An Alternative Cellular Automata Cryptogram

Amr Badr

Faculty of Computers & Information

Dept. of Computer Science

Cairo University

ruaab@rusys.eg.net

Abstract: A cellular automaton-based cryptogram is developed. An alternative approach that combines theories from the Wolfram automata, the cellular automata transforms and reversible cellular automata, is presented. The approach is particularly applicable to session cryptography. It is designed so as to be implemented over simple computer systems and one-to-one data transmission channels. An illustrative example is given for verification.

Keywords: Cryptography, Session Cryptography, Cellular Automata, Wolfram Cellular Automaton, Cellular Automata Transforms, Reversible Cellular Automata.

Amr Badr. : Received his MSc in Computer Science from Cairo Univ. in 1994 in Neural Networks and Evolutionary Computation; PhD in Computer Science in 1998 in Evolutionary Computation and Fuzzy Petri Nets. Currently, he is an Assistant Prof. in Computer Science Dept., Faculty of Computers & Information, Cairo Univ.

1. Introduction

The purpose of cryptography is to hide the contents of messages by encrypting them, so as to make them unrecognizable except by someone who has been given a special decryption key. The purpose of crypto-analysis is then to defeat this by finding ways to decrypt messages without being given the key.

Cryptography has been in use since antiquity, and has been a decisive factor in remarkably large number of military and other campaigns. Typical of early systems was the substitution of cipher of Julius Caesar, in which every letter was cyclically shifted in the alphabet by three positions, with A being replaced by D, B by E and so on.

Starting in 1950s, electronic devices were the primary ones used for cryptography. Linear feedback shift registers & perhaps nonlinear ones seem to have been common, though little is publicly known about military cryptographic systems after World War II. Following the introduction of public-key cryptography in 1975, the idea emerged of basing cryptography not on systems with complicated and seemingly arbitrary construction, but instead on systems derived from well-known mathematical problems. Initially, several different problems were considered. The only ones to survive were those such as the RSA system based essentially on the problem of factoring integers. [2][3][6]

Our purpose is to develop an alternative cryptogram based on cellular automata (CA), in which several CA technologies such as the Wolfram approach [8], the transform-based approach[4][5] and reversible CA[1][7] are combined to form a one cryptogram.

2. A Prologue to Cellular Automata

2.1 What are the Cellular Automata?

A Scientific definition of a Cellular Automaton [8]

Scientists generally define a Cellular Automaton (CA) to be a discrete dynamical system, where space, time, and the states of system are all discrete and have the following properties:

Space is represented by a regular lattice in one, two, or three dimensions.

Each site, or cell, in (or on) the CA lattice can be in one of a finite number of states.

The states are represented by integer number values.

The CA system evolves over a succession of time steps .The values of all the sites in the lattice are updated synchronously in each time steps. While the values of all of the sites are updated in each time step, the value of a site need not change in a given time step.

Sites values are updated using a set of rules (known as a lookup table) that take the values of the site and its neighboring sites into account.

A Computational Definition of a Cellular Automaton: [8]

Since A CA model is always expressed as an algorithm and is invariably implemented as a computer program and run on a computer, it's useful to give a computational definition of CA.

A CA is a computer program in which the following computations are performed in order:

A matrix is created with specific element values (integer, real, symbols list).

A function, or a set of functions that can be used to change the value of a matrix element, based on the values of the element and nearby elements, is defined.

This function is applied (repeatedly) to the matrix, each time changing the values of all the matrix elements simultaneously.

2.2 Lattice:

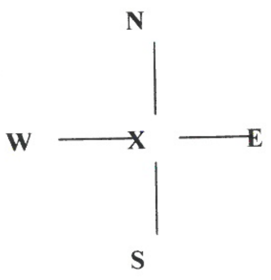
CA Lattice may be one dimensional employing a list or two dimensional employing rectangular lattice .Each lattice sites has a value usually Boolean value 0 or 1.

For example, a simple two-state, one dimensional cellular automaton will consist of a line of cells/sites each of which can take value 0 or 1.

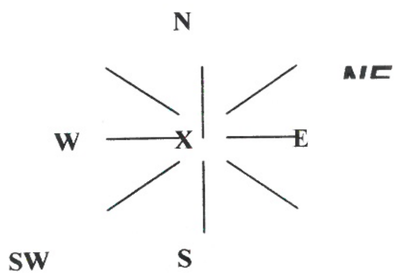
2.3 Neighborhood:

In updating the values of a site in a CA lattice, it is necessary to consider the site's value and the values of the sites in its vicinity. The set of sites that is employed in a CA program depends on the specific "physics" of the system being modeled .here we show some of the neighborhoods that are commonly employed [8].

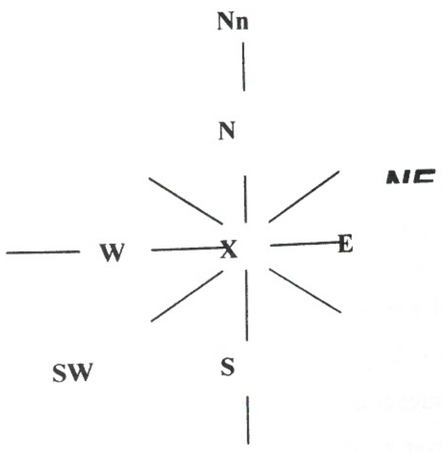
The two most commonly used CA neighborhoods are shown below:



The Von Neumann Neighborhood



The Moore Neighborhood



The Mvon Neighborhood

The theory behind neighborhood is that the value of cell X is updated based on the values of its neighborhood cells.

2.4 Dealing with Boundaries:

The boundary conditions that are used in CA program depend on the specific "physics" of the system being modeled. Traditionally, the lattice sites in the neighborhood of a border site are different for different boundary conditions. For example, under periodic boundary conditions, border sites have sites on opposing border(s) as neighbors (e.g., a sit on the left border has the site in the same row on the right border as its left neighbor) while border site under absorbing boundary conditions have no off–lattice neighbors (e.g. a site on the left border has no left neighbor).

In our computations, lattice sites in the neighborhood of a border site are computed in the same way (which was described in the previous section) under all boundary conditions and different boundary conditions are dealt with in the manner described below.

Periodic Boundaries

This boundary, commonly used in CA models of infinite systems, is the default boundary condition. In a CA model of mobile objects moving in space, the wraparound boundary condition results in objects leaving the system at one border location and reentering at opposing border location.

Absorbing (Open) and Reflecting (Closed) Boundaries and Sinks and Sources:

These boundary conditions are implemented by using a unique value for border sites, different from the values used for interior sites ,and employing rules that specify what happens to the value of a site in the neighborhood of a border site and that leave the value of a border site unchanged . By using unique value for border sites, rules can be written for a CA model of mobile object moving in space that take an object occupying a site adjacent to a border site and freeze the object in place, turn the object around, or remove the object from the system or, if the site ids empty, put an object on site.

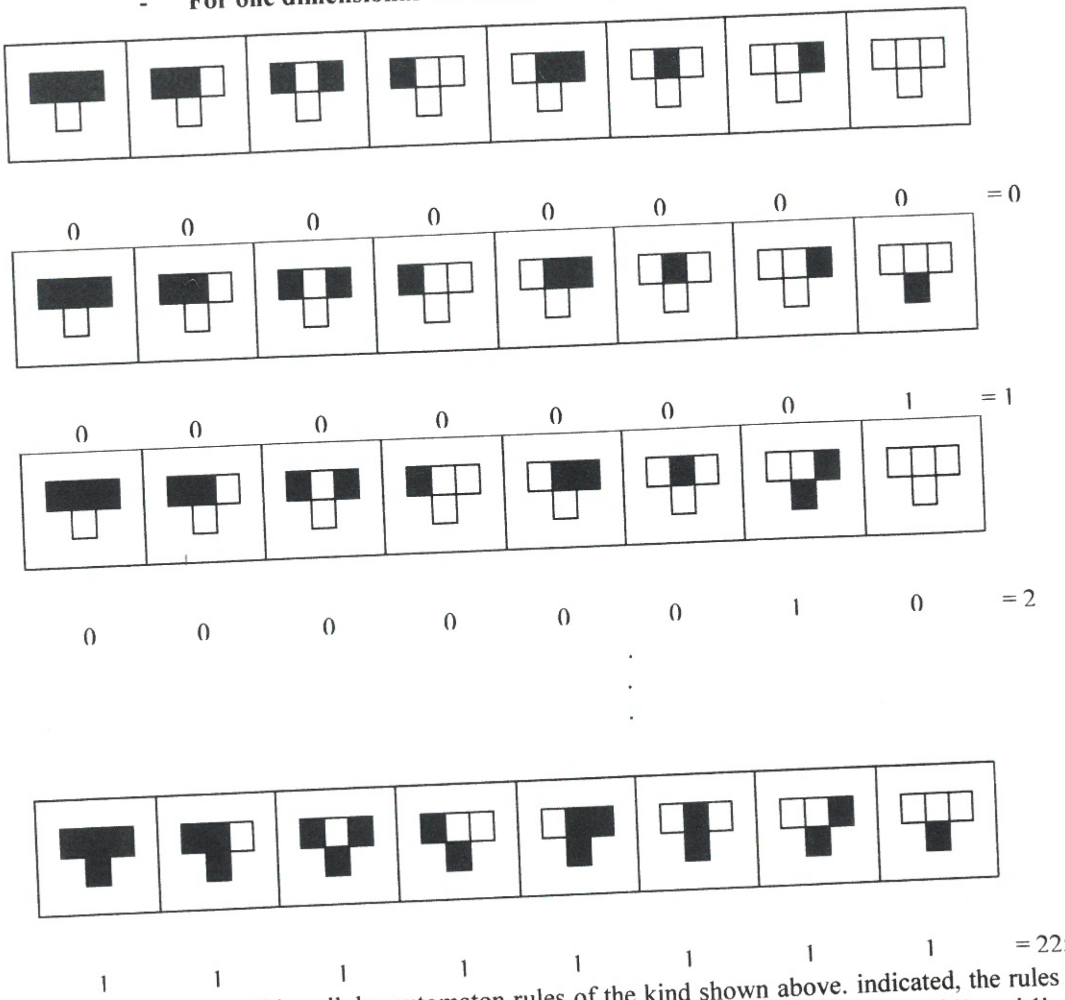
Moving Boundaries

One way to avoid having to deal with specific boundary conditions is to adjust the size of the CA lattice one each time step so that nothing interesting happens on the border sites.

2.5 CA Rules:

These are the rules that govern how each CA lattice is to be updated (evolved).

- **For one dimensional CA the following show some of these rules.**

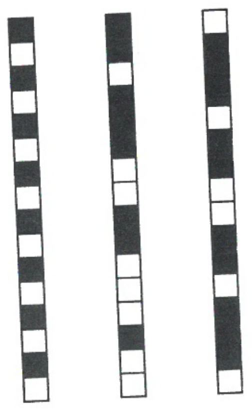


The sequence of 256 possible cellular automaton rules of the kind shown above. indicated, the rules can conveniently be number assigned is such that when written in base 2 it gives a sequence of 0's and 1's that correspond to the sequence of new colors chosen for each of the right possible cases covered by the rule .

There turn out to be a total of 256 possible sets of choices that can be made.

3. Encryption Approaches with CA:

3.1 Wolfram Approach: [8]



(a) (b) (c)

The picture on the left shows a standard method of encrypting messages represented by sequences of black and white squares. The basic idea is to have an encrypting sequence ,shown as column (b) on the left and form the original message (a) to get an encrypted version of the message (c) by reversing the color of every square for which the corresponding square in the encrypting sequence (b) is black.

So if one receives the encrypted message (c), how can one recover the original message (a)? If one knows the encrypting sequence (b) then it is straightforward. For all one needs to do is to repeat the process that was used for encryption and reverse the color of every square in (c) for which the corresponding square in (b) is black.

But how can one arrange that only the intended recipient of the message knows the encrypting sequence (b)? In some situations it may be feasible to transmit the whole encrypting sequence in some secure way. But much more common is to be able to transmit only some short key in a secure way ,and then to have generate the encrypting sequence from this key.

Rule 30 cryptography[8]: Rule 30 is known to have many of the properties desirable for practical cryptography .It does not repeat with any short period or show any obvious structure for almost all keys .Small changes in keys typically leads to large changes in the encrypting sequence .The Boolean expressions which determine the encrypting sequence from the key rapidly become highly complex .And furthermore the system can be implemented very efficiently, particularly in parallel hardware.

3.2 Transform-based Approach: [5]

Consider a data sequence f_i (I=0, 1, 2..., N-1) derived from a message (a plaintext). The steps for encrypting f using this approach are as follows:

1. Select a cellular automata rule set. The rule set includes

Size m of the neighborhood. In the example below m=3.

Maximum state K of the CA must be greater than the magnitude of the largest character in the plaintext. In general we select K=256.

Rules W_j (j=0,1,2...2^m) for evolving the automaton;

Boundary conditions to be imposed. In general, a cyclic condition is imposed on both boundaries.

Maximum time T for which the automaton must be evolved. This time is a function of the rule set W_j (j=0, 1, 2..., 2^m).

Using the sequence f as the initial configuration, evolve the cellular automaton using the rule set selected in (1).

Stop the evolution at time $t=T_f$ when the original message has been recovered. The perfect rule set for the message recovers the message at or before time T. If the original message is not recovered at or before time T, go back to step (1).

Select the time $t=T_o$ at which the states of the n cells achieve maximal entropy or disorderliness as measured against the original message.

Store or transmit the states (N symbols) of the cellular automata at time T_o as the cipher text C_i (I=0, 1, 2..., N-1).

The encryption/decryption keys are:

The CA rule set W_j (j=0, 1, 2..., 2^m).

The quantities T_o and $T_d = T_f - T_o$. In the example below T_d=1.

7. To decrypt the message:

Use the cipher text C_i (I=0, 1, 2..., N-1) as the initial configuration of the CA.

8. With the CA rule set in the encryption/decryption keys, evolve the cellular automata up to the time T_d to recover the original message f_i (I=0,1,2...N-1).

An illustrative example:

Consider the plaintext: This is a test of cellular automata encryption.

This message consists of 47 8-bits characters. To encrypt it we search for a 256-state cellular automaton.

We pre-select time T=64 and assume cyclic conditions at the boundaries. That is, we assume the one-dimensional cells are arranged in a circle; thus making the cell 0 to be a neighbor of cell N-1. We choose a neighborhood size of 3. With W_8=1, we have 8 integers W_j, which define the CA evolution rule.

The coefficients are shown in the following table.

Encryption W-set rule

W_0	W_1	W_2	W_3	W_4	W_5	W_6	W_7
126	81	84	36	10	4	75	0

The original message was recovered after T_f=64 evolution of the CA. The cipher text is:

@_7lNC:_IZL_A~^ga,I_b__Ul_CS_-U__!VyhS_M2,w_@NWhich was obtained from the states of the cells at time t=63. Hence, T_d=1. To decrypt the message, we use the above cipher text as the initial configuration and evolve the CA for T_d=1 time steps.

Although it is described above as a secret-key system, one special feature of this approach is the possibility of a public-key implementation. A second set of CA key-set, U, is required for decoding. The forward evolution (encryption) is carried out, using key-set W, to the termination time T_o at which stage the state of the automaton is the cipher text. For decryption the different set, U, is used to evolve the system for T_d steps with the cipher text serving as the initial configuration. The key-set W then plays the role of a public-key, while U is the private-key. Obviously, the pair of keys (W, U) must be unique. The search for the keys is more involved than in the secret-key implementation.

3.3 Reversible CA Approach: [7]

Encryption:

Encryption continues over m cycles. At each cycle, the following operations are performed in sequence:

1. The noise generator generates 2 noise bits.
2. They are appended to the end of the bit stream S.
   Increment the stream length with 2.
   If the cycle number is 0, then go to step 1.
3. the following operations are performed on the bit stream S from bit indexed i=n-3 to 0:
   3.1 Put the 2 bits (i+1, i+2) in variable G at (1, 0) weight bits respectively.
   3.2 Put bit number i into variable B.
   3.3 If B=0, then get entry F from the rule table whose index is G.

Else get entry F from the rule table whose index is G+4.

   3.4 Set bit number i with the value of F in bit stream S

If cycle number is less than m, then go to step 1.

Else, the encryption cycles are performed successfully, and the bit stream S contains the encrypted message.

Decryption:

Decryption continues over m cycles from cycle m to 0. At each cycle the following steps are performed:

1. If cycle number is 0, then go to step 3.
2. Do the following operations on bit number i on stream S from i= n-3 to 0:
   2.1 Put the 2 bits (i+1, i+2) in variable G at (1, 0) weight bits respectively.
   2.2 Put bit number i into variable B.
   2.3 If B=0, then get entry F from the rule table whose index is G.

Else get entry F from the rule table whose index is G+4.

   2.4 Set bit number i with the value of F in new bit stream S2.

Remove the last 2 noise bits. And decrement the length of the stream S2 by 2.

Assign bits stream S2 to S.

If current cycle number is greater than 0, then go to step 1.

Else the decryption cycles are performed successfully and the decrypted message is contained in bit stream S.

But why use one bit stream in encryption, and two bit streams in decryption?

The reason is:

1. In encryption, we use the previously updated bits into variable G, so we can write on and update the same bit stream.
2. In decryption, we use the old received bits from the bit stream to get the G value. So, if we update and write on those old bits, then we'll get wrong G values in the next sequence.

Regardless of the message contents: As long as the no. of Encryption/Decryption Cycles increases, the generated noise bits increases, so the appended noise bits to the bit stream increases and vice versa... E.g., if we encrypt the following message containing the word "Cryptography"

With 1000 encryption cycle, 2000 noise bits will be added, 250 bytes will be appended to the message, then after encryption:

Message length = original message length + added noise bytes.

Message length = 12 + 250 =262.

The original message length = (12/262)*100 ~~ 4.5 % from the encrypted message length!

The encrypted message will be:

```
خU˜«L`ـzv×ة.:Z«ج]g¢؛o;ª?_¸({%HJªع ©Šÿ'-+H¥…¦²!"©çaCF¬h23Uج×XPCk=¹¦
؟|. ^ج˜„٩ؤ˜¦÷~zš¹ـ%:\kغ•˝ü=eـqـù•Ÿ#*|Cô@!#ژeP¨ûZي+ة¹جï±k•)>šn£d²M¸u§ـ
•‰¶Wك°y!WVyzق`v¥ j ˆg›gـ:"ŒzˆشX•ê½irî@!#:Zأ|"f«¥%?;·ذوxEîéo')î,
«ï+•îî›çf.'!؟إف
```

The Rule Table is an array consists of 8 entries; each entry is a 1-bit code to be used to replacing the current stream bit.

This table must represent any reversible CA rule as Rule 30, Rule 60 in their binary format, each entry is one of the Rule 8-bits.

4. The Proposed CA Cryptogram

Transmitter	Receiver
	1- Generate the key (K) by random generator (128 bit key).
	2- Evolve (K) using CA reversible CA approach for 14 time steps to produce PK.
	3- Send PK to Transmitter.
1- Inverse evolves the received PK to get the initially generated key (K).	
2- Generate two random numbers R1, R2<128.	
3- Evolve K using Wolfram CA approach for R1.	
4- Evolve the message M using transform based CA approach for R2 steps.	
5- Encrypt M with K to produce cipher message C.	
6- Embed R1, R2 into C.	
7- Transmit C to Receiver.	
	4- Get R1, R2 from C.
	5- Evolve K using Wolfram CA for R1.
	6- Decrypt C with K to produce M.
	7- Evolve M for 128-R2 transform CA approach.

An Illustrative example:

Plain text:

"With the increasing digital traffic load on the information highway and the need to secure and protect the integrity of data, cryptography has emerged as a vital technology. Cellular automata provide a robust environment for developing a data encryption standard. Work in the area of CA data encryption has been intense".

The following are the results of the previous steps:

Key: "4A72D200E1D911D6BADECE126DB6FC57".

Evolved key: "□Œع9ؤ©'=c4²ح´¹و—vr$÷ثëRb□P_>,□·""

Send key.

Inverse evolved key : "4A72D200E1D911D6BADECE126DB6FC57"

R1 = 35, R2 = 123.

Evolved key by R1 : "«ء□wwHo»بoo»□□ب»H"Ho□»ب□~"ء"

Evolved message by R2 :

"W5V¼HTp□—†{"EصWμn.غT™{•è□Ntü›، عچ®DW□^H£x□D0‡□ ½ي<#^ثبàà y□h4,,ه½د¬h□iژ
nUéî4<Sx[،G□îA"¼¬èpJ%□غkى|□PQD□düew5^EبCح|#@! H5¦؟ژچ□ژ آط□~x½²□ء ƒ□جف©:§©v□aC
ضا¶□>μh□Nt¼Uë□(›ب\□£د|‹¶œ□رM88‡□9i=¨eن□jC2□@Qن®3؛î،□'غيPفrC؛m©¬h™¼ط´]^R،d-3ف□أ
ETچx½†W®Mj□i□4ذ÷)ذ¬□,گنقW□GŒش□¥p&•¤رE|H□P□□چ□عx½†W®Mj□iG`àà□z²□i¨نE#-(|ن©F"

1. Encrypted message with embedded R1, R2:

"Eu²?Z4d 6گگ5ڠقê□!'Y¨رu□Uك9ل|□ضً...ê000 |□00035ج3-□س™ê,u□Tخت؛P□‰—îd=□'000, -Dى•‡-
`"T/C,p□+,IWà□fب □÷"hR□□ضً'-.□A¤000Bf4ى.e□؛4ك}î¯□آ»C§pWo'□6ت¶îسW§~مU>W-tW¦أ
"W²<²"Vلىشَ P□ىü'²ط^:...pAN|÷-ك™âê□Wیr"□چ´Dà©□123ضي^غhE}`□4F□Uh□}0,□صûثéq¯□ح
d™b{«-000□!□ž"

2. Transmit to receiver.

3. Get R1 = 35, R2 = 123.

4. Evolved key by R1: "«ء□wwHo»بoo»□□ب»H"Ho□»ب□~"ء"

5. Decrypted message:

"W5V¼HTp□—†{"EصWμn.غT™{•è□Ntü›، عچ®DW□^H£x□D0‡□ ½ي<#^ثبàà y□h4,,ه½د¬h□iژ
nUéî4<Sx[،G□îA"¼¬èpJ%□غkى|□PQD□düew5^EبCح|#@! H5¦؟ژچ□ژ آط□~x½²□ء ƒ□جف©:§©v□aC
ضا¶□>μh□Nt¼Uë□(›ب\□£د|‹¶œ□رM88‡□9i=¨eن□jC2□@Qن®3؛î،□'غيPفrC؛m©¬h™¼ط´]^R،دف□أ
-3ETچx½†W®Mj□i□4ذ÷)ذ¬□,گنقW□GŒش□¥p&•¤رE|H□P□□چ□عx½†W®Mj□iG`àà□z²□i¨نE
#-(|ن©F"

6. Evolved message by 128-R2:

"With the increasing digital traffic load on the information highway and the need to secure and protect the integrity of data, cryptography has emerged as a vital technology. Cellular automata provide a robust environment for developing a data encryption standard. Work in the area of CA data encryption has been intense".

So the message was recovered successfully…

5. Conclusion & Enhancements

An alternative cellular automata-based session cryptogram is developed. The cryptogram was tested and verified using an illustrative example. Several future enhancements can be applied as follows:

- A compression option can be added if data is to be transmitted over a network.
- Audio messages are to be verified using this technique so as to allow military personnel to transmit their messages without being decrypted.

REFERENCES

1. ADAMATZKY, A., **Identification of Cellular Automata**, Taylor & Francis, 1994.
2. GUTOWITZ, H., **Cryptography with Dynamical Systems**. In: N. Boccara, E. Goles, S. Martinez and P. Picco (eds), Cellular Automata and Cooperative Phenomena, pp237-274, Kluwer Academic Publishers, 1993.
3. GUTOWITZ, H., **Method and Apparatus for Encryption, Decryption, and Authentication using Dynamical Systems**, U.S. Patent #5,365,589, 1994.
4. LAFE, O., **Method and Apparatus for Data Encryption/ Decryption Using Cellular Automata Transform**, U.S. Patent #5,677,956, 1997.
5. Lafe, O., **Cellular Automata Transforms: Theory and Applications in Multimedia Compression, Encryption and Modeling**, Kluwer Academic Publishers, 2000.
6. SCHEIER, B., **Applied Cryptography**, John Wiley, 1993.
7. TOFFOLI, T.; MARGOLUS, N., **Invertible Cellular Automata**, Physica D, 45, pp229-253, 1990.
8. WOLFRAM, S., **A New Kind of Science**, Wolfram Media Inc, 2002.