Linear Hybrid Cellular Automata (LHCA) Rule 90/150 Based S-Box

Muhammad Fahim Roslan, Kamaruzzaman Seman, Azni Haslizan Ab Halim, M Nor Azizi Md Sayuti

Abstract*: Most modern block cipher today comprises of single nonlinear element which is called as substitution box. It comes with various of size depending on the cipher input and the secret key. Various methods have been proposed in order to construct it. In this paper, a new construction method has been proposed that is based on linear hybrid cellular automata (LHCA) iteration. There are specific number of combinations of rule 150 and rule 90 of elementary cellular automata (ECA) with two states and three neighborhoods that generate maximum cycle sequence which usually adapted as pseudo random number generator. However, the sequence is actually the permutation of finite elements which is suitable for the S-Box construction. Within this paper context, we manipulate the sequence by varying the initial condition to produce about 8192 different S-Box with different security level from 32 different rule combination. The maximum nonlinearity that we could achieve is 108 with differential uniformity value of 8.

Index Terms: Block Cipher, Cellular Automata, Cryptography , Primitive Polynomial

I. I.INTRODUCTION

A frequent discussed topic in block cipher is the construction and implementation of the substitution box (S-Box). The trend of most modern block cipher design only adapt one nonlinear component while the rest use linear transformation or permutation. It can be seen in the implementation of data encryption standard (DES), advanced encryption standard (AES), PRESENT and many others. Hence, it is important to select the best method of S-Box construction and implementation so that the cipher will have efficient operation time and optimum security level. Even though an S-Box acts as the main component of any block cipher, the operation is simply substitutes the input with the predefined value from the fixed table. Hence it also can be viewed as a mapping of the set of finite input to the set of finite output. Besides, its implementation method have also been widely discussed as the evolution of cryptanalysis methods have emerge rapidly. As in this paper, we proposed a method of construction with new implementation. The construction method is based on the linear hybrid cellular automata (LHCA) method. This method is based on the iteration of elementary cellular automata that have been described in detailed in [1].

In the next section, some preliminary concept regarding S-Box as vectorial Boolean function is discussed. Besides, as the same section, the preliminary definition of cellular automata is also presented. At section 3, some relevant studies related to cellular automata and its application in cryptography is presented. Section 4 continue with the methodology of the proposed construction methods together with complete pseudo code. The analysis in term of cryptography properties and the security of constructed S-Box is presented in section 5. The last section concludes the paper.

II. PRELIMINARY CONCEPT

A. Vectorial Boolean Function

This section presents some fundamental aspects of vectorial Boolean function in the application of cryptography block cipher. An S-Box S can be interpreted as vectorial Boolean function which received n -binary input and generate m-binary output. Let $\mathbb{F}_2^n$ be a vector space of n-variables with the cardinality of 2^n. Then an S-Box S can be viewed as map $S: \mathbb{F}_2^n \to \mathbb{F}_2^m$ where each $\bar{x} = (x_1, x_2, \dots, x_n) \in \mathbb{F}_2^n$ and each $\bar{y} = (y_1, y_2, \dots, y_m) \in \mathbb{F}_2^m$ can be assigned as $S(\bar{x}) = \bar{y}$. The map S comprises of m-single Boolean functions which can be defined as $g_i: \mathbb{F}_2^n \to \mathbb{F}_2$ where $i = 1,2,\dots,m$. Most cryptography properties are evaluated based on these functions which also can be referred as coordinate function $S = (g_1, g_2, \dots, g_m)$. In evaluating S-Box, several representations may be used such as truth table, polarity truth table and algebraic normal form. The truth table is simply the collection of all output of Boolean function f with the input vector $\bar{x} \in \mathbb{F}_2^n$ arranged in lexicographical order and has the dimension of 2^n. The polarity truth table on the other hand, is made up of the sign function expressed as $\hat{g} = (-1)^g$ which yield the output of -1 and 1 for the input of 0 and 1 respectively. Despite the truth table, another representation, the algebraic normal form (ANF) represent the Boolean function f in term of polynomial. In general, the ANF of any Boolean function with n variables can be expressed as

$$g(x) = c_0 \bigoplus_{1 \le i \le n} c_i x_i \bigoplus_{1 \le i \le j \le n} c_{ij} x_i x_j \oplus \mathrm{L} \oplus c_{1\dots n} x_1 x_2 \dots x_n \quad (0)$$

where the coefficient $c_0, c_i, c_{ij}, c_{i,\dots,n}$ having the value of either 0 and 1. Each Boolean function f of n variables will have its own unique ANF. The longest term in ANF determine the algebraic degree of the Boolean function f.

Revised Manuscript Received on May 05, 2019.
Muhammad Fahim Roslan, Universiti Sains Islam Malaysia
Kamaruzzaman Seman, Universiti Sains Islam Malaysia
Azni Haslizan Ab Halim, Universiti Sains Islam Malaysia
M Nor Azizi Md Sayuti, Universiti Sains Islam Malaysia

The function is called affine function if the longest term equal to 1, while with the absence of constant term c_0, the function is categorized as linear Boolean function.

B. Cryptography Properties

The cryptography properties are the set of tools that quantify the security of any given substitution box. It comprises of several quantity in which the most essential are the nonlinearity, differential uniformity, algebraic degree and balancedness. The nonlinearity is a measure of distance of any Boolean function f to the set of all affine function $\ell_{a,b}$, that is

$$N_g = \min_{a\in \mathrm{F}_2^n, b\in \mathrm{F}_2} \mathrm{dist}(g, \mathrm{l}_{a,b}) \tag{2}$$

where $\ell_{a,b}(x) = \langle a, x\rangle \oplus b$; $\langle a, x\rangle$ denote the inner product and $\oplus$ as a XOR Boolean operator. The distance mentioned in Eq.2 means the hamming distance. Higher value of N_g is more favorable as it indicates greater distance to the set of all affine function. Frequently, the quantity N_g is evaluated using the Walsh-Hadamard Transform (WHT) that utilize the sign function $\hat{g}$ which can be expressed as

$$W_g(\omega) = \sum_{x\in \mathrm{F}_2^n} (-1)^{\langle x,\omega\rangle \oplus g(x)} \tag{3}$$

with $\forall \omega \in \mathbb{F}_2^n$. The output of Eq.3 is defined as the Walsh spectrum which has the range of $W_g(\omega) \in [-2^n, 2^n]$. The spectrum is then defined the value of nonlinearity of function g by the following equation,

$$N_g = 2^{n-1} - \frac{1}{2}\max_{\omega\in \mathrm{F}_2^n} \left|W_g(\omega)\right| \tag{4}$$

The evaluation of the above equation on any S-Box S is done on each coordinate function g_i where the minimum one is taken as the value of nonlinearity N_g .Differential uniformity is another quantity that defined the security of cryptography algorithm against differential analysis [2] It analyzes the relationship between the difference in plaintext and its corresponding difference in ciphertext. This quantity is defined by the minimum entry of the differential distribution table denoted by parameter δ as follows,

$$\delta = \max_{a,b\in \mathrm{F}_2^n, a\neq 0} \left|x \in \mathrm{F}_2^n : S(x+a) + S(x) = b\right| \tag{5}$$

This parameter only takes even value with the smallest value indicates better resistance against differential analysis. The algebraic degree of an S-Box S is defined as the maximum algebraic degree of all coordinate functions g_i. Higher algebraic degree is preferable in order to thwart higher-order differential attacks.

C. Cellular Automata

The cellular automata (CA) are parallel computation models which have been adapted in plenty of applications. In cryptography, CA is widely used as a part of pseudo-random number generator (PRNG) algorithm. It also have been adapted as nonlinear element in block cipher such as in Keccak [3]. Basically, CA is characterized by the lattice cells of finite dimension which evolve in time synchronously according to the local rule and state of neighborhoods. The most basic CA was characterized briefly by Stephen Wolfram [1] known as elementary cellular automata (ECA) that is build up with one dimensional CA with three neighborhood and two states. Formally, it can be defined as follows,

$$C = (Q, r, f) \tag{6}$$

where $C = \{q_i^t\}$ for $i \in \mathbb{N}$ and $t = [0, T]$ that represent the set of interconnected cells arranged in regular manner as can be seen in Fig.1. Q is the set of states for each cell and r specified the size of neighborhood. ECA only have two states that is represented by binary variables 1 or 0. Each cell q_i evolve in discrete time steps according to local function, f which also known as rule. In details, the state at q_i^{t+1} is determined by the set of neighborhoods which contain the cell q_i^t itself together with the right cell q_{i+1}^t and the left cell q_{i-1}^t. Thus, the next time $t+1$ state q_i^{t+1} can be expressed as a function $q_i^{t+1} = f(q_{i-1}^t, q_i^t, q_{i+1}^t)$. Fig.1 shows the configuration of neighborhood to determine the next time $t+1$ state in a highlighted cell.

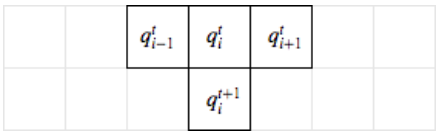

Fig. 1 ECA neighborhood configuration

As each state has two possible values (0 or 1), then there will be 2^r possible combinations for r neighborhoods configuration which also yield the same amount of possible output. Hence, if the neighborhood configuration is arranged in lexicographically binary order, there will be a total of 2^{2^r} distinct local rule f can be created. In [1], these rules known as ECA has been studied extensively in which all $2^{2^3} = 256$ rules was given a reference name that is according to decimal representation of output f arrange in lexicographically order. Table 1 shows two examples of rules with its neighborhood configuration and the output state. The first row represents the neighborhood configuration of left, center, and right state. If all states at time t value is 1 (the last neighborhood configuration), then the state of $t+1$ would be 0 for rule 90 and 1 for rule 150. It follows for all possible configuration. Despite the lookup table as represented in Table 1, each rule may also be represented as combination logic as follows,

$$\begin{aligned} &\text{Rule 90} : q_i^{t+1} = f(q_{i-1}^t, q_i^t, q_{i+1}^t) = q_{i-1}^t \oplus q_{i+1}^t \\ &\text{Rule 150}: q_i^{t+1} = f(q_{i-1}^t, q_i^t, q_{i+1}^t) = q_{i-1}^t \oplus q_i^t \oplus q_{i+1}^t \end{aligned} \tag{7}$$

In this paper, the form represented in Eq.7 has a great usage compared to lookup table as represented in Table 1. Both rules are categorized as linear rule as it only involves XOR operation. If there is involvement of AND operation, the rule is a nonlinear rule.

Table. 1 Lookup Table for Rule 90 and rule 150

r configuration	Rule 90	Rule 150
000	0	0
001	1	1
010	0	1
011	1	0
100	1	1
101	0	0
110	1	0
111	0	1

For practicality, the CA cell array C is set to be finite with length L. This implementation leads to the boundary condition problem as the state at the boundary does not have complete neighborhoods configuration. There are several types of boundary condition (BC) introduced to cater the problem. Periodic BC assume C to have ring shape in which the last state q_L^t is proceeds by the first state q_1^t while the first state precedes by the last state. Another is null BC where it simply set zero value for both end BC ($q_0^t = q_{L+1}^t = 0$). Some type of BC that is rarely in use such as fixed constant BC and mirror BC. Another important term in CA that is useful in this paper is uniform CA (UCA) and hybrid CA (HCA). UCA is composed of the same rule for each cell i while HCA may combine more than two rules for a single cell array C.

III. RELATED WORK

The integration of CA in cryptography algorithm is not a new as it has been integrated for decades ago. In [4] and [5], the author propose a stream cipher using ECA rule 30 as it has been classified in class 3 ECA which has chaotic dynamic. However, it was successfully attacked by [6] as the random number generated by the rule 30 has high correlation which is not a good properties for a stream cipher. Another implementation was proposed by [7] by using two combination of linear CA rule. Rule 90 and 150 combined together in linear hybrid cellular automata (LHCA) to generate maximum cycle of iteration. More recent work on CA integration with stream cipher can be seen in [8] where the author combine the algorithm of CA with nonlinear feedback shift registers (NFSR). Besides stream cipher, CA also adapted in block cipher algorithm in which it become the nonlinear primitive of the algorithm or the entire cryptography scheme itself. A type of CA which is called as reversible cellular automata has been given a spotlight due to its reversible properties that is very suitable with block cipher. In [9], RCA is describe to have unique predecessor and successor in its iteration which is very ideal in block cipher to reverse the encryption process. The authors also proposed a complete algorithm of block cipher which composed of four different CA that is either UCA or RCA. Related work that depend on RCA as the main body of cryptography algorithm is presented in [10] where the process of encryption and decryption is done in parallel. Besides, the author also employed 2nd order RCA schemes in which the next iteration state depends on two predecessor iteration. This scheme was claimed are always reversible even the basic rule CA is not. Similar research that implement 2nd order RCA can be found in [11]. Besides the main algorithm, CA is no exception as the construction of substitution box (S-Box). Construction method of S-Box using CA can be classified as a heuristic method. In [12], iteration of ECA for certain rule acts as the output of the S-Box. The authors listed out the best S-Box using selected rule and number of iterations. The same author extend that idea and proposed the construction of dynamic S-Box for AES and DES in [12]. Low power S-Box architecture was presented in [13] where the S-Box is constructed using 2nd order programmable RCA without storing the S-Box as lookup table as has been done in conventional method. Similar research can be found in [14]. Meanwhile in [15], he authors presented the search of good S-Box using CA and genetic programming for S-Box size 4×4 and 5×5. The genetic programming was used to search for best CA rule in term of nonlinearity, differential uniformity, and the algebraic degree.

IV. EXPERIMENTAL SETUP

A. Linear Hybrid Cellular Automata

LHCA is a type of CA that combine more than two linear rules (local function f) in a single cell array C. Some combination of linear rules such as rule 90 and rule 150 are able to generate full cycle of iteration as mentioned in [16]. Here, full cycle means all possible non-zero binary sequence of length L can be generated by CA iteration based on these rules combination. Hence, for any L, there will be $2^L - 1$ different binary sequences that can be generated. Further, this also means the rules combination may generate the permutation of $2^L - 1$ non-zero elements in decimal representation. Figure 2(a) and 2(b) show the example of full cycle and non-full cycle rule combination respectively with $L = 4$ and null BC.

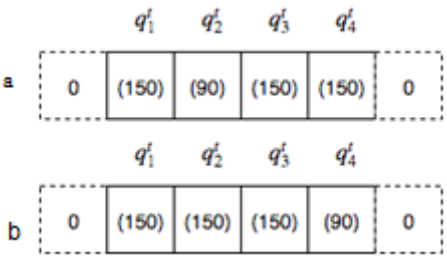

Fig. 2 LHCA configuration for $L = 4$

Figure 2 shows two different rule combinations in which Fig 2(a) has full cycle iteration while 2(b) has the opposite. To show whether it has full cycle, state transition diagram is used where each node represents one of the 2^L possible states in decimal representation. Figure 3(a) and 3(b) show the state transition diagram for configuration 2(a) and 2(b) respectively. It is clear that rule combination in 2(a) has full cycle for all non-zero elements while in 2(b), almost all elements are attracted to the limit cycle that only constitutes of seven elements.

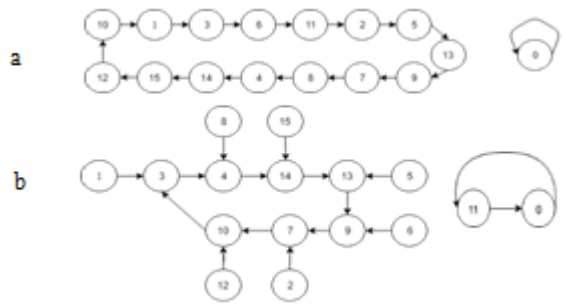

Fig. 3 State transition diagram for LHCA configuration in 2(a) and 2(b) respectively

From this point onward, we present the rule combination as binary vector which is known as the control vector v. Rule 90 is represented as 0 while rule 150 as 1. All the information will be summarized in a control vector as $v = [d_1 d_2 \dots d_L]$ where d_i represent the rule for cell i. Hence, there will be exactly 2^L different configurations that can be made including configuration of $[00 \dots 0]$ and $[11 \dots 1]$ that represent UCA of rule 90 and rule 150 respectively. At some point, we would also represent the control vector v as decimal form for easier representation and reference. In previous example, it is clear that the control vector v can be presented as $[1010]$ or 5 in decimal for example in Fig.2(a).

B. Transition Matrix M and Characteristic Polynomial

In LHCA, the transition of cell array C at time t to $t+1$ can be done graphically where we construct the lattice cell and update each cell according to the rule f. However, for linear rule like rule 90 and 150, this transformation can be done by constant transition matrix. The alternative is by using the transition matrix M in which the iteration of cell array C at time $t+1$ can be expressed as follows,

$$C^{t+1} = M \cdot C^t \qquad (8)$$

where $C^t = [q_1^t, q_2^t, \dots, q_L^t]^T$ and T indicates transpose vector. The transition matrix M can be defined by the next state equation of every cell i. Here, we refer to the combination logic in Eq.7 and example 2(a).

$$\begin{aligned} q_1^{t+1} &= q_1^t + q_2^t \\ q_2^{t+1} &= q_1^t + q_3^t \\ q_3^{t+1} &= q_2^t + q_3^t + q_4^t \\ q_4^{t+1} &= q_3^t \end{aligned}$$

The transition matrix M for example in Fig.2(a) is then can be generated as the following,

$$M = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

However, in general, the transition matrix M can be constructed by setting the control vector v as the diagonal of the tridiagonal matrix as follows,

$$M = \begin{bmatrix} d_1 & 1 & 0 & \text{L} & 0 & 0 & 0 \\ 1 & d_2 & 1 & \text{L} & 0 & 0 & 0 \\ 0 & 1 & d_3 & \text{L} & 0 & 0 & 0 \\ \text{M} & \text{M} & \text{M} & \text{O} & \text{M} & \text{M} & \text{M} \\ 0 & 0 & 0 & \text{L} & d_{L-2} & 1 & 0 \\ 0 & 0 & 0 & \text{L} & 1 & d_{L-1} & 1 \\ 0 & 0 & 0 & \text{L} & 0 & 1 & d_L \end{bmatrix} \qquad (9)$$

The matrix element $M_{1,L}$ and $M_{L,1}$ determine the BC. For null BC, the value is set to 0 as in Eq.11 while for periodic BC, both elements are set to 1. Based on work found in [17], the transition matrix M can be represented by characteristic polynomial $\Delta_k(x)$ and can be referred as CA polynomial. It can be calculated by using LHCA recurrence relation as follows,

$$\begin{aligned} &\Delta_{-1}(x) = 0, \\ &\Delta_0(x) = 1, \\ &\Delta_k(x) = (x + d_k)\Delta_{k-1}(x) + \Delta_{k-2}(x), \quad k \geq 1 \end{aligned} \qquad (10)$$

The following example shows the procedure of deriving characteristic polynomial of example in Fig. 2(a),

$$\begin{aligned} &\Delta_{-1}(x) = 0, \\ &\Delta_0(x) = 1, \\ &\Delta_1(x) = (x + d_1)\Delta_0(x) + \Delta_{-1}(x) = (x+1)(1) + 0 = x+1, \\ &\Delta_2(x) = (x + d_2)\Delta_1(x) + \Delta_0(x) = x^2 + x + 1, \\ &\Delta_3(x) = (x + d_3)\Delta_2(x) + \Delta_1(x) = x^3 + x, \\ &\Delta_4(x) = (x + d_4)\Delta_3(x) + \Delta_2(x) = x^4 + x + 1, \end{aligned} \qquad (11)$$

Therefore, the characteristic polynomial for example in Fig.2(a) is $x^4 + x + 1$. This is also one of the primitive polynomials over $GF(2^4)$. The polynomial p is irreducible if there is no non-constant polynomial a and b such that $p = ab$. It is primitive if it has α as a root that make up a set $\{1, \alpha^2, \alpha^3, \dots, \alpha^{q^n-2}\}$ which is equal to the set of non-zero elements of $GF(q^L)$. Moreover, for any LHCA with full cycle, the characteristic polynomial would be the primitive polynomial. This is based on the theory presented in Cattell & Muzio (1996a), where all primitive polynomial over $GF(2^L)$ are CA polynomial and each primitive polynomial representing exactly two LHCA of rule 90 and rule 150.

C. The S-Box Construction Algorithm

Our proposed S-Box construction are based on permutation of elements generated by LHCA iteration of rule 90 and rule 150. Previous section has clearly brief that some combination of these rules may generate full cycle of iteration which constitutes the permutation of non-zero elements. The zero element however will mapped to itself. Nevertheless, the permutation can be shifted by varying the initial condition of the iteration. Hence the value of nonlinearity, together with other security measure would also change as varying the initial condition lead to new S-Box configuration.

Due to that, we have tested for all combination rule with all possible initial condition. Hence within this section, two algorithms are presented. The first algorithm are dedicated to search for all possible rule combinations that generate full cycle for $L = 8$. The second one is to generate S-Box from all possible initial conditions for each combination searched by the first algorithm and evaluate the S-Box using the cryptography properties as mentioned in section 2. As a result, there will be $2N(p) \times (2^L - 1)$ possible S-Box with different security level that can be generated where $N(p)$ is the total number of primitive polynomials over $GF(2^L)$. Both algorithms are presented as follows.

Algorithm 1: Extracting all possible rule combination

INPUT: Rule combination length L, number of iterations itr, initial condition ic

OUTPUT: List of control vector v that generate full cycle, LIST

```
Set all possible list of control vector v of size 2^L
  For j from 1 to 2^L
    Construct matrix M based on control vector v(j)
    Set itr × L matrix CA for storing the iteration
    Set C^1 = CA(1) = ic
    For t from 1 to itr
      Set C^(t+1) = M · C^t
      Store C^(t+1) in CA(t + 1)
    End
    Set 2^L × 1 matrix DecSq(k) for storing decimal sequence
    For k from 1 to 2^L − 1
      Convert CA(k) into decimal representation
      Store the decimal into DecSq(k)
    End
    Sort DecSq(k) in descending order
    Check for permutation
    If DecSq(k) is a permutation
      Write v(j) in LIST
    End
  End
```

Algorithm 1 shows the procedure to extracting all possible rule combination by testing for all possible control vector v of length L. For simplicity, the initial condition ic is set to $[10000000]$ for all v. The extracted v that able to generate full cycle is then stored for further reference in algorithm 2.

Algorithm 2: Generate and Evaluation of S-Box

INPUT: List of control vector v LIST, Rule combination length L

OUTPUT: Table of S-Box Nonlinearity (N_g table) and Differential Uniformity (δ table)

```
Set empty L × L S-Box S
Set 0 as the first element in S
Set 2^L − 1 × L empty matrix for storing N_f value
For j from 1 to L
  Read v(j) from LIST
  For k from 1 to 2^L − 1
    Set itr = 2^L − 1
    Generate the CA sequence using line 3-9 algorithm 1 with ic = k and v(j)
    Convert CA iteration into decimal sequence
    Store the sequence in S
    Test the Nonlinearity N_g and differential uniformity δ
    Store N_g and δ
  End
End
```

Algorithm 2 shows the procedure to construct S-Box from the extracted v. All generated S-Box is then tested with cryptography properties as discussed in section 2. All results are presented in the next section together with some discussion and comparative analysis.

V. RESULT AND DISCUSSION

A. List of Control Vector Y

This section listed out all v with its respective primitive polynomial extracted by algorithm 1. Based on theorem introduced in [18], one primitive polynomial over $GF(2^L)$ represent two LHCA of rule 90 and 150. Hence, for $L = 8$, the number of primitive polynomial $p(x)$ is [19] which resulted in 32 different possible rule combination. In Table 2, we list out all pair control vector v_1 and v_2 that represent the same primitive polynomial $p(x)$.

Table. 2 Rule combination with its respective primitive polynomial

v_1	v_2	$p(x)$
01100000	00000110	$x^8 + x^4 + x^3 + x^2 + 1$
11110000	00001111	$x^8 + x^6 + x^5 + x^3 + 1$
01010100	00101010	$x^8 + x^7 + x^3 + x^2 + 1$
10110100	00101101	$x^8 + x^6 + x^5 + x^2 + 1$
01101100	00110110	$x^8 + x^6 + x^3 + x^2 + 1$
10011100	00111001	$x^8 + x^6 + x^5 + x^4 + 1$
10100010	01000101	$x^8 + x^7 + x^5 + x^3 + 1$
11010010	01001011	$x^8 + x^6 + x^5 + x + 1$
11011010	01011011	$x^8 + x^7 + x^6 + x^3 + x^2 + x + 1$
10111010	01011101	$x^8 + x^7 + x^6 + x + 1$
11111010	01011111	$x^8 + x^5 + x^3 + x + 1$
11101110	01110111	$x^8 + x^5 + x^3 + x^2 + 1$
11001001	10010011	$x^8 + x^6 + x^4 + x^3 + x^2 + x + 1$
11010101	10101011	$x^8 + x^7 + x^6 + x^5 + x^4 + x + 1$
11010011	11001011	$x^8 + x^7 + x^6 + x^5 + x^2 + x + 1$
11110111	11101111	$x^8 + x^7 + x^2 + x + 1$

It can be seen from Table 2 that the rule combination pair v_1 and v_2 for any $p(x)$ have the common relationship in which there are in reverse order of each other. As mention in [18], both v with the same $p(x)$ have the same structure except the index of all cell relabeled in the opposite order. If we set the initial condition ic for both v in opposite binary configuration, say $[10000000]$ and $[00000001]$ for v_1 and

υ_2 respectively, then the evolution of next time state q_i^{t+1} pattern would follow to have the opposite pattern for every t. This lead the generated S-Box to have the same set of coordinate function g_i but arranged in the opposite order. As we start the iteration with invariant initial condition ic, this opposite pattern is not clearly appear, however all generated S-Box from both υ would have similar Boolean structure. It leads to another result which is the distribution of high and low N_g S-Box for υ_1 and υ_2 would be the same.

B. Nonlinearity

Table 3 tabulate the N_g distribution for all listed υ in Table 2. As mentioned before, for easier reference, we list the control vector υ in decimal representation with the same arrangement as in Table 2. Based on Table 3, the value of N_g spread out ranging from the lowest of 92 up to maximum of 108.

There are 16 pairs of υ represented by the same $p(x)$ that have the same N_g distribution as can be seen from the table. This is due to the reason that stated before. After all, about 65% of all constructed S-Box achieve at least $N_g = 100$ and only a useful four out of 8160 constructed S-Box that achieve the value of 108. Majority of the constructed S-Box have the value N_g of 100, 102 and 104. Only two rules combination that able to generate maximum N_g of 108 which are $\upsilon = [01101100]$ and $\upsilon = [00110110]$ that represented as 54 and 108 in decimal notation respectively. From all generated S-Box, we have selected 10 different S-Box which show high value of N_g. Table 4 shows in detail the value N_g of each coordinate function g_i for those good S-Box.

Table. 3 N_g distribution for all υ

υ_1	υ_2	N_g								
		92	94	96	98	100	102	104	106	108
6	96	0	0	0	0	80	39	121	15	0
15	240	0	0	0	0	83	76	92	4	0
42	84	31	0	49	72	55	33	10	5	0
45	180	0	0	0	0	0	56	129	70	0
54	108	0	0	0	0	0	59	129	65	2
57	156	0	0	0	0	31	128	83	13	0
69	162	0	0	0	0	93	132	28	2	0
75	210	31	0	49	40	101	21	10	3	0
91	218	0	0	0	0	83	120	50	2	0
93	186	0	0	8	0	32	151	58	6	0
95	250	0	0	0	0	141	59	53	2	0
119	238	0	0	0	0	95	84	74	2	0
147	201	0	0	8	0	57	51	96	43	0
171	213	0	0	8	55	0	45	103	44	0
203	211	0	0	0	0	90	126	31	8	0
239	247	0	0	8	0	85	75	83	4	0
Total		**124**	**0**	**260**	**334**	**2052**	**2510**	**2300**	**576**	**4**

Table. 4 Details of selected good S-Box

Ref	υ	ic	g_1	g_2	g_3	g_4	g_5	g_6	g_7	g_8	Average
S1	54	72	108	108	108	108	108	108	108	108	108
S2	108	18	108	108	108	108	108	108	108	108	108
S3	54	107	108	108	108	108	108	108	108	108	108
S4	108	214	108	108	108	108	108	108	108	108	108
S5	15	17	108	112	106	108	106	106	106	106	107.25
S6	240	136	106	106	106	106	108	106	112	108	107.25
S7	15	56	106	106	106	106	108	106	112	108	107.25
S8	240	28	108	112	106	108	106	106	106	106	107.25
S9	45	90	110	108	112	108	106	106	106	106	107.75
S10	180	90	106	106	106	106	108	112	108	110	107.75

The initial condition is in decimal representation. Based on that table, we have selected the generated S-Box that have the value of range from 106 to 112. The arrangement in Table 4 are made in pair, in which both S-Box have opposite arrangement of (S1 & S2, S3 & S4 and so on). Six out of 10 S-Box have maximum for at least one comparable to original AES S-Box. However, those S-Box also contain the minimum value of $N_g = 106$. .

C. Differential Uniformity

As for differential uniformity δ, the value ranges from maximum 14 to minimum 8. Lowest value are more favorable in which $\delta = 2$ represent almost perfect nonlinear (APN) Boolean function. Original AES S-Box achieve $\delta = 4$ and known as differential 4-uniform function. Basically, each rule combination v with the same $p(x)$ would have the same value of δ for all possible initial condition ic. The following table (Table 5) shows the result of differential uniformity δ for each rule combination.

Table. 5 δ for all rule combination δ

v_1	v_2	δ
6	96	8
15	240	14
42	84	8
45	180	8
54	108	8
57	156	8
69	162	10
75	210	8
91	218	8
93	186	8
95	250	10
119	238	10
147	201	8
171	213	8
203	211	8
239	247	8

D. Algebraic Degree

We only test for algebraic degree for the selected S-Box in table 4. All S-Box from the list achieve maximum value of algebraic degree which is comparable to the one in original AES construction. The summarization of algebraic degree is shown in Table 6.

E. Comparative Analysis

Finally, we conduct a comparative analysis in order to compare our S-Box performance with other construction. We gather the work of S-Box construction using various of work including CA and other heuristic methods. Table 6 list out selected work for comparison in term of nonlinearity N_g, differential uniformity δ and the algebraic degree.

Table. 6 Comparative Analysis

S-Box	N_g	δ	AD	Technique
S1	108	8	7	LHCA with $v = 54$ and $ic = 72$
S2	108	8	7	LHCA with $v = 108$ and $ic = 18$
S3	108	8	7	LHCA with $v = 54$ and $ic = 107$
S4	108	8	7	LHCA with $v = 108$ and $ic = 214$
S5	106	14	7	LHCA with $v = 15$ and $ic = 17$
S6	106	14	7	LHCA with $v = 240$ and $ic = 136$
S7	106	14	7	LHCA with $v = 15$ and $ic = 56$
S8	106	14	7	LHCA with $v = 240$ and $ic = 28$
S9	106	8	7	LHCA with $v = 45$ and $ic = 90$
S10	106	8	7	LHCA with $v = 180$ and $ic = 90$
AES	112	4	7	Finite Field Inversion
(Ahmad, Bhatia, & Hassan, 2015)	106	NP	NP	Ant Colony Optimization Based Scheme
(Zhang, Chen, Chen, Xu, & Hu, 2018)	108	10	7	I-Ching operator
(Ye & Zhimao, 2018)	104	10	7	Six-Dimensional Fractional Lorenz-Duffing System
(Ahmed, Zolkipli, & Ahmad, 2018)	106	10	7	Firefly algorithm and Discrete chaotic map
(Gangadari & Ahamed, 2016)	108	18	7	Programmable 2nd order RCA

NP-Not Provided

VI. CONCLUSION

Our new proposed S-Box generally are based on hybrid CA that produced full cycle of iteration. The construction presented in this paper manage to achieve the value of nonlinearity with maximum of 108 and minimum of 92. Besides, the method also manage to generate S-Box with differential uniformity δ of 8 and maximum algebraic degree of 7. Compared to other method in the class, this method may have better performance S-Box in term of maximum nonlinearity as we can detect single coordinate function with maximum value of 112, that is comparable to AES S-Box. In the future, this method can be extend to other size of S-Box and with better planning and algorithm, dynamic S-Box could also be implemented using the proposed methods. This is due to the variability of initial condition and control vector together with simplest algorithm compared to original one that implement finite field inversion.

ACKNOWLEDGMENT

The authors would like to express special thanks to Ministry of Education Malaysia for supporting and financing this research project under the Transdisciplinary Research Grant Scheme (TRGS/FKAB/50216/59). The author would also thanks to anonymous reviewers for their helpful comment and suggestion.

REFERENCES

1. Wolfram, S. (2002). A New Kind of Science. Champaign: Wolfram media.
2. Biham, E., & Shamir, A. (1990). Differential Cryptanalysis of DES-like Cryptosystems. In Advances in Cryptology - CRYPTO '90, 10th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1990, Proceedings (Vol. 537, pp. 2–21).
3. Bertoni, G., Daemen, J., Peeters, M., Assche, G. V. Van, Guido, B., Joan, D., … Gilles, V. A. (2011). The keccak reference. Submission to NIST (Round 3), 1–14.
4. Wolfram, S. (1986a). Cryptography with Cellular Automata. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics).
5. Wolfram, S. (1986b). Random sequence generation by cellular automata. Advances in Applied Mathematics (Vol. 7).
6. Meier, W., & Staffelbach, O. (1991). Analysis of Pseudo Random Sequence Generated by Cellular Automata.
7. Serra, M., Slater, T., Muzio, J. C., & Miller, D. M. (1990). The Analysis of One-Dimensional Linear Cellular Automata and Their Aliasing Properties. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 9(7), 767–778.
8. Raut, L., & Hoe, D. H. K. (2013). Stream cipher design using cellular automata implemented on FPGAs. 45th Southeastern Symposium on System Theory, 146–149.
9. Seredynski, M., & Bouvry, P. (2005). Block Cipher based on Riversible Cellular Automata. New Gener Comput, 23(3), 245–258.
10. Kamel Mohamed, F. (2014). A parallel block-based encryption schema for digital images using reversible cellular automata. Engineering Science and Technology, an International Journal, 17(2), 85–94.
11. Li, K., Sun, M., Li, L., & Chen, J. (2017). Image Encryption Algorithms Based on Non-uniform Second-Order Reversible Cellular Automata with Balanced Rulesn. In Intelligent Computing Theories and Application. ICIC 2017. Lecture Notes in Computer Science (Vol. 10362, pp. 445–455). Springer, Cham.
12. Szaban, M., & Seredynski, F. (2008). Application of cellular automata to create S-box functions. IPDPS Miami 2008 - Proceedings of the 22nd IEEE International Parallel and Distributed Processing Symposium, Program and CD-ROM.
13. Gangadari, B. R., & Ahamed, S. R. (2016). Low Power S-Box Architecture for AES Algorithm using Programmable Second Order Reversible Cellular Automata: An Application to WBAN. Journal of Medical Systems, 40(12).
14. Gangadari, B. R., & Rafi Ahamed, S. (2016). Design of cryptographically secure AES like S-Box using second-order reversible cellular automata for wireless body area network applications. Healthcare Technology Letters, 3(3), 177–183.
15. Picek, S., Mariot, L., Yang, B., Jakobovic, D., & Mentens, N. (2017). Design of S-boxes defined with cellular automata rules. ACM International Conference on Computing Frontiers 2017, CF 2017, 409–414.
16. Cattell, K., & Muzio, J. C. (1996b). Synthesis of one-dimensional linear hybrid cellular automata. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 15(3), 325–335.
17. Serra, M., Slater, T., Muzio, J. C., & Miller, D. M. (1990). The Analysis of One-Dimensional Linear Cellular Automata and Their Aliasing Properties. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 9(7), 767–778.
18. Cattell, K., & Muzio, J. C. (1996a). Analysis of One-Dimensional Linear Hybrid Cellular Automata over GF(q). IEEE Transactions on Computers, 7(July 1996).
19. Mullen, G. L., & Panario, D. (2013). Handbook of finite fields. CRC Press.