International Journal of Computer and Communication Technology

Volume 3 | Issue 2 Article 7

April 2012

Applications of Two Dimensional Cellular Automata rules for Block Cipher in Cryptography

Sambhu Prasad Panda
C V Raman Computer Academy, Bidyanagar, Mahura, Janla Bhubaneswar-752054, Orissa, India, sambhu.prasad.panda@gmail.com

Madhusmita Sahu
C V Raman Computer Academy, Bidyanagar, Mahura, Janla Bhubaneswar-752054, Orissa, India, madhu_sahu@yahoo.com

Manas Kumar Swain
C V Raman College of Engineering3 Bhubaneswar-752054, mkswain2004@yahoo.co.in

Applications of Two Dimensional Cellular Automata rules for Block Cipher in Cryptography

Sambhu Prasad Panda [1], Madhusmita Sahu [2], Manas Kumar Swain [3]
C V Raman Computer Academy [1,2] , C V Raman College of Engineering[3]
Bhubaneswar-752054
sambhu.prasad.panda@gmail.com [1], madhu_sahu@yahoo.com [2], mkswain2004@yahoo.co.in [3]

Abstract--**Cellular Automaton is an idealized parallel processing machine which is an array (1-D, 2-D) of numbers or symbols called cell values together with an updating rule. A cell value is updated based on this updating rule, which involves the cell value as well as other cell values in a particular neighborhood. A fundamental objective of cryptography is to enable two people to communicate over an insecure channel (a public channel such as internet) in such a way that any other person is unable to recover their message (called the plaintext) from what is sent in its place over the channel (the cipher text). The transformation of the plaintext into the cipher text is called encryption, or enciphering. The transformation of the cipher text into the plaintext is called decryption, or deciphering. In this paper we present a new encryption and decryption algorithm based on the linear (periodic boundary-PB) and nonlinear Cellular Automata rules. First we apply PB CA rules to plain text and key. The result of both plain text and key is XORed. Then the result of XOR operation is fed to substitution box(S-box) and again PB CA rules are applied for exchange and shift operations. At the end Complement operation is applied for encryption of plain text. The decryption process is carried out just similar to encryption but in the reverse way. Both the process of encryption and decryption is performed for 8 numbers of rounds in order to avoid the dependency between the plain text and cipher text.**

1. INTRODUCTION

Cellular Automata were originally proposed by John von Neumann as formal models of self reproducing organisms. The structure studied was most on one and two dimensional infinite grids, through higher dimensions were considered. Later physicists and biologists began to study cellular automata for the purpose of modeling in their respective domains. In the present era, cellular automata are being studied from many widely different angles, and the relationship of these structures to existing problems are being constantly sought and discovered. Cellular automata have wide area of applications such as in cryptography, computer graphics, neural network etc.

Cryptography is an important and vital application in security, defense, medical, business and many other application areas. The effective measure of a cryptosystem is how long it can be used to encrypt and decrypt messages without the 'key' being broken using cellular automata (CA) rules. A class of cellular automata (CA) based encryption algorithms presents a particular promising approach to cryptography, since the initial state of the CA is the key to the encryption, evolving a complex chaotic system from this 'initial state' which cannot be predicted. However, software implementations of CA cryptography have the disadvantage that special purpose hardware can, be applied to break the code. The pseudo randomness brought by the available mathematical systems should not be blindly trust. The designing techniques developed by cryptographic community are always optimal. Here we use the symmetric key cryptosystem where the sender and the receiver share only one key. The sender uses the key to encrypt the message. The actual message is called plain text and the encrypted message is called cipher text. Again the receiver decrypt the message using the same key (used by sender) to find the actual message.

The remainder of the paper is organized as follows. Section 2 introduces the concept of Cellular Automata. In Section 3, we discuss some works of cryptography applied to one dimensional Cellular Automata. Section 4 describes our encryption and decryption algorithm.

2. BACKGROUND

2.1. Boolean Function and its properties

A **Boolean algebra** is an algebraic structure (a collection of elements and operations on them obeying defining axioms) that captures essential properties of both set operations and logic operations. Specifically, it deals with the set operations of intersection, union, complement; and the logic operations of AND, OR, NOT.

Any Boolean Function f in n variables is defined as a map

$$f:\{0,1\}^n \longrightarrow \{0,1\}$$

There are 2^{2^n} boolean functions out of which 2^n are linear boolean functions and $2^{2^n} - 2^n$ are nonlinear boolean functions.

For example:- n=2,
Then $f:\{0,1\}^2 \longrightarrow \{0,1\}$
This implies $f: \{0,1\}x \{0,1\} \longrightarrow \{0,1\}$
and $f : \{00,01,10,11\} \longrightarrow \{0,1\}$
Then the total numbers of possible functions are shown in Figure 1.

Dec value	A	B	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
1	0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
2	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
3	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Figure 1: Boolean Functions

Thus $2^{2^n} = 2^4 = 16$ boolean functions in two-variables appears.

The boolean functions are classified into two categories

- Group of linear functions.
- Group of non-linear functions.

A boolean function f in n-variable is said to be linear if it satisfies the following linearity property:
$f(x+y)=f(x)+f(y)$,
where $x = (x_1,x_2,---,x_n)$ and $y = (y_1,y_2,---,y_n)$
There are 2^n linear boolean functions in n-variables.
Those are

- The zero function $f(x_1,x_2,---,x_n) = 0$ is always a linear function and is termed as Rule 0.
- $f(x_1,x_2,---,x_n) = x_i$, (i= 1,2,3,---,n) are n-linear boolean functions which are termed as fundamental linear rules.
- The combinations of these n-linear functions taking some or all at a time, give rest of the 2^n-1 linear boolean functions.

For n variables, there are 2^{2^n} boolean functions, out of which 2^n linear boolean functions and rest are non-linear boolean functions.

From Figure 1, it is clear that out of 16(=2^4) boolean functions (f_0, f_1, f_2, …, f_{15}) in 2 variables (A, B), there are 4(=2^2) linear Boolean functions (f_0,f_6,f_{10} and f_{12}) and 12 (=2^4-2^2) nonlinear Boolean functions (f_1,f_2,f_3,f_4,f_5,f_7,f_8,f_9,f_{11},f_{13},f_{14},f_{15}).

Functions of degree at most one are called *affine* functions. An affine function with constant term equal to zero is called a *linear* function. The set of all n-variable affine functions is denoted by A(n). The nonlinearity of a n-variable function f is the minimum distance between the affine functions f and g which is given by

$$nl(f) = \min_{G \quad A(n)} (d(f, g))$$

All affine functions are non linear.

2.2. Cryptographic criteria for Boolean functions

Use of nonlinear Boolean functions in encryption algorithm creates substitution or confusion whereas use of linear boolean functions in encryption algorithm creates permutation or diffusion. A secure cryptographic algorithm should contain both nonlinear (substitution or confusion) and linear (permutation or diffusion) Boolean functions.

The following factors are considered as important properties of Boolean functions.

- **Balancedness:** A Boolean function must output zeroes and ones with the same probabilities.
- **Good non-linearity:** The Boolean function must be at the sufficiently high distance from any affine function.
- **High algebraic degree:** The Boolean function must be at high algebraic degree.
- **Simple implementation in hardware:** Hardware implementation should be very simple.

2.3. Cellular Automata (CA)

The increasing prominence of computers has led to a new way of looking at the world. This view sees nature as a form of computation. That is, we treat objects as simple computers, each obeying it's own set of laws.

The "Cellular automaton" extends this analogy to provide a way of viewing whole populations of interacting "cells", each of which itself a computer (automaton). By building appropriate rules into a cellular automaton, we can simulate many kinds of complex behavior, from turbulence in fluids to pattern in biological growth.

Cellular Automata (CA) were originally conceived by Ulman and Von Neumann in the 1940. CA provides a formal framework for investigating the behavior of complex and extended system. Cellular automata are simple mathematical idealizations of natural system. This is an idealized parallel processing machine, which is an array (1-D, 2-D, 3-D) of numbers or symbols called cell values together with an updating rule. A cell value is updated based on this updating rule, which involves the cell value as well as other cell values in a particular neighborhood. Figure 2 and Figure 3 show the structure of one and two dimensional cellular automata neighborhood cells respectively.

Figure 2: 1-D neighborhood

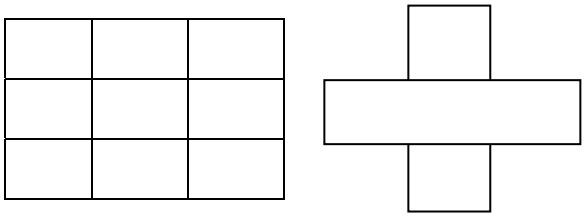

Figure 3: 2-D neighborhood

2.3.1. CA rules as boolean functions

A one-dimensional cellular automaton (CA) consists of two things: a row of "cells" and a set of "rules". Each of the cells can be in one of several "states". The number of possible states depends on the automaton. Think of the states as colors. In a two-state automaton, each of the cells can be either black or white. Over time, the cells can change from state to state. The cellular automaton's **rules** determine how the states change. It works like this: When the time comes for the cells to change state, each cell looks around and gathers information on its neighbors' states. (Exactly which cells are considered "neighbors" is also something that depends on the particular CA.) Based on its own state, its neighbors' states, and the rules of the CA, the cell decides what its new state should be. All the cells change state at the same time.

Two-dimensional cellular automaton consists of an infinite (or finite) grid of cells, each in one of a finite number of states. Time is discrete and the state of a cell at time t is a function of the states of its neighbors at time t-1. For cryptographic application 8-neighborhood CA rules are newly introduced. Table 1 shows all the rules of two dimensional cellular automata.

Table 1: 8-neighborhood CA rules

64	128	
32	1	2
16	8	4

In 2-D eight neighborhood CA the next state of a particular cell is affected by the current state of itself and seven cells in its nearest neighborhood (figure). Such dependencies are accounted by various rules. The central cell represents the current cell (i.e. the cell being considered) and all other cells represent the seven neighbors of that cell. The number within each cell represents the rule number(i.e. Rule 1, Rule 2, Rule 4, Rule 8, Rule 16, Rule 32, Rule 64 and Rule 128) characterizing the dependency of the current cell on that particular neighbor only. These 8 rules are called fundamental rules of cellular automata and are known as linear rules of cellular automata. In case the cell has dependency on two or more neighboring cells, the rule number will be the arithmetic sum of the numbers of the relevant cells, which gives the linear rules of cellular automata. So XOR operation is also linear rules of CA.

For example the 2D CA rule 170 (=2+8+32+128) refers to the 4 neighborhood dependency of the central cell on right, bottom, left and top. The number of such rules is ${}^8C_0+{}^8C_1+....+{}^8C_8=256$.

2.3.2. Definitions

Null Boundary: A null boundary CA is the one in which the extreme cells are connected to logic - 0 states.

Periodic boundary: A periodic boundary (PB) CA is the one in which the extreme cells are connected to each other.

3. RELATED WORK

Carlet [1] performed study on Boolean functios for cryptography. He utilized cryptographic criteria to identify the linearity (diffusion) and nolinearity (confusion) operations involved in Data Encryption Standard (DES) and Advanced Encryption Standard (AES) algorithm. Wolfram [2] applied the one-dimensional cellular automata rules on stream cipher for security. Maitra et. al. [3] applied the method of generating key stream sequences for stream ciphers by combining the outputs of several linear feedback shift registers (LFSR) using a combined Boolean function.

All these works were based on one dimensional cellular automata for stream cipher. Also very few works have been carried out on two dimensional cellular automata for block cipher [9].

4. Encryption & Decryption algorithm using cellular Automata Rule

4.1. Introduction

Cryptography is the heart of security. If we need to create privacy, we need to encrypt our message at the sender site and decrypt it at the receiver site. There are different types of encryption and decryption algorithm like substitution, monoalphabetic or polyalphabetic, Vigenere cipher, DES, AES etc. We use Two-dimensional Cellular Automata (CA) rule for encryption and decryption algorithm. Here both the sender and receiver use the same key.

4.2. Overall Structure

Table 2: Parameter of Algorithm

Key size(words/bytes/bits)	4/16/128
Plain text block size(words/bytes/bits)	4/16/128
Number of rounds	8
Round key size(words/bytes/bits)	4/16/128

Table 2 shows the parameters of the encryption and decryption algorithm. This algorithm contains substitution (non-linear cellular automata rule), permutation (linear cellular automata rule), complement (non-linear cellular automata rule), and XOR (linear cellular automata rule) operations. In crypto system, use of non-linear rule is more secure then use of linear rule. But if we take the entire non-linear CA rule for encryption, then we get more secure algorithm. Nobody can attack the message. But it is a wrong idea, because if we apply the entire non-linear CA rules then we can not decrypt the message in reverse way of encryption algorithm. If the encryption

and decryption algorithms differ, then extra space is needed for the decryption. Also, whether the two algorithms are same or not, there may be timing difference between encryption and decryption. Figure 4 shows the Encryption & Decryption algorithm applying CA rule. Figure 5 shows the one round of encryption algorithm.

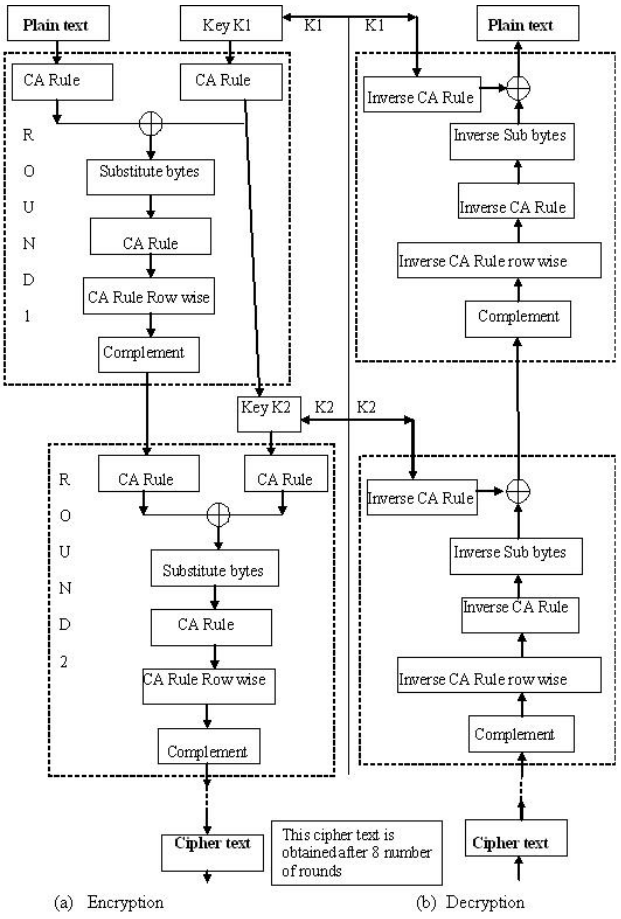

Figure 4: Encryption and Decryption algorithm applying CA rule

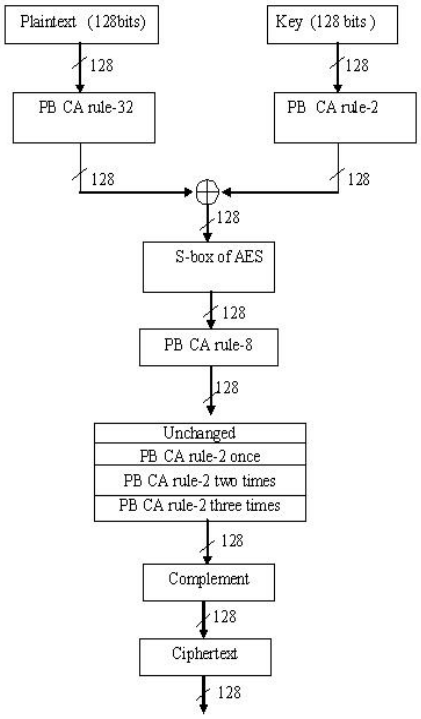

Figure 5: one-round of encryption algorithm applying CA rule

4.3. Encryption Algorithm

4.3.1. XOR Operation and PB CA rule2 and CA rule32

We take the length of the plain text as 128 bits and the length of key as 128 bits. First we convert the plain text as 4x4 matrix with each cell containing 1 bytes (= 8 bits). Now we apply Cellular Automata (CA) Rule-32 on each bit. Similarly we write the 128-bits key in 4x4 matrix and apply CA rule 2 on each bit. Table 3 shows the 4x4 matrix of 128 bits plaintext and Table 4 shows the 4x4 matrix of 128 bits key.

Table 3: 128- bits plaintext

A_1	A_2	A_3	A_4
A_5	A_6	A7	A_8
A_9	A_{10}	A_{11}	A_{12}
A_{13}	A_{14}	A_{15}	A_{16}

Table 4: 128-bits key

K_1	K_2	K_3	K_4
K_5	K_6	K7	K_8
K_9	K_{10}	K_{11}	K_{12}
K_{13}	K_{14}	K_{15}	K_{16}

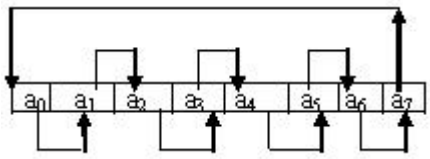

Figure 6: CA rule-32 Periodic Boundary on each cell of plaintext

Figure 6 shows the use of CA rule-32 Periodic Boundary on each cell of plaintext. For example, in the following matrix, the values in the first column have been shifted to second column, values in the second column have been shifted to third column and so on and values in the last column have been shifted to fist column after applying periodic boundary (PB) CA rule-32 to each value in the cell of the matrix.

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \xrightarrow{\text{PB CA rule-32}} \begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

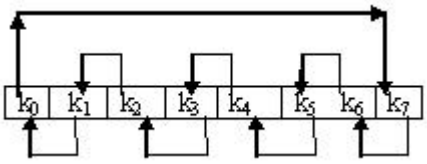

Figure 7: CA rule-2 Periodic Boundary on each cell of key

Figure 7 shows the use of CA rule-2 Periodic Boundary on each cell of key. For example, in the following matrix, the values in the second column have been shifted to first column, values in the third column have been shifted to second column and so on and values in the first column have been shifted to last column after applying periodic boundary (PB) CA rule-2 to each value in the cell of the matrix.

$$\begin{bmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix} \xrightarrow{\text{PB CA rule-2}} \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Hence PB CA rule-32 and PB CA rule-2 are reversible. So we encrypt the message by applying PB CA rule-32 and we decrypt the message by applying PB CA rule-2.

After applying CA rule-32 on each bit of plaintext, we get again 4 x 4 matrix and each cell contain 8-bits. Similarly we apply CA rule-2 on each bit of key. Again we get 4 x 4 matrix and each cell contain 8-bits. Here we change the position of the bit which is known as permutation. Rule-2 and Rule-32 both are linear rule of CA. After applying Rule-32 on plaintext and applying Rule-2 on key, we have two 4 x 4 matrix and each cell of the matrix contains 8-bits. To combine these two matrices we apply bitwise XOR operation. In bitwise XOR operation both the bits of the matrices are same then we write zero, otherwise write one. Now we get 4 x 4 matrices and each cell contains 8-bits. Since we have taken the secret key (K1) at the beginning of the encryption process, the encryption algorithm is found to be more secure.

4.3.2. Substitute Bytes Transformation(S-Box)

Forward and Inverse Transformations: The forward substitute byte transformation, called SubBytes, is a simple table lookup below. This table is same as AES table. In future we try to develop a S-box which is balanced, high non-linear, high algebraic degree. But AES S-box is balanced, non-linear, and high algebraic degree. AES defines a 16 x 16 matrix of byte values, called S-box that contains a permutation of all possible 256 8-bit values. Each individual byte of state is mapped into a new byte in the following way. The leftmost 4 bits of the byte are used as a row value and the rightmost 4 bits are used as a column value. These row and column values serve as indexes into the S-box to select a unique 8-bit output value. For example, the hexadecimal value {95} references row 9, column 5 of the S-box, which contains the value {2A}. Accordingly, the value {95} is mapped into the value {2A}.

Figure 8 shows an example of the SubBytes transformation. Figure 9 shows the general form of Substitute byte transformation.

EA	04	65	85
83	45	5D	96
5C	33	98	B0
F0	2D	AD	C5

→

87	F2	4D	97
EC	6E	4C	90
4A	C3	46	E7
8C	D8	95	A6

Figure 8: SubBytes transformation

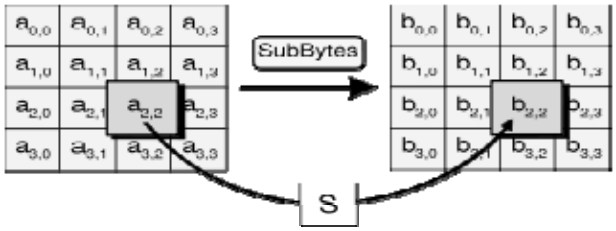

Figure 9: General form of Substitute byte transformation

4.3.3.PB CA rule-8 , CA rule1(unchanged) and Cyclic shifts using PB CA rule2

Applying periodic boundary CA rule-8 to the output bits of S-box, the first row becomes second row, second row becomes third row, third row became fourth row and fourth row became first row. This rule is also linear rule. Only the matrix can be permutated and we obtain a 4 x 4 matrix. After getting 4 x 4 matrix, first row remains unchanged, apply CA rule-2 on second row, apply rule-2 two-times on third row, apply rule-2 three-times on fourth row. The output matrix again gives a 4 x 4 matrix. But in cryptography, only one dimensional Cellular Automata rule is applied till now. But here we apply two-dimensional Cellular Automata which gives more

security. Figure 10 shows the use of Periodic Boundary CA rule-8 to the matrix.

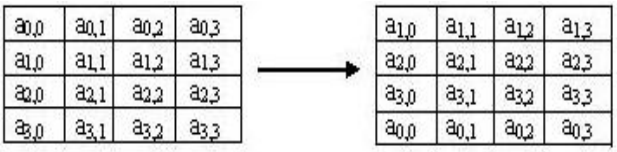

Figure 10: Periodic Boundary CA rule-8

For example, in the following matrix, the values in the second row have been shifted to first row, values in the third row have been shifted to second row and so on and values in the first row have been shifted to last row after applying periodic boundary (PB) CA rule-8 to each value in the cell of the matrix.

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \xrightarrow{\text{PB CA rule-8}} \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

For example, in the following matrix, the values in the first row have been shifted to second row, values in the second row have been shifted to third row and so on and values in the last row have been shifted to first row after applying periodic boundary (PB) CA rule-128 to each value in the cell of the matrix.

$$\begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix} \xrightarrow{\text{PB CA rule-128}} \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Hence PB CA rule-8 and PB CA rule-128 are reversible. So we encrypt the message by applying PB CA rule-8 and we decrypt the message by applying PB CA rule-128. Figure 11 shows the shifting of rows by PB CA rule-2.

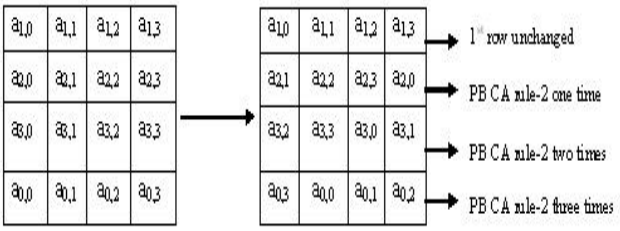

Figure 11: Shifting of rows by PB CA rule-2

For example, in the following matrix, the values in the first row have been unchanged, values in each cell of the second row have been shifted towards left by one cell with value in the first cell being shifted to last cell, values in each cell of the third row have been shifted towards left by two cells with values in the first and second cells being shifted to last but one and last cell respectively and so on after applying periodic boundary (PB) CA rule-2 twice to each value in the cell of the matrix.

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \xrightarrow[\text{(PB CA rule-2)}]{\text{shift rows}} \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

(once,twice)

For example, in the following matrix, the values in the first row have been unchanged, values in each cell of the second row have been shifted towards right by one cell with value in the last cell being shifted to first cell, values in each cell of the third row have been shifted towards right by two cells with values in the last and last but one cells being shifted to next to first and first cell respectively and so on after applying periodic boundary (PB) CA rule-32 twice to each value in the cell of the matrix.

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \xrightarrow[\text{(PB CA rule-32)}]{\text{shift rows}} \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

(once,twice)

4.3.4. Complement matrix

Now we exchange zero to one and one to zero to get the complement of matrix obtained in exchange and shift rows during one round for encryption. Complement of matrix provides substitution which is called as non linear rule of cellular automata. Next we continue this round till the output cipher text has no dependence with the original plain text. But we know with minimum six numbers of rounds the output cipher text has some dependency. That is why we take eight rounds to get the cipher text so that there will be no dependence between the plain text and cipher text.

4.4. Decryption algorithm

Since compliment, PB rule 8, S-box, XOR and PB rule 32 are reversible (i.e. the inverse matrix of PB CA rule -32 and PB CA rule-8 are PB CA rule-2 and PB CA rule-128 respectively), we can easily decrypt the cipher text to get the plain text by using the operations of encryption in reverse way which is shown in Figure 12.

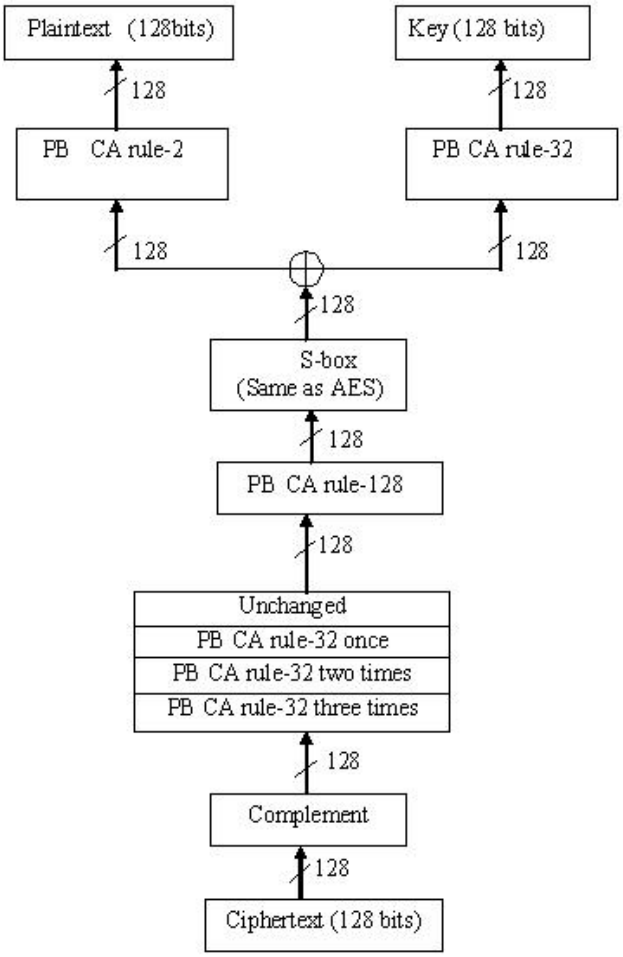

Figure 12: one-round of decryption algorithm applying CA rule

5. CONCLUSION AND FUTURE WORK

Our algorithm deals with both linear as well as the non-linear boolean functions. This algorithm is more secure than DES because the key is used in the beginning of algorithm unlike it's in DES at a later stage and the number of rounds in DES is 16 whereas it is 8 in our algorithm. This algorithm, being based on concept of CA, helps parallel processing of text. Besides, due to availability of chip level design CA machine (CAM), this algorithm can encrypt and decrypt the text at very high speed in the order of nano seconds.

We have planned to develop a new substitute box (S-Box) where total 9 bit texts can be passed and which satisfies all the cryptography properties. Also our aim is to study on nonlinear cellular automata rules and its inverse. In this paper, we have not compared the strength of Advanced Encryption Standard (AES) with our algorithm. But in future we will compare this.

6. REFERENCES

[1] Claude Carlet, "Boolean functions for cryptography and error correcting codes". PhD Thesis.

[2] Stephen Wolfram, "Cryptography with cellular automata". Lecture Notes in Computer Science, 218 (Springer-Verlag, 1986) , pages 429-432, 1986.

[3] Subhamoy Maitra and Enes Pasalic, "Further Constructions of Resilient Boolean Functions With Very High Nonlinearity".*IEEE Transactions on Information Theory*, Vol. 48(7), July 2002.

[4] William Stallings and Lawrie Brown, *Computer Security: Principles and Practice*, Pearson Education Inc., New Delhi.

[5] Charles P. Pfleeger and Shari Lawrence Pfleeger, *Security in Computing*. Pearson Education Inc., New Delhi.

[6] XIA Xuewen, LI Yuanxiang, XIA Zhuliang, and WANG Rong, "Data Encryption Based on Multi-Granularity Reversible Cellular Automata". Proceedings of *International Conference on Computational Intelligence and Security*, 2009, pages: 192-196.

[7] Debasis Das and Abhishek Ray, "A Parallel Encryption Algorithm for Block Ciphers Based on Reversible Programmable Cellular Automata". *Journal of Computer Science and Engineering*, Volume 1, Issue 1, May 2010, pages: 82-90.

[8] Anirban Kundu, Alok Ranjan Pal, Tanay Sarkar, Moutan Banerjee, Sutirtha Kr. Guha, and Debajyoti Mukhopadhyay, "Comparative Study on Null Boundary and Periodic Boundary 3-Neighborhood Multiple Attractor Cellular Automata for Classification". Proceedings of *third International Conference on Digital Information Management, ICDIM 2008*, pages: 204-209.

[9] Irfan Siap, Hasan Akin, Ferhat Sah, "Garden of eden configurations for 2-D cellular automata with rule 2460 N. *Information Sciences*, Volume 180, Issue 18, 15 September 2010, Pages 3562-357.