CRYPTOSYSTEMS BASED ON CELLULAR AUTOMATA: A SHORT SURVEY

LUCIA SIRBU

RISC-Linz

June 2001

Abstract

Cellular automata are massively parallel computing devices modeling natural systems that consist of large collections of simple objects (cells) interacting locally with each other. They have recently gained popularity as discrete dynamical systems (a discrete dynamical system being a system whose state evolves over time).

The low speed of decryption/encryption causes often big problems with practical implementation of cryptosystems. Cellular automata are inherently parallel which makes their parallel implementations natural. They can be counted on to deliver high encryption/decryption rates at low cost.

This paper offers a brief state of the art in this area.

1. Cryptography

The ancient and basic problem of cryptography is secure communication over an insecure channel. Part A wants to send to part B a secret message over a communication line which may be tapped by an adversary.

1.1. Private vs. Public Key Encryption

The traditional solution to this problem is called **private key encryption**. In private key encryption, A and B hold a meeting before the remote transmission takes place and agree on a pair of *encryption* and *decryption algorithms* E and D and an additional piece of information S, *the common secret key*, to be kept secret. The adversary may know the encryption and decryption algorithms D and E which are being used, but does not know S.

After the initial meeting, when A wants to send B the *cleartext* (or *plaintext*) message m over the insecure communication line, A encrypts m by computing the ciphertext $c=E(S,m)$ and sends c to B. Upon receipt, B decrypts c by computing $m=D(S,c)$. The line-tapper or adversary, who does not know S, should not be able to compute m from c.

A rigorous theory of perfect secrecy based on information theory is developed by Shannon in 1943. In this theory, the adversary is assumed to have unlimited computational resources. Shannon shows that an encryption system can be secure only if the secret key S is as large as the number of secret bits ever to be exchanged remotely using the encryption system.

Modern cryptography abandons the assumption that the adversary has infinite computing resources, and assumes instead that the adversary's computation is resource bounded in some reasonable way (e.g. that the encryption and decryption algorithms designed are probabilistic and run in polynomial time). Accordingly, we speak in modern cryptography of the infeasibility of breaking the encryption system.

The development of **public key cryptography** in the seventies [3] enable one to drop the requirement that A and B must share a key. The receiver B can publish authenticated information (called the public key) for anyone including the adversary (e.g. in a phone book). The encryption system is no longer intended to be used by a pair of specified users, but by many senders wishing to send secret messages to a single recipient. The receiver keeps secret (to himself alone!) information (called the receiver's secret key) about the public key, which enables him to decrypt the ciphertext he receives. This is possible given a trapdoor function: a one-way function for which there exists some trapdoor information known to the receiver alone, with which the receiver can invert the function. Therefore, the public key is used to encrypt the message and the private key is used to decrypt the resulted ciphertext.

A cryptographer's ideal encryption scheme is an operation on a message which renders the message fully meaningless to anyone who does not possess a decryption key, yet in no way degrades the meaning extractable by anyone who does possess the key. An ideal practical code is in addition fast on both encryption and decryption, uses a key of manageable size, and produces no expansion of the data upon encryption [5].

A provable ideal practical encryption method in this sense in not yet known. For the state of the art see [1].

1.2. Attacks

To break a cryptosystem means to discover the meaning of messages encrypted by the system without being handed the secret key. It is generally assumed in academic cryptology that the mechanism of encryption in all its detail is known to the cryptanalyst, the only information lacking being the secret key. Typically, breaking a cryptosystem means reconstructing the key through observations of the cryptosystem in operation. The type of observations on, and manipulations of the cryptosystem which are allowed the cryptanalyst determine the mode of attack [5].

The first kind of attack is **passive attack**, in which the cryptanalyst can only make observations on the cryptosystem as it performs. In a ciphertext-only attack, the cryptanalyst has access only to a stream of ciphertext coming from a cryptosystem loaded with its secret key. The cryptanalysts attempts to find statistical regularities in the stream of ciphertext, departures from randomness which might reveal the nature of the key. All but the most naive cryptosystems produce ciphertext with a high degree of randomness, so that a cryptosystem which falls prey to this kind of attack is considered very weak.

A stronger passive attack allows the cryptanalyst observations both of a stream of ciphertext and the corresponding message stream which produced it. This is called a known-plaintext attack. Again, cryptology has progressed to the point where cryptosystems susceptible to a known-plaintext attack hold little interest [5].

More important are the **active attacks**. Here cryptanalysts can opt to have plaintext of their choosing encrypted and see the ciphertext which results (a chosen-plaintext attack). Similarly, a chosen-ciphertext attack permits ciphertext chosen by the cryptanalyst to be compared with the corresponding plaintext. By current cryptographic standards, a good cryptosystem must resist attacks which permit both plaintext and ciphertext to be chosen, and according to any strategy preferred by the cryptanalyst.

It would be interesting to consider the cryptanalysis of the so-called Caesar cipher [5], used by Caesar to communicate with his troops. It consists of a pair of concentric rings. On each ring the letters of the alphabet are written in order. The key of the system is the displacement of the outer ring with respect to the inner ring. To send an encrypted message, the sender emits in sequence the letters on the inner ring which correspond to the letters on the outer ring contained in the message. The receiver reverses the process, reading off from the outer ring letters which correspond to the letters on the inner ring received. While a fair amount of ciphertext might be required in a passive ciphertext-only attack before the key is guessed, a ciphertext-plaintext pair for a single letter reveals the key in any other attack.

The area of activity in modern cryptology closest to dynamical systems theory concerns so-called iterated cryptosystems. An *iterated cryptosystem* is one in which a cryptographically weak transformation is applied repeatedly to a message, so that the composed transformation is strong [5]. The most well-known and well-used cryptosystem is an iterated cryptosystem. It is known as the Data Encryption Standard, or DES. The Data Encryption Standard specifies a FIPS (Federal Information Processing Standards) approved cryptographic algorithm for encrypting (enciphering) and decrypting (deciphering) binary coded informationb [2]. The DES encryption/decryption algorithm consists of 16 rounds of a transformation designed to fully mix message information together with random key information, which is made up of 64 binary digits ("0"s or "1"s) of which 56 bits are randomly generated and used directly by the algorithm and the other 8 bits, which are not used by the algorithm, are used for error detection. .

The security of the DES has recently been seriously challenged using a technique known as differential cryptanalysis.

Whereas classical cryptography was confined to the art of designing and breaking various encryption schemes (or ‘‘secrecy codes”), modern cryptography is concerned with the rigorous analysis of any system which should withstand malicious attempts to abuse it. We emphasize two aspects of the transition from classical to modern cryptography:

(1) the widening of scope from one specific task to an utmost wide general class of tasks; and

(2) the move from an engineering-art which strives on ad-hoc tricks to a scientific discipline based on rigorous approaches and techniques.

2. Cellular Automata

Cellular automata are massively parallel computing devices that recently have gained popularity as discrete dynamical systems modeling natural systems consisting of large collections of simple objects interacting locally with each other [9].

A cellular automaton (CA) is therefore a *discrete dynamical system*, a system whose state evolves over time. Space, time, and the states of the system are *discrete*. Each point in a *regular spatial lattice*, called a *cell*, can have any one of a finite number of *states*. The states of the cells in the lattice are updated according to a *local rule*. That is, the state of a cell at a given time depends only on its own state one time step previously, and the states of its nearby neighbors at the previous time step. All cells on the lattice are updated *synchronously*. Thus the state of the entire lattice advances in discrete time steps. Since the update rule is simple, local, and discrete, it can be executed in easily-constructed massively-parallel hardware at extraordinary speeds.

Formally, a d-dimensional cellular automaton consists of a d-dimensional (ideally infinite) array of identical cells. The positions of the cells are indexed by $\mathbf{Z}^d$, the set of d-tuples of integers. Each cell is always in one state from a finite state set S, that is, it carries a fixed finite amount of information. The cells can read the states of some nearby cells, their so-called neighbours. The *neighbourhood vector*

$$N = (x_1, x_2, ..., x_n)$$

of the CA, consisting of n different elements of $\mathbf{Z}^d$, tells where the neighbours of each cell are situated. The neighbours of a cell in position $x \in \mathbf{Z}^d$ are the cells in positions

$$x+x_i \text{ for } i=1,2,...,n.$$

The cells alter their states syncronously on discrete time steps according to the *local rule* f of the CA. The local rule f is a function from S^n into S that gives the new state of each cell as a function of the old states of its neighbors. The array is homogeneous in the sense that all cells operate under the same local rule.

As seen, in order to define a CA, one has to specify four items: the dimension d, the state set S, the neighborhood vector N and the local rule f. We say that the CA is defined by the 4- tuple (d,S,n,f).

Above it was assumed that the array containing the cells is infinite. In practice the space has to be made finite. This can be easily achieved by connecting the cells in a shape of a finite d-dimensional rectangle and by wrapping it around, that is, by attaching the cells on the opposite edges of the rectangle to each other, thus obtaining the surface of a d-dimensional torus. The CA with finitely many cells operate exactly as the ones with an infinite number of cells, except for the fact that when determining the neighbors of cells the additions of the components of indices are done cyclically.

The states of all cells in the array (finite or infinite) are described by a *configuration* of the CA. A configuration is a function

$$c:X \rightarrow S,$$

where $X= \mathbf{Z}^d$ (if there are infinitely many cells) or $X=X(M_1, M_2, ...,M_d)$ (if the number of cells is finite, and M_i are positive integers). In either case, the configuration assigns states to all cells. Let $C(X, S)$ denote the set of all configurations. The local rule f determines the global function

$$G_f:C \rightarrow C$$

that describes the dynamics of the CA. At each step a configuration c is transformed into a new configuration $G_f(c)$ where $G_f(c)(x)=f(c(x+x_1), c(x+x_2), ..., c(x+x_n))$ for all x in X.

If no restrictions are made on the local rule f, the CA it defines can lose information, that means that there can be several configurations that are transformed into the same configuration in one time step. However, some local rules give rise to CA that are information preserving - each configuration has a unique predecessor. Their global transition function is one-to-one. Such CA are termed *injective* [9].

A CA is called *reversible* if there exists another CA, called the *inverse CA*, that retraces the computation steps of the first one backwards in time. The global functions of the two CA are inverses of each other. Naturally a reversible CA has to be injective – otherwise no inverse of the global function exists –, and the converse is also true, so *a CA is reversible if and only if it is injective*.

States under a *reversible rule* always have one and only one preimage. States under an *irreversible rule*, on the other hand, may have either many or no preimages. If states can have no preimages under a given cellular automaton, then the cellular automaton cannot always be *inverse iterated*. Such rules are to be avoided in applications to cryptography.

3. Cryptosystems based on Cellular Automata – why?

Dynamical systems are often described as „unpredictable" or „complex" as aspects of their behavior may bear a cryptic relationship with the simple evolution laws which define them [5]. Some theorists work to quantify this complexity in various ways. Others try to turn the cryptic nature of dynamical systems to a practical end: encryption of messages to preserve their secrecy.

Cryptosystems constructed from *cellular automaton* primitives can be implemented in simply constructed massively parallel hardware. They can be counted on to deliver high encryption/decryption rates at low cost.

The low speed of decryption/encryption causes often big problems when practical implementation of cryptosystems are built. This is true especially in connection with public-key cryptosystems, but in some context higher speed of secret-key cryptosystems would be necessary as well. Normally, the first solution to this problem is to replace software implementations with specialized hardware. If this is not enough, a usual approach is to make parallel implementation of encryption/decryption algorithms. In practice this can be difficult. If the cryptosystem is not especially constructed with parallelism in mind it might be impossible to find efficient parallel algorithms for it. And even if parallelization is possible, in order to get considerable speed-up, a big amount of dedicated hardware is required, which can make the cost of implementing the system very high [9].

Cellular automata are inherently parallel which makes their parallel implementations natural. The simplicity and locality of their operations make it possible to build cheap and fast devices containing even hundreds of thousands of cells (“processors”) working in parallel.

In addition to these practical features, cellular automaton cryptosystems may help illuminate some foundational issues in both dynamical systems theory and cryptology, since each of these disciplines rests heavily on the meanings given to the intuitive notion of complexity [5].

... and how?

The future state of a (chaotic) dynamical system depends sensitively on its initial state. After enough time has elapsed the initial configuration is forgotten. Yet, since the system is deterministic, the same trajectory will always be traced out from the same initial configuration. Thus, if *the key of the cryptosystem is the initial state of a public known dynamical system*, a collection of users who share a secret key can send secret messages to each other by combining (using for instance an XOR operation) these messages in some way with some part of the trajectory traced out by the secret initial state under the action of the dynamical system. Anyone who does not know secret initial state would not be able to recreate the trajectory and thus would not be able to disentangle it from the encrypted message.

At least two concrete proposals have been based on this idea, one using a continuous dynamical system, the other a discrete dynamical system (CA-based) [5].

The CA-based system uses iteration of a cellular automaton to generate the bit string. The cellular automaton chosen, known as rule 30, seems according to numerical evidence to generate temporal sequences which have a high degree of randomness. As in the continuous dynamical system approach, the secret key is the initial state of the system, and a message can be encrypted and decrypted by combining it with the temporal sequences generated by the dynamical system using an XOR operation.

The essential observation that is using a known-plaintext attack, the effective key space can be considerably reduced in size. The aspects of these system which should be retained are

1) the *key* is *a state of the system*,
2) the message is encrypted by combining it with an information stream generated by *forward* iteration of the system,
3) the message is decrypted by combining the ciphertext with an information stream again generated by *forward* iteration of the system.

Consideration of *inverse* as well as forward iteration of dynamical systems opens up some new ways to use dynamical systems for encryption. One possibility, which again has occurred independently to a number of investigators, is to concentrate on *reversible* *dynamical systems* [5]. Using a reversible dynamical system, a message can be encrypted by encoding it as a state of the system and then running the system forward in time some distance. The resulting state is the ciphertext.

To decrypt the ciphertext, the system is inverse iterated the same number of time steps as were used in encryption, recovering the plaintext as a state of the system. Note the contrast with the systems considered above in which only forward iteration is used. In those systems, the key is a state of the system and the system is fixed. When forward and inverse iteration is used, *the key is the dynamical system itself*. The key operates directly on the message to encrypt and decrypt it, while in the previous systems the information generated by the dynamical system is combined indirectly, so to speak, externally, with the message information.

3. Some Models Of Cryptosystems Based On Cellular Automata

1. Howard Gutowitz – CA-1.1 - A Massivelly Parallel Cryptosystem Based on Cellular Automata
2. Stephen Wolfram – A Stream Cipher Based On A One-Dimensional Cellular Automata
3. Jarkko Kari – Reversible Cellular Automata as Efficient Encryption/Decryption Devices
4. Jesus Urias – Cryptography Primitives Based On Cellular Automata

Other significant work in the area of encryption using cellular automata has been done by P. Guan [4], who introduces a public key cryptosystem based on a cellular automaton, Habutsu et.al. [8], about a secret key cryptosystem by iterating a chaotic map, M. Mihaljevic et. al., about fast dedicated one-way functions based on cellular automata [13].

3.1. Howard Gutowitz – CA-1.1 - A Massivelly Parallel Cryptosystem Based on Cellular Automata [6]

Howard Gutowitz has developed a cryptosystem, called CA-1.1, which illustrates some of the principles of encryption with cellular automata. In CA-1.1, a message is encoded as a state of the system, and various cellular automaton rules applied to it in order to produce a new state which is the ciphertext. To decrypt, the same cellular automaton rules are applied in reverse order. The cellular automata employed are derived from a secret key.

CA-1.1 uses both *reversive* and *irreversible* cellular automaton rules.

Under a reversible rule, each state of the lattice comes from a unique predecessor state, while under an irreversible rule, each state can have many predecessor states. During encryption, irreversible rules are iterated backwards in time. To go backward from a given state, one of the possible predecessor states is selected at random. This process can be repeated many times. Backward iteration thus serves to mix random information with the message information.

CA-1.1 uses *a particular kind of partially linear irreversible rule* which is such that a random predecessor state for any given state can be rapidly built. Reversible rules are also used for some stages of encryption. The reversible rules (simple parallel permutations on sub-blocks of the state) are fully non-linear, while the irreversible rules are partially linear. The irreversible rules are derived entirely from information in the key, while the reversible rules depend both on key information and on the random information inserted during the stages of encryption with irreversible rules.

CA-1.1 features an authentication mechanism which renders chosen-ciphertext attack difficult. This authentication mechanism ensures that only ciphertext produced by the system loaded with a given key is decrypted using the same key.

3.2. Stephen Wolfram – A Stream Cipher Based On A One-Dimensional Cellular Automata [10]

Stephen Wolfram discusses a stream cipher based on a simple one-dimensional cellular automaton. The cellular automaton consists of a circular register with cells, each having a value equal to 0 or 1. The values are updated synchronously in discrete time steps according to the rule

(1a) $a'_i = a_{i-1} \text{ XOR } (a_i \text{ OR } a_{i+1})$,

or, equivalently,

(1b) $a'_i = (a_{i-1} + a_i + a_{i+1} + a_i\, a_{i+1}) \bmod 2$.

The initial state of the register is used as a *seed* or *key*. The values $a^{(t)}$ attained by a particular cell through time can then serve as a random sequence. Ciphertext C can be obtained from binary plaintext P as usual according to

$$C_i = P \text{ XOR } a^{(i)};$$

the plaintext can be recovered by repeating the same operation, but only if the sequence $a^{(i)}$ is known.

Cellular automata such as (1) have been investigated in studies of the origins of randomness in physical systems [10]. They are related to non-linear feedback shift registers, but have slightly different boundary conditions.

The security of a cryptographic system based on (1) relies on the difficulty of finding the seed from a time sequence of cell values. No systematic algorithm for its solution is currently known that takes a time less than exponential in N. No statistical regularities have been found in sequences shorter than the cycle length.

3.3. Jarkko Kari – Reversible Cellular Automata as Efficient Encryption/Decryption Devices [9]

The use of *reversible rules* in cryptography has been championed by Kari [9]. Kari considers in particular public-key cryptosystems based on reversible cellular automata. In such systems security depends on the difficulty of finding the inverse cellular automaton given the forward cellular automaton. An advantage of reversible cellular automata for cryptography is that both encryption and decryption can be performed in an entirely data-parallel fashion.

Reversible cellular automata (RCA for short) can be used as cryptosystems in the most natural way. Let $A=(d,S,N_1,f_1)$ be a RCA and let $A^{-1}=(d,S,N_2,f_2)$ be its inverse RCA. The plaintext to be encrypted is written on a d-dimensional array usoing the symbols of S. If the array is of size $M_1 x M_2 x ... x M_d$ then the plaintext constitutes a cofiguration $p \in C(X(M_1,M_2,...,M_d),S)$. The encryption is done by applying the RCA A on p for k time steps. The number k can be either a fixed positive integer or it can depend on the size of the array. The resulting configuration $c=G^k_{f1}(p)$ is the ciphertext. The plaintext p can be obtained from c simply by applying the inverse RCA A^{-1} on c for k time steps, that is, $p= G^k_{f2}(c)$.

RCA can be also used as *public-key cryptosystems*.

According to [9], „*for every algorithm that finds the inverses of all two-dimensional RCA given as input, and for every computable function f, there is an RCA A such that the algorithm does not find the inverse of A in time f(s) where s is the ‚size‘ of A (the number of its states for example).*“ According to the corrolary above, in general, one can make public the RCA *A* used in encryption without sacrificing the secrecy of the inverse RCA A^{-1} used in decryption. It is, however, necessary that the RCA used is at least two-dimesional, since all the one-dimensional RCA can effectively be inverted.

To build a practical public-key cryptosystem from the idea above one has to give a procedure for constructing RCA that are difficult to invert. It can be proved that if several RCA that are easy to invert are composed, the resulting RCA can be hard to invert.

The main problem with RCA as public-key cryptosystems is the fact that the encryption key tends to get large [9]. If the RCA used for encryption has n states and the size of the neighborhood is n, then storing its local rule - if represented simply as a table, requires $s^n \log_2 s$ bits. That is why it is important to keep the neighborhood size of the RCA used for encryption small. It is also possible that the encryption RCA has a smaller neighborhood than the decryption RCA. For the decryption RCA, it is even favorable to have a large neighborhood, because it makes the cryptanalysis harder, and in any case, it is more economical to remember the simple RCA that the decryption RCA is composed of than to remember the possibly large decryption RCA.

3.4. Jesus Urias – Cryptography Primitives Based On A Cellular Automaton [11]

The article „Cryptography Primitives Based on a Cellular Automaton“ [11] considers a class of *block cryptosystems* that consist of an indexed family of permutations of binary words and a deterministic generator of pseudorandom sequences of indices that are used to select sequences of permutations.

The family of permutations is

$$\Psi=\{\psi_k:M\rightarrow C \mid k\in K\}$$

With all three sets M, C, and K the set of binary words of length N, Z_2^N, each. The binary alphabet is $Z_2=\{0,1\}$. The words in M are the *clearblocks*; C is the set of enciphered words or *cipherblocks*; and the words in the set of indices K are the *enciphering keys*. For each $k\in K$, every $m\in M$ has a cipherblock expression $\psi_k(m)$.

To encrypt a long plain text, it is factored out into a succession of clearblocks m^0, m^1, m^2,... that are transformed sequentially to $\psi_{k0}(m^0)$, $\psi_{k1}(m^1)$, $\psi_{k2}(m^2)$, ... by following the sequence of keys k^0, k^1, k^2, ... yielded by the pseudorandom generator.

To disclose from the sequence of cipherblocks the plain text, we need to know the seed that was used to generate the pseudorandom sequence of keys, as to reproduce it, and have access to the family of inverse permutations $\Phi=\{\phi_k:C\rightarrow M \mid k\in K\}$ such that for every $k\in K$, $m=\phi_k(\psi_k(m))$ for every m.

The phenomenon of syncronization in coupled cellular automata is used in [9] to implement the families of permutations Ψ and Φ and a pseudorandom generator for cryptography applications. The primitives are implemented in the form of a single bidimesional array of XOR gates, the unit cipher (UC).

References

[1] E. Denning, *Cryptography and Date Security*, Addison Wesley, 1982

[2] Federal Information Processing Standards Publication 46-2, 1993 December 30 *Announcing the Standard for DATA ENCRYPTION STANDARD (DES)*

[3] Shafi Goldwasser, Mihir Bellare, *Lecture Notes on Cryptography*

[4] P. Guan, *Cellular Automaton Public-Key Cryptosystems*, Complex Systems Vol.1, 1987

[5] Howard Gutowitz, *Cryptography with Dynamical Systems*, in *Cellular Automata and Cooperative Phenomena*, Eds. E. Goles and N. Boccara, Kluwer Academic Press (1993)

[6] Howard Gutowitz, *A Massively Parallel Cryptosystem Based on Cellular Automata*, (1994)

[7] Howard Gutowitz, *Cellular Automata and the Sciences of Complexity* (part II), (November 1995)

[8] T. Habutsu, Y. Nishio, I. Sasase, S. Mori, *A Secret Key Cryptosystem by Iterating a Chaotic Map*, Proceedings of Eurocrypt ’91 (127-140), 1991

[9] Jarkko Kari, *Cryptosystems Based on Reversible Cellular Automata*, University of Turku, Finland preprint (April 1992)

[10] Stephen Wolfram, *Cryptography with Cellular Automata*, Proceedings of *Crypto ’85*, pp. 429-432 (1985)

[11] Jesus Urias, *Cryptography Primitives Based on a Cellular Automaton*, Universidad Autonoma de San Luis Potosi, Mexico

[12] Jesus Urias, Gelasio Salazar, Edgardo Ugalde, *A cryptosystem based on cellular automata*, *Chaos*, 8, 814-818 (1998)

[13] M. Mihaljevic et. al., *About Fast Dedicated One-Way Functions Based On Cellular Automata*, IEICE Trans. Fundamentals, Vol. E82-A, No. 1 January 1999