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 $O(n^22^n)$ 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.pasalic@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 \(AI_f = [n/2]\). 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 $O(D^{2+\varepsilon})$ 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 $O(n^{2n}D)$ 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 $GF(2^n)$ denote the binary Galois field and $GF(2)^n$ an $n$-dimensional vector space spanned over $GF(2)$. The operation “⊕” 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 $f : GF(2)^n \rightarrow GF(2)$. The set of all Boolean functions $f(x_1, \ldots, x_n)$ is denoted by $B_n$.

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

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

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 $f \in B_n$ is called the all term function. If $\deg(f) \leq 1$, a Boolean function $f \in B_n$ 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 $f \in B_n$ 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 $D_i = \{L(X'_n) \mid L(X''_{n-i}) = c \cdot X''_{n-i} \oplus b, c \in GF(2)^{n-i}, b \in GF(2)\}$. Then any nonlinear Boolean function $f \in \mathbb{B}_n$ can be decomposed and represented as below:

$$f(X) = f(X'_i, X''_{n-i}) = \sum_{i=1}^{n-1} \sum_{\sigma = \sigma_1 \ldots \sigma_i} \prod_{s=j_1}^{j_i} (x_s \oplus \sigma_s \oplus 1) \cdot \varphi_{i, [\sigma]}(X''_{n-i}), X'_i \in B_i,$$

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 $X'_i = (x_{j_1}, \ldots, x_{j_k}) = a \in GF(2)^i$, 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 $B_i = \{X'_i \mid f_{X'_i=a}(X''_{n-i}) \in \mathbb{B}_{n-i}, X'_i = a \in GF(2)^i\}$. 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 $X'_{i+1} \in GF(2)^i \setminus B_{i+1}$, 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 $||B_i|| \neq 0$. \[\Box\]

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. $\sum_{i=1}^{n-1} ||B_i|| \times 2^{n-i} = 2^n$, $||B_n|| \leq 2^n$.
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 $f(x_1, \ldots, x_4) = x_1 \oplus x_4 \oplus x_1 x_2 \oplus x_1 x_2 x_3$. 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 $x_3 = 1$ 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 $D_1 = \{x_1 \oplus x_4\}$. 

5
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 \(B_2^{(x_2, x_3)} = \{(1, 0), (1, 1)\}\).

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:
\[
f(x_1, x_2, x_3, x_4) = \sum_{\sigma=(x_2)\in B_1} \left( \prod_{s=2}^{3} (x_s \oplus \sigma_s \oplus 1) \right) \cdot \varphi_{1,[\sigma]}(B_1)
\]
\[
\oplus \sum_{\sigma=(x_2, x_3)\in B_2} \left( \prod_{s=2}^{3} (x_s \oplus \sigma_s \oplus 1) \right) \cdot \varphi_{2,[\sigma]}(B_2)
\]
\[
= (x_2 \oplus 1)(x_1 \oplus x_4) \oplus x_2(x_3 \oplus 1)x_4 \oplus x_2x_3(x_1 \oplus x_4).
\]

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 \((x_1 \rightarrow x_2 \rightarrow \ldots \rightarrow x_{n-1})\).

**Step 2** For each \(x_1 = a_1 \in GF(2)\), randomly choose different \(2^k\) pairs \((a_{n-1}^j, b_{n-1}^j)\),
\[ \alpha_{n-1}^j, \beta_{n-1}^j \in GF(2)^{n-1}, \text{ for } j = 1, \ldots, 2^k. \] 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 \( g_2(x_3, \ldots, x_n) = f_{x_1 = a_2}(x_3, \ldots, x_n) \) and again for each pair \((\alpha_{n-2}^j, \beta_{n-2}^j)\) perform \(2^k\) linear relation tests:

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

(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 \(i \rightarrow i + 1\) 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 \((\alpha_{n-i}^j, \beta_{n-i}^j) \in GF(2)^{n-i-1} \times GF(2)^{n-i-1}\), 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 \(\binom{2}{1} = 6\) 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.,

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

Moreover, we have

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

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 $\left(\begin{array}{c}n \\ n-1 \end{array}\right) \times (n - 1)! = n!$ 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 $\lambda = \min\{i \mid ||B_i|| \neq 0, i = (1, \ldots, n - 1)\}$.

**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

$$g(X) = g(X_i', X''_{n-i}) = \sum_{i=1}^{n-1} \sum_{\sigma = (\sigma_1, \ldots, \sigma_j) \in B_i} \left( \prod_{s=j_1}^{j_1} (x_s \oplus \sigma_s \oplus 1) \right) \cdot \phi_{i,\sigma}(X''_{n-i}), X'_i \in B_i, \quad (3)$$

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 $\lambda = \min\{i \mid ||B_i|| \neq 0, i = (1, \ldots, n - 1)\}$.

8
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 $|B_3| = 1, |B_4| = 4, |B_5| = 32, |B_6| = 0, i \neq (2, 4, 6)$. 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 $AI = 2$.

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 small 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 $T_B^j = 0$, and tables $C_i^j$, 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 $(\alpha^l_{n-1}, \beta^l_{n-1}), (\alpha^l_{n-1}, \beta^l_{n-1}) \in GF(2)^{n-1}$. Let $g = f_{x_1 = a_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_n}$ and for each pair $(\alpha^l_{n-1}, \beta^l_{n-1}), \ell = 1, \ldots, 2^k$, perform the linear relation test below:

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

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

(2) Let $I_1 = \{ j \mid \max_{j=1}^{n} T_B^j \}$.

**Step 3** Randomly choose $j^*_1 \in I_1$, for each $j \in \{1, \ldots, n\}$, $(j^*_1 \neq j)$ and for each $(a_1, x_j) = a_2 \in GF(2)^2, a_1 \in C_i^j$, randomly choose $2^k$ pairs $(\alpha^l_{n-2}, \beta^l_{n-2}), (\alpha^l_{n-2}, \beta^l_{n-2}) \in GF(2)^{n-2}$. Let $g_2 = f_{(x_1, x_j) = a_2}$ and for each pair $(\alpha^l_{n-2}, \beta^l_{n-2}), \ell = 1, \ldots, 2^k$, perform the linear relation test below:

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

(1) If all $2^k$ pairs $(\alpha^l_{n-2}, \beta^l_{n-2})$ pass the linear test, then let $T_{B_2}^j = T_{B_2}^j + 1$. Otherwise, save corresponding $a_2$ to table $C_i^j$.

(2) Let $I_2 = \{ j \mid \max_{j=1}^{n} T_B^j \}$. 

9
Similarly, repeat the Step 3 above, by increasing the size of input variables, i.e.,

\[(a_{i-1}, x_j) = a_i \in GF(2)^i, \text{ and } (i = 3, \ldots, n-k), j \in \{1, \ldots, n\}, j \neq j^*, j_i^* \in I_i, t = (1, \ldots, i - 1).\]

In general, for \(i = (n-k+1, \ldots, n-2),\) randomly choose different \(2^{n-i}\) pairs

\((\alpha_{n-i}^\ell, \beta_{n-i}^\ell), \alpha_{n-i}^\ell, \beta_{n-i}^\ell \in GF(2)^{n-i}, \ell = 1, \ldots, 2^{n-i},\) and check whether all \(2^{n-i}\) pairs can pass the linear relation test or not, where \(g_i = f(x_{i-1}^* x_j) = a_i.\)

**Step 5** Return the values of \(\|B_i\| = T_{B_i^*},\) 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.,

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

Moreover, we have

\[
T_{\text{complexity}} = 2^k \sum_{i=1}^{n-k} 2^i \times (n + 1 - i) + \sum_{j=n-k+1}^{n-1} 2^j \times 2^{n-j} \times (n + 1 - j)
\]

\[
< (2^n - 2^k + 2^n \times (n - 1 - (n - k + 1) + 1)) \times n
\]

\[
< (k \times 2^n) \times n,
\]

where \(k = \log_2 n.\) Therefore, the time complexity of this algorithm is about \(O(n 2^n \times \log_2 n)\) operations. The memory complexity is only about \(O(n 2^{n-1})\) 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.

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

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

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

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,

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

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

$$u'_{i,[\sigma]}(X''_{n-i}) = a_0^\sigma \oplus a_1^\sigma x_{i+1} \oplus \cdots \oplus a_{n-r_i}^\sigma x_n \oplus \cdots \oplus a_{n-r_i}^\sigma x_{i+r_i} \oplus \cdots \oplus a_{n-r_i+1}^\sigma x_{n-r_i+1} \cdots x_n. \quad (7)$$

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

Then we try to select the coefficients $a_i^\sigma \in GF(2)$ 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 $d$.
is at most $r_i + d_i$, $(1 \leq i < n - 1)$ since $\deg(u'_{i,\sigma}(X'_{n-i})) + \deg(\varphi_{i,\sigma}(X'_{n-i})) + \deg(x_j \cdots x_j) \leq r_i + 1 + d - 1 = r_i + d$, for $\sigma \in B_i$. If we are able to obtain such a choice of the coefficients $a_{q_i}^r \in GF(2)$ 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 $\sigma$ runs through all $B_i, (1 \leq i < n - 1)$, we obtain in total $\sum_{1 \leq i < n-1, |B_i| \neq 0} \prod_{i \in B_i} (x_i \oplus \sigma_i \oplus 1)$ 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 $\sum_{i}^{n} \binom{n}{i}^2$ 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., $\sum_{1 \leq i < n-1, |B_i| \neq 0} \prod_{i \in B_i} (x_i \oplus \sigma_i \oplus 1)$) 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 $a_{q_i}^r \in GF(2)$) with $r' + d \leq \deg(g^*) \leq r + d$, where $r = \max_{i=0}^{n-1} \{ r_i \} \text{ and } r' = \min_{i=0}^{n-1} \{ r_i \}$. 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:**

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

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

$$g'(X'_1, X''_{n-1}) = \sum_{i=1}^{n-1} (\prod_{\sigma=(\sigma_1, \ldots, \sigma_i) \in B_i, l=1}^{i} (x_l \oplus \sigma_l \oplus 1) \cdot \xi_{i, [\sigma]}(X''_{n-1})),$$

where each $\xi_{i, [\sigma]}(X''_{n-1})$, for any $\sigma \in B_i$, is a degree $r_i$ function given by,

$$\xi_{i, [\sigma]}(X''_{n-1}) = b'_0 \oplus b'_1 x_{i+1} \oplus \cdots \oplus b'_{n-1} x_n \oplus \cdots \oplus b'_{n-r_i+1, \ldots, n} x_{n-r_i+1} \cdots x_n. \quad (11)$$

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 $b'_q \in GF(2)$ in (11) in such a way that the terms in the ANF of $g'$ containing $x_j, \ldots, x_{j_1}$ are all cancelled for any $s < q \leq i < n$, where $s$ is a fixed integer and $\{j_1, \ldots, j_q\} \subset \{1, \ldots, i\}$. This also implies that the degree of

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

would be at most $r_i + s$ for each $B_i$ and $1 \leq i < n - 1$. If we are able to obtain such a choice of the coefficients $b'_q \in GF(2)$ in $\xi_{i, [\sigma]}(X''_{n-1})$, 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 $\sigma$ 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 $b'_q \in GF(2)$. On the other hand, to cancel all the terms in the ANF of $g'$ containing $x_{j_1}, \ldots, x_{j_q}$, will induce certain restrictions on the coefficients $b'_q \in GF(2)$ in the resulting system of homogeneous linear equations (involving these $b'_q \in GF(2)$) whose total number is given by,

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

where $i - s - 1 \geq 0$, $i - l \geq 0$. 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 $x_{j_1}, \ldots, x_{j_q}$ in $\sum_{\sigma \in B_i} \prod_{l=1}^{i} (x_l \oplus \sigma_l \oplus 1)$ section (where $\{j_1, \ldots, j_q\} \subset \{1, \ldots, i\}$), 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 $\xi_{i, [\sigma]}(X''_{n-1})$ 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 $B_i \neq \emptyset$, $(1 \leq i < n - 1)$.

Thus, if the

**Condition 1:**

$$\sum_{1 \leq i \leq n-1, ||B_i|| \neq 0} ||B_i|| \times \sum_{j=0}^{r_i} \binom{n-i}{j} \geq \sum_{1 \leq i \leq n-1, ||B_i|| \neq 0} \left( \sum_{l=0}^{i-s-1} \binom{i}{i-l} \sum_{j=0}^{r_i} \binom{n-i}{j} \right),$$

13
is at most $i$ that the ANF of $fg$ is at most $e$.

It is clear that the algebraic degree of $B$ into account the number of terms of the form $x_i$ in the ANF of equality (14). The first binomial term of the double binomial sum takes $\sum_{e=0}^{n-1} \sum_{l=0}^{i-s-1} \binom{i}{i-l}$ to counting all the terms in the ANF of equality (14) whose degree (denoted by $j$) is satisfied, then we will obtain at least one Boolean function $g'$ (by solving the system for unknown $b_i^\sigma \in GF(2)$) with $r' + s \leq \deg(g') \leq r + s$, where $r = \max_{i=0}^{n-1} \{ r_i \}$ and $r' = \min_{i=0}^{n-1} \{ r_i \}$.

(2) Secondly, we note that

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

where each $\deg(\varphi_{i,\sigma}(X''_{n-i})) \leq 1$, for any $\sigma \in B_i$, $1 \leq i \leq n - 1$.

It is clear that the algebraic degree of

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

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

$$\sum_{\sigma=(\sigma_1, \ldots, \sigma_i) \in B_i} \prod_{l=1}^{i} (x_l + \sigma_l + 1) \times \varphi_{i,\sigma}(X''_n)$$

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 $b_i^\sigma \in GF(2), \sigma \in B_i, 1 \leq i \leq n - 1$, which in total induces at most $\Lambda_1$ equations, where

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

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

The sum (i.e., $\sum_{\{1 \leq i \leq n-1, ||B_i|| \neq 0\}}$) in $\Lambda_1$ refers to counting all homogeneous linear equations for all $B_i \neq \emptyset, (1 \leq i < n - 1)$. The binomial sum $\sum_{j=e+1}^{i+r_i+1} \binom{n}{j}$ in $\Lambda_1$ 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 $r_i + 2 \leq v_1 \leq n - i$ and $i + 1 \leq j_1 < j_2 < \cdots < j_{v_1} \leq n$, since clearly $\varphi_{i,\sigma}(X''_n) \times \xi_{i,\sigma}(X''_{n-i})$ 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 $x_{j_1^*} \cdots x_{j_{l_1-v_1}}$ where $l_1 - v_1 > 0, e + 1 \leq l_1 \leq r + 1 + i$ and $1 \leq j_1^* < j_2^* < \cdots < j_{l_1-v_1}^* \leq i$, 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:**

$$\sum_{\{1 \leq i \leq n-1, ||B_i|| \neq 0\}} ||B_i|| \times \sum_{j=0}^{r_i} \binom{n-i}{j} \geq \Lambda_1,$$

where

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

(16)

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

### 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

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

and

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

for AA and FAA, where $\lambda = \min\{i \mid ||B_i|| \neq 0, i = 1, \ldots, n - 1\}$.

**Step 3** Repeat Step 1 and Step 2 $\left\lceil \frac{n}{\log_2 n} \right\rceil$ 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 $O\left(\left\lceil \frac{n}{\log_2 n} \right\rceil \times \log_2 n \times n 2^n \right) \approx O(n^2 2^n)$ operations, if Algorithm 2 is used to search for the values of $||B_i||$, $i = 1, \ldots, n - 1$. The memory complexity is about $O(n 2^n)$ 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 \(|B_i|\) ≠ 0 for small i, an optimal decomposition for \(B_i, (i = 1, \ldots, n - 1)\) in Algorithm 3 is still an open problem.

<table>
<thead>
<tr>
<th>The ability against AA or FAA</th>
<th>The time complexity</th>
<th>Resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>AA</td>
<td>(O(D^2))</td>
<td>[4]</td>
</tr>
<tr>
<td>AA</td>
<td>(O(n^d))</td>
<td>[7]</td>
</tr>
<tr>
<td>AA</td>
<td>(O(D^4))</td>
<td>[1]</td>
</tr>
<tr>
<td>FAA</td>
<td>(O(D^2E))</td>
<td>[1]</td>
</tr>
<tr>
<td>FAA</td>
<td>(O(DE^2 + D^2))</td>
<td>[2]</td>
</tr>
<tr>
<td>AA or FAA</td>
<td>(O(n^2nD))</td>
<td>[8]</td>
</tr>
<tr>
<td>AA or FAA</td>
<td>(O(D^{2+ε}))</td>
<td>[14]</td>
</tr>
<tr>
<td>AA or FAA</td>
<td>(O(n^22^n))</td>
<td>new</td>
</tr>
</tbody>
</table>

\[(D = \sum_{i=0}^{d} \binom{n}{i}, E = \sum_{i=0}^{ε} \binom{n}{i}, ε = 0.5).\]

| Table 2: A time complexity comparison for 30 ≤ n ≤ 40. |
| --- | --- | --- | --- | --- | --- | --- |
| n   | 30  | 31  | 32  | 33  | 34  | 35  | 36  | 37  | 38  | 39  | 40  |
| 281.63 | 284.39 | 287.49 | 290.36 | 293.36 | 296.24 | 299.24 | 302.12 | 305.12 | 308.01 | 311.01 |
| 2.3.00 | 2.3.54 | 2.5.32 | 2.5.67 | 2.5.62 | 2.6.24 | 2.6.00 | 2.5.68 | 2.5.62 | 2.5.01 | 2.5.01 |
| 2.54.42 | 2.145.54 | 2.157.16 | 2.150.12 | 2.171.09 | 2.172.05 | 2.183.20 | 2.184.16 | 2.196.40 | 2.197.42 | 2.199.88 |
| 2.98.02 | 2.70.41 | 2.72.91 | 2.75.30 | 2.78.80 | 2.80.20 | 2.82.70 | 2.85.10 | 2.87.60 | 2.89.01 | 2.92.54 |
| 2.40.81 | 2.42.04 | 2.43.74 | 2.45.18 | 2.46.68 | 2.48.12 | 2.49.62 | 2.51.06 | 2.52.56 | 2.54.01 | 2.55.51 |
| 2.22.81 | 2.22.81 | 2.24.09 | 2.24.08 | 2.25.17 | 2.25.17 | 2.26.34 | 2.26.34 | 2.27.50 | 2.27.50 | 2.28.63 |

\((\| : ε = 1, \| : D^{2+ε}, \| : D^{2-ε})\)

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 ≤ \(n\) ≤ 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., $||B_0|| = 132$, $||B_7|| = 102$, $||B_i|| = 0$, $i \neq (6, 7)$, when using the canonical order of fixing the input variables, thus $(x_1 \rightarrow x_2 \ldots \rightarrow x_{11})$. Using Algorithm 3 to estimate the theoretical lower and upper bound on AA, we found $5 \leq AI(f) \leq 6$, ($\Delta_{AA}^{lower} = r' + d = 1 + 4 = 5$, $\nabla_{AA}^{upper} = r + d = 1 + 5 = 6$), which is consistent to the real value $AI(f) = 5$. Moreover, we have found another decomposition for this function, i.e., $||B_0|| = 512$, $||B_i|| = 0$, $i \neq 9$, if the order of fixing the input variables is $(x_1, x_2, x_3, x_{10}, x_{11}, x_4, x_5, x_6, x_7, x_8, x_9)$. Using Algorithm 3 to estimate the theoretical lower and upper bound on the ability against FAA, we found $6 \leq \deg(g) + \deg(h) \leq 7$, ($\Delta_{FAA}^{lower} = r' + s + e = 0 + 1 + 5 = 6$, $\nabla_{FAA}^{upper} r + s + e = 1 + 1 + 5 = 7$), which is also consistent to the real value $\deg(g) + \deg(h) \geq 7$.

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 $\Delta_{FAA}^{lower} = \nabla_{FAA}^{upper} = 13$, 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 $\lambda$ 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 \( \Delta_{AA}^{lower} = \nabla_{AA}^{upper} = 7 \) 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 \( ||B_{2^n}|| = 2^{2^n - 1}, ||B_{2^n+1}|| = 2^{2^n - 1}, ||B_{2^n+2}|| = 2^n \), (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>
<thead>
<tr>
<th>( n )</th>
<th>( r' )</th>
<th>( d )</th>
<th>( r + d )</th>
<th>Optimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>1</td>
<td>5</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>14</td>
<td>1</td>
<td>6</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>16</td>
<td>1</td>
<td>7</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>18</td>
<td>1</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>20</td>
<td>1</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>22</td>
<td>1</td>
<td>9</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>24</td>
<td>1</td>
<td>9</td>
<td>10</td>
<td>12</td>
</tr>
<tr>
<td>26</td>
<td>1</td>
<td>10</td>
<td>11</td>
<td>13</td>
</tr>
<tr>
<td>28</td>
<td>1</td>
<td>11</td>
<td>12</td>
<td>14</td>
</tr>
<tr>
<td>30</td>
<td>1</td>
<td>11</td>
<td>12</td>
<td>15</td>
</tr>
<tr>
<td>32</td>
<td>1</td>
<td>12</td>
<td>13</td>
<td>16</td>
</tr>
<tr>
<td>34</td>
<td>1</td>
<td>13</td>
<td>14</td>
<td>17</td>
</tr>
<tr>
<td>36</td>
<td>1</td>
<td>13</td>
<td>14</td>
<td>18</td>
</tr>
<tr>
<td>38</td>
<td>1</td>
<td>14</td>
<td>15</td>
<td>19</td>
</tr>
<tr>
<td>40</td>
<td>1</td>
<td>15</td>
<td>16</td>
<td>20</td>
</tr>
</tbody>
</table>

### 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.

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

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


Advances in Cryptology–CRYPTO 2003 (Lecture Notes in Computer Science), vol. 2729. 

Related to Fast Algebraic Attacks,” in International Workshop on Boolean Functions: 


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.

\[
\begin{array}{c}
\text{7781 42ae f2d3 8ac3 f3e5 8b57 802c 88c9 ce89 297b 4ce5 d599 bd82 922b 55fc 3a16 f30a} \\
\text{55f1 4eb7 a053 2e6a fe3d 6ebc c48c b22b 1485 7433 3237 d922 8b8e 748f 9bf5 f561} \\
\text{56a9 3b3e c55a c06e 2065 d239 d1e3 a264 a2b6 7fe2 a67b 950e 008e 069f 92ef d309 0717} \\
\text{1f67 1aa1 9695 7e4f 8ca5 e200 d4a5 b60f 63f4 eb32 a274 4cb0 6ee6 d30e 3073 a13c} \\
\text{c25e 5830 91f1 1ebc 8c1f eb9d cb9f 249f 4d25 917c 8572 c2b2 9b6c d5f6 8d11 81a8 c235} \\
\text{0e64 2b18 872a b266 49a6 fbdc 18c5 5cbf 71cb dbaa d167 a080 c782 0bad b799 1a25 b4bb} \\
\text{4735 0698 ffe7 b9b7 312c 1390 890c 40a4 9344 8855 c417 c5b7 bb84 f631 abbe 6b80 fa0b} \\
\text{6c9b 801b a05c 4b70 6f1b af4a aca8 721e 95c7 0231 58ef ec7f 04ee d85d a353 049d b0db 87e8} \\
\text{a144 edb5 554e 5e97 b48e ab12 132f 698a d589 861c 4886 8020 b72b 3006 8191 3e44 e0b2 7653} \\
\text{f7af 92d5 9738 888f 78b2 35b5 90fc 0a9b 2fac e83f b2ef 02e5 f309 dc3f 9bc1 df2f c573 45d9} \\
\text{203b 51e1 81b0 1e0c 0e02 1ccd a308 790c 3f3c 65a5 0812 37e9 9e58 9877 ba62 28ad ea76 5011 73ee cc7e 841b} \\
\text{3997 9bc7 f825 bc1b a239 db33 3790 3b0b 5450 258e e031 53f3 db34 061b 8721 0b8b 8d35 f4b6 07bc e86c} \\
\text{a0b3 2b03 0df6 2106 122f de79 d411 e718 fafc a8ae 60f1 888f a40e e68e 3062 faa9 399b fd11 a816 b4f4} \\
\text{9bef 69da 7bca 74f5 f94e 5566 d381 77c3 f922 3d0b b68a 4ddf 2b13 f1ca 1920 3efb 5a83} \\
\text{3016 9eeb 3a77 a0f6 8e53 d371 fcab 704c ce36 91c9 bc18 3f15 7107 a27e 7ee9 7772 9549} \\
\text{1671 4268 67ed f431 bb6f e79f 36fd 0d31 0ff6 6c21 ded9 1a9d 71b3 4eb7 ba37 0c06 5a25} \\
\text{aac2 a8ac ce09 b8e1 c023 7283 46de 5361 4d8a 6e7c a514 0382 5777 bee7 b505 cb68 68e8} \\
\text{8f26 1544 c6aa 6125 6a0e 4c58 76f6 4b41 188c 0257 1626 6345 71a6 0f0c 2209 8d8a 59e7} \\
\text{1219 328f e78b 543d e9a5 c28e 5ad6 d44e 9551 97e1 c67d 9a78 4ebf 8de1 e1ae 23a5 59ef} \\
\text{10ff 6b03 60e3 ae27 1a7e 5f51 d6e2 e9f9 04d5 39dd c4f0 93ed 8dce 940e 5b8e 61f5 0123 a09f} \\
\text{aedb 0469 91b7 a86a e9a2 6e25 8208 b4a0 89fa 7fb5 bb50 6618 d243 f387 5577 f083 0cd8} \\
\text{b4ec 802f aa5c d930 83e7 5a99 c6ef 2abe 29a0 f0e8 5ec8 ffd8 7f18 a99e 30cd d5a4 codb 08cd} \\
\text{e626 7138 c026 b6f0 4217 2b09 c37c 7007 8008 fda5 798e 9439 6ff4 109f 4878 1918 9bbf faea} \\
\text{6933 d666 cd06 6bbd 8d3a 7a82 e878 1d9b faec 84ef 7419 3261 1583 5a33 13d3 d501 08a6 3038} \\
\text{1a78 d253 0e84 596a}
\end{array}
\]