An algorithm for evaluating the linear redundancy and the factor of inertial groups of S-box

Nghi Nguyen Van

***Abstract*— This paper presents an algorithm for evaluating the linear redundancy and the factor of inertial groups with small computational complexity. Specifically, the article introduces the concept of the factor of inertial groups, the relationship between the factor of inertial groups and the linear redundancy of S-box. Thus, it is recommended to use S-boxes that do not possess linear redundancy and have the factor of inertial groups equal to 1 to have better cryptographic properties, and also provide an algorithm for searching such large size S-boxes.**

***Tóm tắt*— Bài viết này đưa ra thuật toán đánh giá độ dư thừa tuyến tính và hệ số quán tính của S-hộp với độ phức tạp tính toán nhỏ. Khái niệm hệ số quán tính, mối liên quan giữa hệ số nhóm quán tính và độ dư thừa tuyến tính của S-hộp được phân tích cụ thể. Qua đó, đưa ra khuyến nghị nên sử dụng các S-hộp không sở hữu độ dư thừa tuyến tính và có hệ số quán tính bằng 1 để có tính chất mật mã tốt hơn, đồng thời cũng đưa ra thuật toán để tìm các S-hộp kích thước lớn như vậy.**

***Keywords*— S-boxes; affine equivalence; inertial group; non-linear; linear redundancy; Boolean function.**

***Từ khóa*— S-Hộp; tương đương affine; nhóm quán tính; phi tuyến; độ dư tuyến tính; hàm Boolean.**

I. INTRODUCTION

Block ciphers are widely used to ensure information security on every application, platform in the information system today. The security of block cipher depends mainly on the non-linear component contained in its structure – which is also the S-box.

A strong S-box needs to satisfy various of properties, such as: the resistance against linear cryptanalysis, differential cryptanalysis, high algebraic degree, balancedness, non-linear structure, no fixed point…. In which, the two first attended properties are non-linearity (the resistance against linear cryptanalysis) and differential characteristic (the resistance against differential cryptanalysis). In fact, the popular S-box generation methods can not construct powerful S-boxes which can resist all block cipher attacks as well as optimize hardware or software performance in block cipher installation [1].

Affine equivanlence is attended problem in evaluating S-boxes. Through affine equivanlence for the original S-box, we create a class of S-boxes that have two same properties: non-linearity and differential characteristic. In this affine equivalent S-box class, cryptographers can select a S-box representation that is effectively installed on the hardware and software or against the side channel attacks better than the original S-box. This selection problem is executed by many different methods and the most common one is the algorithm presented in [2].

This paper presents the research results of the inertial group properties of S-box through the affine equivalence inherited and developed from [3]. The layout of the article is as follows:

- After the introduction, part II introduces the concept of the factor of inertial groups and the demonstration model of affine equivalent S-boxes.
- Part III presents the relationship between the factor of inertial groups and linear redundancy of S-box. At the same time, it is the necessary and sufficient condition for S-box to have an factor of inertial groups equal to 1.

This manuscript is received on December 2, 2018. It is commented on December 16, 2018 and is accepted on December 22, 2018 by the first reviewer. It is commented on December 18, 2018 and is accepted on December 27, 2018 by the second reviewer.

- Part IV of this paper presents the algorithm for searching S-box that does not possess linear redundancy and has an factor of inertial groups equal to 1. At last is the conclusion and the direction of future research.

II. THE FACTOR OF INERTIAL GROUPS OF S-BOX

A. Affine equivalence of S-box

The first problem is how to obtain a affine equivalent S-box with the given S-box? Considering the S-box of n = 4 bits given in Table 1 (denoted by S_1), to perform the affine equivalence, the value of S-box must be represented by the space of binary vectors V_4 in the primitive polynomial modulo $h(x)$.

TABLE 1. THE INITIAL S-BOX

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
0	1	9	E	D	B	7	6	F	2	C	5	A	4	3	8

Then the S-box class S_2 and S_1 are affine equivalent with if exists linear bijections A, B (two inverse binary matrixes $\in GL(n, \mathbb{F}_2)$) and vector constants $a, b \in \mathbb{F}_2^n$ such that,

$$S_2 = B \cdot S_1(A \cdot x \oplus a) \oplus b \ . \qquad (1)$$

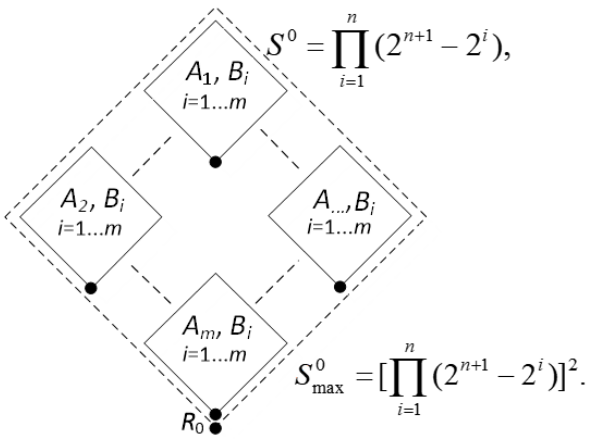

Fig 1. Representation model of affine equivalent S-box set

Having grouped S-boxes (denoted by S^0) received from S_1 by the affine equivalence respect to the value of *A* matrix and vector *a*, let us make up representation model of affine equivalent S-box set (Fig 1).

n bits S-box is presented in the form of the truth tables and have *n* component Boolean functions. Then the weight of S-box is calculated as the decimal value of component Boolean function in the bit order from top to bottom [2].

S-box R_0 on Figure 1 has the smallest weight and is called S-box representative of this representation model. S-boxes in each diamond (subsets of affine equivalent S-boxes) are obtained by affine equivalence by all possible values of *B* matrix and *b* vector while fixing the value of *A* matrix and *a* vector. S-box is represented by a dark dot at the lowest peak of the diamonds that have the smallest weight in each diamond, known as local representative.

B. The number of local representative

By fixing the value of *A* matrix and *a* vector while selecting all values of *B* matrix and *b* vector that can be obtained, the number of local representative (denoted by N_{lm}) or the number of affine equivalent S-box set is not exceeded the limit (2)

$$N_{lm} \le \prod_{i=1}^{n} (2^{n+1} - 2^i) \qquad (2)$$

For different *n* values we shall calculate maximum possible number of representatives and introduce them into Table 2.

TABLE 2. MAXIMUM NUMBER OF REPRESENTATIVES (ADJACENT CLASSES)

n	***max* N_{lm}**
4	322560
5	319979520
6	1290157424640
7	20972799094947840
8	1369104324918194995200
16	2,191516442724341427197177313e+81

C. The number of local representative and nonlinearity of S-box

Nonlinearity of S-box (denoted by N_{Sbox}) can be determined as the minimum Hamming distance between component functions defining S-box as well as their linear combinations and the whole set of their affine functions. See [4] for futher details on calculating its nonlinearity. Compare the number of local representative for

the affine non equivalent S-boxes that have different nonlinearity in Table 3.

TABLE 3. THE NUMBER OF LOCAL REPRESENTATIVE FOR S-BOXES 4×4

***S*-box**	N_{Sbox}	N_{lm}
D2781EB45AF093C	0	1
0FA5C369872D4BE1	0	1
01C86F4E3DBA2975	2	5376
019EDB76F2C5A438	4	5376
0123468A5BCF7E9D	4	80640
C462A5B9E8D703F1(GOST)	4	322560

As the results in Table 3 and experimental results for a large number of different S-boxes, we have commented that:

- S-boxes with nonlinearity equals to 0 always have the number of local representative as 1.
- S-boxes with the same high nonlinearity have different number of local representative and greater than 1.

The above is the basis for "new" property of S-box with the term "the factor of inertial groups" given in definition 1.

Definition 1: *The factor of inertial groups of S-box is the number of affine equivalent S-box set from that S-box and is denoted by* k_{ig} *received in accordance with the expression* (3),

$$k_{ig} = \frac{N_{lm}}{\max N_{lm}} \tag{3}$$

In Table 3, it is found that the S-box of Russian Federation's GOST 24.12-2015 "Magma" block cipher has a high nonlinearity $N_{Sbox}=4$ and has the largest number of local representative with the factor of inertial groups as $k_{ig}=1$. In conclusion, S-box with the factor of inertial groups equal to 1 need to be considered and evaluated based on other properties of S-box.

III. THE RELATIONSHIP BETWEEN THE FACTOR OF INERTIAL GROUPS AND LINEAR REDUNDANCY OF S-BOX

A. Linear redundancy of S-box

In [5], Fuller and Milan proposed a new standard to consider the possibility of choosing S-boxes that have good cryptographic properties, the notion of "non-possession of linear redundancy" is defined as follows:

Definition 2: *A* $n\times m$ *bit S-box has linear redundancy, if there exists at least two Boolean functions* $g(x)=a_1f_1\oplus a_2f_2\oplus...\oplus a_mf_m$ *and* $h(x)=b_1f_1\oplus b_2f_2\oplus...\oplus b_mf_m$ *with* $a=(a_1,a_2,...,a_m)$ *and* $b=(b_1,b_2,...,b_m)\in\mathbb{B}^m\setminus\{0\}$ *are affine equivalent.*

In contrast, S-box doesn't possess linear redundancy. And the number of pairs of different Boolean component function is affine equivalent, called linear redundancy of S-box.

When all Boolean component functions of S-box (excluding linear combination 0) are affine equivalent, S-box possesses the complete linear redundancy (see [7] for proof). In contrast, when there are two non-trivial linear combinations of affine equivalent coordinate functions, the S-box doesn't possess linear redundancy. A small size S-box would be easy to have linear redundancy because of a few affine equivalent classes when the number of variables in Boolean function is small. However, for larger size S-boxes, the number of equivalent classes increases dramatically. The presence of linear redundancy in substitution boxes is mentioned as an indicator of non-randomness, and according to some cryptanalyst's evaluation, this is a potential source for the cryptanalysts to exploit.

Some suggestions for the cryptanalysts to exploit were provided by Fuller et al as follows:

- Differential attacks are possible on round reduced ciphers due to S-boxes have linear redundancy. More research is needed to find out how the surrounding structures affect equivalent properties on multiple loops. Whenever the redundancy persists on some rounds, cipher does not show a random appearance and can easily be distinguished from the random one.
- Linear redundancy can be exploited to reduce the plaintext requirements of some existing attack techniques, propose new types of key related attacks, or improve the effectiveness of cryptanalysis such as using non-linear approximation, higher degree

derivative, interpolation attacks, square/integral attacks, algebraic attacks.

- The unique formula to performing Rijndael, presented at SAC2001 [6], is simplified by this result, since the divide function in the fractional multiplier is actually the inverse in the finite field. Fuller invites the cryptanalysts to check how this redundancy affects the complexity of solving the equations from [6].
- Some ciphers, including Rijndael that use the inverse of S-box in the key scheme lead to doubts about the effect of linear redundancy on round keys on the effectiveness of linear cryptanalysis and differential cryptanalysis.

B. The relationship between the factor of inertial groups and linear redundancy of S-box

Based on the results of the research about linear redundancy in [7] and experimental results with a large number of affine equivalent S-box classes, the paper gives the following theorem:

Theorem 1: *A $n\times n$ bit S-box S_1 doesn't contain any pairs of Boolean affine equivalent component functions, then it has the factor of inertial group as $k_{ig}=1$.*

Proof:

Since S_1 and S_2 are two affine equivalent S-boxes then there exist invertible maps $A=\left(a_{ij}\right)_{n\times n}$, $\mathrm{B}=\left(b_{ij}\right)_{n\times n}\in \mathrm{GL(n,}\mathbb{F}_2)$, and constants $c, d \in \mathbb{F}_2^n$ such that

$$S_1 = B\cdot S_2(A\cdot x\oplus c^T)\oplus d^T$$

Let $f_{n-1},...,f_0$, $g_{n-1},...,g_0$ with f_i, g_i are Boolean component functions of S_1 and S_2.

Then, we have the following expression derived from affine equivalent relationship:

$$f_i\ x\ =\sum_{j=0}^{n-1}b_{ij}g_j\ A\ x\ \oplus c^{\mathrm{T}}\ \oplus d_i$$

$$\Leftrightarrow f_i(x)=\sum_{j=0}^{n-1}b_{ij}g_j\left(Ax^T\oplus c^T\right)\oplus d_i,\ i=0,...,\ n-1 \quad (4)$$

Assuming S_1 contains two Boolean component functions f_λ, f_β in which $f_{n-1},...,f_0$ are affine equivalent, then there exist two sets $\lambda=\ \lambda_{n-1},...,\lambda_0$, $\beta=\ \beta_{n-1},...,\beta_0$ of the same matrix $\mathcal{D}\in GL\ n,\mathbb{F}_2$, n-dimensional vectors $u,l\in\mathbb{F}_2^n$ and $m\in\mathbb{F}_2$ such that:

$$\sum_{i=0}^{n-1}\lambda_i f_i\ x\ =\sum_{i=0}^{n-1}\beta_i f_i\ \mathcal{D}x^{\mathrm{T}}\oplus u^{\mathrm{T}}\ \oplus lx^{\mathrm{T}}\oplus m,\forall x\in\mathbb{F}_2^n \quad (5)$$

Substituting (4) into (5), we obtain:

$$\sum_{i=0}^{n-1}\lambda_i\left(\sum_{j=0}^{n-1}b_{ij}g_j\ Ax^{\mathrm{T}}\oplus c^{\mathrm{T}}\ \oplus d_i\right)=$$

$$\sum_{i=0}^{n-1}\beta_i\left(\sum_{j=0}^{n-1}b_{ij}g_j\ A\ \mathcal{D}x^{\mathrm{T}}\oplus u^{\mathrm{T}}\ \oplus c^{\mathrm{T}}\ \oplus d_i\right)\oplus lx^{\mathrm{T}}\oplus m$$

Follows,

$$\sum_{j=0}^{n-1}\sum_{i=0}^{n-1}\lambda_i b_{ij}g_j\ Ax^{\mathrm{T}}\oplus c^{\mathrm{T}}\ \oplus\sum_{i=0}^{n-1}\lambda_i d_i=$$

$$\sum_{j=0}^{n-1}\sum_{i=0}^{n-1}\beta_i b_{ij}g_j\ A\ \mathcal{D}x^{\mathrm{T}}\oplus u^{\mathrm{T}}\ \oplus c^{\mathrm{T}}\ \oplus\sum_{i=0}^{n-1}\beta_i d_i\oplus lx^{\mathrm{T}}\oplus m$$

Let $\lambda_j^*=\sum_{i=0}^{n-1}\lambda_i b_{ij},\beta_j^*=\sum_{i=0}^{n-1}\beta_i b_{ij}$ and $m^*=m\oplus\sum_{i=0}^{n-1}\lambda_i d_i\oplus\sum_{i=0}^{n-1}\beta_i d_i$, we have

$$\sum_{j=0}^{n-1}\lambda_j^* g_j\ Ax^{\mathrm{T}}\oplus c^{\mathrm{T}}\ =$$

$$\sum_{j=0}^{n-1}\beta_j^* g_j\ A\ \mathcal{D}x^{\mathrm{T}}\oplus u^{\mathrm{T}}\ \oplus c^{\mathrm{T}}\ \oplus lx^{\mathrm{T}}\oplus m^*$$

Next,

$$\sum_{j=0}^{n-1}\lambda_j^* g_j\ Ax^{\mathrm{T}}\oplus c^{\mathrm{T}}\ =$$

$$\sum_{i=0}^{n-1}\beta_j^* g_j\ A\mathcal{D}A^{-1}\ Ax^{\mathrm{T}}\oplus c^{\mathrm{T}}\ \oplus A\mathcal{D}A^{-1}c^{\mathrm{T}}\oplus Au^{\mathrm{T}}\oplus c^{\mathrm{T}}\ \oplus lx^{\mathrm{T}}\oplus m^*=$$

$$\sum_{i=0}^{n-1}\beta_j^* g_j\ A\mathcal{D}A^{-1}\ Ax^{\mathrm{T}}\oplus c^{\mathrm{T}}\ \oplus A\mathcal{D}A^{-1}c^{\mathrm{T}}\oplus Au^{\mathrm{T}}\oplus c^{\mathrm{T}}\ \oplus lx^{\mathrm{T}}\oplus m^*$$

Let $\mathcal{D}^*=A\mathcal{D}A^{-1},c^*=A\mathcal{D}A^{-1}c\oplus Au^{\mathrm{T}}\oplus c$, then we get the following equation:

$$\sum_{j=0}^{n-1}\lambda_j^* g_j\ Ax^{\mathrm{T}}\oplus c^{\mathrm{T}}\ =$$

$$\sum_{i=0}^{n-1}\beta_j^* g_j\ \mathcal{D}^*\ Ax^{\mathrm{T}}\oplus c^{\mathrm{T}}\ \oplus c*^{\mathrm{T}}\ \oplus lx^{\mathrm{T}}\oplus m^*$$

Since $A \in GL\ n,\mathbb{F}_2$, if x gets in turn all values in the space $\mathbb{F}_2^n$ then the transformation $A\ x\ \oplus c$ or $A^{\mathrm{T}}x \oplus c$ also gets in turn all values in $\mathbb{F}_2^n$. Then denoting $X = A^{\mathrm{T}}x \oplus c$, we have:

$$\sum_{j=0}^{n-1} \lambda_j^* g_j\ \ X\ =$$

$$\sum_{i=0}^{n-1} \beta_j^* g_j\ \ \mathcal{D}^* X^{\mathrm{T}} \oplus c^{\mathrm{T}}\ \ \oplus l x^{\mathrm{T}} \oplus m^*, \forall X \in \mathbb{F}_2^n \quad (6)$$

Equation (6) shows that S_2 has two non-trivial affine equivalent linear components. It is easy to see that the transformation and role of S_1 and S_2 in each equality are equivalent because $A, B, \mathcal{D}$ are invertible maps in the general linear group $GL(n,\mathbb{F}_2)$. Therefore, for any two linear affine equivalent components of S_2, we always obtain two corresponding linear affine equivalent component combinations of S_1, in other words, the numbers of pairs (λ,β) and (λ^*,β^*) are equal.

- From (4), calculate the number of local representative in Section II.A by exhausting all values of the inverse matrix B and all values of the constant d, it means that consider the case $B = E$ (unit matrix), then the set of affine equivalent S-boxes S_1 containing S_2 characterized by $\#L = 2^{n+1} - 2$ linear combinations [3] will contain Boolean affine equivalent functions $g_{L-2},...,g_0$ in pairs with only one Boolean function in $f_{L-2},...,f_0$.
- Assume S_1 does not contain any pair of Boolean affine equivalent component functions, i.e. there is no formula (6), then the sets of Boolean affine equivalent functions contain any pair of different (f_i, g_i). Thus, there exists $\prod_{i=1}^{n}\left(2^{n+1} - 2^i\right)$ different subsets and which S_1 are affine equivalent or S_1 has $k_{ig} = 1$. ■

Corollary 1: *S-boxes that do not possess linear redundancy have the factor of inertial groups equal to 1.*

Conclusion: It is recommended to use S-boxes that do not possess linear redundancy and have the factor of inertial groups $k_{ig} = 1$ to avoid linear attacks based on the Boolean component functions.

IV. ALGORITHM FOR SEARCHING S-BOX THAT DOES NOT POSSESS LINEAR REDUNDANCY AND HAS THE FACTOR OF INERTIAL GROUPS EQUAL TO 1

Based on the conclusions in Sections II and III of this paper, the problem of searching S-boxes that do not possess linear redundancy and have the factor of inertial groups equal to 1 is highly practical. However, calculating the number of local representative in Section II.A by exhausting all values of the inverse matrix B would be difficult for large value n $(n \geq 16)$. Therefore, this part of the paper will present the algorithm for finding large size S-box with the factor of inertial groups equal to 1 and does not possess linear redundancy based on the mathematical expression given by theorem 1, the proposed algorithm turns into the algorithm for testing affine equivalence of two Boolean component functions in S-box. And the mathematical basis for this algorithm is given in the following definitions and theorems.

Definition 3: *Two n bit Boolean functions f and g are affine equivalent if there exist invertible binary matrix $\mathcal{D}$, two binary vectors with n elements a, b and binary constant c such that:*

$$g(x) = f(\mathcal{D}(x) \oplus a^T) \oplus b \cdot x^T \oplus c, \quad (7)$$

with $b \cdot x^T = b_{n-1}x_{n-1} \oplus b_{n-2}x_{n-2} \oplus ... \oplus b_0x_0$ *is symbol of linear function of* x *is selected by b.*

From the above definition, it is natural to conclude that two functions f and g are not affine equivalent. We need to consider *#M(n)* (*M(n)* is a set of matrices that exist invertible matrix with the order n) matrix $\mathcal{D}$, 2^n vector a, 2^n vector b and two values c that result (3.1) do not occur. Because of the above condition and storing a square matrix $\mathcal{D}$ with the oder n needs n^2 bits, the following clause will be given.

Clause 1:

- The computational complexity of (7), denoted by T_1, is evaluated according to the following formula:

$$T_1 = \#M(n)\cdot 2^{2n+1} = \prod_{i=0}^{n-1}(2^n - 2^i)\cdot 2^{2n+1} \quad (8)$$

with $\#M(n) = \prod_{i=0}^{n-1}(2^n - 2^i)$ [9].

- The storage space of [7], denoted by R_1, is evaluated according to the following formula:

$$R_1 = \#M(n)\cdot n^2 = \prod_{i=0}^{n-1}(2^n - 2^i)\cdot n^2 \text{ bit} \quad (9)$$

A. Necessary conditions for two n bit Boolean functions f and g that are affine equivalent

Theorem 2: *Two n bit Boolean functions f and g are affine equivalent if the absolute distribution table of the Walsh-Hardamard Spectrum for the two functions is identical.*

Proof: The Walsh-Hardamard spectrum of Boolean function g at the point $u \in V_n$ is calculated by the following formula [10]:

$$W_g(u) = \sum_{x\in\mathbb{F}_2^n} (-1)^{g(x)\oplus\langle x,u\rangle} \quad (10)$$

Two Boolean component functions f and g are affine equivalent, then there exist the matrix $\mathcal{D} \in GL\ n,\mathbb{F}_2$, n-dimensional vectors $a,b \in \mathbb{F}_2^n$ and $c \in \mathbb{F}_2$ such that:

$$g(x) = f\ (\mathcal{D}x^{\mathrm{T}} \oplus a^{\mathrm{T}}\ \oplus bx^{\mathrm{T}} \oplus c, \forall x \in \mathbb{F}_2^n \quad (11)$$

Substituting (11) and $\langle u,x\rangle = u\cdot x^T$ into (10) we have:

$$W_g(u) = \sum_{x\in\mathbb{F}_2^n} (-1)^{u\cdot x^T \oplus f\left(\mathcal{D}x^T \oplus a^T\right)\oplus bx^T \oplus c} \quad (12)$$

Let:

- $\mathcal{D}x^{\mathrm{T}} \oplus a^{\mathrm{T}} = z^T \Leftrightarrow x^T = \mathcal{D}^{-1}\cdot(z^T \oplus a^{\mathrm{T}})$
- $k\cdot x^{\mathrm{T}} = (u \oplus b)x^{\mathrm{T}}$

And substituting into (10) we have:

$$W_g(u) = (-1)^{k\mathcal{D}^{-1}a^T\oplus c} \sum_{x\in\mathbb{F}_2^n} (-1)^{k\mathcal{D}^{-1}\cdot z^T \oplus f\left(z^T\right)}$$

$$= (-1)^{k\mathcal{D}^{-1}a^T\oplus c}\cdot W_f(k\mathcal{D}^{-1})$$

Know that:

- Since matrix $\mathcal{D}$ is invertible, then $h = k\cdot\mathcal{D}^{-1}$ gets in turn all values in $\mathbb{F}_2^n$.
- $(-1)^{k\mathcal{D}^{-1}a^T\oplus c} = \pm 1$.

So we have:

$$(12) \Rightarrow\ W_g(u) = \pm W_f(\mathrm{h}).$$

or

$$\left|W_g(u)\right| = \left|W_f(\mathrm{h})\right|.$$

B. Sufficient conditions for two n bit Boolean functions f and g that are affine equivalent

From theorem 2, consider the affine equivalence of two functions *f* and *g* according to the following algorithm.

Algorithm 1: *Consider the affine equivalence of two Boolean component functions of S-box*

INPUT: Two *n* bit Boolean component functions *f* and *g*.

OUTPUT: Message about the affine equivalence of two Boolean functions *f* and *g*.

Details of all steps of the algorithm as following:

1) Input is two *n* bit Boolean functions *f* and *g*.

2) Calculate Walsh-Hardamard spectrums of two Boolean functions *f* and *g*.

3) Calculate the frequency distribution table of absolute values of Walsh-Hardamard spectrums.

4) Compare the frequency distribution table of absolute values of Walsh-Hardamard spectrums for functions *f* and *g*. Where the comparing results are the same, the two functions may be affine equivalent. In different cases, it is concluded that *f* and *g* are not affine equivalent.

With the above algorithm, we only need to calculate and store two tables *WH(f)* and *WH(g)*. The calculation of each table takes 2^n times to be calculated according to (10), and each table consists of 2^n+1 integers in the range from 0 to 2^n (those numbers are not exceeding $n+1$ bits) so need $(2^n+1)(n+1)$ bits to storage. In short, we have results in clause 2 below.

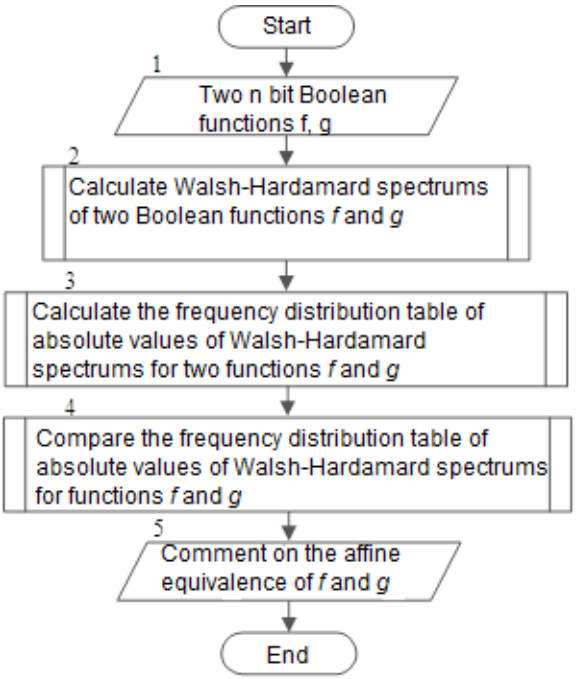

Fig 2. Algorithm for testing affine equivalence of two Boolean component functions of S-box

Clause 2:

- The computational complexity of algorithm 1, denoted by T_2 using the fast Walsh-Hardamard spectrums algorithm in [8], is evaluated according to the following formula:

$$T_2 = n \cdot 2^{2n} \qquad (13)$$

- The storage space of algorithm 1, denoted by R_2, is evaluated according to the following formula:

$$R_2 = 2(2^n + 1) \cdot (n+1) \qquad (14)$$

The consideration of sufficient conditions for two n bit Boolean functions f and g that are affine equivalent means that the functions f and g have the same table of absolute values of Walsh-Hardamard spectrums or the opposite of the theorem 2, the author executed the brute force with $n = 3$ bits as follows:

- Sort $2^8 = 256$ Boolean 3 bit functions into classes with the same table *WH(f)* called *WH* class. Make table of the numbers of *WH* class and the number of elements in each *WH* class.

- In each class, checking the affine equivalence of boolean function pairs according to (7) are exhausted by all inverse matrices $\mathcal{D}$, two binary vectors with n elements a, b and a binary constant c. Results are:

+ Class with *WH = (7, 0, 0, 0, 0, 0, 0, 0, 1)* includes 16 Boolean functions, of which there are 14 balanced functions (represented by an integer or weight of Boolean function), is *{0, 15, 51, 60, 85, 90, 102, 105, 150, 153, 165, 170, 195, 204, 240, 255}.*

+ Class with *WH = (0, 0, 7, 0, 0, 0, 1, 0, 0)* includes 128 functions, of which there is no balanced function, is *{1, 2, 4, 7, 8, 11, 13, 14, 16, 19, 21, 22, 25, 26, 28, 31, 32, 35, 37, 38, 41, 42, 44, 47, 49, 50, 52, 55, 56, 59, 61, 62, 64, 67, 69, 70, 73, 74, 76, 79, 81, 82, 84, 87, 88, 91, 93, 94, 97, 98, 100, 103, 104, 107, 109, 110, 112, 115, 117, 118, 121, 122, 124, 127, 128, 131, 133, 134, 137, 138, 140, 143, 145, 146, 148, 151, 152, 155, 157, 158, 161, 162, 164, 167, 168, 171, 173, 174, 176, 179, 181, 182, 185, 186, 188, 191, 193, 194, 196, 199, 200, 203, 205, 206, 208, 211, 213, 214, 217, 218, 220, 223, 224, 227, 229, 230, 233, 234, 236, 239, 241, 242, 244, 247, 248, 251, 253, 254}.*

+ Class with *WH = (4, 0, 0, 0, 4, 0, 0, 0, 0)* includes 112 functions, of which there are 56 balanced functions, is *{3, 5, 6, 9, 10, 12, 17, 18, 20, 23, 24, 27, 29, 30, 33, 34, 36, 39, 40, 43, 45, 46, 48, 53, 54, 57, 58, 63, 65, 66, 68, 71, 72, 75, 77, 78, 80, 83, 86, 89, 92, 95, 96, 99, 101, 106, 108, 111, 113, 114, 116, 119, 120, 123, 125, 126, 129, 130, 132, 135, 136, 139, 141, 142, 144, 147, 149, 154, 156, 159, 160, 163, 166, 169, 172, 175, 177, 178, 180, 183, 184, 187, 189, 190, 192, 197, 198, 201, 202, 207, 209, 210, 212, 215, 216, 219, 221, 222, 225, 226, 228, 231, 232, 235, 237, 238, 243, 245, 246, 249, 250, 252};*

NOTE: Boolean functions are represented by integers like the type of representation of a vector. It means that:

$$f = \left(f(0), f(1), \ldots, f(2^n - 1) \right)$$

$$= f(0), f(1) \cdot 2, \ldots, f(2^n - 1) \cdot 2^{2^n - 1}$$

+ Obtain several pairs of f and g functions that have the absolute values of Walsh-Hardamard spectrums but are not affine

equivalent. These pairs *f* and *g* are unbalanced (see example 1).

+ The pairs of *f* and *g* functions that have the absolute values of Walsh-Hardamard spectrums and are balanced are affine equivalent.

Example 1: Two Boolean 3 bits functions *f* and *g* have the same distribution table of absolute values of Walsh-Hardamard spectrums but are not affine equivalent (Table 4).

TABLE 4. 3 BIT F AND G FUNCTIONS

x	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
f(x)	1	0	0	0	0	0	0	0
g(x)	0	0	0	0	0	0	0	1

Thus, it has been proved that with *n = 3* theorem 2 has the opposite to satisfy the necessary and sufficient condition for two Boolean functions *f* and *g* to be affine equivalent when there is more condition that both *f* and *g* are balanced. Predict that the above is also true for any *n*. And note that the Boolean component functions in S-boxes are balanced so algorithm 1 is applied to check the affine equivalence of Boolean functions in S-boxes.

The installing program using algorithm 1, works well with *n = 8, 16, 24* bits and has computational complexity of clause 2 is quite small compared to the exhausting algorithm in clause 1 and is also smaller than the algorithm for checking the affine property of two Boolean functions presented in [5].

V. CONCLUDE

In the article, the author presents a new property of S-box as the factor of inertial groups, the relationship between it and the linear redundancy that suggets the use of S-box does not possess linear redundancy and has the factor of inertial groups equal to 1 to avoid attacks based on the linear relationship between Boolean component functions. In the last part of this paper, we suggested the algorithm for searching S-boxes that do not possess linear redundancy and has the factor of inertial groups equal to 1 based on the algorithm for testing the affine equivalence of two Boolean component functions with a small computational complexity. This is also an algorithm that plays an important role in determining the linear redundancy of S-box.

The proof of mathematical theory in contrast to theorem 2 is the problem that needs to be solved for the author in fucture research.

REFERENCES

[1]. Panasenko, “Encryption Algorithms, Specialized book”, BHV-Petersburg, pp. 576, 2009.

[2]. Alex Birykov, Christophe De Cannere, An Braeken, and Barn Prenell, “A Toolbox for Cryptanalysis: Linear and Affine Equivalent Algorithms”, Advances in Cryptology – EUROCRYPTO 2003. Springer, Vol. 2656, pp. 33–50, 2003.

[3]. N. P. Borisenko, “ Using search algorithm of affine equivalent S-boxes set for their quality assessment”, Ban Cơ yếu Chính Phủ, Nghiên cứu Khoa học và Công nghệ trong lĩnh vực An toàn thông tin, Hà Nội, pp.11-16, 2016.

[4]. O. A.Logachev, A. A. Salnikov, S. V. Smyshlyaev [and another]. “Boolean functions in coding theory and cryptology”, Moscow: LENAND, pp. 576, 2015.

[5]. Joanne Fuller and William Millan, “Linear redundancy in S-boxes” in Fast Software Encryption. Springer, pp. 15, 2003.

[6]. Niels Ferguson, Richard Schroeppel, and Doug Whiting “A Simple Algebraic Representation of Rijndael”. SAC 2001, LNCS 2259, pp. 103–111, 2001.

[7]. Nguyen Bui Cuong, Nguyen Van Long, Hoang Dinh Linh, “Analyzing the influence of linear redundancy in S-boxes with affine equivalence within XSL-like round functions”, Yaroslavl: CTCrypt, pp. 9, 2016.

[8]. Stjepan Picek, “Applications of Evolutionary Computation to Cryptology”, Radboud University, Netherlands, pp. 184, 2015.

ABOUT THE AUTHOR

BS. Nghi Nguyen Van

Workplace: FSO Academy, Russia
Email: nghivn25@gmail.com

The education process: graduated from FSO academy in Russia in 2013, specializing in “Information security”, a PhD student at FSO academy in Russia since October 2016, specializing in “Information security”.
Research today: Cryptography science.