ALGEBRAIC DEGREE ESTIMATION OF BLOCK CIPHERS USING RANDOMIZED ALGORITHM; UPPER-BOUND INTEGRAL DISTINGUISHER.

Haruhisa Kosuge and Hidema Tanaka
National Defense Academy of Japan, Yokosuka, Japan

ABSTRACT

Integral attack is a powerful method to recover the secret key of block cipher by exploiting a characteristic that a set of outputs after several rounds encryption has (integral distinguisher). Recently, Todo proposed a new algorithm to construct integral distinguisher with division property. However, the existence of integral distinguisher which holds in additional rounds cannot be denied by the algorithm. On the contrary, we take an approach to obtain the number of rounds which integral distinguisher does not hold (upper-bound integral distinguisher). The approach is based on algebraic degree estimation. We execute a random search for a term which has a degree equals the number of all inputted variables. We propose an algorithm and apply it to PRESENT and RECTANGLE. Then, we confirm that there exists no 8-round integral distinguisher in PRESENT and no 9-round integral distinguisher in RECTANGLE. From the facts, integral attack for more than 11-round and 13-round of PRESENT and RECTANGLE is infeasible, respectively.

KEYWORDS

Chosen plaintext attack, Integral attack, Algebraic normal form, Algebraic degree, PRESENT, RECTANGLE

1. INTRODUCTION

1.1 BACKGROUND

Integral attack was proposed as SQUARE attack [1], and Knudsen et al. formalized it as integral attack [2]. The attacker can recover the secret key by exploiting integral distinguisher which is obtained by $2^n$ chosen plaintexts, where $n$ must be less than the block length $M$ ($n < M$). A set of chosen plaintexts is encrypted for multiple rounds to make a set of outputs and we compute integration (XOR summation) of the set. If there exist a bit whose integrated value is constant for any secret key, we can define integral distinguisher.

Conventionally, we exploit two characteristics to construct integral distinguisher. One characteristic is based on algebraic degrees of boolean functions [3], and the other is on bijective functions inside the cipher function [2]. Recently, Todo proposed an algorithm to construct integral distinguisher by division property [4] which can exploit both characteristics. Using division property, full-round integral attack on MISTY1 was announced in CRYPTO2015 [5].

By overestimating algebraic degrees of nonlinear functions, the algorithm can effectively construct integral distinguisher. Owing to the overestimation, however, it is impossible to deny integral distinguisher in additional rounds [6]. For example, algebraic degrees of all output bits of S-box are regarded as 3 in PRESENT, though there is an output bit which has a degree 2. Division property can obtain 6-round integral distinguisher [4], however, Wu et al. showed 7-round integral distinguisher using the fact [7]. Because of the output bit with degree 2, the
algebraic degree increases unevenly when a round function of PRESENT is iterated [7]. Also in SIMON family [8], there is the same problem. In SIMON32, there is a gap between integral distinguisher constructed by division property [4] and one experimentally constructed in [9].

In order to solve the problem, the same author proposed similar algorithm using bit-based division property, and applied it to SIMON family [6]. As for SIMON32 (32-bit block cipher), the algorithm succeeded to construct the same integral distinguisher of [6]. However, it can not be applied if the block length is large since time and memory complexity increases in exponential order with the block length. Because of the complexity, it can not be applied to SIMON48.

Aside from division property, Perrin et al. proposed high-degree indicator matrix (HDIM) which can be used to search for integral distinguisher [10] in FSE2016. This is experimental method whose time and memory complexity increases in exponential order with the block length. Therefore, it is infeasible to compute HDIM when the block length is large as well as bit-based division property. They showed the existence of some patterns in the HDIM which holds any Feistel structure, however, their application is restricted.

In spite of remarkable advances in this research field, the feasible algorithm to obtain supremum integral distinguisher is still an open problem. Note that we define supremum integral distinguisher as $\gamma$-round integral distinguisher such that no $(\gamma + 1)$-round one can be constructed.

1.2 CONTRIBUTION

As a new solution, we propose an algorithm taking following three approaches.

(i) We focus on analyzing integral distinguisher which we cannot construct. We define such integral distinguisher as upper-bound integral distinguisher. Using it, we can obtain the number of rounds which integral attack is invalid. As we can construct integral distinguisher in additional rounds by increasing the number of chosen plaintexts [2], we consider integral distinguisher which is obtained by the maximum number of chosen plaintexts. It is $2^{M-1} (n = M - 1)$ in $M$-bit block cipher.

(ii) In order to achieve the objective of (i), we search for intermediate bits whose integrated values change depending on the secret key. We define such bits as unbalanced bits. We regard each intermediate bit as a boolean function expressed by the algebraic normal form (ANF). If there is a term in the ANF such that its degree is $n$ and its coefficient changes depending on the secret key, the bit is unbalanced bit. We aim to show the existence of such a term in the ANF. We propose term-expression set and term-expression chain to express any term in the ANF effectively. By expressing terms, we can check their degrees and coefficients.

(iii) Since the number of all terms increases in exponential order with the number of variables and rounds, we limit the number of terms we check depending on the computer environments. In order to search the terms explained in (ii), we propose a randomized algorithm to determine whether the bit is unbalanced or not. When we apply the proposal algorithm to all $M$ output bits in $\gamma$-th round, we can determine if we can construct $\gamma$-round integral distinguisher or not. If we cannot construct it, upper-bound integral distinguisher is $\gamma$-round one. number of terms.

We demonstrate the proposal algorithm in two 64-bit block ciphers PRESENT [11] and RECTANGLE [12]. As a result, we find that all bits in 8th round of PRESENT and 9th round of RECTANGLE are unbalanced bits. Therefore, upper-bound integral distinguisher is 8-round and
9-round integral distinguisher, respectively. In other word, there is a possibility that the attacker can construct 7-round and 8-round integral distinguisher, respectively. Using the most effective key-recovery technique for these two ciphers proposed by Zhang et al. [13], we estimate the number of rounds in which integral attack is invalid. Then, we confirm that it is infeasible to attack more than 11-round of PRESENT and 13-round of RECTANGLE. Hence, we conclude that full-round PRESENT (31-round) and RECTANGLE (25-round) are secure enough for integral attack.

Conventionally, we show the number of rounds we can attack to determine the security margin. Our approach is different from the concept. We aim to show the maximum number of rounds we cannot attack. The parameter is more effective to determine the security margin. To obtain the parameter, the algorithm obtain upper-bound integral distinguisher by estimating algebraic degrees.

2. PRELIMINARIES

2.1. ALGEBRAIC NORMAL FORM

Let $f$ be a boolean function of $n$ variables $(f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2)$. The algebraic normal form (ANF), polynomial representation over $\mathbb{F}_2$, of $f$ is written as

$$f(x) = \sum_{I \subseteq \{0, 1, \ldots, n-1\}} \alpha_I \prod_{i \in I} x_i,$$

(1)

where $P(I_N)$ denotes the power set of $I_N = \{0, 1, \ldots, n-1\}$ and $X = (x_0, x_1, \ldots, x_{n-1}) \in \mathbb{F}_2^n$. Note that $\alpha_I \prod_{i \in I} x_i$ is called term and it is composed of coefficient $\alpha_I$ and monomial $\prod_{i \in I} x_i$. A term whose monomial is 1 is called a constant term. For any $I$, we can compute the coefficient $\alpha_I$ as

$$\alpha_I = \sum_{X \in \mathbb{F}_2^n(\supp(X) \subseteq I)} f(X),$$

(2)

where $\supp(X) = \{i \in I_N \mid x_i = 1\}$. Let $\deg(f)$ be an algebraic degree of $f$ and it is defined as the maximum $|I|$, s.t., $\alpha_I = 1$.

2.2. PRESENT AND RECTANGLE

PRESENT [11] and RECTANGLE [12] are 64-bit lightweight block ciphers based on SPN structure which has three layers in each round: AddRoundkey, SBoxLayer and PerLayer. Both ciphers has two variations of key lengths, 80 and 128 bits. PRESENT and RECTANGLE iterate 31 rounds and 25 rounds, respectively. In AddRoundKey, a round key $k^{r-1} \in \mathbb{F}_2^{64}$ is XORed in $r$-th round. Then, a 4-bit Sbox $s$ is applied to each 4 bits of 64-bit block in parallel in SBoxLayer. Finally, a linear permutation $\pi : \mathbb{F}_2^4 \rightarrow \mathbb{F}_2^4$ is applied in PerLayer. S-boxes of PRESENT and RECTANGLE are bijective function $(S : \mathbb{F}_2^4 \rightarrow \mathbb{F}_2^4)$, and it is a vector boolean function from input bits $(x_0, x_1, x_2, x_3)$ to output bits $(y_0, y_1, y_2, y_3)$. The ANF of S-box of PRESENT is:

$$y_0 = 1 \oplus x_0 \oplus x_2 \oplus x_3 \oplus x_1 x_2 x_3 \oplus x_0 x_2 x_3 \oplus x_0 x_1 x_3$$
$$y_1 = 1 \oplus x_0 \oplus x_1 \oplus x_0 x_2 \oplus x_1 x_2 x_3 \oplus x_0 x_2 x_3 \oplus x_0 x_2 x_3 \oplus x_0 x_1 x_3$$
$$y_2 = x_0 \oplus x_2 \oplus x_0 x_2 \oplus x_0 x_1 \oplus x_1 x_2 x_3 \oplus x_0 x_2 x_3 \oplus x_0 x_1 x_3$$
$$y_3 = x_0 \oplus x_1 \oplus x_3 \oplus x_1 x_2.$$
Also, one of RECTANGLE is:

\[
\begin{align*}
    y_0 &= 1 \oplus x_1 \oplus x_2 \oplus x_1 x_2 \oplus x_1 x_2 x_1 \\
    y_1 &= x_0 \oplus x_1 \oplus x_2 \oplus x_2 \oplus x_0 x_3 \\
    y_2 &= x_1 \oplus x_2 \oplus x_1 x_3 \oplus x_0 x_3 \\
    y_3 &= 1 \oplus x_0 \oplus x_1 \oplus x_3 \oplus x_0 x_2 \oplus x_2 x_1 \oplus x_1 x_2 x_3.
\end{align*}
\]  

(4)

2.3. INTEGRAL DISTINGUISHER

If an integrated value is constant for all \( K \), we can define \(-\)round integral distinguisher since our focus is only on integral distinguisher, see [14] and [13] for techniques to recover round-key values.

![Figure 1: Round Function of PRESENT.](image1)

![Figure 2: Round Function of RECTANGLE.](image2)

2.3. INTEGRAL DISTINGUISHER

We denote an output of \( r \)-th round by \( X^r = (y_0^r, y_1^r, \ldots, y_{M-1}^r) \), where the plaintext is denoted by \( X^0 \).

The attacker chooses \( n \) bits \( X \in \mathbb{F}_2 \) from \( X^0 \) as variable \((n < M)\) and \( C \in \mathbb{F}_2^{M-n} \) as constant. Let \( I_N \) be an index set of variable bits and \( I_C \) one of constant bits \((|I_N| = n, |I_C| = M-n)\). For example, we set \( I_N = \{1, 2, \ldots, M-1\} \) and \( I_C = \{0\} \) when all bits other than leftmost bit \( x_0^0 \) is chosen as variable. The attacker prepares a set of \( 2^n \) chosen plaintexts in which \( X \) takes on all values through the set and \( C \) is an arbitrary constants. Let \( K \in \mathbb{F}_2^M \) be all round-key values used for encryption. Also we denote \( K \) and \( C \) as a set \( K = \{K, C\} \). The value of \( K \) is unknown for the attacker and one of \( C \) is known.

Suppose that \( l \)-th bit of \( y \)-th round is denoted by \( x_i^l = f_k(X) \), where \( f_k : \mathbb{F}_2 \rightarrow \mathbb{F}_2 \) is a boolean function defined by the structure of cipher function and the values of \( K \). We define integral of \( x_i^l \) as an XOR summation for all chosen plaintexts. An integrated value of \( x_i^l \) has following property.

\[
\bigoplus_{X \in \mathbb{F}_2} f_k(X) = \bigoplus_{X \in \mathbb{F}_2} \bigoplus_{x \in \mathbb{F}_2} a_l x_i^l \tag{5}
\]

\[
= \begin{cases} 
0 & \text{if } \deg(f_k) < n \\
1 & \text{if } \deg(f_k) = n 
\end{cases}
\]

If an integrated value is constant for all \( K \), we can define \(-\)round integral distinguisher. Since our focus is only on integral distinguisher, see [14] and [13] for techniques to recover round-key values.
2.4. BALANCED / UNBALANCED BIT

We introduce following three cases related to Eq.(5).

\[
\begin{align*}
\text{case 1 : } & \forall \mathbb{K}, \deg(f_K) < n \iff \forall \mathbb{K}, a_I = 0 \\
\text{case 2 : } & \forall \mathbb{K}, \deg(f_K) = n \iff \forall \mathbb{K}, a_I = 1 \\
\text{case 3 : } & \text{Otherwise } \Rightarrow \exists \mathbb{K}, a_I = 0 \text{ and } \exists \mathbb{K}, a_I = 1
\end{align*}
\]

Note that \(a_I\) is a coefficient for a monomial \(x^2 I\) in the boolean function \(f_K\) (see Eq.(5)). In case 1, the algebraic degree is upper-bounded by \(n - 1\) and the attacker can predict an integrated value as 0. In case 2, it is always \(n\) and the attacker can predict an integrated value as 1. Since the attacker can predict an integrated value of a certain bit both in case 1 and 2, he use the fact to recover some round-key values. If a bit \(x^2 I\) is in case 1 or 2, we express \(x^2 I\) is balanced bit. In case 3, the integrated value of \(x^2 I\) is unpredictable, and we express \(x^2 I\) is unbalanced bit.

3. RELATED WORKS

We show three works related to integral distinguisher, division property [4], bit-based division property [6] and HDIM [10]. These three works have similarity with our method, since they use the ANF and estimate algebraic degree to construct integral distinguisher. We use three cases shown in Sec.2.4 for clarifying objectives of these works.

3.1. DIVISION PROPERTY [4]

The major difference between division property and bit-based division property [6] is in the way to regard \(M\)-bit block. The former assumes it is composed of \(l\) \(m\)-bit words \((M = l \_ m)\), and the latter simply assumes it is composed of \(M\) bits. Generally, \(m\)-bit words are defined by nonlinear function such as S-box. We construct integral distinguisher in word unit by using division property. Specifically, we can determine all bits of each word are in case 1 or not exploiting the structure of round function. The structure is used in the propagation rule of division property, and division property holds independently of round-key and constant values in \(K\). The objectives of division property (bit-based division property) and HDIM differ in this point.

Let \(X^r = (x_0^r, x_1^r, ..., x_{M-1}^r) \in \mathbb{F}_2^M\) be an output value of \(r\)-th round, and \(X^r\) its set which is obtained by \(2^n\) times encryption in the integral attack scenario shown in Sec.2.3. Using \(m\)-bit words, we rewrite it as \(X^r = (x_0^r, x_1^r, ..., x_{M-1}^r)\), where \(x_i^r \in \mathbb{F}_2^m\). We denote \(i\)-th bit of \(x_i^r\) by \(x_i^{r[j]}(= x_{i_0}^{r[j]})\). Let \(U = (u_0, u_1, ..., u_{M-1}) \in \mathbb{F}_2^M\) be parity words, and \(r\)-th bit of \(u_j\) is written as \(u_j^{[r]}\). Also, \(W(U) = (w(u_0), w(u_1), ..., w(u_{M-1})) \in \mathbb{Z}^l\) is a vectorial Hamming weight of \(U\). We define division property by \(D_V\), where \(V\) is a set of vectorial integers \(V = (V_0, V_1, ..., V_{l-1}) \in \mathbb{Z}^l\). If \(w(u_j) \geq V_j\) for all \(j\), we define the relation as \(W(U) \geq V\). Bit product of \(X^r\) for \(U\) is defined as \(\prod_{j=0}^{l-1} (x_i^{r[j]}(u_j^{[r]})^k)^{[r]}\), where \((x_i^{r[j]}(0 = 1\) and \((x_i^{r[j]}(1 = x_i^{r[j]}\). An integrated value of bit product for all \(X^r\) has following property when \(X^r\) has division property \(D_V\).

\[
\bigoplus_{X \in X^r} \bigoplus_{j=0}^{l-1} [x_i^{r[j]}(u_j^{[r]})^k, l, l] = \begin{cases} \text{unknown} & \exists V \in V, W(U) \geq V \\ 0 & \text{otherwise} \end{cases}
\]
In $D_V$, if $V, V' \in V$ satisfy $V' \supseteq V$, $V'$ is removed from $V$ as $V \setminus \{V'\}$.

Each bit $x_i$ is expressed as a boolean function, and bit product $\bigwedge_{i=0}^{M-1} (x_i)^{\mu_i}$ is also a boolean function. Then, we consider three cases shown in Sec. 3 in the boolean function. If an integrated value equals 0 in Eq.(6), then the boolean function is in case 1. This is what Eq.(6) implies, and it does not consider case 2.

In [4], the author shows some rules for propagation of division property in iteration of a round function. Especially in $m$-bit non-linear function, we need to assume that all output bits have the same algebraic degree, since we consider the propagation in word unit. We assume that all output degree is overestimated, otherwise, Eq.(6) does not hold, since algebraic degree of bit product is underestimated. As mentioned in Sec. 1, it is impossible to construct 7-round integral distinguisher by using division property from the above.

After some iteration, if $V_i \geq 2$ for $V \in V$, all bits of $i$-th word is in case 1 (balanced), and the attacker can exploit it. If there is no $V \in V$, s.t., $V_i \geq 2$, we assume that all bits of $i$-th word is in case 3 (unbalanced). We apply the propagation rules of the round function until we confirm that all bits of all words are unbalanced. Note that the number of elements $|V|$ increases in the iteration. However, it is feasible to store all of them and apply the propagation rules in realistic time, since we can truncate redundant elements in $V$.

### 3.2. BIT-BASED DIVISION PROPERTY [6]

As mentioned in Sec. 3.1, word is not defined in bit-based division property. Therefore, we define bit-based division property by redefining Eq.(6). We use $X = (x_0, x_1, ..., x_{M-1}) \in \mathbb{F}_2^M$ and $U = (u_0, u_1, ..., u_{M-1}) \in \mathbb{F}_2^M$ defined in Sec. 3.1. Bit product of $X$ for $U$ is redefined as $\bigwedge_{i=0}^{M-1} (x_i)^{\mu_i}$, where $(x_i)^{\mu_i} = 1$ and $(x_i)^{\mu_i} = x_i$. Also, we define bit-based division property by $D_{V,W}$, where $V$ and $W$ are sets of $M$-bit values $V \in \mathbb{F}_2^M$ and $W \in \mathbb{F}_2^M$. An integrated value the bit products has following property in $D_{V,W}$.

$$\bigoplus_{X \in \mathbb{F}_2^M} \bigwedge_{i=0}^{M-1} (x_i)^{\mu_i} = \begin{cases} \text{unknown} & V \in V, \supp(U) \supseteq \supp(V) \\ 1 & W \in W, U = W \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

In $D_{V,W}$, if $V \in V$ and $W \in W$ satisfy $\supp(V) \subseteq \supp(W)$, $W$ is removed from $W$ as $W \setminus \{W\}$. Also, if $V, V' \in V$ satisfy $\supp(V') \subseteq \supp(V')$, $V'$ is removed from $V$ as $V \setminus \{V'\}$.

The basic idea is the same as division property, however, we consider case 2 in bit-based division property. We consider three cases shown in Sec. 3 in the boolean function $\bigwedge_{i=0}^{M-1} (x_i)^{\mu_i}$. If an integrated value equals 0 or 1 in Eq.(7), then $\bigwedge_{i=0}^{M-1} (x_i)^{\mu_i}$ is in case 1 or 2, respectively. This is what Eq.(7) implies.

In [6], the authors show some rules for propagation of bit-based division property in iteration of a round function. After some iteration, if $V_i = 1$ and $W_i = 1$ for $V \in V$, $i$-th and $r$-th bits are in case 1 (balanced) individually, and the attacker can exploit it. This is equivalent to determination of balanced bits in division property. In addition, we consider case 2 in bit-based division property. If two values $W_i = 1$ and $W_r = 1$ in $W \in \mathbb{F}_2$, $i$-th and $r$-th bits are in case 2 (balanced), and the authors used them for obtaining balanced bits in additional rounds. If $V_i = 1$ and all other values are 0 for $V \in V$, $i$-th bit is in case 3 (unbalanced).

The way to estimate complexity to compute bit-based division property has not yet been clarified. However, it needs much more complexity than the conventional division property. The
complexity of bit-based division property increases depending on the block length $M$. Therefore, the authors only show integral distinguisher of SIMON32 (32-bit block length).

### 3.3. High-Degree Indicator Matrix [10]

High-degree indicator matrix (HDIM) $\hat{H}$ for a bijective function $E_K : F^M_2 \to F^M_2$ is defined as follows.

$$\hat{H}(E_K)[i,j] = \bigoplus_{x \in F^M_2} a_i(x^j),$$

where $a_i : F^M_2 \to F_2$ is an $i$-th coordinate of $E_K$ and $x^0 = (x^0_1, x^0_2, \ldots, x^0_M)$ is a plaintext in which only $x^0_0$ is always 0. Note that $E_K$ is a cipher function defined by round-key values $K$ (see Sec.2.3). Computation of each coordinate Eq. (8) is equivalent to Eq. (2). If $\hat{H}(E_K)[i,j] = 1$, then $a_i$ has a term $[1]_{E_K} \cdot x^j_i$, where $E_K = [0, 1, \ldots, M - 1] / [j]$. Therefore, $a_i : F^M_2 \to F_2$ has degree $M - 1$. If $\hat{H}(E_K)[i,j] = 0$ holds for all $K$, case 1 holds ($n = M - 1$). Therefore, $i$-th output bit is balanced when all $M - 1$ bits other than $j$-th bit are chosen as variable. We can say the same thing for $\hat{H}(E_K)[i,j] = 0$ and case 2. In order to show case 3, two keys $K$ and $K'$, s.t., $\hat{H}(E_K)[i,j] = 0$ and $H(E_K)[i,j] = 1$ should be found.

In order to compute HDIM, it needs $M \cdot 2^{M-1}$ times encryption, and it is infeasible when $M$ is large. Also such computation should be repeated by changing the value of $K$ adequate number of times to conform integral distinguisher [9]. As a feasible technique, the existence of some patterns in the HDIM of Feistel structures is shown. However, the pattern cannot deny the existence of integral distinguisher which holds in additional rounds. Also for other structures, we still have to compute Eq. (8). Therefore, application of HDIM is highly restricted.

### 4. New Method To Express Terms

As shown in Sec.2.4, the coefficient $a_{k_i}$ will decide whether $x^j_i = f_2(x)$ is balanced/unbalanced. Using HDIM, $a_{k_i}$ can be obtained if we repeat computation of Eq. (8) by changing the value of $K$, however, it is infeasible as mentioned in Sec.3.3. Therefore, in the same manner with bit-based division property, we exploit the structure of round function and obtain a property which holds for any $K$.

Generally, the ANF of single round function is easily obtained, however, it is difficult to obtain the ANF after some round iteration. From plaintext side to ciphertext side, the ANF is obtained by repetition of substitution and expansion of polynomials. We call this method straightforward method. If we can execute straightforward method, we can express the coefficient $a_{k_i}$ as an AND product of round-key values $K$ and constant bits $C$ in $K$. Since it is infeasible when the number of rounds is large, we take another approach for the same objective.

We introduce term-expression set and term-expression chain in Sec.4.1 which are used to express single term of the ANF. In the expression, we can check a degree of this term, and we search for a term whose existence implies case 3 shown in Sec.2.4. We use the fact that case 3 holds if there is a term which has a degree $n$ and its coefficient $a_{k_i}$ varies with $K$. If we can find at least one term as above, case 3 holds and we can determine $x^j_i$ is unbalanced bit. As opposed to the related works shown in Sec.3 or other works [2][15][14][16], we aim to search for unbalanced bits. However, we need to consider bijective functions in the cipher function [2], therefore, we modify the proposal method in Sec.4.2. Also, we show the way to reduce the complexity in Sec.5.1.
4.1. Term-Expression Set And Term-Expression Chain

By the calculation from ciphertext side to plaintext side, we can express any of terms of $x_j^r$ as follows. Let $x^r = (x_0, x_1', ..., x_{3u-1}')$ be an output of $r$-th round. We use the ANF of a round function. For example in PRESENT [11], 3rd output bit of $r$-th round $x_j^r$ is obtained as

$$x_j^r = (k_0^r \oplus k_1^r \oplus k_2^r \oplus k_3^r k_4^r) \oplus x_0^r$$
$$\oplus (1 \oplus k_5^r) x_1^r \oplus k_6^r x_2^r \oplus x_3^r \oplus x_4^r x_5^r,$$  \hfill (9)

where $k_i^r \in F_2$ is a round-key value XORed with $x_i^r$ in AddRoundKey. Let $p_j^r$ be any of all terms in $x_j^r$, and there are 6 ways to express $p_j^r$ from Eq (9). Let $I[w]$ be an index set which is given by indexes of variables in $w$-th term of the ANF of round function, e.g., $I_0[2] = [1]$ and $I_0[5] = [1, 2]$.

We denote $a_w$ as the number of all terms in the ANF of round function ($w \in [0, 1, ..., a_w - 1]$). Note that $I[w]$ is an empty set when the term is constant, e.g., $I_0[0] = \emptyset$. Using $I[w]$, $p_j^r$ is obtained as

$$p_j^r = a_{I[w]} \prod_{\alpha \in I[w]} p_j^{r, \alpha}$$ \hfill (10)

Note that $a_{I[w]}$ is a coefficient for the monomial given by $I[w]$, e.g., $a_{I[4]} = 1 \oplus k_5^{r-1}$ and $a_{I[5]} = 1$. In Eq. (10), the AND product is applied for $p_j^{r, \alpha}$, since an AND product of $x_i^r$ becomes a polynomial and it contradicts with the definition of $p_j^r$.

Each $p_j^{r, \alpha}$ in Eq. (10) is expressed as an AND product of terms in the ANF of coordinates in $x_j^{r-2}$. Therefore, $p_j^r$ is obtained as

$$p_j^r = a_{I[w]} \prod_{\alpha \in I[w]} a_{I[w]} \prod_{\alpha \in I[w]} p_j^{r, \alpha}.$$ \hfill (11)

For all $t \in I[w]$, we define $I_t[w]$ by choosing $w \in (0, 1, ..., a_w - 1)$. We can uniquely obtain a coefficient $a_{I[w]}$ and monomial $\prod_{t \in I[w]} p_j^{r, \alpha}$ from $I_t[w]$.

The above procedure can be repeated. If it is repeated for $r$ times, $p_j^r$ can be expressed as $a_{I_{r+1}} \prod_{t \in I_{r+1}} x_j^t$, where $I \subseteq I_{r+1}$ (see Sec.2.3 for $I_0$). To formalize the procedure to express a term, we define following variables ($r, r' \in [1, 2, ..., y])$.

$$P^{-1} = \left\{ I_{[w]} \mid t \in \bigcup_{L \in [w]} I_{[w]}, w \in (0, 1, ..., a_{y-1}) \right\}$$
$$P = \{ P, P^{-1}, ..., P \}$$
$$\alpha^{-1} = \prod_{P \in P} \left( \prod_{I_{[w]} \in P} a_{I[w]} \right)$$
$$p^{-1} = \prod_{P \in P} \left( \prod_{I_{[w]} \in P} p_j^{r, \alpha} \right)$$
$$p_j^r = \alpha^{-1} p^{-1}$$ \hfill (12)
To obtain a term $p^0_i$ of $x^0_i = f_k^i(X)$, we divide it into a coefficient $a^0 \in \mathbb{F}_2$ and a monomial $p^0 \in \mathbb{F}_2$.

As we set $\Gamma' = \{\omega\}$ by choosing $w$ successively, we calculate as $\Gamma' \rightarrow \Gamma'^{-1} \rightarrow \ldots \rightarrow \Gamma$. Then, we can obtain $\Gamma'$. We call $\Gamma'$ term-expression set and $\Gamma'$ term-expression chain. As shown in Eq.(12), the monomial $p^0$ requires $\Gamma'$, however, the coefficient $a^0$ and the term $p^0_i$ require $\Gamma'$.

We need to choose $w$ out of all $\omega_i$ possible terms in the ANF of round function when we obtain $\Gamma'^{-1}$ from $\Gamma'$. Note that we regard term-expression set $\Gamma'$ as a multiset. For example, we suppose $\Gamma' = \{l_t[w], l_t[w]\}$, then $p^0$ is obtained as:

$$p^0 = \prod_{i \in [L]} p^0_i$$

From the definition of $p^0_i$, $p^0_i = p^0_i$ equals an AND of any two terms in $x^0_i$. If we regard $\Gamma'$ as a set, $p^0_i$ equals any single term of $x^0_i$ and we can not fully express the monomial $p^0$. Hence, $\Gamma'$ should be a multiset. Note that we regard $\bigcup_{l_t[w] \in \Gamma'} \{l_t[w]\}$ as a multiset for the same reason in Eq.(12). We can derive $l_t[w], e_{l_t[w]}$ and $\omega_i$ by the ANF of round function such as Eq.(9).

By repeating the procedure shown in Eq.(12) for $\gamma$ times, $p^0_i$ is expressed as $a^0 p^0_i$, and $p^0$ is written as:

$$p^0 = \prod_{l_t[w] \in \Gamma'} \left( \prod_{i \in L} x^0_i \right)$$

We express $p^0$ using $n$ variable bits $X$ in plaintext $X$ (see Sec.2.3). Then, Eq.(14) is rewritten as:

$$p^0 = \prod_{l_t[w] \in \Gamma'} \left( \prod_{i \in L} x^0_i \right) = a_C p_X$$

where

$$a_C = \prod_{l_t[w] \in \Gamma'} c_0, \quad p_X = \prod_{l_t[w] \in \Gamma'} x^0_i.$$ 

Hence, we can express a monomial $p_X$ using $\Gamma'$ and a term $p^0_i$ using $\Gamma'$. Since the number of variables included in $p_X$ equals the degree of $p^0_i$, it is easily checked. Also, we check if $a_C$ varies with the values of $K$. Then, we can determine whether the existence of this term implies case 3 or not.

### 4.2. CONSIDERATION OF BIJECTIVE FUNCTIONS

If we find a term which implies case 3, $x^0_i = f_k^i(X)$ may be balanced because of the existence of bijective functions in the cipher function [2]. We show the reason and the way to consider them.
Let $F : \mathbb{F}_2^n \to \mathbb{F}_2$ be a bijective function included in the cipher function. Note that $F$ is a random function defined by $K$. If a boolean function $f_K$ is a composition of $F$ and $f_K^F$ ($f_K = f_K^F \circ F$), we can express $f_K(X)$ by the ANF as follows.

$$f_K(X) = f_K^F(F(X)) = f_K^F(X') = \sum_{(r,t) \in R_N} a_{r,t} \prod_{j=1}^{2^n} x_j^r$$

(16)

where $R_N$ is an index set $(r,t)$, s.t., $x_j^r$ equals any of coordinates of $X' = F(X) (|X'| = n)$. Since $n$ bits $X'$ are output bits of intermediate rounds, we use $(r,t)$ as an element of $R_N$. As a toy example, we present a difference between $I_N$ and $I_N^I$ using two rounds of 16-bit PRESENT-like cipher in Fig.3, where $S : \mathbb{F}_2^4 \to \mathbb{F}_2^4$ is a bijective function. In this case, $F$ is composed of three S-boxes other than the leftmost one.

An integrated value is computed as

$$\bigoplus_{X \in \mathbb{F}_2^n} f_K(X) = \bigoplus_{X' \in \mathbb{F}_2^n} f_K^F(X')$$

(17)

where $I_N^I = \{0\}$ if $deg(f_K^F) < n$ and $I_N^I = \{1\}$ if $deg(f_K^F) = n$.

\[
\begin{array}{c|c|c}
\hline
K^0 & S & S \\
\hline
K^1 & S & S \\
\hline
\end{array}
\]

\[
\begin{array}{c|c|c}
\hline
K^0 & S & S \\
\hline
K^1 & S & S \\
\hline
\end{array}
\]

Figure 3: Difference between $IN$ and $I'$

We can apply case 1 to 3 to $f_K$ and $X'$. Suppose that we can find a term with degree $n$ and a coefficient $a_{r,t}$, varying with $K$ in $x_j^r = f_K^F(X')$. If case 1 or 2 holds in $x_j^r = f_K^F(X')$, then $x_j^r$ is balanced bit from Eq (17). Also, there are multiple ways to define $F$ and $f_K^F$ depending on the structure of the cipher function. If at least one combination $(F, f_K^F)$ satisfies case 1 or 2 for $x_j^r = f_K^F(X')$, $x_j^r$ is balanced bit. Hence, for all $(F, f_K^F)$, we need to show that $x_j^r$ is in case 3.

In order to obtain a term of $x_j^r = f_K^F(X')$, we need to modify Eq (12) by the new definition of $F$ as follows.
\[ p^{r-1} = p_X^{-1} p_X^{-1} \cdots p_X^{-1} \prod_{L \in \mathfrak{L}} \left( \prod_{e \in \mathfrak{E} \cap L} \prod_{i \in L} \prod_{j \in L_i} p_t^{-1} \right) \]

\[ p_X^{-1} = \prod_{L \in \mathfrak{L}} \left( \prod_{e \in \mathfrak{E} \cap L} \prod_{i \in L_i} \prod_{j \in L_j} x_i^{-1} \right) \]

By repeating the procedure shown in Eq. (12) for \( \gamma \) times, \( p_t^\gamma \) is expressed as \( a^0 p^0 \), and \( p^0 \) is written as

\[ p^0 = p_X^0 p_X^{r-1} \cdots p_X^{0} \prod_{L \in \mathfrak{L}} \left( \prod_{e \in \mathfrak{E} \cap L} \prod_{i \in L_i} x_i^0 \right) = a_C p_X^0 p_X^{r-1} \cdots p_X^{0}. \]  

5. RANDOMIZED ALGORITHM FOR ALGEBRAIC DEGREE ESTIMATION

By using the method shown in Sec.4, we can uniquely express a term in the ANF. Since the number of all possible ways to define term-expression chain \( J_1 \) equals one obtained in straightforward method, the complexity to check all the terms is easily estimated. We show the way based on straightforward method. As mentioned in Sec.4, the number of terms increases by the iteration of round function. Note that we do not consider combination of coefficients with the same monomial.

```
Algorithm 1 Counting algorithm for the number terms.
procedure COUNTTERMS(I_G, x^r)
    for all \((r, i) \in I_G\) do \(r_i^* = 1\) \(\triangleright \) Set 1 for only variable bits.
    end for
    for all \((r, i) \notin I_G\) do \(r_i^* = 0\)
    end for
    for \(r = 1 \rightarrow \gamma\) do \(\triangleright \gamma\) rounds iteration.
        for \(l = 0 \rightarrow M - 1\) do
            \(r_i^* = c_l (r_i^{l-1}, r_i^{l-1}, \ldots, r_i^{l-1})\) \(\triangleright c_l : Z^N \rightarrow Z\)
        end for
    end for
    return \(r_i^*\)
end procedure
```

(similar terms), e.g., it is counted as two terms when there is \(a_{[0,1]} x_0 x_1 \oplus a_{[0,1]} x_0 x_1\) in the ANF. If we combine similar terms, the number of all possible terms is \(2^n\) at most. However, we do not combine them, and the number is not therefore upper bounded.

To count the number of terms in \(x_i^r = f^r_G\), we use functions \(c_0, c_1, \ldots, c_{N-1}\) \(c_l : Z^N \rightarrow Z\) which calculate the number of terms included in an output bit from ones of input bits. Such functions can easily derived from the ANF of a round function. For example in PRESENT, we can derive \(c_3\) from Eq. (9) as
\[ x'_3 = (k_0^{-1} \oplus k_1^{-1} \oplus k_3^{-1} \oplus k_2^{-1} k_1^{-1}) \oplus x_0^{-1} \]
\[ \oplus (1 \oplus k_2^{-1} x_1^{-1} \oplus k_1^{-1} x_2^{-1} \oplus x_3^{-1} \oplus x_1^{-1} x_2^{-1}). \]
\[ = f'_3 = c_0 x'_3, c_1 x'_3, ..., c_{N-1} x'_3 \]
\[ = 1 + f'_3 \oplus f'_4 + f'_5 + f'_6 + f'_7 \times f'_8, \]

where \( f'_j \in \mathbb{Z} \) is the number of terms included in \( x'_j \). Using \( \{c_0, c_1, ..., c_{N-1}\} \), the number of terms \( f'_j \)
can be counted by Algorithm 1.

If the number \( f'_j \) is small enough to execute brute force search, we execute it. We try all possible ways to define \( J \in \{\mathbb{P}^1, \mathbb{P}^{-1}, ..., \mathbb{P}\} \) by using recursive algorithm. Then, we can determine to which cases shown in Sec.2.4 \( x'_j \) belongs to. If it is infeasible, however, we cannot help but limit the number of trials considering our computational resources. Let \( r \) be the number of trials in the search. Among \( r \) terms, if we can find a term which has a degree \( n \) and its coefficient varies with \( \mathbb{K} \), case 3 holds in \( x'_j = f'_2 (x'') \). If case 3 holds for all \( \{F, f'_2\} \), \( x'_j \) is unbalanced bit. Otherwise, we can not determine whether it is balanced or unbalanced bit, and we express \( x'_j \) is uncertain bit.

When \( r < f'_j \), we need to execute the search effectively as follows.

(i) We use randomized algorithm to search for a term which implies case 3. When we obtain \( \mathbb{P}^{-1} \) from \( \mathbb{P} \), we use a random number in choosing \( w \) in \( h[w] \) (see Eq (12)).

(ii) We reduce the number of all possible ways to define \( J \). In other word, we reduce the search space. We achieve it by truncating redundant terms in the ANF of round function, and we show the method in Sec. 5.1.

(iii) We execute brute force search by recursive algorithm before randomized algorithm. The combination of two algorithms are dedicated to block ciphers with simple structure. We show the method in Sec. 5.2.

### 5.1. Search Space Reduction

We search for a term which implies case 3 among \( f'_j \) terms using the method shown in Sec.4.1. Note that we only combine on finding term with degree \( n \), and consider its coefficient thereafter. Since we do not combine similar terms, there are multiple terms with the same monomial. Let \( h'_j \) be the number of terms with degree \( n \). Note that, it is still an open problem to obtain \( h'_j \) without the brute force search. If we try \( f'_j = h'_j \) terms, at least one term with degree \( n \) is found. If we can reduce \( f'_j \), we can reduce the number as well. Let \( f'_j \) and \( h'_j \) be the numbers of the terms after the reduction. If \( h'_j \) becomes 0 by the reduction, we can not find the objective term. Therefore, we should reduce the terms effectively. By truncating redundant terms from the ANF of round function, we achieve the above reduction.

We define redundant term as a term whose monomial is included in another term in the ANF. For example in Eq (9), \((1 \oplus k_1^{-1})x_1^{-1} \) has a monomial \( x_1^{-1} \) which is included in another term \( x_1^{-1} x_2^{-1} \). Therefore, we regard \((1 \oplus k_1^{-1})x_1^{-1} \) as a redundant term. Also, constant terms such as \((k_0^{-1} \oplus k_1^{-1} \oplus k_3^{-1} \oplus k_2^{-1}) \) is redundant, since their monomial is included in any variable term.

If \( h'_j > 0 \), we always have \( h'_j > 0 \) if we only truncate redundant terms. For example in Eq (9), we truncate redundant terms as

\[ x'_3 = (k_0^{-1} \oplus k_1^{-1} \oplus k_3^{-1} \oplus k_2^{-1} k_1^{-1}) \oplus x_0^{-1} \]
\[ \oplus (1 \oplus k_2^{-1} x_1^{-1} \oplus k_1^{-1} x_2^{-1} \oplus x_3^{-1} \oplus x_1^{-1} x_2^{-1}). \]
\[ = x'_3 = x_0^{-1} \oplus x_3^{-1} \oplus x_1^{-1} x_2^{-1}. \]
In this truncation, $h_{ij}^j - h_{ij}^i = h_{ij}^{i-1} + h_{ij}^{r-1}$. Even if terms with degree $n$ only exist in $x_{i-1}^1$ and $x_{r-1}^1$ ($h_{ij}^{i-1} > 0, h_{ij}^{r-1} > 0, h_{ij}^{r-1} = h_{ij}^{r-1} = 0$, $h_{ij}^{i-1}$ is more than 0, since the number included in $x_{i-1}^1x_{r-1}^1$ is more than $h_{ij}^{i-1}x_{r-1}^1 + h_{ij}^{r-1}x_{i-1}^1$. Since we can cut off $1 + x_{i-1}^1 + x_{r-1}^1$ terms as a whole, we can effectively reduce the search space by truncating redundant terms.

We use the above truncation technique to express the ANF of round functions. Let $\pi_P$ and $\pi_R$ be linear permutations of $\text{Player}$ of PRESENT and RECTANGLE, respectively [11][12]. We express the ANF of round function of PRESENT for $0 \leq j \leq 15$ as:

$$x_{\pi_P(4j)} = x_{4j_2}^{-1}x_{4j_3}^{-1} \oplus x_{4j_4}^{-1}x_{4j_5}^{-1} \oplus x_{4j_6}^{-1}x_{4j_7}^{-1} \oplus x_{4j_8}^{-1}x_{4j_9}$$
$$x_{\pi_P(4j+1)} = x_{4j_3}^{-1}x_{4j_4}^{-1}x_{4j_5}^{-1} \oplus x_{4j_6}^{-1}x_{4j_7}^{-1}x_{4j_8}^{-1}$$
$$x_{\pi_P(4j+2)} = x_{4j_4}^{-1}x_{4j_5}^{-1}x_{4j_6}^{-1} \oplus x_{4j_7}^{-1}x_{4j_8}^{-1}x_{4j_9}^{-1}$$
$$x_{\pi_P(4j+3)} = x_{4j_5}^{-1}x_{4j_6}^{-1}x_{4j_7}^{-1} \oplus x_{4j_8}^{-1}x_{4j_9}^{-1}x_{4j_1}^{-1}$$

(22)

Also, one of RECTANGLE $(0 \leq j \leq 15)$ is:

$$x_{\pi_R(4j)} = x_{4j_1}^{-1}x_{4j_2}^{-1} \oplus x_{4j_3}^{-1}x_{4j_4}^{-1}x_{4j_5}^{-1}$$
$$x_{\pi_R(4j+1)} = x_{4j_2}^{-1}x_{4j_3}^{-1} \oplus x_{4j_4}^{-1}x_{4j_5}^{-1}x_{4j_6}^{-1}$$
$$x_{\pi_R(4j+2)} = x_{4j_3}^{-1}x_{4j_4}^{-1}x_{4j_5}^{-1} \oplus x_{4j_6}^{-1}x_{4j_7}^{-1}x_{4j_8}^{-1}$$
$$x_{\pi_R(4j+3)} = x_{4j_4}^{-1}x_{4j_5}^{-1}x_{4j_6}^{-1} \oplus x_{4j_7}^{-1}x_{4j_8}^{-1}x_{4j_9}^{-1}$$

(23)

To show the effectiveness of the technique, we compute $f'$ for $1 \leq \gamma \leq 4$ and $0 \leq l \leq 63$ by two methods in PRESENT and RECTANGLE (see Table 1). The column “#terms (before truncation)” shows the results using the ANF of the actual round functions of the ciphers [11][12]. And “#terms (after truncation)” shows ones using the ANF of Eq.(22) and (23). Note that we input $f = (0, 0, 0.1, ..., 0.63)$ to CountTerm for simplicity, and we show only maximum and minimum number of terms among all 64 bits. From Table 1, we succeed to reduce the search space by the truncation of redundant terms, which enables us to estimate algebraic degrees in additional rounds.

Since there is no coefficient which varies with $K$ in Eq.(22) and (23), the term obtained by these equation has always a coefficient equals 1 ($a_0 = 1$) and the existence of the term does not imply case 3. We use a technique to use a redundant element of $I^l$. We rewrite Eq.(19) as
\[ p^0 = a_c p_n^r p_n^{-1} \cdots p_0^1 \prod_{l_v \in \mathbb{I}^1} \left( \prod_{a, l_v \in (0, \ldots, n)} x^0_i \right). \]  

(24)

If there exist at least one \( l_v[\bar{w}] \in \mathbb{I}^1 \) such that

\[
\begin{align*}
p^0 &= a_c p_n^r p_n^{-1} \cdots p_0^1 \prod_{l_v[\bar{w}] \in \mathbb{I}^1} \left( \prod_{a, l_v \in (0, \ldots, n)} x^0_i \right) \\
&= a_c p_n^r p_n^{-1} \cdots p_0^1 p_0^0 \\
&= a_c \prod_{(0, \ldots, n)} x^0_i.
\end{align*}
\]

(25)

The equation means that \( p^0 \) has a degree \( n \) without using \( l_v[\bar{w}] \in \mathbb{I}^1 \). In other word, \( l_v[\bar{w}] \in \mathbb{I}^1 \) is redundant to determine its degree. We use it for consideration of the coefficient \( a^0 \). We fix \( \bar{w} \) such that \( a^0 \) varies with \( \bar{w} \). Note that \( l_v[\bar{w}] \) indicates a term which is not in Eq. (22) and (23). If we use terms whose coefficients are always 1 such as Eq. (22) and (23) except for \( l_v[\bar{w}] \in \mathbb{I}^1 \), we have

\[
a^0 = \prod_{\bar{w} \in \mathbb{I}^1} \left( \prod_{a, l_v \in (0, \ldots, n)} a_{l_v[\bar{w}]} \right)
\]

(26)

Then, \( p^0_I = a_c a_{l_v[\bar{w}]} \prod_{(0, \ldots, n)} x^0_i \) holds. Therefore, the existence of such term implies case 3.

5.2. COMBINATION OF RECURSIVE AND RANDOM SEARCH

When we consider \( \mathcal{P} = \{P, P_1, \ldots, P_j\} \) which is an intermediate value of the process to obtain \( \mathcal{I}^1 \), it is possible that a degree of a term \( p^0_I \) depends on \( \mathcal{P} \). Especially for block ciphers with simple structure, we conjecture the above characteristic is eminent. Let \( \mathcal{P}_j \) be a certain term-expression chain. We regard \( \mathcal{P}_j \) as a box in which there are terms expressed by \( \mathcal{P}_j = \{P_j, P_1, \ldots, P_j\} \) with fixed \( P_j \).

The maximum number which \( J \) takes becomes large depending on the value of \( \mathcal{P}_j \). When \( \tau < \mathcal{P}_j \), there is a possibility that we cannot check any term in \( \mathcal{P}_j \) and it only has the terms with degree \( n \). In such case, we can not search the term effectively. To avoid the problem which occurs in simple block ciphers, we execute brute force search by recursive search before the randomized search mentioned in Sec.5. We obtain all possible term-expression chain \( \mathcal{P}^{\gamma} \) (\( \gamma \in [0, 1, \ldots, \gamma] \)), and we use them as start points of the random search. The value of \( \gamma \) is the number of rounds which we execute the recursive search.

Let \( \mathcal{B}(\mathcal{P}^{\gamma}) \) be a set of all possible \( \mathcal{P}^{\gamma} \). We can obtain \( |\mathcal{B}(\mathcal{P}^{\gamma})| \) by Algorithm 1. We execute the random search for \( \tau \) times from each start point, where \( \tau = |\mathcal{B}(\mathcal{P}^{\gamma})| \). The maximum value of \( \gamma \) is calculated as the maximum integer satisfying \( \tau > |\mathcal{B}(\mathcal{P}^{\gamma})| \). We determine the value of \( \gamma \) depending on the structure of cipher function. We discuss the effectiveness of this technique by the experimental results in Sec 7.
6. SPECIFICATION OF THE PROPOSAL ALGORITHMS

We show the specification of the algorithm using the method shown in Sec.4. Since we use the truncation technique for the reduction of the terms shown in Sec.5.1, we need to consider the existence of the coefficients which varies with $K$ shown in Eq.(25).

In order to make the algorithm simple, we define $\mathcal{P}_{\text{sum}}$ as follows.

$$\mathcal{P}_{\text{sum}} = \bigcup_{L \in \mathcal{L}[v]} L'$$  \hfill (27)

Using $\mathcal{P}'$, we can uniquely obtain $\mathcal{P}_{\text{sum}}$. We use two notations $\mathcal{P}'$ and $\mathcal{P}_{\text{sum}}$ as needed. As mentioned in Sec.4.1, $\mathcal{P}'$ and $\mathcal{P}_{\text{sum}}$ are regarded as matisets.

As mentioned in Sec.4.2, for all combinations $\{F, F'\}$, we need to execute this algorithm and show $x_i^\gamma = F'(x^\gamma)$ is in case 3 (unbalanced). Also, we apply it to all output bits of $\gamma$-th round to construct integral distinguisher. The proposal algorithm is shown in Algorithm 2. Note that we only use term-expression set $\mathcal{P}'$, since we mainly consider monomial which can be obtained only by term-expression set. Also, coefficients can be checked only by $\mathcal{P}'$ from Eq.(25) and (26). If there is case that we need to consider coefficients specifically, term-expression chain is required. We describe each procedure of Algorithm 2 as follows.

Degree Estimation: We can determine “$x_i^\gamma$ is unbalanced” when $n$ is substituted for a variable $flag_0$ once. Otherwise, “$x_i^\gamma$ is uncertain” is outputted. Note that $flag_0$ is a global variable as used in C/C++ language. DegreeEstimation call RecursiveSearch, and RecursiveSearch call RandomSearch.

RecursiveSearch: At first, the value of $flag_0$ is checked. If $flag_0 = 1$, RecursiveSearch is not called anymore and DegreeEstimation returns “$x_i^\gamma$ is unbalanced”. Next, $r > \gamma - \gamma'$ is checked. If $r \leq \gamma - \gamma'$, $\mathcal{P}_{\text{sum}} (r = \gamma - \gamma')$ becomes an input for RandomSearch. Otherwise, two procedures are repeated recursively: (i) Make each element $i \in \mathcal{P}_{\text{sum}}$ corresponds to $L[w] \in \mathcal{P}^{-1}$ (recursion in the same round). (ii) Make $\mathcal{P}^{-1}$ from $\mathcal{P}_{\text{sum}}^{-1}$ (recursion for a next round). By repeating procedure (i), we can try all possible $w$ for $L[w]$ ($i \in \mathcal{P}_{\text{sum}}$) recursively. When we set $L[w]$ for $i$, $i$ is removed from $\mathcal{P}_{\text{sum}}$ as $\mathcal{P}_{\text{sum}} \setminus \{i\}$. If there is no more element to try ($\mathcal{P}_{\text{sum}} = \emptyset$), then we advance to procedure (ii). In procedure (ii), $\mathcal{P}_{\text{sum}}^{-1}$ is uniquely obtained by $\mathcal{P}^{-1}$ as shown in Eq.(27), and $\mathcal{P}_{\text{sum}}^{-1}$ becomes an input for procedure (i) in the next round.

RandomSearch: In RandomSearch, $\mathcal{P}_{\text{sum}}^{-1}$ is generated from $\mathcal{P}_{\text{sum}} (r = \gamma - \gamma')$, and the procedure is iterated $\tau'$ times. We use a random number $rand$ to define a term-expression set $\mathcal{P}^{-1}$, where
Algorithm 2 Randomized Algorithm For Algebraic Degree Estimation

```
procedure DEGREESTIMATION(τ', γ', I_N, x_i)
    Y_sum ← [1], τ' = τ / CountTerms(τ', x_i)
    global flag0 ← 0  
    RecursiveSearch(τ', γ', I_N, Y_sum, φ)
    if flag0 = 1 then return "x_i is unbalanced."
    else return "x_i is uncertain."
end procedure

procedure RecursiveSearch(τ', γ', I_N, Y_sum, φ)
    if flag0 = 0 then  
        if τ > γ then
            if Y_sum ≠ φ then
                for all i ∈ Y_sum do Y_sum ← Y_sum \ {i}
                for w = 0 → ω - 1 do RecursiveSearch(τ', γ', I_N, Y_sum ∪ {I(w)})
            end for
        else for all I_τ[w] ∈ Y - 1 do
            for all i ∈ I_τ[w] do Y_sum ← Y_sum ∪ {i}
        end for
    RecursiveSearch(τ', γ', I_N, Y_sum, φ)
    else RandomRound(τ', I_N, Y_sum)
end if
end procedure

procedure RandomSearch(τ', I_N, Y_sum)
    for count = 0 → τ' - 1 do Y_sum ← Y_sum, flag1 ← 1, flag2 ← 0
    for τ' = τ + 2 do
        for all (τ' - 1, 1) ∈ I_N do
            if i ∉ Y_sum then flag1 = 0
        end if
    end for
    τ' - 1 ← φ, Y_sum - 1 ← φ
    for all τ' ∈ Y_sum do Y' - 1 ← Y' - 1 ∪ {I_τ[rand]}
    end for
    for all I_τ[w] ∈ Y' - 1 do
        for all i ∈ I_τ[w] do Y_sum ← Y_sum ∪ {i}
    end for
    for all I_τ[ω] ∈ Y' - 1 do
        for all (0, 1) ∈ I_τ do
            if i ∉ Y_sum then flag3 = 0
        end if
    end for
    if flag3 = 1 then flag2 = 1
    end if
    if flag1 = 1 and flag2 = 1 then flag0 = 1
    end if
end for
end procedure
```
Algorithm 3 Bijective Functions of PRESENT / RECTANGLE.

procedure BijectiveFunction($I_N, x_i^j$)  
for all $i \in I_N$ do $\hat{I}_N = \hat{I}_N \cup \{(0, i)\}$  
end for  
for $r = 0 \to \gamma - 1$ do  
for $j = 0 \to 15$ do  
if $\{(r, 4j), (r, 4j + 1), (r, 4j + 2), (r, 4j + 3)\} \subseteq \hat{I}_N$ then  
$\hat{I}_N = \hat{I}_N \setminus \{(r, 4j), (r, 4j + 1), (r, 4j + 2), (r, 4j + 3)\}$  
$\hat{I}_N = \hat{I}_N \cup \{(r + 1, \pi(4j))\}$  
$\hat{I}_N = \hat{I}_N \cup \{(r + 1, \pi(4j + 1))\}$  
$\hat{I}_N = \hat{I}_N \cup \{(r + 1, \pi(4j + 2))\}$  
$\hat{I}_N = \hat{I}_N \cup \{(r + 1, \pi(4j + 3))\}$  
end if  
end for  
end for  
return $\hat{I}_N$  
end procedure

$\text{rand} \in \{0, 1, \ldots, \omega_t - 1\}$. In the iteration, we check if $i$ is in $\hat{I}^r$ for all $(r - 1, i) \in I_N$. If at least one element is not in $\hat{I}^r$, the term we check does not include a variable $x_i^{r-1}$ and its degree is less than $n$. We substitute 0 for $\text{flag1}$ to imply the fact. In order to show that this term has a degree $n$, $\text{flag1} = 1$ is required.

Next, we consider the coefficient of this term. As mentioned in Sec.5.1, if there is no coefficient which varies with $K$ in the ANF of round function $g_2[i_0] = 1$ for any $I[w]$, the term has always a coefficient 1. Therefore, the existence of the term does not imply case 3 although this term has a degree $n$ ($\text{flag1} = 1$). If we can find $I[w] \in \hat{I}^r$ satisfying Eq. (25), the term has a coefficient which varies with $K$. We substitute 1 for $\text{flag2}$ to imply the fact. Both $\text{flag1} = 1$ and $\text{flag2} = 1$ are satisfied, we substitute 1 for $\text{flag0}$, and it implies that $x_i^j$ is unbalanced.

7. Application to Present and Rectangle

7.1. Bijective Functions of PRESENT and RECTANGLE

As mentioned in Sec.4.2, we consider bijective functions of PRESENT and RECTANGLE. We use a dedicated algorithm shown in Algorithm 3. Let $\{F, f_k\} (f_k = f_k \circ F)$ be a set of functions in which $X = F(X)$ corresponds to $I_N$, and the function BijectiveFunction in Algorithm 3 is a procedure to obtain $I_N$ from $I_N$. As mentioned in Sec.4.2, we need to find a term which implies case 3 for all possible $\{F, f_k\}$, however, we only need to consider $\{\hat{F}, f_k\}$ in PRESENT and RECTANGLE. Cipher functions of PRESENT and RECTANGLE is composed of multiple bijective functions defined by $G_j : (x_{i_0}^{j-1}, x_{i_1}^{j-1}, x_{i_2}^{j-1}, x_{i_3}^{j-1}) \rightarrow (x_{i_0}^{j-1}, x_{i_1}^{j-1}, x_{i_2}^{j-1}, x_{i_3})$, where $\pi$ is a linear permutation of $\text{Layer}$. Also, bijective functions $\hat{F}$ has the same structure. The bijective function $\hat{F}$ is one of $F$ which satisfies that $G_j \circ \hat{F}$ is not bijective for any $G_j$. In other word, $\hat{F}$ is the most comprehensive bijective function which consider all bijective functions.
7.2. Experimental Results

We applied Algorithm 2 and 3 to PRESENT and RECTANGLE. From the attack scenario of integral attack, 63 is the maximum number of variables. By setting 63 variables, we can construct integral distinguisher which holds in the maximum number of rounds. Therefore, we set all possible conditions for 63 variables. Let \( \lambda_\ell \in \{0, 1, \ldots, 63\} \) be an index of the constant bit in plaintext \( x^0 \). Therefore, \( F_\ell \) becomes \( \{0, 1, \ldots, 63\} \backslash \{\lambda_\ell\} \) and \( \hat{I}_X \) obtained by Algorithm 3 varies with \( \lambda_\ell \). When the index set \( \hat{I}_X \) indicates output bits of intermediate rounds, and the maximum number of rounds they exist \( \max(r | (r, t) \in \hat{I}_X) \) directly affects upper bound of integral distinguisher. We find that \( \max(r | (r, t) \in \hat{I}_X) = 3 \) in PRESENT, and \( \max(r | (r, t) \in \hat{I}_X) = 4 \) in RECTANGLE for all \( \lambda_\ell \).

First, we set \( \tau = 2^{22} \) considering our computer environment (CPU: Intel Core i7-3770 3.40GHz and memory: 16Gbyte), since we can check all 64 output bits for all 64 possible \( \lambda_\ell \) in a month. Note that \( 2^{22} \gg \tau \) for any \( \tau \) when \( \gamma > 5 \) in both ciphers. Next, we decide \( \gamma \) in Algorithm 2. For simplicity, we use \( \gamma = 0 \) or \( \gamma = \gamma_{\max} \), where \( \gamma \) is the maximum integer satisfying \( 2^{22} > |\mathcal{B}_F(\gamma_{\max})| \) (see Sec.5.3). Note that, \( \gamma_{\max} \) varies depending on the position of bits \( t \) from the result of Algorithm 1.

The above experimental results show that all 64 output bits of 8th round are unbalanced for all \( \lambda_\ell \) in PRESENT. Also in RECTANGLE, we find that all 64 output bits of 9th round are unbalanced for all \( \lambda_\ell \). Hence, we confirm that there is no 8-round integral distinguisher in PRESENT and no 9-round integral distinguisher in RECTANGLE. As shown in RecursiveSearch in Algorithm 2, if a term which implies that \( x^\ell \) is in case 3, the algorithm stops. Let \( \hat{t} (\hat{t} \leq \tau) \) be the number of terms already checked when the algorithm stops. When \( \gamma = 0 \), we have \( \hat{t} < 2^{12} \) in PRESENT and \( \hat{t} < 2^{20} \) in RECTANGLE for all \( \lambda_\ell \). On the other hand, when \( \gamma = 0 \), \( \hat{t} < 2^{20} \) in PRESENT and \( \hat{t} < 2^{13} \) in RECTANGLE for all \( \lambda_\ell \). From these results, the combination of recursive and random search is only effective in RECTANGLE. Since RECTANGLE has a more simple PLayer, the above results support our conjecture shown in Sec.5.3.

As an additional experiment, we obtain the number of unbalanced bits in 7th round of PRESENT and 8th round of RECTANGLE. We set \( \gamma = 0 \) for PRESENT and \( \gamma = \gamma_{\max} \) for RECTANGLE, and we set \( \tau = 2^{22} \). Let \( \mu \) be the number of unbalanced bits in a round \( (0 \leq \mu \leq 64) \). Table 2 shows \( \mu \) for all \( \lambda_\ell \) in PRESENT and RECTANGLE. The number of uncertain bits is \( 64 - \mu \). Therefore, the number of balanced bits can be \( 64 - \mu \) at most. In other words, there is a possibility that attacker can exploit \( 64 - \mu \) bits as integral distinguisher. When \( \mu \) is small, integral distinguisher is more advantageous for attacker. In PRESENT, there are 9 possible \( \lambda_\ell \) to have \( \mu = 45 \) from Table 2. On the contrary, we have \( \mu = 6 \) for all \( \lambda_\ell \) in RECTANGLE. Hence, integral distinguisher of PRESENT depends on a position of a constant bit.
7.3. DISCUSSION

Table 3 shows integral distinguisher shown in previous works. These results show the number of rounds in which integral distinguisher holds $\gamma$ and balanced bits $\beta$ when the number of chosen plaintexts is $2^n$. Note that our result shows the number of uncertain bits $64 - \mu$, and it may be less than $\beta$. Since $\gamma = 7$ in PRESENT and $\gamma = 8$ in RECTANGLE at most, our results are consistent with previous works.

From the results shown in Sec.7.2, we estimate the number of rounds which integral attack is

<table>
<thead>
<tr>
<th>$i_c$</th>
<th>$\mu$</th>
<th>$i_c$</th>
<th>$\mu$</th>
<th>$i_c$</th>
<th>$\mu$</th>
<th>$i_c$</th>
<th>$\mu$</th>
<th>$i_c$</th>
<th>$\mu$</th>
<th>$i_c$</th>
<th>$\mu$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>45</td>
<td>8</td>
<td>50</td>
<td>16</td>
<td>46</td>
<td>24</td>
<td>48</td>
<td>32</td>
<td>46</td>
<td>40</td>
<td>46</td>
</tr>
<tr>
<td>1</td>
<td>47</td>
<td>9</td>
<td>50</td>
<td>17</td>
<td>46</td>
<td>25</td>
<td>48</td>
<td>33</td>
<td>45</td>
<td>41</td>
<td>48</td>
</tr>
<tr>
<td>2</td>
<td>48</td>
<td>10</td>
<td>49</td>
<td>18</td>
<td>46</td>
<td>26</td>
<td>47</td>
<td>34</td>
<td>48</td>
<td>42</td>
<td>47</td>
</tr>
<tr>
<td>3</td>
<td>49</td>
<td>11</td>
<td>47</td>
<td>19</td>
<td>46</td>
<td>27</td>
<td>47</td>
<td>35</td>
<td>45</td>
<td>43</td>
<td>47</td>
</tr>
<tr>
<td>4</td>
<td>45</td>
<td>12</td>
<td>45</td>
<td>20</td>
<td>47</td>
<td>28</td>
<td>46</td>
<td>36</td>
<td>47</td>
<td>44</td>
<td>45</td>
</tr>
<tr>
<td>5</td>
<td>49</td>
<td>13</td>
<td>45</td>
<td>21</td>
<td>47</td>
<td>29</td>
<td>46</td>
<td>37</td>
<td>47</td>
<td>45</td>
<td>48</td>
</tr>
<tr>
<td>6</td>
<td>49</td>
<td>14</td>
<td>45</td>
<td>22</td>
<td>46</td>
<td>30</td>
<td>48</td>
<td>38</td>
<td>46</td>
<td>46</td>
<td>45</td>
</tr>
<tr>
<td>7</td>
<td>50</td>
<td>15</td>
<td>45</td>
<td>23</td>
<td>47</td>
<td>31</td>
<td>47</td>
<td>39</td>
<td>47</td>
<td>47</td>
<td>46</td>
</tr>
</tbody>
</table>

Table 2: Results on PRESENT And RECTANGLE.

<table>
<thead>
<tr>
<th>$\gamma$</th>
<th>$\beta$</th>
<th>$n$</th>
<th>reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.5</td>
<td>16</td>
<td>4</td>
<td>[16]</td>
</tr>
<tr>
<td>5</td>
<td>16</td>
<td>32</td>
<td>[15]</td>
</tr>
<tr>
<td>6</td>
<td>16</td>
<td>12</td>
<td>[17]</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
<td>60</td>
<td>[4]</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>16</td>
<td>[7] [13]</td>
</tr>
<tr>
<td>7</td>
<td>*19</td>
<td>63</td>
<td>our result</td>
</tr>
</tbody>
</table>

Table 3: Previous Results on PRESENT and RECTANGLE.

* the number of uncertain bits $(64 - \mu \leq \beta)$. 
8. CONCLUSION

We propose the randomized algorithm for algebraic degree estimation in order to obtain upper bound integral distinguisher. By demonstrating the algorithms in PRESENT and RECTANGLE, we confirm that 8- and 9-round integral distinguisher cannot be constructed, respectively. From these facts, we confirm that integral attacks for 11-round PRESENT and 13-round RECTANGLE are invalid. Hence, we conclude that both ciphers are secure enough for integral attack considering the specifications of the number of rounds. The upper bound integral distinguisher helps designers to decide security margin. The proposal algorithms enables them to obtain such security parameter by algebraic degree estimation.

9. ACKNOWLEDGMENT

This work was supported by JSPS KAKENHI Grant Number 24560491.

REFERENCES


