(11,00), (11,01), (11,10), (00,01), (00,10), (01,10)\}$. Obviously the probability of passing the six linear relation tests is 0, for any 2-variable nonlinear Boolean function g_{n-2}. Therefore, the success rate of this algorithm is about $p = 1$.

On the other hand, the time complexity of this algorithm is dominated by Step 2, i.e.,

$$T_{complexity} = \sum_{i=1}^{n-k} 2^i \times 2^k + \sum_{j=n-k+1}^{n-2} 2^j \times 2^{n-j}.$$

Moreover, we have

$$\begin{aligned} T_{complexity} \quad &= 2^k \sum_{i=1}^{n-k} 2^i + \sum_{j=n-k+1}^{n-2} 2^j \times 2^{n-j} \\ &= 2^{n+1} - 2^{k+1} + 2^n \times (n-2-(n-k+1)+1) \\ &= k \times 2^n - 2^{k+1} \\ &< k \times 2^n. \end{aligned}$$

where $k = \log_2 n$. Therefore, the time complexity of this algorithm is about $(\log_2 n) \times 2^n$ operations. The memory complexity is only about $O(2n)$ n-bit, which is mainly used to save the parameters T_{B_i} and $X \in GF(2)^n$.

Remark 1 *In Step 1, for different orders of $(n-1)$-variable, this algorithm will return different values of $||B_i||$, for $i = 1, \ldots, n-1$. It is clear that there are $\binom{n}{n-1} \times (n-1)! = n!$ ordered choices for a given n-variable function f. Therefore, there are $n!$ different values for $||B_i||$. However, it is computationally infeasible to calculate all these values if n is relatively large. We also notice that the approach taken in [7], which employs small subfunctions of f, allows that subfunctions are also nonlinear. Our algorithm, due to strict linear decomposition, does not allow the use of nonlinear subfunctions.*

Algorithm 1 essentially provides an upper bound (for a fixed decomposition order) on the algebraic degree of annihilators of f due to the following result.

Theorem 2 *With the same notation used in Theorem 1, if Boolean function $f \in \mathbb{B}_n$ can be decomposed (with $B_i, i = (1, \ldots, n-1)$) by using Algorithm 1, then there is at least an annihilator $g \in \mathbb{B}_n$ with $deg(g) \leq \lambda + 1$ such that $f \cdot g = 0$, where $\lambda = \min\{i \mid ||B_i|| \neq 0, i = (1, ..., n-1)\}$.*

Proof. Let $D^*_\lambda = \{L(X''_{n-\lambda}) \oplus 1, L(X''_{n-\lambda}) \in D_\lambda\}$, and $D^*_i = \{0\}, (i \neq \lambda)$. Moreover, let

$$g(X) = g(X'_i, X''_{n-i}) = \sum_{i=1}^{n-1} \sum_{\sigma=(\sigma_{j_1}, \ldots \sigma_{j_i}) \in B_i} \left(\prod_{s=j_1}^{j_i} (x_s \oplus \sigma_s \oplus 1) \right) \cdot \phi_{i,[\sigma]}(X''_{n-i}), X'_i \in B_i, \quad (3)$$

where $\phi_{i,[\sigma]}$ are injective mappings from B_i to D^*_i, and $||B_i|| \neq 0$. Note that $\varphi_{i,[\sigma]}(X''_{n-i}) \cdot \phi_{i,[\sigma]}(X''_{n-i}) = 0$ for all $(X'_i, X''_{n-i}) \in GF(2)^n$. It is easily verified that $f \cdot g = 0$ and $\deg(g) \leq \lambda + 1$, where $\lambda = \min\{i \mid ||B_i|| \neq 0, i = (1, \ldots, n-1)\}$. ∎

Example 2 *Consider an $n = 8$ variable Boolean function $f(x)$ whose truth table is given below. Using the existing algorithm in [1], we can easily calculate the exact AI value of this function, getting $AI = 2$. On the other hand, using our algorithm we find a decomposition for this function, where $||B_2|| = 1, ||B_4|| = 4, ||B_6|| = 32, ||B_i|| = 0, i \neq (2, 4, 6)$. Using Theorem 2, to estimate the theoretical upper bound on AI value, we found $AI \leq 3, (\lambda + 1 = 2 + 1 = 3)$, which is consistent to the exact value $AI = 2$.*

0001000100010001000100010100010000010001000100010001000101000100000
1000100010001000100010100010000010001000100010001000100111011100010 0
0100010100000100010100000100010001000101000001000101000001000100010
0010100000100010100000100100010001001110010001001110010

Note that the number of elements in the set of affine subfunctions on $(n-i)$-variable is $||B_i|| \times 2^{n-i}$ (for those a_i for which g_i passes the linearity test) over $GF(2)^n$, for $i = 1, \ldots, n-1$. It is clear that $||B_i|| \times 2^{n-i}$ will be relatively large if i is relatively s-mall and $||B_i|| \neq 0$. To estimate the maximal size of $||B_i|| \neq 0$ for small i, we propose an optimized algorithm below. In difference to Algorithm 1, where a fixed decomposition of $n-1$ variables gives unique (fixed) sets B_i, Algorithm 2 selects the best decomposition in terms of maximal cardinality of B_i. It implies that in each step we select a decomposition which for a fixed choice of the positions of input variables gives maximal number of affine subfunctions.

Algorithm 2 (Optimized Partial Linear Decomposition)

Step 1 For a given n-variable Boolean function $f \in \mathbb{B}_n$, let $k = \lceil \log_2 n \rceil$. Set counters $T^j_{B_i} = 0$, and tables C^j_t, where $i = 1, \ldots, n-1$ and $j = 1, \ldots, n$.

Step 2 For each $x_j = a_1 \in GF(2)$, $j = 1, \ldots, n$, randomly choose different 2^k pairs $(\alpha^\ell_{n-1}, \beta^\ell_{n-1})$, $\alpha^\ell_{n-1}, \beta^\ell_{n-1} \in GF(2)^{n-1}$. Let $g = f_{x_j = a_1}(x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_n)$ and for each pair $(\alpha^\ell_{n-1}, \beta^\ell_{n-1})$, $\ell = 1, \ldots, 2^k$, perform the linear relation test below:

$$g_1(\alpha^\ell_{n-1} \oplus \beta^\ell_{n-1}) = g_1(\alpha^\ell_{n-1}) \oplus g_1(\beta^\ell_{n-1}) \oplus g_1(0, \ldots, 0).$$

(1) If all 2^k pairs $(\alpha^\ell_{n-1}, \beta^\ell_{n-1})$ can pass through the linear test, then let $T^j_{B_1} = T^j_{B_1} + 1$. Otherwise, save corresponding a_1 to table C^j_1.

(2) Let $I_1 = \{j \mid \max^n_{j=1} T^j_{B_1}\}$.

Step 3 Randomly choose $j^*_1 \in I_1$, for each $j \in \{1, \ldots, n\}, (j^*_1 \neq j)$ and for each $(a_1, x_j) = a_2 \in GF(2)^2, a_1 \in C^{j^*_1}_1$, randomly choose 2^k pairs $(\alpha^\ell_{n-2}, \beta^\ell_{n-2})$, $\alpha^\ell_{n-2}, \beta^\ell_{n-2} \in GF(2)^{n-2}$. Let $g_2 = f_{(x_{j^*_1}, x_j) = a_2}$ and for each pair $(\alpha^\ell_{n-2}, \beta^\ell_{n-2})$, $\ell = 1, \ldots, 2^k$, perform the linear relation test below:

$$g_2(\alpha^\ell_{n-2} \oplus \beta^\ell_{n-2}) = g_2(\alpha^\ell_{n-2}) \oplus g_2(\beta^\ell_{n-2}) \oplus g_1(0, \ldots, 0).$$

(1) If all 2^k pairs $(\alpha^\ell_{n-2}, \beta^\ell_{n-2})$ pass the linear test, then let $T^j_{B_2} = T^j_{B_2} + 1$. Otherwise, save corresponding a_2 to table C^j_2.

(2) Let $I_2 = \{j \mid \max^n_{j=1, j \neq j^*} T^j_{B_2}\}$.

Step 4 Similarly, repeat the Step 3 above, by increasing the size of input variables, i.e., $(a_{i-1}, x_j) = a_i \in GF(2)^i$, and $(i = 3, \ldots, n-k), j \in \{1, \ldots, n\}, j \neq j_t^*, j_t^* \in I_t, t = (1, \ldots, i-1)$. In general, for $i = (n-k+1, \ldots, n-2)$, randomly choose different 2^{n-i} pairs $(\alpha^\ell_{n-i}, \beta^\ell_{n-i})$, $\alpha^\ell_{n-i}, \beta^\ell_{n-i} \in GF(2)^{n-i}$, $\ell = 1, \ldots, 2^{n-i}$, and check whether all 2^{n-i} pairs can pass the linear relation test or not, where $g_i = f_{(x_{j^*_{i-1}}, x_j) = a_i}$.

Step 5 Return the values of $||B_i|| = T_{B_i^{j_t^*}}$, for $i, t = 1, \ldots, n-1$.

Similarly to the analysis of Algorithm 1, the success rate of this algorithm is also about $p = 1$. Step 2 requires about $2 \times 2^k \times \binom{n}{1}$ operations, whereas Step 3 needs about $2^2 \times 2^k \times \binom{n-1}{1}$ operations. The time complexity of this algorithm is dominated by Step 2-4, i.e.,

$$T_{complexity} = \sum_{i=1}^{n-k} 2^i \times 2^k \times \binom{n+1-i}{1} + \sum_{j=n-k+1}^{n-2} 2^j \times 2^{n-j} \times \binom{n+1-j}{1}.$$

Moreover, we have

$$\begin{aligned} T_{complexity} \quad &= 2^k \sum_{i=1}^{n-k} 2^i \times (n+1-i) + \sum_{j=n-k+1}^{n-1} 2^j \times 2^{n-j} \times (n+1-j) \\ &< (2^n - 2^k + 2^n \times (n-1-(n-k+1)+1)) \times n \\ &< (k \times 2^n) \times n, \end{aligned}$$

where $k = \log_2 n$. Therefore, the time complexity of this algorithm is about $O(n2^n \times \log_2 n)$ operations. The memory complexity is only about $O(n2^{n-1})$ bits, which is mainly used to save the tables C_t^j, for $t = 1, \ldots, n-1, j = 1, \ldots, n$. Notice also that Theorem 1 is valid for Algorithm 2, thus an upper bound on AI can be derived using either Algorithm 1 or Algorithm 2.

Remark 2 *One may notice that both Algorithms 1 and 2 start with fixing one coordinate. In the case of highly nonlinear functions we do not expect to get affine subfunctions by fixing some small number of variables. Therefore, one may run these algorithms backwards, i.e., to start with a selection of $n-2$ or $n-3$ fixed coordinates. By fixing, say $n-2$ coordinates, it is quite likely that we get many affine subfunctions. However, further selection of fixed coordinates for sets B_i ($i < n-2$) is highly affected by certain complicated properties of these sets which are not mentioned in Corollary 1. Thus, finding an explicit non-probabilistic algorithm which provides a complete description of sets B_i, which result in a decomposition (2) of an arbitrary input function f, we leave as an open problem. Note that the existence of decomposition (2) of an arbitrary function f is guaranteed by Theorem 1.*

4 Estimation the resistance of Boolean function on n variables against AA and FAA

In this section, the resistance of Boolean functions against (fast) algebraic attack is discussed. The fact that our algorithms provide an upper bound on AI is not sufficient for efficient estimation of the algebraic properties of a given function. Indeed, in the first place we need a lower bound on AI and furthermore a tight lower and upper bound concerning the algebraic degree of deg(g)+deg(h) in the relation $fg = h$ is necessary. These bounds are derived in this section (through the set of conditions relating the main decomposition parameters) which then along with the use of Algorithm 2 gives us an efficient algorithm for estimating the algebraic properties of a given function.

4.1 Resistance to AA

Without loss of generality, we assume $X = (x_1, \ldots, x_n) \in GF(2)^n$, $X'_i = (x_1, \ldots, x_i) \in GF(2)^i$, $X''_{n-i} = (x_{i+1}, \ldots, x_n) \in GF(2)^{n-i}$, and then the equality (2) has the form below.

$$f(X'_i, X''_{n-i}) = \sum_{i=1}^{n-1} \sum_{\sigma=(\sigma_1,\ldots,\sigma_i)\in B_i} \prod_{l=1}^{i} (x_l \oplus \sigma_l \oplus 1) \cdot \varphi_{i,[\sigma]}(X''_{n-i}), \tag{4}$$

where $\varphi_{i,[\sigma]}$ are injective mappings from B_i to D_i, $X'_i \in B_i$, and $||B_i|| \neq 0$.

Note that any annihilator of f can be represented as

$$g^*(X'_i, X''_{n-i}) = \sum_{i=1}^{n-1} \sum_{\sigma=(\sigma_1,\ldots,\sigma_i)\in B_i} \prod_{l=1}^{i} (x_l \oplus \sigma_l \oplus 1) \cdot u_{i,[\sigma]}(X''_{n-i}), \tag{5}$$

where $u_{i,[\sigma]}(X''_{n-i})$ is any annihilator of $\varphi_{i,[\sigma]}(X''_{n-i})$, i.e., $u_{i,[\sigma]}(X''_{n-i}) \cdot \varphi_{i,[\sigma]}(X''_{n-i}) = 0$, $\sigma \in B_i$.

Let us restrict the degree of g^* to a fixed value $r + d \leq n/2$. If we need to cancel the terms in the ANF of g^* containing $x_{j_1} \cdots x_{j_q}$ for any q in the range $d-1 < q \leq i < n$, where d is a fixed integer and $\{j_1, \cdots, j_q\} \subset \{1, \cdots, i\}$, then the sufficient condition is that,

$$\sum_{i=1}^{n-1} \sum_{\sigma\in B_i} u_{i,[\sigma]}(X''_{n-i}) = \sum_{i=1}^{n-1} \sum_{\sigma\in B_i} (\varphi_{i,[\sigma]}(X''_{n-i}) \oplus 1) \times u'_{i,[\sigma]}(X''_{n-i}) = 0, \tag{6}$$

where each $u'_{i,[\sigma]}(X''_{n-i})$, for any $\sigma \in B_i$, is at most of degree r_i given by,

$$\begin{aligned} u'_{i,[\sigma]}(X''_{n-i}) &= a_0^\sigma \oplus a_1^\sigma x_{i+1} \oplus \cdots \oplus a_{n-i}^\sigma x_n \oplus \cdots \\ &\oplus a^\sigma_{1,\ldots,r_i} x_{i+1} \cdots x_{i+r_i} \oplus \cdots \oplus a^\sigma_{n-r_i+1,\ldots,n} x_{n-r_i+1} \cdots x_n. \end{aligned} \tag{7}$$

Note that $\deg(\varphi_{i,[\sigma]}(X''_{n-i})) \leq 1$, for any $\sigma \in B_i$.

Then we try to select the coefficients $a_\iota^\sigma \in GF(2)$ in (7) in such a way that the terms in the ANF of g^* containing $x_{j_1} \cdots x_{j_q}$ are all cancelled for any q in the range $d-1 < q \leq i < n$, where d is a fixed integer and $\{j_1, \cdots, j_q\} \subset \{1, \cdots, i\}$. This also implies that the degree of

$$\sum_{\sigma=(\sigma_1,\ldots\sigma_i)\in B_i} \prod_{l=1}^{i} (x_l \oplus \sigma_l \oplus 1) \times (\varphi_{i,[\sigma]}(X''_{n-i}) \oplus 1) \times u'_{i,[\sigma]}(X''_{n-i}),$$

is at most $r_i+d, (1 \leq i < n-1)$ since $\deg(u'_{i,[\sigma]}(X''_{n-i}))+ \deg(\varphi_{i,[\sigma]}(X''_{n-i}))+ \deg(x_{j_1}\cdots x_{j_q}) \leq r_i+1+d-1 = r_i+d$, for $\sigma \in B_i$. If we are able to obtain such a choice of the coefficients $a^{\sigma}_{\iota} \in GF(2)$ in $u'_{i,[\sigma]}(X''_{n-i})$, then the degree of g^* would be at most $\max_{i=0}^{n-1}\{r_i\}+d$ (but at least $\min_{i=0}^{n-1}\{r_i\}+d$), for all $B_i, (1 \leq i < n-1)$. Notice that when σ runs through all $B_i, (1 \leq i < n-1)$, we obtain in total $\sum_{\{1\leq i\leq n-1, ||B_i||\neq 0\}} ||B_i|| \times \sum_{j=0}^{r_i}\binom{n-i}{j}$ unknown coefficients $a^{\sigma}_{\iota} \in GF(2)$. On the other hand, cancelling all the terms in the ANF of g^* containing $x_{j_1}\cdots x_{j_q}$, will induce certain restrictions on the coefficients $a^{\sigma}_{\iota} \in GF(2)$ in the resulting system of homogeneous linear equations (involving these $a^{\sigma}_{\iota} \in GF(2)$) whose total number is given by,

$$\sum_{\{1\leq i\leq n-1, ||B_i||\neq 0\}} \sum_{l=0}^{i-d}\binom{i}{i-l}\sum_{j=0}^{r_i+1}\binom{n-i}{j}, \tag{8}$$

where $i-d \geq 0, i-l \geq 0$. The binomial sum term $\sum_{l=0}^{i-d}\binom{i}{i-l}$ in the above equation refers to counting all the terms that contain $x_{j_1}\cdots x_{j_q}$ in $\sum_{\sigma\in B_i}\Pi_{l=0}^{i}(x_i \oplus \sigma_i \oplus 1)$ section (where $\{j_1,\cdots,j_q\} \subset \{1,\cdots,i\}$), whose degree q is in the range d to i. The binomial sum term $\sum_{j=0}^{r_i+1}\binom{n-i}{j}$ counts all the possible terms that are involved in the $\varphi_{i,[\sigma]}(X''_{n-i}) \times u'_{i,[\sigma]}(X''_{n-i})$ portion. Moreover, the summation (i.e., $\sum_{\{1\leq i\leq n-1, ||B_i||\neq 0\}}$) in the above equation takes into account all homogeneous linear equations for all $||B_i|| \neq 0, (1 \leq i < n-1)$.

It is obvious that there will be solutions to this homogeneous system of equations if the number of equations is less or equal than the number of unknowns.

Thus, if the condition below is satisfied, then we will obtain at least one Boolean function g^* of degree $r'+d \leq \deg(g^*) \leq r+d$ (by solving the system for unknown $a^{\sigma}_{\iota} \in GF(2)$) with $r'+d \leq \deg(g^*) \leq r+d$, where $r = \max_{i=0}^{n-1}\{r_i\}$ and $r' = \min_{i=0}^{n-1}\{r_i\}$. This gives us both a lower and upper bound on the value of AI and these bounds appear to be tight for randomly selected Boolean functions.

Condition 0:

$$\begin{aligned}\sum_{\{1\leq i\leq n-1, ||B_i||\neq 0\}} ||B_i|| \times \sum_{j=0}^{r_i}\binom{n-i}{j} \geq \\ \sum_{\{1\leq i\leq n-1, ||B_i||\neq 0\}} (\sum_{l=0}^{i-d}\binom{i}{i-l} \times \sum_{j=0}^{r_i+1}\binom{n-i}{j})\end{aligned} \tag{9}$$

It seems to be difficult to obtain a concise expression for the optimal choice of the parameters r', r and d.

Remark 3 *From equality (9), we know that the values of $||B_i||, (i = 1,\ldots,n-1)$ have a positive impact on the AI value. In particular, larger $||B_i||$ and smaller i usually implies smaller AI.*

4.2 Resistance to FAA

Our main objective now is to confirm the existence of a low degree Boolean function g' such that the function fg' also has a low degree, though more precisely our goal is to minimize

$\deg(g') + \deg(fg')$, where f has the form given by (4). Let us restrict the degree of g' with a fixed value $r + s$, by considering

$$g'(X'_i, X''_{n-i}) = \sum_{i=1}^{n-1}\{ \sum_{\sigma=(\sigma_1,\ldots\sigma_i)\in B_i} \prod_{l=1}^{i}(x_l \oplus \sigma_l \oplus 1) \cdot \xi_{i,[\sigma]}(X''_{n-i})\}, \tag{10}$$

where each $\xi_{i,[\sigma]}(X''_{n-i})$, for any $\sigma \in B_i$, is a degree r_i function given by,

$$\begin{aligned} \xi_{i,[\sigma]}(X''_{n-i}) &= b_0^\sigma \oplus b_1^\sigma x_{i+1} \oplus \cdots \oplus b_{n-i}^\sigma x_n \oplus \cdots \\ &\oplus b_{1,\ldots,r_i}^\sigma x_{i+1}\cdots x_{i+r_i} \oplus \cdots \oplus b_{n-r_i+1,\ldots,n}^\sigma x_{n-r_i+1}\cdots x_n. \end{aligned} \tag{11}$$

There are two basic conditions that need to be satisfied so that both g' and fg' are of low degree.

(1) Firstly, we need to specify g' to be of low algebraic degree. We try to select the coefficients $b_\iota^\sigma \in GF(2)$ in (11) in such a way that the terms in the ANF of g' containing $x_{j_1}\cdots x_{j_q}$ are all cancelled for any q in the range $s < q \le i < n$, where s is a fixed integer and $\{j_1, \cdots, j_q\} \subset \{1, \cdots, i\}$. This also implies that the degree of

$$\sum_{\sigma=(\sigma_1,\ldots\sigma_i)\in B_i} \prod_{l=1}^{i}(x_l \oplus \sigma_l \oplus 1) \cdot \xi_{i,[\sigma]}(X''_{n-i})$$

would be at most $r_i + s$ for each B_i and $1 \le i < n-1$. If we are able to obtain such a choice of the coefficients $b_\iota^\sigma \in GF(2)$ in $\xi_{i,[\sigma]}(X''_{n-i})$, then the degree of g' would be at most $\max_{i=0}^{n-1}\{r_i\} + s$ (but at least $\min_{i=0}^{n-1}\{r_i\} + s$), for all B_i. Notice that when σ runs through all B_i, $1 \le i < n-1$, we obtain in total $\sum_{\{1\le i\le n-1, ||B_i||\neq 0\}} ||B_i|| \times \sum_{j=0}^{r_i}\binom{n-i}{j}$ unknown coefficients $b_\iota^\sigma \in GF(2)$. On the other hand, to cancel all the terms in the ANF of g' containing $x_{j_1}\cdots x_{j_q}$, will induce certain restrictions on the coefficients $b_\iota^\sigma \in GF(2)$ in the resulting system of homogeneous linear equations (involving these $b_\iota^\sigma \in GF(2)$) whose total number is given by,

$$\sum_{\{1\le i\le n-1, ||B_i||\neq 0\}} (\sum_{l=0}^{i-s-1}\binom{i}{i-l}\sum_{j=0}^{r_i}\binom{n-i}{j}), \tag{12}$$

where $i - s - 1 \ge 0, i - l \ge 0$. The binomial sum term $\sum_{l=0}^{i-s-1}\binom{i}{i-l}$ in the above equation refers to counting all the terms containing $x_{j_1}\cdots x_{j_q}$ in $\sum_{\sigma\in B_i}\Pi_{l=0}^{i}(x_i \oplus \sigma_i \oplus 1)$ section (where $\{j_1, \cdots, j_q\} \subset \{1, \cdots, i\}$), whose degree q is in the range $s+1$ to i. The binomial sum term $\sum_{j=0}^{r_i}\binom{n-i}{j}$ counts all the possible terms that are involved in the $\xi_{i,[\sigma]}(X''_{n-i})$ portion. Moreover, the summation (i.e., $\sum_{\{1\le i\le n-1, ||B_i||\neq 0\}}$) in the above equation takes into account all homogeneous linear equations for all $B_i \neq \emptyset, (1 \le i < n-1)$.

Thus, if the
Condition 1:

$$\sum_{\{1\le i\le n-1, ||B_i||\neq 0\}} ||B_i|| \times \sum_{j=0}^{r_i}\binom{n-i}{j} \ge \sum_{\{1\le i\le n-1, ||B_i||\neq 0\}} \Big(\sum_{l=0}^{i-s-1}\binom{i}{i-l}\sum_{j=0}^{r_i}\binom{n-i}{j}\Big),$$

i.e.,

$$\sum_{\{1\le i\le n-1,||B_i||\neq 0\}} ||B_i|| \geq \sum_{\{1\le i\le n-1,||B_i||\neq 0\}} \sum_{l=0}^{i-s-1} \binom{i}{i-l} \tag{13}$$

is satisfied, then we will obtain at least one Boolean function g' (by solving the system for unknown $b_\iota^\sigma \in GF(2)$) with $r'+s \le$deg$(g') \le r+s$, where $r = \max_{i=0}^{n-1}\{r_i\}$ and $r' = \min_{i=0}^{n-1}\{r_i\}$.

(2) Secondly, we note that

$$f(X_i', X_{n-i}'') \times g'(X_i', X_{n-i}'') = \sum_{i=1}^{n-1} \sum_{\sigma=(\sigma_1,\ldots\sigma_i)\in B_i} \prod_{l=1}^{i}(x_l \oplus \sigma_l \oplus 1) \times \varphi_{i,[\sigma]}(X_{n-i}'') \times \xi_{i,[\sigma]}(X_{n-i}''),$$

where each $\deg(\varphi_{i,[\sigma]}(X_{n-i}'')) \le 1$, for any $\sigma \in B_i$, $(1 \le i \le n-1)$.
It is clear that the algebraic degree of

$$\sum_{\sigma=(\sigma_1,\ldots\sigma_i)\in B_i} \prod_{l=1}^{i}(x_l \oplus \sigma_l \oplus 1) \times \varphi_{i,[\sigma]}(X_{n-i}'') \times \xi_{i,[\sigma]}(X_{n-i}'') \tag{14}$$

is at most $r_i + i + 1$, due to the fact that the degree of $\xi_{i,[\sigma]}(X_{n-i}'')$ is r_i and the degree of

$$\sum_{\sigma=(\sigma_1,\ldots\sigma_i)\in B_i} \prod_{l=1}^{i}(x_l \oplus \sigma_l \oplus 1) \times \varphi_{i,[\sigma]}(X_{n-i}'')$$

is at most $i+1$. Moreover, to restrict the degree of fg' not to be larger than e, we require that the ANF of fg' contains the terms of algebraic degree at most e. In other words, in the ANF of fg', the coefficients of the terms of algebraic degree greater than e must be equal to zero. Notice that the coefficients of these terms can be expressed as some linear equations of the unknowns $b_\iota^\tau \in GF(2), \sigma \in B_i$, $1 \le i \le n-1$, which in total induces at most Λ_1 equations, where

$$\Lambda_1 \;\; = \;\; \sum_{\{1\le i\le n-1,||B_i||\neq 0\}} (\sum_{j=e+1}^{i+r_i+1} \binom{n}{j} - \sum_{l_1=e+1}^{i+r_i+1} \binom{i}{l_1 - v_1} \sum_{v_1=r_i+2}^{n-i} \binom{n-i}{v_1}),$$

$l_1 - v_1 \geq 0$, and $l_2 - v_2 \geq 0$.

The sum(i.e., $\sum_{\{1\le i\le n-1,||B_i||\neq 0\}}$) in Λ_1 refers to counting all homogeneous linear equations for all $B_i \neq \emptyset, (1 \le i < n-1)$. The binomial sum term $\sum_{j=e+1}^{i+r_i+1}\binom{n}{j}$ in Λ_1 refers to counting all the terms in the ANF of equality (14) whose degree (denoted by j) ranges from $e+1$ to $i+r_i+1$. From this part we have to subtract those equations, corresponding to the double binomial sum (i.e., $\sum_{l_1=e+1}^{i+r_i+1}\binom{i}{l_1-v_1}\sum_{v_1=r_i+2}^{n-i}\binom{n-i}{v_1}$), that cannot appear in the ANF of equality (14). The first binomial term of the double binomial sum takes into account the number of terms of the form $x_{j_1}\cdots x_{j_{v_1}}$, where $r_i+2 \le v_1 \le n-i$ and $i+1 \le j_1 < j_2 < \cdots < j_{v_1} \le n$, since clearly $\varphi_{i,[\sigma]}(X_{n-i}'') \times \xi_{i,[\sigma]}(X_{n-i}'')$ in equality (14) is of degree at most r_i+1 in $x_{i+1},\ldots,x_n$. The second binomial term of the double binomial sum takes care of the number of terms of the form $x_{j_1^*}\cdots x_{j_{l_1-v_1}^*}$ as a constituent part of non-appearing terms of the form $x_{j_1}\cdots x_{j_{v_1}}x_{j_1^*}\cdots x_{j_{l_1-v_1}^*}$, where $l_1-v_1>0, e+1\le l_1\le r+1+i$ and $1\le j_1^*<j_2^*<\cdots<j_{l_1-v_1}^*\le i$, since clearly $\sum_{\sigma=(\sigma_1,\ldots\sigma_i)\in B_i}\prod_{l=1}^{i}(x_l\oplus\sigma_l\oplus 1)$ in equality (14) is of degree at most i in $x_1,\ldots,x_i$.

Therefore, we will obtain at least one Boolean function g' such that the degree of fg' is e if Condition 2 below is satisfied.

Condition 2:

$$\sum_{\{1\le i\le n-1,||B_i||\ne 0\}}||B_i||\times\sum_{j=0}^{r_i}\binom{n-i}{j}\ge\Lambda_1, \tag{15}$$

where

$$\Lambda_1 = \sum_{\{1\le i\le n-1,||B_i||\ne 0\}}(\sum_{j=e+1}^{i+r_i+1}\binom{n}{j}-\sum_{l_1=e+1}^{i+r_i+1}\binom{i}{l_1-v_1}\sum_{v_1=r_i+2}^{n-i}\binom{n-i}{v_1}), \tag{16}$$

$l_1-v_1\ge 0$, and $l_2-v_2\ge 0$.

4.3 Estimating the resistance of Boolean functions against AA and FAA

In this section, an algorithm for estimating the resistance of Boolean functions against both AA and FAA is introduced. It uses previously described algorithms for finding a good decomposition of a Boolean function, thus the sets B_i is found by using either Algorithm 2 or Algorithm 1.

Algorithm 3
Step 1 For a given n-variable Boolean function f, use Algorithm 2 (or Algorithm 1) to calculate the values of $||B_i||$, for $i=1,\ldots,n-1$.
Step 2 Use Conditions $0-2$ to calculate

$$\triangle_{AA}^{lower}=\min\{\lambda+1,\lceil n/2\rceil,r'+d\},\ \ \nabla_{AA}^{upper}=\min\{\lceil n/2\rceil,r+d\},$$

and

$$\triangle_{FAA}^{lower}=\min\{n-1,r'+s+e\},\nabla_{FAA}^{upper}\min\{n-1,r+s+e\}$$

for AA and FAA, where $\lambda=\min\{i\mid ||B_i||\ne 0, i=(1,\ldots,n-1)\}$.
Step 3 Repeat Step 1 and Step 2 $[\frac{n}{\log_2 n}]$ times, and return the minimum values of upper and lower bound: $(\triangle_{AA}^{lower},\nabla_{AA}^{upper})$ and $(\triangle_{FAA}^{lower},\nabla_{FAA}^{upper})$, respectively.

The time complexity of this algorithm is about $O([\frac{n}{\log_2 n}]\times\log_2 n\times n2^n)\approx O(n^2 2^n)$ operations, if Algorithm 2 is used to search for the values of $||B_i||$, $i=1,\ldots,n-1$. The memory complexity is about $O(n2^n)$ bits.

Remark 4 *Algorithm 3 only gives a theoretical upper and lower bound on both AI and $r+s+e$ for FAA. On the other hand, although Algorithm 2 proposes an approach for calculating the maximum $||B_i|| \neq 0$ for small i, an optimal decomposition for $B_i, (i = 1, \ldots, n-1)$ in Algorithm 3 is still an open problem.*

Table 1: The time complexity of our algorithm versus previous works.

The ability against AA or FAA	The time complexity	Resource
AA	$O(D^3)$	[4]
AA	$O(n^d)$	[7]
AA	$O(D^2)$	[1]
FAA	$O(D^2E)$	[1]
FAA	$O(DE^2+D^2)$	[2]
AA or FAA	$O(n^{2n}D)$	[8]
AA or FAA	$O(D^{2\pm\varepsilon})$	[14]
AA or FAA	$O(n^2 2^n)$	new

$(D = \sum_{i=0}^{d} \binom{n}{i}, E = \sum_{i=0}^{e} \binom{n}{i}, \varepsilon = 0.5)$.

Table 2: A time complexity comparison for $30 \leq n \leq 40$.

n	[4]	[7]	[2]§	[8]	[14]†	[14] ‡	new
30	$2^{81.63}$	$2^{73.60}$	$2^{54.42}$	$2^{144.42}$	$2^{68.02}$	$2^{40.81}$	$2^{22.81}$
31	$2^{84.49}$	$2^{74.31}$	$2^{56.33}$	$2^{145.37}$	$2^{70.41}$	$2^{42.24}$	$2^{22.81}$
32	$2^{87.49}$	$2^{80.00}$	$2^{58.33}$	$2^{157.16}$	$2^{72.91}$	$2^{43.74}$	$2^{24.00}$
33	$2^{90.36}$	$2^{80.71}$	$2^{60.24}$	$2^{158.12}$	$2^{75.30}$	$2^{45.18}$	$2^{24.00}$
34	$2^{93.36}$	$2^{86.49}$	$2^{62.24}$	$2^{171.09}$	$2^{77.80}$	$2^{46.68}$	$2^{25.17}$
35	$2^{96.24}$	$2^{87.20}$	$2^{64.16}$	$2^{172.05}$	$2^{80.20}$	$2^{48.12}$	$2^{25.17}$
36	$2^{99.24}$	$2^{93.06}$	$2^{66.16}$	$2^{183.20}$	$2^{82.70}$	$2^{49.62}$	$2^{26.34}$
37	$2^{102.12}$	$2^{93.77}$	$2^{68.08}$	$2^{184.16}$	$2^{85.10}$	$2^{51.06}$	$2^{26.34}$
38	$2^{105.12}$	$2^{99.71}$	$2^{70.08}$	$2^{196.46}$	$2^{87.60}$	$2^{52.56}$	$2^{27.50}$
39	$2^{108.01}$	$2^{100.42}$	$2^{72.01}$	$2^{197.42}$	$2^{90.01}$	$2^{54.01}$	$2^{27.50}$
40	$2^{111.01}$	$2^{106.44}$	$2^{74.01}$	$2^{209.88}$	$2^{92.51}$	$2^{55.51}$	$2^{28.64}$

(§ : $e = 1$, † : $D^{2+\varepsilon}$, ‡ : $D^{2-\varepsilon}$)

Table 1 describes the time complexity of previous works and of our algorithm for estimating the resistance of random n-variable Boolean functions against AA and FAA. In particular, Table 2 describes a comparison of the time complexity for $30 \leq n \leq 40$. For instance, for $n = 40$, the best previous known time complexity is about $2^{55.51}$ operations in [14]. However, the time complexity of our algorithm is only about $2^{28.64}$ operations. It is evident that our new algorithm has a more favourable time complexity than other methods though being probabilistic it may not succeed in outputting the best possible decomposition choice which may result in lose lower and upper bound.

Example 3 *Choose an $n = 12$ variable Boolean function $f(x)$. The truth table of this function in the hexadecimal format is given below. Using algorithms in [1], we could easily verify the actual resistance of this function against AA and FAA to be $AI(f) = 5$, $deg(g)+deg(h) \geq 7$, ($deg(f) = 8$) for nonzero Boolean functions g and h such as $fg = h$. On the other hand, we obtained decomposition for this function, i.e., $||B_6|| = 13, ||B_7|| = 102, ||B_i|| = 0, i \neq (6, 7)$, when using the canonical order of fixing the input variables, thus $(x_1 \to x_2 \ldots \to x_{11})$. Using Algorithm 3 to estimate the theoretical lower and upper bound on AA, we found $5 \leq AI(f) \leq 6$, $(\triangle_{AA}^{lower} = r' + d = 1 + 4 = 5, \nabla_{AA}^{upper} = r + d = 1 + 5 = 6)$, which is consistent to the real value $AI(f) = 5$. Moreover, we have found another decomposition for this function, i.e., $||B_9|| = 512, ||B_i|| = 0, i \neq 9$, if the order of fixing the input variables is $(x_1, x_2, x_3, x_{10}, x_{11}, x_{12}, x_4, x_5, x_6, x_7, x_8, x_9)$. Using Algorithm 3 to estimate the theoretical lower and upper bound on the ability against FAA, we found $6 \leq deg(g)+deg(h) \leq 7$, $(\triangle_{FAA}^{lower} = r' + s + e = 0 + 1 + 5 = 6, \nabla_{FAA}^{upper} r + s + e = 1 + 1 + 5 = 7)$, which is also consistent to the real value $deg(g)+deg(h) \geq 7$.*
6666 9999 6666 6666 6666 6666 6666 9999 6666 9999 6666 6666 9999 9999 9999 6666 6699 6699 6699 9966 6699 6699 6699 9966 9966 9966 6699 9966 6699 6699 9966 6699 33cc 33cc 33cc cc33 33cc 33cc 33cc cc33 cc33 33cc cc33 cc33 33cc cc33 33cc 33cc 0f0f f0f0 0f0f f0f0 00ff ff00 00ff ff00 0f0f 0f0f f0f0 f0f0 00ff 00ff ff00 ff00 55aa aa55 55aa 55aa 55aa aa55 55aa 55aa 55aa 55aa 55aa aa55 aa55 aa55 aa55 55aa 6996 6996 9669 6996 6996 6996 9669 6996 6996 6996 9669 6996 9669 9669 6996 9669 6969 6969 6969 9696 9696 9696 6969 9696 6969 6969 6969 9696 6969 6969 9696 6969 0000 ffff ffff 0000 55aa 55aa 5555 aaaa 6666 6666 5a5a 5a5a 3c3c 3c3c 33cc 33cc 5a5a 5a5a 5a5a a5a5 5a5a 5a5a 5a5a a5a5 5a5a 5a5a 5a5a a5a5 a5a5 a5a5 a5a5 5a5a 3cc3 c33c 3cc3 3cc3 3cc3 c33c 3cc3 3cc3 3cc3 c33c 3cc3 3cc3 c33c 3cc3 c33c c33c 0ff0 0ff0 f00f 0ff0 0ff0 0ff0 0ff0 f00f 0ff0 0ff0 f00f 0ff0 f00f f00f f00f 0ff0 5555 aaaa aaaa 5555 0f0f f0f0 f0f0 0f0f 3333 cccc cccc 3333 00ff ff00 ff00 00ff 3c3c 3c3c 3c3c c3c3 c3c3 3c3c c3c3 c3c3 3c3c 3c3c 3c3c c3c3 3c3c c3c3 3c3c 3c3c 5555 aaaa 5555 aaaa 3333 cccc 3333 cccc 5555 5555 aaaa aaaa 3333 3333 cccc cccc 5aa5 5aa5 a55a 5aa5 5aa5 5aa5 a55a 5aa5 5aa5 5aa5 5aa5 a55a a55a a55a a55a 5aa5 3333 cccc 0ff0 0ff0 3cc3 c33c 6699 9966 0f0f f0f0 6969 6969 5aa5 a55a 6996 9669

Example 4 *Choose an $n = 14$ variables Boolean function $f(x)$. The truth table of this function is given in appendix. Similarly, using algorithms in [1], we easily verified the real resistance of this function against FAA is $deg(g)+deg(h) \geq 13$, ($deg(f) = 14$) for nonzero Boolean functions g and h such as $fg = h$. On the other hand, we obtained a decomposition for this function, i.e., $||B_{11}|| = 132, ||B_{12}|| = 1761, ||B_{13}|| = 4142, ||B_i|| = 0, i \neq (11, 12, 13)$. Using Algorithm 3 to estimate the theoretical lower bound on the ability against FAA, we found $r' + s + e = 0 + 6 + 7 = 13$, and $\triangle_{FAA}^{lower} = \nabla_{FAA}^{upper} = 13$, which is also completely consistent to the actual value $deg(g)+deg(h) \geq 13$.*

Remark 5 *Some simulations for randomly chosen Boolean functions $f(x)$ with $n = 14$ variables, were also performed using Algorithm 3. We found the estimation of theoretical upper and lower bounds on AI and FAA to be consistent to the actual values. In other words, the actual values belong to a small range given by the estimated theoretical lower and upper bound (using Algorithm 3). In particular, Algorithm 3 may return an exact theoretical value, if the decomposition of these functions always occur so that λ is too close to $n - 1 = 13$ or*

$n-2=12$, where $\lambda=\min\{i \mid ||B_i|| \neq 0, i=(1,\ldots,n-1)\}$. (In this case, it usually means that a Boolean function has quite good algebraic properties). For instance, in example 4, we could easily verify that the actual resistance of this function against AA is $AI(f)=7$. Moreover, using Algorithm 3 to estimate the theoretical lower and upper bound on AA, we found $r'+d=0+8=8$, $r+d=1+8=9$ and $\triangle_{AA}^{lower}=\nabla_{AA}^{upper}=7$ which also completely consistent to the actual value $AI(f)=7$.

Example 5 *Use Algorithm 3 to check the theoretical upper bound on the resistance of functions in [19] against AA and FAA, where $||B_{\frac{n}{2}}||=2^{\frac{n}{2}-1}$, $||B_{\frac{n}{2}+1}||=2^{\frac{n}{2}-1}$, $||B_{\frac{n}{2}+2}||=2^{\frac{n}{2}}$, (for even $n=12$ to 40). In Table 3 and Table 4 we compare the upper bounds on AA and FAA, respectively, for this class of functions to their optimal values. It is clear that the resistance of functions designed in [19] against AA and FAA are not optimal or suboptimal, for even $n=18$ to 40.*

Table 3: Estimation the upper bound on the AI values of functions in [19].

n	r	d	$r+d$	Optimal
12	1	5	6	6
14	1	6	7	7
16	1	7	8	8
18	1	7	8	9
20	1	8	9	10
22	1	9	10	11
24	1	9	10	12
26	1	10	11	13
28	1	11	12	14
30	1	11	12	15
32	1	12	13	16
34	1	13	14	17
36	1	13	14	18
38	1	14	15	19
40	1	15	16	20

5 Conclusions

In this paper it is shown that an arbitrary Boolean function can be decomposed into many linear (affine) subfunctions by using the disjoint sets of input variables. Two probabilistic algorithms for finding these decompositions are presented and a theoretical framework for finding lower and upper bounds that employ these decompositions are given.

This algorithm only requires about $O(n^2 2^n)$ operations to establish tight lower and upper bounds which essentially estimate the resistance of a function to (fast) algebraic cryptanal-

Table 4: Estimation the upper bound on the resistance of functions in [19] against FAA.

n	r	s	e	$r+s+e$	Suboptimal
12	1	3	6	10	11
14	1	4	7	12	13
16	1	4	8	13	15
18	1	5	8	14	17
20	1	5	9	15	19
22	1	6	10	17	21
24	1	6	10	17	23
26	1	7	11	19	25
28	1	7	12	20	27
30	1	8	12	21	29
32	1	8	13	22	31
34	1	9	14	24	33
36	1	9	14	24	35
38	1	10	15	26	37
40	1	10	16	27	39

ysis. It remains a challenging task to optimize the search for the best decomposition which may further improve the tightness of the derived bounds.

References

[1] F. Armknecht, C. Carlet, P. Gaborit, S. Knzli, W. Meier, and O. Ruatta, "Efficient Computation of Algebraic Immunity for Algebraic and Fast Algebraic Attacks," in *Advances in Cryptology–EUROCRYPT 2006* (Lecture Notes in Computer Science), vol. 4004. Berlin, Germany: Springer, 2006, pp. 147–164.

[2] A. Braeken, J. Lano, and B. Preneel, "Evaluating the Resistance of Stream Ciphers with Linear Feedback Against Fast Algebraic Attacks," in *Information Security and Privacy* (Lecture Notes in Computer Science), vol. 4058. Berlin, Germany: Springer-Verlag, 2006, pp. 40–51.

[3] C. Carlet and K. Feng, "An Infinite Class of Balanced Functions with Optimal Algebraic Immunity, Good Immunity to Fast Algebraic Attacks and Good Nonlinearity, " in *Advances in Cryptology-ASIACRYPT 2008* (Lecture Notes in Computer Science), vol.5350. Berlin, Germany: International Association for Cryptologic Research, 2008, pp. 425–440.

[4] N. T. Courtois and W. Meier,"Algebraic Attacks on Stream Ciphers with Linear Feedback," in *Advances in Cryptology–EUROCRYPT 2003* (Lecture Notes in Computer Science), vol. 2656. Berlin, Germany: International Association for Cryptologic Research, 2003, pp. 345–359.

[5] N. T. Courtois, "Fast Algebraic Attacks on Stream Ciphers with Linear Feedback," in *Advances in Cryptology–CRYPTO 2003* (Lecture Notes in Computer Science), vol. 2729. Berlin, Germany: Springer-Verlag, 2003, pp. 176–194.

[6] D. Dalai, K. Gupta, and S. Maitra, "Notion of Algebraic Immunity and its Evaluation Related to Fast Algebraic Attacks,"in *International Workshop on Boolean Functions: Cryptography and Applications*, pp. 13–15, 2006.

[7] F. Didier, and J. Tillich, "Computing the Algebraic Immunity Efficiently," in *Fast Software Encryption* (Lecture Notes in Computer Science), vol. 4047. Berlin, Germany: Springer-Verlag, 2006, pp. 359–374.

[8] F. Didier, "Using Wiedemanns Algorithm to Compute the Immunity Against Algebraic and Fast Algebraic Attacks," in *Progress in Cryptology–INDOCRYPT 2006* (Lecture Notes in Computer Science), vol. 4329. Berlin, Germany: Springer-Verlag, 2006, pp. 236–250.

[9] X. Feng and G. Gong, "On Algebraic Immunity of Trace Inverse Functions over Finite Fields with Characteristic Two," in *Cryptology ePrint Archive*. [Online]. Available: http://www.eprint.iacr.org/2013/585.pdf

[10] S. Fischer and W. Meier, "Algebraic Immunity of S-Boxes and Augmented Functions," in *Fast Software Encryption* (Lecture Notes in Computer Science), vol. 4593. Berlin, Germany: Springer-Verlag, 2007, pp. 366–381.

[11] S. Fu, C. Li, and L. Qu, "A Recursive Construction of Highly Nonlinear Resilient Vectorial Functions," *Information Sciences*, vol. 269, pp. 388–396, 2014.

[12] F. Liu and K. Feng, "Efficient Computation of Algebraic Immunity of Symmetric Boolean Functions," in *Theory and Applications of Models of Computation* (Lecture Notes in Computer Science), vol. 2247. Berlin, Germany: Springer-Verlag, 2007, pp. 318–329.

[13] F. Liu and K. Feng, "Perfect Algebraic Immune Functions," in *Advances in Cryptology - Asiacrypt 2012* (Lecture Notes in Computer Science), vol. 7658. Berlin, Germany: Springer-Verlag, 2012, pp. 172–189.

[14] L. Jiao, B. Zhang, and M. Wang, "Revised Algorithms for Computing Algebraic Immunity against Algebraic and Fast Algebraic Attacks," *Information Security* (Lecture Notes in Computer Science), vol. 8783. International, Switzerland: Springer International, 2014, pp. 104–119.

[15] W. Meier, E. Pasalic, and C. Carlet, "Algebraic Attacks and Decomposition of Boolean Functions," in *Advances in Cryptology–EUROCRYPT 2004* (Lecture Notes in Computer Science), vol. 3027. Berlin, Germany: Springer-Verlag, 2004, pp. 474–491.

[16] S. Mesnager, "A Note on Linear Codes and Algebraic Immunity of Boolean Functions," in *21st Int. Sym. Math. Theory Net. Sys.*(Groningen, the Netherlands), pp. 923–927, 2014.

[17] Y. Nawaz, G. Gong, and K. Gupta, "Upper Bounds on Algebraic Immunity of Boolean Power Functions," in *Fast Software Encryption* (Lecture Notes in Computer Science), vol. 4047. Berlin, Germany: Springer-Verlag, 2006, pp. 375–389.

[18] F. Zhang, C. Carlet, Y. Hu, and T. Cao, "Secondary Constructions of Highly Nonlinear Boolean Functions and Disjoint Spectra Plateaued Functions," *Information Sciences*, vol. 283, pp. 94–106, 2014.

[19] W. Zhang and E. Pasalic, "Generalized Maiorana-McFarland Construction of Resilient Boolean Functions with High Nonlinearity and Good Algebraic Properties," *IEEE Trans. Inf. Theory*, vol. 60, no. 10, pp. 6681–6695, Oct. 2014.

Appendix

The truth table is described in the hexadecimal format, in particular, the most significant bit is the leftmost bit, e.g. (0001) = 1, etc.

7781 42ae f22d 8aec fd3e 8b57 802c 88c9 ce89 297b 4cce d599 bd82 922b 55fc 3a16 f30a
55f1 4eb7 a053 2e6a fe64 efc8 6ebc c48c b22b 1485 7433 3273 d922 8bae 7489 9b5f f561
56a9 3b3e c55a c06e 2065 d239 d1e3 a264 a2b6 7fe2 a678 950e 008f 0695 92ef d039 0717
1fc6 71aa b196 8995 7e4f 8ca5 e200 d4a5 b60f 63f4 eb32 2a74 4cb0 ee6c d30e 3078 a31c
c25c 5830 91f1 1ebc 8cf1 eb9d cb91 249f 4d25 917c 8572 e2bb 296e d5df 8dd1 81a8 c235
0e64 21d8 872a b366 49a6 fbbc 18c5 5cbf 71cb bdaa d167 a080 c782 0bad b799 1a25 b4bb
4735 0698 fe72 9ab7 312c 1390 890c 40a4 9344 8855 c4c1 c5b7 bb84 f631 abbe 6b80 fa0b
6c9b f01b af4a aca8 721e 95c7 0231 58ef ecf7 04ee d85d a353 049d 4b08 db7e 3fef d10c 1844
edb5 554e 5e97 b48e ab12 132f 698a df59 861c 4f86 8020 b72b 3006 8191 3e44 e0b2 7653
f7af 9f25 d973 888f 78b2 355b 9f0c 0a9b 2fac e83f b2ef 02e5 f309 dc3f 9bc1 df2f c573 4d59
203b 5c16 81b0 1ec5 e0c2 1ccd 8304 79cc 37a3 8c55 61ff c490 3fdf 9913 7e29 8657 c8b0 3f38
6530 0812 37e9 9e58 9877 ba62 28ad ea76 5601 f73e cc7e 841b 3997 9bc7 f825 bc1b a239
db33 3790 3b0b 5450 2586 e031 53fc db34 061b 8721 0b8b 8d35 f4b6 07bc e86c a023 b203
0dfd 2106 122f de79 d841 e718 fafc a8ae 60f1 888f aa40 e68e 3062 faaa 399b fd11 a816 b4f4
9bef 69da 7bca 74f5 f94e 5566 d381 77c3 f922 3d06 b68a 4ddf 2b13 f1ca 1920 3efb 5a83
3016 9ceb 3a77 a0f6 8c53 d371 fcab 704c ce36 91c9 bc18 3f15 7107 a27e 7ec9 7772 9549
1671 4268 67ed f431 bb6f e79f 36fd 0d31 0f0f 6c21 ded9 1a9d 71b3 4eb7 ba37 0c06 5a25
aac2 a8ac ce09 b8e1 c023 7283 46de 5361 4d8a 6e7c a514 0382 5f77 bee7 b05c 0bf6 68e8
8f26 1544 c6aa 6125 6a0e c458 7f6b 4b41 188c 0257 1626 6345 71a6 0ffc 2209 8d8a 59e7
1219 328f e78b 543d e9a5 c2c8 5ad6 d44e 9551 97e1 c67d 9a78 4efb 8de1 e1ae 23fa 5967
1f0f 6803 60e3 ae27 1a7e 5f51 d6e2 e9f9 04d5 39dd c4f0 93ed 8dc5 940e 5b8e 6f15 0023 a091
aedb 0469 91b7 a86a e9a2 6e25 8208 b40a 89fa 7fb5 bb50 6618 d243 f387 5577 f083 0cd8
b4cc 802f aa5c d930 bf3c 5a99 c06c 8dea 29a6 0fe8 5ec8 ffd8 7f18 a99e 30cd d5a4 c0b0 d8cd
e626 7138 c026 b6f0 4217 2b09 c37c 7007 8008 fda5 7f98 9439 6ff4 109f 4878 1918 f9bd faea
6933 d666 cd06 6b9c 75d3 ac81 f138 1edb 9af2 cec6 a84f 3734 d2bf 13d0 c475 c1c0 a266
9755 cfa8 7adf af7e 84cf 7419 3261 1583 5a33 13d3 d501 08a6 3038 1a78 d253 0c84 596a

2bf6 18ca db4d 2590 5a1b 32c5 0ffd e21f 9129 a18c 6930 8bbf a26e 70d4 3880 994d ad0f
222c 2a96 c6b0 28a3 c424 e694 7f03 beef 108b 370b 7c02 1f49 b18f ee33 8b8e 92ca bdfe 35de
352a 7853 21c8 5788 1a4a 9b9f 3fec 0cb8 a8aa 7457 c3cf b66f f9bd 5545 0cc7 e8d3 9f6b 02f5
d92d 4b35 9588 0080 cd38 fe53 23d4 ac46 2ff3 9b3e a218 bf1b 2b85 6fba 3000 c683 6682
4fdd f861 21d5 9967 c98a f8d0 4b7d 0cb5 bbef 23d7 88d1 1652 6cc9 2712 0189 e538 1667
0e33 684e 8872 029d 474c efd1 e884 ebf8 357e b193 e41c 17da 0810 6907 9ca0 8d2f 73c5
832f 3088 f062 ca22 947e b220 8219 a4bf 908f b40d 0c1a e187 7372 8404 6394 7794 8190
e57b bafa 3d34 62f2 830c 617e 2bf3 de54 8a52 d89d 212a 2167 95b2 4f51 aeb9 22a7 9afa
86fb 4b9d df88 64e0 8da0 ebda fff9 60ed 518c b477 cd5a 9226 7b8f 6b9c 565a 619b 41c8
0c45 3c28 c774 34d1 1872 5d73 9027 fcad 59b6 46f2 ec07 75e4 3a57 27bb 12a9 f365 d6fe
4e43 8f51 340e 4bb4 b5b8 95c4 8d66 7723 2b5a 8fb4 43ba 96f2 51a7 a694 a037 efb2 b226
6146 8ccd e566 4b93 0d77 86a2 ebf0 aec6 8d36 3f3e 04c9 3d78 c809 cb35 bee8 b430 846e
d2d8 f71e 6544 02aa 6292 28b7 3375 2f0e 8d39 0494 8162 1e47 8ac6 a838 d36f 60d0 991a
e8b0 5df4 3c66 26c7 dde2 d730 a238 a79e 9272 43dd e834 d173 bf99 8334 6c23 6080 2bfc
1394 d8a3 c6c2 37c0 f841 24ee 3b6e 30b0 b26f ed0a 50c2 645b 5bed 18cc 69da fd96 1278
ce7d 04af d6ac ec0c e916 b9d4 8b43 4d29 1f44 5ea9 5e1c af72 2882 cee4 2405 8681 29a8
d263 f3a6 cd74 d8ef 18bf eeca e074 3338 e1ab 6beb 6d96 6583 7ab9 027e 5c4a 3548 0810
632f 8890 de52 aa75 6c7c ecbe 3b9d 6313 6f56 6cad f00d 5a12 cefe 25c7 49de cad6 1149 3f71
4474 4fde e1b2 e454 0fd5 72ff ac52 0dfd a431 f830 67f7 acc8 7f57 8a2c a5e4 d246 efc6 dea6
e811 6010 1358 07c9 d60e 8705 a59f bb44 036d 8976 c1be 5764 7150 a12e 1e38 ae78 ba6c
acae be65 ceaf d3b9 ec8b ebea f99a 2b4a e777 dd69 cf46 5c39 8364 e18d 2072 e2b3 a147
122d 7f18 fb35 5e34 b95e 6249 8554 a956 e2fc 7950 8c40 734a 1be7 1bbb d8da 93d3 b127
0e2f acf8 1279 6572 7e7b 4cb4 ef99 dad8 547c 15f9 7329 6a89 31ca 0d74 1e41 1aa0 5fc8 21fd
1759 626e a379 fc08 35f8 374e 1520 a3d6 f8e6 c6d6 67eb a280 4702 b832 b3e6 9a87 fdec 0f28
ce0d 5975 7789 8a2b 71f9 b2c5 d04a 533d 014c 0872 6d1f c7cf e94d 1938 e4b7 e55b 8096
d03f 9878 7046 0138 9237 d1f1 2112 0073 e208 4c28 1b24 62ac a506 db48 6806 69b2 d9c1
df7f d4c5 463a 064c 8952 99e5 2277 42e6 78cf f304 4c75 2f7e acf3 8a6b c688 dbcf 96be 0eed
ec01 2168 eee3 340d 0134 0f3f 7ad2 4dd0 0496 a347 7e99 8769 6310 7edb 7b56 b99b 43e2
8691 eeb4 923f 4d7e 0bf4 1bc6 8ef2 f74d 6f4d c730 b9ff 2553 d224 87a0 db1c 1e1c b774 5f25
fe70 9f99 cbb8 49fc b34b 9e4c af01 73a8 dd68 dbd4 5214 5b58 0724 9416 4e3f 08d5 7d1f
d3da 84c7 24b2 d7bc 019a 2dc5 ca18 623b 67a6 df7f c59a e9c9 f230 a971 587a a07a a8ba
7229 a793 10ef c7aa 549a c6cd 311a d4af a40e 99cb c87b 8522 6106 435d bd9e 8cae c26e
e078 15bc 5d31 194d f731 9eb8 12bd c597 7d42 8c24 6e1d 9043 880c d49c ac85 33c9 c489
a7b6 cf6b cf11 8996 5021 16fd