Letter to the editor

A simple linearization of the self-shrinking generator by means of cellular automata

Amparo Fúster-Sabater[a,*], M. Eugenia Pazo-Robles[b], Pino Caballero-Gil[c]

[a] Institute of Applied Physics, CSIC, 144, Serrano 28006 Madrid, Spain
[b] Instituto Tecnológico de Buenos Aires (ITBA), Av. E. Madero 399, C1106ACD Buenos Aires, Argentina
[c] Faculty of Mathematics, DEIOC, University of La Laguna, 38271 Tenerife, Spain

Keywords:
Self-shrinking generator
Cellular automata
Linearization
Stream cipher
Cryptography

ABSTRACT

In this work, it is shown that the output sequence of a well-known cryptographic generator, the so-called self-shrinking generator, can be obtained from a simple linear model based on cellular automata. In fact, such a cellular model is a linear version of a nonlinear keystream generator currently used in stream ciphers. The linearization procedure is immediate and is based on the concatenation of a basic structure. The obtained cellular automata can be easily implemented with FPGA logic. Linearity and symmetry properties in such automata can be advantageously exploited for the analysis and/or cryptanalysis of this particular type of sequence generator.

1. Introduction

Nowadays, stream ciphers are the fastest among the encryption procedures so that they are implemented in many technological applications e.g. algorithms A5 in GSM communications (see GSM webpage) or the encryption algorithm E0 (see Bluetooth specifications). From a short secret key (known only by the two interested parties) and a public algorithm (the sequence generator), stream cipher procedure consists in generating a long sequence of seemingly random bits. Such a sequence is called the *keystream sequence*. For encryption, the sender executes a bit-wise XOR operation among the bits of the plaintext and the keystream sequence. The result is the ciphertext that is going to be sent. For decryption, the receiver generates the same keystream, executes the same bit-wise XOR operation between the received ciphertext and the keystream sequence and recovers the original message.

Most keystream generators are based on maximal-length Linear Feedback Shift Registers (LFSRs) (Golomb, 1982) whose output sequences (the *PN*-sequences) are combined in a nonlinear way. Combinational generators, nonlinear filters, clock-controlled generators, multi-speed generators are just some of the most popular sequence generators with applications in cryptography. All these structures produce keystream sequences with high linear complexity, long period and good statistical properties (see Caballero-Gil & Fúster-Sabater, 2004; Fúster-Sabater, 2004).

On the other hand, bit sequences generated by a kind of one-dimensional linear binary Cellular Automata (CA) have been found (Cattell & Muzio, 1996) to be exactly the same *PN*-sequences as those of the LFSRs above mentioned. In this sense, maximal-length linear binary CA can be considered as alternative generators to the maximal-length LFSRs, as shown in Chang, Lee, Kim, and Song (1997). In fact, the current interest of these CA stems from the lack of correlation between the bit sequences generated by adjacent cells, see Cho, Un-Sook, and Yoon-Hee (2004).

The relevance of the one-dimensional binary linear CA used in this letter is due to the fact that some cryptographic generators designed as LFSR-based nonlinear structures can be modeled as CA-based linear structures. Such a result was first stated in Fúster-Sabater and Caballero-Gil (2006). Indeed, that paper might be considered a preliminary and general study where no specific generator was analyzed. On the other hand, a well known cryptographic generator, the so-called Self-Shrinking Generator (SSG) (Meier & Staffelbach, 1994) was first analyzed in Fúster-Sabater, Caballero-Gil, and Delgado (2008) with tools that are similar to the ones used here. However, this letter constitutes an advanced formalization where difference equations are defined in combination with the CA-based linearization of the SSG. In fact, the linearization procedure to convert a given SSG into a linear cellular model here proposed is quite immediate as it is to implement the cellular automaton with simple FPGA logic.

[*] Corresponding author. Tel.: +34 91 5631284; fax: +34 91 4117651.
E-mail addresses: amparo@iec.csic.es (A. Fúster-Sabater), eugepazorobles@gmail.com (M.E. Pazo-Robles), pcaballe@ull.es (P. Caballero-Gil).

The proposed idea can be generalized to other cryptographic generators similar to the SSG. Therefore, discrete synchronous neural networks, as a generalization of CA, might also be used for modeling since the local transition of a neural network applied in parallel and synchronously to all cells leads to a global transformation of the vector that describes the state of the networks, which is similar to the global map of CA.

2. Fundamentals and basic notation

First of all, different features of the two basic structures (SSG and linear binary CA) considered in this paper are briefly introduced.

2.1. *The self-shrinking generator*

The SSG was designed by Meier & Staffelbach for potential use in stream cipher applications. The SSG is attractive by its simplicity as it involves a unique LFSR in a very simple way. This generator consists of a maximal-length LFSR (Golomb, 1982) of L stages whose PN-sequence $\{c_n\}$ is self-decimated giving rise to the *self-shrunken sequence* $\{a_j\}$ or output sequence of the SSG. The decimation rule is quite simple. In fact, let (c_{2i}, c_{2i+1}) $(i = 0, 1, 2, \ldots)$ be pairs of consecutive bits of the sequence $\{c_n\}$, then we proceed as follows:

If $c_{2i} = 1$, then $a_j = c_{2i+1}$.

If $c_{2i} = 0$, then c_{2i+1} is discarded.

The key of this generator is the initial state of the LFSR and the feedback polynomial (also recommend as a part of the key). Periods, linear complexities and statistical properties (Meier & Staffelbach, 1994) make the self-shrunken sequences very adequate for their application in stream cipher. In brief, the SSG is a simplified version of the Shrinking Generator, suggested by Coppersmith, Krawczyk and Mansour (1993), which satisfies the same decimation rule but includes two maximal-length LFSRs.

2.2. *Cellular automata*

CA are particular forms of finite state machines defined as uniform arrays of identical cells in an n-dimensional space (Kari, 2005). The cells change their states (contents) synchronously at discrete time instants. The next state of each cell depends on the current states of the neighbor cells according to a *state transition rule*. In this work, our attention is focused on one-dimensional linear CA with binary contents whose time evolution is determined by two simple linear transition rules:

- rule 90 $\rightarrow$ $x_{t+1}^{i} = x_t^{i-1} \oplus x_t^{i+1}$
- rule 150 $\rightarrow$ $x_{t+1}^{i} = x_t^{i-1} \oplus x_t^{i} \oplus x_t^{i+1}$

where Wolfram's (1986) notation has been used.

Indeed, $x_{t+1}^{i}$is the content of the i-th cell at time $t + 1$ for $(i = 1, \ldots, N)$ where N represents the automaton's length and the symbol $\oplus$ the XOR logic operation. Recall that both rules are linear and that just involve the addition of either two bits (rule 90) or three bits (rule 150). The state of the automaton at time t is the binary content of the N cells at such an instant. Moreover, the CA here considered will be *hybrid* (different cells evolve under different transition rules) and *null* (cells with null content are adjacent to the automaton extreme cells). For a cellular hybrid null extreme automaton of length $N = 6$ cells, transition rules (90, 150, 90, 150, 150, 90) and initial state (0, 0, 0, 1, 1, 1), Table 1 illustrates the behavior of this structure: the formation of its output sequences $\{x_t^i\}$ $(i = 1, 2, \ldots, 6)$ (binary sequences read vertically) as well as the state succession (binary configurations of 6 bits read horizontally). All the output sequences in a state cycle have the same period, linear complexity as well as characteristic polynomial, see Fúster-Sabater and Caballero-Gil (2006).

Table 1
A linear 90/150 automaton of 6 cells.

90	150	90	150	150	90
0	0	0	1	1	1
0	0	1	0	1	1
0	1	0	0	0	1
1	1	1	0	1	0
1	1	1	0	1	1
⋮	⋮	⋮	⋮	⋮	⋮

A natural form of representation for this kind of automaton is a binary N-tuple (*rule vector*), notated $\Delta_N = (d_1, \ldots, d_N)$, where $d_i = 0$ if the i-th cell satisfies the rule 90 while $d_i = 1$ if the i-th cell satisfies rule 150. In fact, the characteristic polynomial $P_N(x)$ of an N-cell automaton can be easily obtained from its rule vector as $P_N(x) = (x + d_1)(x + d_2) \ldots (x + d_N)$. In addition, $P_N(x)$ is also the characteristic polynomial of the output sequences and determines their recurrence linear relationship.

3. Modeling the SSG in terms of CA

First self-shrunken sequences are presented as solutions of linear difference equations. Then the CA that linearize the class of SSGs are introduced.

3.1. *Self-shrunken sequences and difference equations*

According to Meier and Staffelbach (1994), over $GF(2)$ the characteristic polynomial of the self-shrunken sequence generated by a maximal-length LFSR of length L can be written as:

$$P(x) = (x + 1)^p \quad 2^{L-2} < p \leq 2^{L-1}. \tag{1}$$

This implies a linear recurrence relationship of the form:

$$(E + 1)^p \, a_n = 0. \tag{2}$$

E being the one-sided shift operator that acts on the sequence terms (i.e. $Ea_n = a_{n+1}, \; E^k a = a_{n+k}$). The Eq. (2) represents a linear binary constant coefficient difference equation whose characteristic polynomial (1) has a unique root $\lambda = 1$ with multiplicity p. The solutions of this equation are binary sequences $\{a_n\}$ whose generic term (Lidl & Niederreiter, 1986) is given by:

$$a_n = \binom{n}{0} c_0 1 + \binom{n}{1} c_1 1 + \cdots + \binom{n}{p-1} c_{p-1} 1, \tag{3}$$

where $c_i \in GF(2)$ are binary coefficients, 1 is the root with multiplicity p and the $\binom{n}{i}$ $i \geq 0$ are binomial coefficients mod 2. In fact, each binomial coefficient defines a succession of binary values with a constant period T_i. Table 2 depicts the first binomial coefficients with their corresponding binary sequences and periods.

The 2^p possible choices of coefficients c_i provide us with the different binary sequences $\{a_n\}$ that satisfy the Eq. (2). Particular choices of the c_i give rise to the self-shrunken sequences generated by SSGs of L stages. Recall that all the solutions of the difference equation (2), included the self-shrunken sequences, are just the bit-wise sum of the basic sequences coming from the binomial coefficients and weighted by the coefficients c_i.

3.2. *Self-Shrinking Generators and CA*

Now in order to model Self-Shrinking Generators in terms of CA, we proceed as follows.

Table 2
Binomial coefficients, binary sequences and periods.

Bin. coeff.	Binary seq.	T_i
$\binom{n}{0}$	1, 1, 1, 1, 1, 1, 1, 1, 1, 1, ...	1
$\binom{n}{1}$	0, 1, 0, 1, 0, 1, 0, 1, 0, 1, ...	2
$\binom{n}{2}$	0, 0, 1, 1, 0, 0, 1, 1, 0, 0, ...	4
$\binom{n}{3}$	0, 0, 0, 1, 0, 0, 0, 1, 0, 0, ...	4
$\binom{n}{4}$	0, 0, 0, 0, 1, 1, 1, 1, 0, 0, ...	8
$\binom{n}{5}$	0, 0, 0, 0, 0, 1, 0, 1, 0, 0, ...	8
$\binom{n}{6}$	0, 0, 0, 0, 0, 0, 1, 1, 0, 0, ...	8
...	...	...

Since the characteristic polynomial of a shrunken sequence is a unique factor $(x+1)$ multiplied by itself p times, it seems quite natural to construct its corresponding automaton by p successive concatenations of the basic automaton associated to the factor $(x+1)$. In fact, in Fúster-Sabater and Caballero-Gil (2006) it is proved the following relationship between corresponding characteristic polynomials $P_L(x)^{2^i}$ and rule vectors Δ_{2^iL}:

$$
\begin{aligned}
&P_L(x) \rightarrow \Delta_L = (d_1, d_2, \ldots, d_L)\\
&P_L(x)^2 \rightarrow \Delta_{2L} = (d_1, d_2, \ldots, \bar{d}_L, \bar{d}_L, \ldots, d_2, d_1)\\
&P_L(x)^4 \rightarrow \Delta_{4L} = (d_1, \ldots, \bar{d}_L, \bar{d}_L, \ldots, \bar{d}_1, \bar{d}_1, \ldots, \bar{d}_L, \bar{d}_L, \ldots, d_1)\\
&\ddots
\end{aligned}
$$

Notice that the basic automaton Δ_L associated to the factor $(x+1)$ is concatenated with its reversal version after the complementation of the last rule. Then, successive applications of this result provide us with CA of characteristic polynomials:

$$P_L(x)^2, P_L(x)^{2^2},\ P_L(x)^{2^3}, \ldots, P_L(x)^{2^q}$$

and their corresponding lengths:

$$2L, 2^2L, 2^3L, \ldots, 2^qL.$$

As the automaton corresponding to $(x+1)$ is a simple rule 150 that is $\Delta_1 = (1)$, then the application of the previous result allows us to derive the following relationships polynomials-rule vectors:

$$
\begin{aligned}
&(x+1) \rightarrow \Delta_1 = (1)\\
&(x+1)^2 \rightarrow \Delta_2 = (0, 0)\\
&(x+1)^4 \rightarrow \Delta_4 = (0, 1, 1, 0)\\
&(x+1)^8 \rightarrow \Delta_8 = (0, 1, 1, 1, 1, 1, 1, 0)\\
&\vdots\\
&(x+1)^{2^{L-1}} \rightarrow \Delta_{2^{L-1}} = (0, 1, 1, \ldots, 1, 1, 0).
\end{aligned}
$$

In this way, rule vectors corresponding to 90/150 CA whose characteristic polynomials are products of $(x+1)$ are easily obtained. The last rule vector corresponds to the required automaton. In this way, we have obtained a linear cellular automaton able to generate the self-shrunken sequences generated by SSGs of L stages.

Let us see an illustrative example.

Example. Let $\{a_j\} = \{0, 0, 0, 1, 1, 1, 1, 0\}$ be the shrunken sequence generated by a LFSR of length $L = 4$, feedback polynomial $x^4 + x + 1$ and initial state $(1, 0, 0, 0)$. The sequence $\{a_j\}$ has period $T = 8$, linear complexity $LC = 5$ and characteristic polynomial $P(x) = (x+1)^5$, see Meier and Staffelbach (1994).

Table 3
A linear 90/150 automaton generating a self-shrunken sequence.

90	150	150	150	150	150	150	90
0	0	0	1	0	1	1	1
0	0	1	1	0	0	1	1
0	1	0	0	1	1	0	1
1	1	1	1	0	0	0	0
1	1	1	0	1	0	0	0
1	1	0	0	1	1	0	0
1	0	1	1	0	0	1	0
0	0	0	0	1	1	1	1

According to the previous results, the 90/150 linear cellular automaton that generates such a sequence has as rule vector $\Delta_8 = (0, 1, 1, 1, 1, 1, 1, 0)$ and the self-shrunken sequence is a particular solution of the difference equation:

$$(E+1)^8\ a_n = 0.$$

Table 3 depicts the automaton's state succession, starting at the state $(0, 0, 0, 1, 0, 1, 1, 1)$, for which the sequence $\{a_j\}$ is generated at the most left cell (in bold) under law 90. Recall that the adjacent cell (law 150) generates exactly the same sequence but shifted one position upwards. The reverse initial state would generate at the most right cell the same sequence $\{a_j\}$.

It must be also noticed the symmetry of these CA, since sequences produced at symmetric cells are the same sequences but shifted $T/2$ positions.

Now, different considerations regarding the realization of these CA can be stated:

(1) Notice that the forms of the computed CA are standard: rule 90 at the extremes and rule 150 at all intermediate positions.

(2) The automaton $\Delta_{2^{L-1}}$ generates all the sequences that are solutions of the difference equation:

$$(E+1)^{2^{L-1}}\ x_n = 0. \tag{4}$$

That is:

$$x_n = \sum_{i=0}^{2^{L-1}-1} \binom{n}{i} c_i 1. \tag{5}$$

Thus, the self-shrunken sequences are just particular solutions with $c_i = 0\ \forall i \geq p$.

(3) The automaton $\Delta_{2^{L-1}}$ generates all the self-shrunken sequences produced by all maximal-length LFSRs of length L. In this case, the LFSR feedback polynomial is not necessary as the automaton is exactly the same. Thus, the knowledge of such a polynomial which is a part of the key is useless.

(4) The automaton $\Delta_{2^{L-1}}$ generates all the self-shrunken sequences corresponding to LFSRs of lengths $< L$. That is the longest automaton always includes all the sequences corresponding to shorter automata by starting at symmetric initial states.

(5) The implementation of these 90/150 linear models is easy and very adequate for FPGA logic. This characteristic makes it suitable for developments where time execution is relevant as in stream ciphers and in communication systems with high transmission rates.

(6) The linearity of the CA-based model as well as the encountered symmetries (see Table 3) can be exploited to mount a cryptanalytic attack based on the (partial) reconstruction of the keystream sequence from portions of intercepted sequence.

4. Conclusions

Self-shrunken sequences are particular solutions of linear difference equations and can be generated by means of a particular kind of CA. In this way, a popular cryptographic sequence generator the Self-Shrinking Generator conceived and designed as a nonlinear LFSR-based generator can be linearized in terms of cellular models. The key idea is that the characteristic polynomial of these sequences is a unique factor multiplied by itself a number of times. Therefore, the concatenation of the cellular automaton associated to such a factor allows one to easily determine the cellular model. Since this is the case for many other cryptographic sequences, the so-called interleaved sequences, the linearization procedure is general and can be applied to many cryptographic examples in a range of practical applications.

Acknowledgements

This work was supported in part by CDTI (Spain) and the companies INDRA, Unión Fenosa, Tecnobit, Visual Tool, Brainstorm, SAC and Technosafe under Project Cenit-HESPERIA as well as supported by Ministry of Science and Innovation (Spain) and European FEDER Fund under Project TIN2008-02236/TSI.

References

Bluetooth. Specifications of the Bluetooth system. Version 1.1. 2001. Retrieved from http://www.bluetooth.com/.

Caballero-Gil, P., & Fúster-Sabater, A. (2004). A wide family of nonlinear filter functions with a large linear span. *Information Sciences*, *164*(4), 197–207.

Cattell, K., & Muzio, J. C. (1996). Synthesis of one-dimensional linear hybrid cellular automata. *IEEE Transactions on Computers-Aided Design*, *15*(3), 325–335.

Chang, T., Lee, M. J., Kim, Y. H., & Song, I. (1997). Some properties of maximum length cellular automaton sequences. In *Proc. 1997 ICICS, 2* (pp. 1124–1128). IEEE.

Cho, S., Un-Sook, C., & Yoon-Hee, H. (2004). Computing phase shifts of maximum-length 90/150 cellular automata sequences. In *LNCS*: *Vol. 3305*. *Proc. ACRI'04* (pp. 31–39). Springer-Verlag.

Coppersmith, D., Krawczyk, H., & Mansour, Y. (1993). *LNCS*: *Vol. 773*. *The shrinking generator (Proc. CRYPTO'93)* (pp. 23–39). Springer-Verlag.

Fúster-Sabater, A. (2004). Run distribution in nonlinear binary generators. *Applied Mathematics Letters*, *17*(12), 1427–1432.

Fúster-Sabater, A., & Caballero-Gil, P. (2006). Concatenated automata in cryptanalysis of stream ciphers. In *LNCS*: *Vol. 4173*. *Proc. ACRI '06* (pp. 611–616). Springer-Verlag.

Fúster-Sabater, A., Caballero-Gil, P., & Delgado, O. (2008). On the use of linear cellular automata for the synthesis of cryptographic sequences. In *LNCS*: *Vol. 5271*. *Proc. HAIS 2008* (pp. 475–482). Springer-Verlag.

Golomb, S. (1982). *Shift-register sequence*. New York: Aegean Press.

GSM. (1999). Global Systems for Mobile Communications. Retrieved from http://cryptome.org/gsm-a512.html.

Kari, J. (2005). Theory of cellular automata: A survey. *Theoretical Computer Science*, *334*(3), 3–33.

Lidl, R., & Niederreiter, H. (1986). *Introduction to finite fields and their applications*. Cambridge University Press.

Meier, W., & Staffelbach, O. (1994). The Self-Shrinking Generator. In *LNCS*: *Vol. 950*. *Proc. eurocrypt'94* (pp. 205–214). Springer-Verlag.

Wolfram, S. (1986). Random sequences with cellular automata. *Advances in Applied Mathematics*, *7*, 123.