International Journal of Advanced Research in

Computer Science Engineering and Information Technology

Volume: 2 Issue: 3 08-Apr-2014,ISSN_NO: 2321-3337

2D- Cellular Automata Linear Rules for Cryptography Based on Pattern Evolution

Wani Shah Jahan [1], Khan K. A.[2], Peer M. A.[3]

[1] Department of Computer Sciences, University of Kashmir, J & K, India.
[2] Principal Govt. Degree College,Tral Pulwama, J & K, India.
[3] Department of Computer Sciences, University of Kashmir, J & K, India.

.

ABSTRACT— *In the fast growing information and communication technology (ICT) challenge of data security is emerging out predominantly due to flow of vital data on the wired and wireless networks. Cryptosystems have been designed with various techniques of text encryption and decryption. Easy key encryptions have low level of attack immunity while as complex keys although more resistant to attacks have other drawbacks of occupying more memory space and low speed of encoding and decoding. Cryptosystems have been designed with the help of pseudo sequence generation property of cellular automata too and found relevant due to easy VLSI implementation. We have observed a large number of 2-dimensional cellular automata linear rules in odd groups having an ability to generate the original data in the forward iterations of applied rule. This property is having a versatile fitness for cryptographic applications. Since almost all additive CA linear rules have been found to have irreversible character both encryption and decryption of data are possible in forward direction with the difference of iterations only. The model we can call as Pattern Cryptography, like other CA based models provides easy VLSI implementation with elegant properties of compact size, high efficiency and high speed at low cost. Since all odd groups contain such rules to achieve the cryptographic objectives, although with varied complexity, an additional feature of varied complexity can be incorporated in its VLSI design implementation.*

Keywords: Cryptography, Cellular Automata, Ciphertext, Block cipher, Stream Cipher, Chaotic pattern.

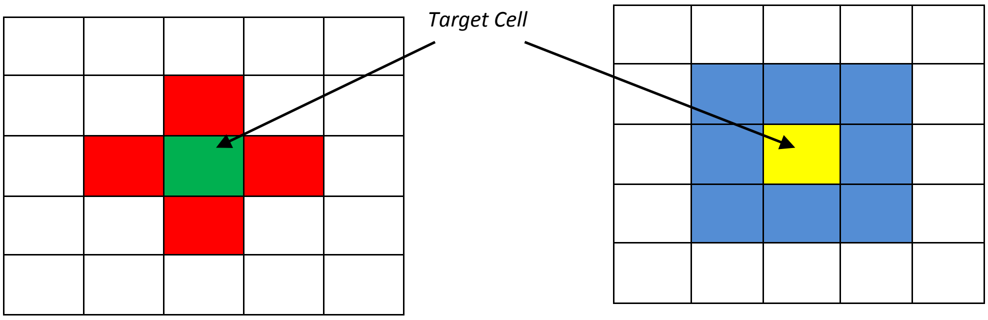
1 INTRODUCTION

Cryptography may be considered as old as communication and defined as an art and science for producing ciphers. A cipher can be defined as the deliberately distorted message produced for the security of the original message. In cryptography ciphers are produced in such a way that the cracking is almost impossible without knowing the key. The art and science of breaking ciphers into original messages is known as cryptanalysis and the study of both processes is called cryptology. The basic message that needs to be converted to a cipher is called the plaintext. The process of converting plaintext to a cipher is known as encryption and the process of converting ciphertext to plaintext is termed as decryption. The other familiar terms one can encounter are block cipher, stream cipher, shared key, public key, digital signature etc. Block cipher is one in which the message is broken into successive blocks that are encrypted using single key or multiple keys. Stream ciphers are those ciphers in which message is broken into bits or characters and then the string of bits or characters is encrypted using key stream. Block ciphers may have a single key for both encryption and decryption called as shared key also known as secret key or symmetric, or may have separate keys for encryption and decryption known as public key or asymmetric. Digital signature schemes also come under asymmetric primitives.

An ideal cryptosystem must have no data expansion at encryption process. It must have fast encoding algorithm, low dimension key, fast decoding and must produce correct message after decryption. The system also needs to be fully attack resistant. In practical systems it is not possible to achieve all objectives simultaneously. It is always required to have complexity in coding for the security reasons; hence for an ideal system for cryptography one cannot have fast encoding and attack immune cipher production in one configuration.

2 CELLULAR AUTOMATA

Cellular Automata (CA) model is composed of a universe of cells in a state having neighborhood and a local rule. With the advancement of time in discrete steps the cell changes its value in accordance to the state of its neighbors. Thus the rules of the system are local and uniform. In one-dimensional CA cells are like a linear canvas and values of the canvas cells change due to application of a local rule in discrete advancing time steps. In two-dimensional CA the cells form a canvas plane and the changes take place in two dimensions while as in three dimensional CA volumetric changes take place by the application of local rule with advancement of time. As the application is two dimensional, here we use 2DCA model where cells are arranged in a two dimensional matrix having interaction with neighboring cells. The central space represents the target cell (cell under consideration) and all spaces around represent its eight nearest neighbors. The structure of the neighbors mostly discussed and applied include Von Neumann neighborhood and Moore neighborhood, are shown in Figure (1). In Von Neumann neighborhood, four cells are positioned at the orthogonal positions of the target cell while as Moore neighborhood is extension of Neumann structure with additional four cells placed diagonally at the four corner positions. For simplicity Von Neumann neighborhood cells can be termed as orthogonal neighbors and the additional cells by Moore can be called as corner neighbors.



Von Neumann Neighborhood Moore Neighborhood

Figure (1)

In general two dimensional Cellular Automata are represented by the equation in as shown below:

$$[a_{i,j}]_{t+1} = R[\, a_{i,\,j} , a_{i,\,j+1} , a_{i+1,\,j} , a_{i,\,j-1} , a_{i-1,\,j} \,]_{t} \qquad \text{--(I)}$$

For Additive Cellular Automata the implementation of the famous totalistic rule in Von Neumann and Moore neighborhoods, the representative equation can be written as follows:

$$[a_{i,j}]_{t+1} = \mathrm{XOR}[\, a_{i,\,j} , a_{i,\,j+1} , a_{i+1,\,j} , a_{i,\,j-1} , a_{i-1,\,j} \,]_{t} \qquad \text{--(II)}$$

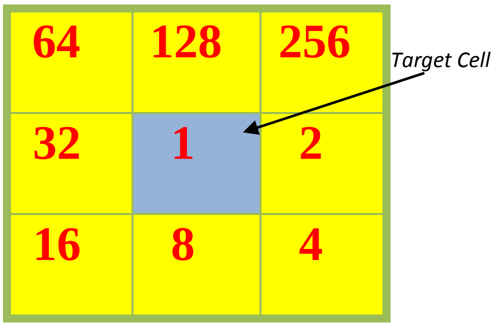
Since the exploring worksheet/canvas is practically limited, researchers have defined some boundary conditions to facilitate the protection of data overflow outside the edges of the worksheet. On the basis of the applied boundary conditions the cellular automata have been divided into three main categories, briefly defined as follows:

Null Boundary Cellular Automata (NBCA): Under null boundary conditions the extreme edge cells are having zero values. For 1DCA the extreme right cell and the extreme left cell are considered to be having a value of binary zero ’0’.

Periodic Boundary Cellular Automata (PBCA): Under periodic boundary conditions the canvas is considered to be folded so that the extreme cells are taken to be adjacent to each other. For 1DCA the extreme right cell is considered to be adjacent to extreme left cell and the extreme left cell is considered to be adjacent to extreme right cell.

Intermediate Boundary Cellular Automata (IBCA): Under intermediate boundary conditions the left neighbor of the leftmost cell is regarded as the second right neighbor. Right neighbor of the rightmost cell is considered as the second left neighbor.

After Stephen Wolfram various studies have also been carried out by Pabitra Pal Choudhury et. el. [1], who classified the cellular automata rules in Moore neighborhood by assigning the rule values to different cells as shown in Figure (2). The rules are generated by the interaction of target cell with itself and with the 8-neighbors around it. These nine rules are said to be basic or fundamental rules and group rules are derived from their combination, Group 2 are rules generated by addition of two basic rules, Group 3 by the combination of three basic rules, Group 4 by the combination of four basic rules, Group 5 by the combination of five basic rules and so on. The group 9 rule (only rule in the group) is the combination of all basic rules. All the combinations are additive (EX-OR operation).



64	128	256
32	1	2
16	8	4

Figure (2)

Examples:

Rule 3 = Rule 2 ⊕ Rule 1 --- (Group 2)

Rule 11= Rule 8 ⊕ Rule 2 ⊕ Rule 1 --- (Group 3)

Rule 43 = Rule 32 ⊕ Rule 8 ⊕ Rule 2 ⊕ Rule 1 --- (Group 4)

Rule 171 = Rule 128 ⊕ Rule 32 ⊕ Rule8 ⊕ Rule 2 ⊕ Rule 1 --- (Group 5)

According to criteria of applying cellular automata rules to a group of data in any neighborhood, the cellular automata have been divided into two types:

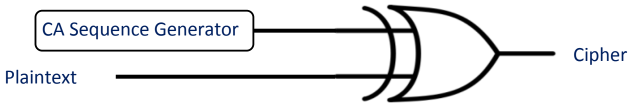
i) Uniform Cellular Automata
ii) Non-Uniform Cellular Automata

Uniform CA also known as Linear CA is applied uniformly on a data matrix of cells. All the cells in matrix get operated with the same rule. The Non-Uniform CA also known as Hybrid CA is one in which all the cells of a matrix have their own local rule that may be different from the rule applied to other cells of that matrix.

3 BRIEF OVERVIEW

Wolfram while reporting stream cipher generation based on one dimensional cellular automata in 1985 [2] also indicated towards high potential of the cellular automata for cryptographic applications. He also explained its integrated circuit implementation and use for high-bandwidth cryptographic applications. This model is based on simple 1DCA model and represented by the following equation;

$$[\ a_i\ ]_{t+1} = [\ a_{i-1}\ \text{XOR}\ \ a_i\ ]_t$$



Circuit Concept

Eduard Franti et. el. [3,4] reported different ways of cipher generation using key stream. They have also proposed a cryptosystem for VLSI implementation. But their projection of cryptosystem based on irreversible automata [3] has hardness of reasoning as backward turning in irreversible CA seems to have algorithm problems. Sabater A. Fuster et. el. have reported use of non-linear interleaved sequence generator in terms of linear cellular automata [5] for symmetric cryptography. Reversible CA application for cryptography has also been proposed by Seredynski Franciszek et. el [6] and Tripathy Somanath et. el [7] using reversible cellular automata pseudo sequence generation. Kishore M. Phani Krishna et. el [8] reported use of reversible CA application on data layers. All these reports have used block or stream cipher techniques for creation of their cryptosystems.

4 PROPOSED METHOD

This proposed work generates block cipher by using 2D cellular automata based secret key rule. While generating and studying chaotic patterns with help of linear additive 2D cellular automata rules, we observed recovery of data block under study in the forward periodic iterations. In the lower group rule instance it seems to be reversibility associated with uniform cellular automata but with increased complexity we observed some chaotic patterns also associated with the recovered data after similar number of rule iterations. The recovery of the original data is possible even after a limited overflow of data along the edges of the experimentation matrix. The first application of these results is reported in this paper on cryptography. Instead of piling linear data blocks to form a two dimensional data sheet we have used a null matrix worksheet (canvas) and loaded plaintext data block at its centre. The data block is allowed to grow for patterns with the application of 2DCA linear rule (Odd group). The rule application produces a cipher and repeated application of the rule increases complexity of the produced cipher. In the forward iterations of the applied rule a stage is reached when the original plaintext is recovered at its starting position in the experimental worksheet. The process algorithm is given below followed by a flowchart in Figure (3);

Encryption:

```
        Start
        Load a Null Matrix
        Input Plaintext at the centre
        Start Complexity Counter
Label   Apply CA Rule
        Decrement Complexity Counter
        Loop to Label on Counter Condition
        Output Resulting Cipher
        Stop
```

Decryption:

```
        Start
        Load a Null Matrix
        Input Cipher
        Start Decryption Counter
Label   Apply CA Rule
        Decrement Decryption Counter
        Loop to Label on Counter Condition
        Output Resulting Plaintext
        Stop
```

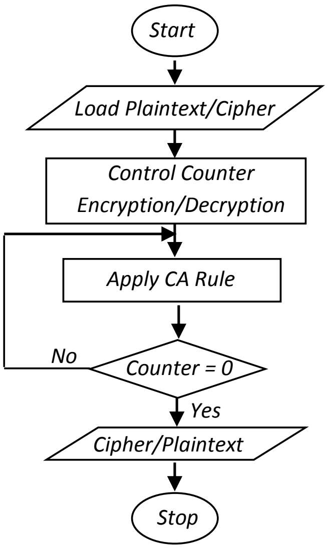
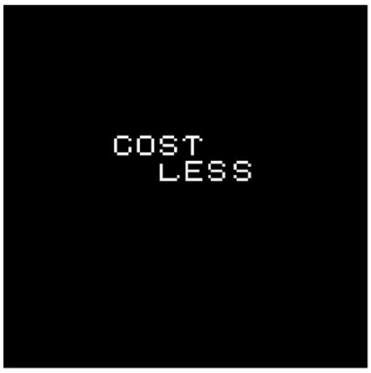


Figure (3)

4.1 IMPLEMENTATION

A 2D canvas of (128×128) null data matrix was loaded with a (48×16) binary data matrix of plaintext at its centre with the help of Matlab programming. The graphical representation of the data matrix is shown below

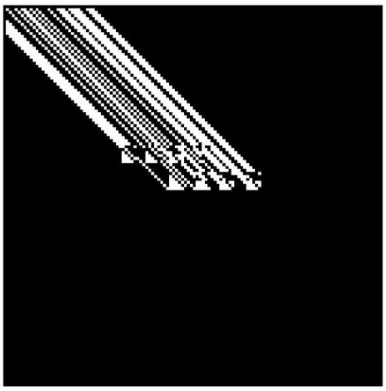


Binary Plaintext

The graphical text ‘COST LESS’ may not be considered as the plaintext directly but graphical representation of 96 bytes of binary data values of plaintext. The data values have been deliberately selected to demonstrate the recovery of data after decryption. The iteration of the applied rule produces a cipher and more iterations increase complexity of the cipher. For the decryption process the cipher is again loaded at the centre of the null matrix and same rule is applied, a stage is reached in the forward iterations when the cipher produces plaintext at its original position.

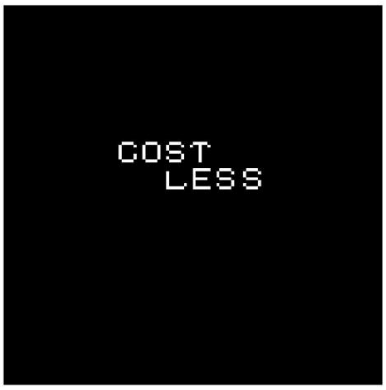
4.2 RESULTS

Group 1: Among group-1 rules we selected rule 4 for the purpose and result produced for the most complex cipher is given below.



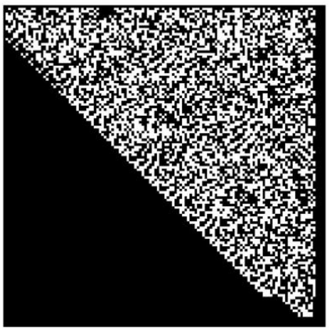
Rule 4(1G)--63

The cipher generation with group-1 rules is less complex, fast and occupying low memory. The speed and memory limitation can be controlled with lower number of iterations but at the cost of complexity. The above cipher is produced after 63 iterations of rule 4(G1). The dereption of the cipher was achieved on the 64th iteration and is shown below;

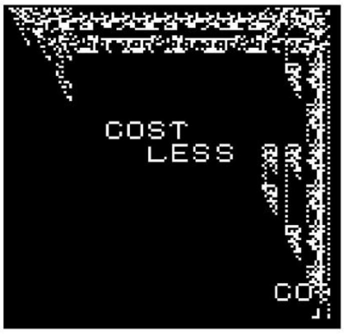


Rule 4(1G)--64

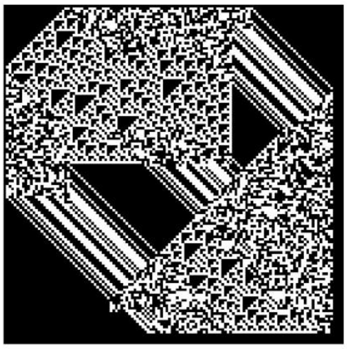
In the same manner Group 3 and Group 5 rules were applied on sample and following results were recorded for demonstration. In this paper we have only shown the most complex ciphers despite of having generated 63 ciphers for each rule application.



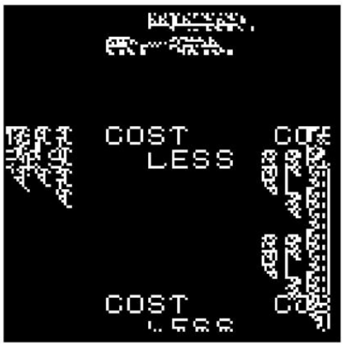
Rule 84(3G)--63



Rule 84(3G)--64



Rule 186(5G)--63



Rule 186(5G)--64

The observation of chaotic patterns around the decrypted ciphers have been observed but the plaintext has been thoroughly analyzed and found correct and at its original position.

4.3 VLSI IMPLEMENTATION

The implementation scheme for the proposed method requires a data block generator which, converts plaintext to a binary block of data. With the help of sync control the binary data along with neighborhood is given to operation register where rule implementation is carried out with the help of control unit. The encrypted data flows to ciphertext block and when a complete block of data transmits out of the ciphertext block an acknowledgement signal comes to the control unit to generate block separation sync followed by sync pulse for data block generator to initialize process for the next block of data. The block diagram for this implementation is given in Figure (4) below;

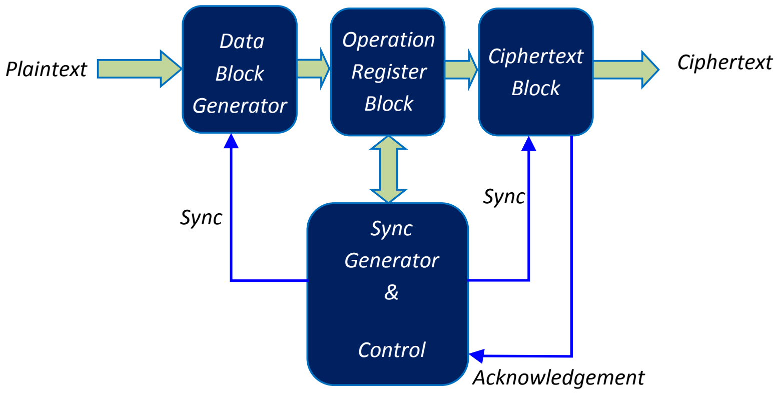


Figure (4)

An alternative way for implementation can be achieved by using a microcontroller interface at encryption and decryption. The device at encryption can be programmed to accept plaintext data block and apply CA rule to create a cipher. The device at decryption can be programmed to accept ciphertext and apply the same CA rule to get the plaintext back.

5 CONCLUSION

The results achieved here have been found according to most standard ones. The introduction of more complexity in terms of iterations and higher Group rule selection has been observed to cause more memory requirements for produced ciphers, but this is not a demerit as the system has less complexity options also available in each configuration. The cryptographic system can cover everybody's application need with two-level and multi-option encryption. This method is versatile in producing large number of cipher complexity options in the same go.

6 REFERENCES

[1] Choudhury P. P. et al. Classification of CA Rules based on their Properties., IJCC Dec. 2010.

[2] Wolfram S. , "Cryptography with Cellular Automata". Springer Verlag 1998.

[3] Eduard Franti et. el, "Design of CA Based Cipher Schemes", ICCC-2004.

[4] Eduard Franti et. el, "Hardware Implementation on FPGA Plateform for CA Cryptosystem", WSEAS-2006.

[5] Sabater A. Fuster et. el, "Efficient Application of Hybrid 150/90 CA to Symmetric Cryptography" Springer Verlag-2006.

[6] Franciszek Seredynski et. el, "Secret Key Cryptography with CA", (IPDPS) IEEE, 2003.

[7] Trpathy Somanath et. el, "Lightweight CA Based Symmetric Cryptography", IJNS, March-2009.

[8] Kishore M. Phani Krishna et. el, "A Novel Encryption System using Layered CA", WCE, July-2011.