Simple Cellular Automata-Based Linear Models for the Shrinking Generator

Amparo Fúster-Sabater(1) and Dolores de la Guía-Martínez(2)
(1) Instituto de Física Aplicada, C.S.I.C.
Serrano 144, 28006 Madrid, Spain
amparo@iea.csic.es
(2) Centro Técnico de Informática, C.S.I.C.
Pinar 19, 28006 Madrid, Spain
lola@cti.csic.es

Abstract
Structural properties of two well-known families of keystream generators, Shrinking Generators and Cellular Automata, have been analyzed. Emphasis is on the equivalence of the binary sequences obtained from both kinds of generators. In fact, Shrinking Generators (SG) can be identified with a subset of linear Cellular Automata (mainly rule 90, rule 150 or a hybrid combination of both rules). The linearity of these cellular models can be advantageously used in the cryptanalysis of those keystream generators.

1 Introduction
Cellular Automata (CA) are discrete dynamic systems characterized by a simple structure but a complex behavior [1, 2, 3]. This configuration makes them very attractive to be used in the generation of pseudorandom sequences. In this sense, CA are studied in order to obtain a characterization of the rules (mapping to the next state) producing sequences with maximal length, balancedness and good distribution of 1’s and 0’s. From a cryptographic point of view, it is fundamental to analyze some additional characteristics of these generators, such as linear complexity or correlation-immunity. The results of this study point toward the equivalence between the sequences generated by CA and those obtained from Linear Feedback Shift Registers-based models [4].

In this paper, CA hybrid configurations constructed from combinations of rules 90 and 150 are considered. In fact, a linear model that describes the

---

[1] Proceedings 2003 IEEE Information Theory Workshop, pp. 143-146. La Sorbonne, Paris, 31 March - 4 April, 2003.
behavior of a kind of pseudorandom sequence generator, the so-called shrinking generator (SG), has been derived. In this way, the sequences generated by SG can be studied in terms of CA. Thus, all the theoretical background on CA found in the literature can be applied to the analysis and/or cryptanalysis of shrinking generators.

2 General description of the basic structures

The two basic structures are introduced:

2.1 The Shrinking Generator

It is a very simple generator with good cryptographic properties [5]. This generator is composed by two LFSRs: a control register, called \( R_1 \), that decimates the sequence produced by the other register, called \( R_2 \). The sequence produced by the LFSR \( R_1 \), that is \( \{a_i\} \), controls the bits of the sequence produced by \( R_2 \), that is \( \{b_i\} \), which are included in the output sequence \( \{c_j\} \) (the shrunken sequence), according to the following rule:

1. If \( a_i = 1 \Rightarrow c_j = b_i \)
2. If \( a_i = 0 \Rightarrow b_i \) is discarded.

Example 1: Consider the following LFSRs:

1. \( R_1 \) with length \( L_1 = 3 \), feedback polynomial \( 1 + D + D^3 \) and initial state \( (1,0,0) \). The sequence obtained is \( \{0,0,1,1,1,0,1\} \) with period 7.
2. \( R_2 \) with length \( L_2 = 4 \), feedback polynomial \( 1 + D^3 + D^4 \) and initial state \( (1,0,0,0) \). The sequence obtained is \( \{0,0,0,1,0,0,1,1,0,1,0,1,1,1,1\} \) with period 15.

The output sequence \( \{c_j\} \) will be determined by:

- \( \{a_i\} \rightarrow 0\,0\,1\,1\,1\,0\,1\,0\,0\,1\,1\,0\,1\,0 \ldots \)
- \( \{b_i\} \rightarrow \underline{0}\,\underline{0}\,0\,1\,0\,0\,1\,1\,\underline{1}\,0\,1\,1\,1\,1 \ldots \)
- \( \{c_j\} \rightarrow 0\,1\,0\,1\,1\,0\,1\,1 \ldots \)

The underlined bits \( \underline{0} \) or \( \underline{1} \) in \( \{b_i\} \) are discarded.

According to [5], the period of the shrunken sequence is

\[
T = (2^{L_2} - 1)2^{(L_1-1)}
\]

and its linear complexity, notated \( LC \), satisfies the following inequality

\[
L_2 \, 2^{(L_1-2)} < LC \leq L_2 \, 2^{(L_1-1)}.
\]
A simple calculation allows one to compute the number of 1’s in the shrunken sequence. Such a number is

\[ \text{No.1's} = 2^{(L_2-1)}2^{(L_1-1)}. \]  

Thus, the shrunken sequence is a quasi-balanced sequence. Since simplicity is one of its most remarkable characteristics, this scheme is suitable for practical implementation of efficient stream cipher cryptosystems.

### 2.2 Cellular Automata

An one-dimensional cellular automaton can be described as a \( n \)-cell register, whose binary stages are updated at the same time depending on a \( k \)-variable function also called rule. If \( k = 2r + 1 \) input variables are considered, then there is a total of \( 2^k \) different neighborhood configurations. Therefore, for a binary cellular automaton there can be up to \( 2^{2^k} \) different mappings to the next state. Such mappings are the different rules \( \Phi \). In fact, the next state \( x_{t+1}^i \) of the cell \( x_t^i \) depends on the current state of \( k \) neighbor cells

\[ x_{t+1}^i = \Phi(x_{t-r}^i, \ldots, x_t^i, \ldots, x_{t+r}^i) \tag{4} \]

If these functions are composed exclusively by XOR and/or XNOR operations, then CA are said to be additive. In this case, the next state \( (x_{t+1}^1, \ldots, x_{t+1}^n) \) can be computed from the current state \( (x_t^1, \ldots, x_t^n) \) such as follows:

\[ (x_{t+1}^1, \ldots, x_{t+1}^n) = (x_t^1, \ldots, x_t^n).A + C \tag{5} \]

where \( A \) is an \( n \times n \) matrix with binary coefficients and \( C \) is the complementary vector.

In CA, either all cells evolve under the same rule (uniform case) or they follow different rules (hybrid case). At the ends of the array, two different boundary conditions are possible: null automata whether cells with permanent null content are supposed adjacent to the extreme cells or periodic automata whether extreme cells are supposed adjacent.

In this paper, all automata considered will be null hybrid CA with rules 90 y 150. For \( k = 3 \), these rules are described such as follows:

- rule 90 \( \rightarrow x_{t+1}^i = x_{t-1}^i + x_{t+1}^i \)
  
  \[
  \begin{array}{cccccccc}
  111 & 110 & 101 & 100 & 011 & 010 & 001 & 000 \\
  0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\
  \end{array}
  \]

  01011010 (binary) = 90 (decimal).

- rule 150 \( \rightarrow x_{t+1}^i = x_{t-1}^i + x_t^i + x_{t+1}^i \)
  
  \[
  \begin{array}{cccccccc}
  111 & 110 & 101 & 100 & 011 & 010 & 001 & 000 \\
  1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
  \end{array}
  \]

  10010110 (binary) = 150 (decimal).

The main idea of this work is to write a given SG in terms of a hybrid cellular automaton, where at least one of its output sequences equals the SG output sequence.
3 A shrinking generator linear model in terms of cellular automata

In this section, an algorithm to determine the one-dimensional linear hybrid CA corresponding to a particular shrinking generator is presented. Such an algorithm is based on the following facts:

Fact 1: The characteristic polynomial of the shrunken sequence [5] is of the form

\[ P(D)^N \]  

where \( P(D) \) is a \( L_2 \)-degree primitive polynomial and \( N \) satisfies the inequality \( 2^{(L_1-2)} < N \leq 2^{(L_1-1)} \).

Fact 2: \( P(D) \) depends exclusively on the characteristic polynomial of the register \( R_2 \) and on the length \( L_1 \) of the register \( R_1 \). Moreover, \( P(D) \) is the characteristic polynomial of cyclotomic coset \( 2^{L_1} - 1 \), see [4]. This result can be proved in the same way as the lower bound on the LC is derived in reference [5].

Fact 3: Rule 90 at the end of the array in a null automaton is equivalent to two consecutive rules 150 with identical sequences. Reciprocally, rule 150 at the end of the array in a null automaton is equivalent to two consecutive rules 90 with identical sequences.

According to the previous facts, the following algorithm is introduced:

Input: Two LFSR’s \( R_1 \) and \( R_2 \) with their corresponding lengths, \( L_1 \) and \( L_2 \), and the characteristic polynomials \( P_2(D) \) of the register \( R_2 \).

Step 1: From \( L_1 \) and \( P_2(D) \), compute the polynomial \( P(D) \). In fact, \( P(D) \) is the characteristic polynomial of the cyclotomic coset \( E \), where \( E = 2^0 + 2^1 + \ldots + 2^{L_1-1} \). Thus, \( P(D) = (D + \alpha E)(D + \alpha^2 E)\ldots(D + \alpha^{2^{L_1-1}} E) \) \( \alpha \) being a primitive root in \( GF(2^{L_2}) \).

Step 2: From \( P(D) \), apply the Cattell and Muzio synthesis algorithm [6] to determine the two linear hybrid CA whose characteristic polynomial is \( P(D) \). Such CA are written as binary strings with the following codification: \( 0 = \text{rule 90} \) and \( 1 = \text{rule 150} \).

Step 3: For each one of the previous binary strings representing the CA, we proceed:

3.1 Complement its least significant bit. The resulting binary string is notated \( S \).

3.2 Compute the mirror image of \( S \), notated \( S^* \), and concatenate both strings \( S_c = S \ast S^* \).

3.3 Apply steps 3.1 and 3.2 to \( S_c \) recursively \( L_1 - 1 \) times.
Output: Two binary strings codifying the CA corresponding to the given SG.

Remark that the characteristic polynomial of the register $R_1$ is not needed. Due to the particular form of the shrunken sequence characteristic polynomial, it can be noticed that the computation of the CA is proportional to $L_1$ instead of $2^{L_1}$. Consequently, the algorithm can be applied to SG in a range of cryptographic interest (e.g. $L_1, L_2 \approx 64$). In order to clarify the previous steps a simple numerical example is presented.

Example 2: Consider the following LFSRs: $R_1$ with length $L_1 = 2$ and $R_2$ with length $L_2 = 5$ and characteristic polynomial $P_2(D) = 1 + D + D^3 + D^4 + D^5$.

Step 1: $P(D)$ is the characteristic polynomial of the cyclotomic coset 3. Thus, $P(D) = 1 + D^2 + D^5$.

Step 2: From $P(D)$ and applying the Cattell and Muzio synthesis algorithm, two linear hybrid CA whose characteristic polynomial is $P(D)$ can be determined. Such CA are written as:

\[
\begin{array}{ccccccc}
0 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 0 \\
\end{array}
\]

Step 3: The two binary strings of length $L = 10$ representing the required CA are:

\[
\begin{array}{ccccccc}
0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\end{array}
\]

with the corresponding codification above mentioned. The procedure has been carried out once as $L_1 - 1 = 1$.

From $L = 10$ known bits of the shrunken sequence $\{c_j\}$, the whole sequence, whose period $T = 62$, can be easily reconstructed. In fact, let $\{c_j\}$ be of the form

\[
\{c_j\} = \{0 1 0 1 1 0 1 0 0 1 ...\},
\]

then the initial state of the cellular automaton can be computed from right to left (or vice versa), according to the corresponding rules 90 and 150. Tables 1 depicts the computation of the initial state for the first automaton. The shrunken sequence is placed at the most right column.
Table 1 - The shrunken sequence is at the most right column

Once the corresponding initial states are known, then the cellular automata will produce their corresponding output sequences and the shrunken sequence can be univocally determined.

In fact, CA computed by the previous algorithm will generate all the possible sequences \( \{x_i\} \) that are solutions of the difference equation

\[
[P(E)]^{2^{k_{1}-1}} \{x_i\} = 0
\]

\( E \) being the shifting operator on \( x_i \) (i.e. \( Ex_i = x_{i+1} \)). The shrunken sequence \( \{c_j\} \) is just a particular solution of the previous equation. The different sequences \( \{x_i\} \) are distributed into the different state cycles of each automaton. Once a specific sequence is fixed in a particular cell, e.g. the shrunken sequence at the most right cell in the previous example, the location of the other sequences is univocally determined. In addition, every particular solution \( \{x_i\} \) can be generated by every automaton cell depending on the state cycle considered. In terms of LFSR-based generators, the solution sequences \( \{x_i\} \) correspond to sequences generated by different combinations of LFSRs: clock-controlled shrinking generators [7], shrinking generators with distinct rules of decimation, irregular clocking of the register \( R_2 \) based on particular stages of the register \( R_1 \) etc. All these generators are included in a simple automaton.

4 Applications of the CA-based model to the cryptanalysis of the shrinking generator

Since a linear model describing the behavior of the SG has been derived, the cryptanalysis of such a generator can be considered from different points of view:

- Cryptanalysis based on the SG linear complexity: attacking the SG through its linear complexity requires the knowledge of \( LC \) bits of the shrunken sequence, \( LC \) being its linear complexity, or equivalently, the length of the cellular automaton. Remark that this is just half the sequence required by the Berlekamp-Massey algorithm [8] to reconstruct the original sequence.
Crytanalysis based on the Linear Consistency Test (LCT): The linear consistency test [9] is a general divide-and-conquer cryptanalytic attack that can be applied to the SG on the basis of the linear models provided by the cellular automata. This attack would require the exhaustive search through all possible initial states of the LFSR \( R_2 \).

A new attack that exploits the weaknesses inherent to the CA-based linear model can be also considered. Such an attack will be specified in next section.

5 Phaseshift analysis of CA sequences

If the Bardell's algorithm to phaseshift analysis of CA [10] is applied, then it is possible to calculate the relation among the sequences obtained from CA. In fact, in [10] it was shown that the characteristic equation determines the recursion relationship among the bits in the output sequences of a hybrid 90/150 CA. A shift operator was used in conjunction with a table of discrete logarithms to determine the phaseshift analytically.

Although the characteristic equation in [10] is a primitive polynomial \( P(D) \), it can be proved that the algorithm is valid for \( P(D)^n \) too.

Example 3: Let us consider a CA with the following characteristics:

- The automaton length \( L = 10 \)
- Rule distribution: 0011001100
- \( P(D) = (1 + D + D^2 + D^4 + D^5)^2 \).

Let \( S \) be a shift operator defined in \( GF(2) \) which operates on \( X_i \), the state of the \( i \)-th cell, such as follows:

\[
X_i(t + 1) = SX_i(t) \quad (8)
\]

we can write

\[
X_1(t + 1) = X_2(t) \quad (9)
\]
as

\[
SX_1(t) = X_2(t) \quad (10)
\]
or simply

\[
SX_1 = X_2. \quad (11)
\]
The difference equation system is as follows:

\[
\begin{align*}
SX_1 &= X_2 \\
SX_2 &= X_1 + X_3 \\
SX_3 &= X_2 + X_3 + X_4 \\
SX_4 &= X_3 + X_4 + X_5 \\
SX_5 &= X_4 + X_6 \\
SX_6 &= X_5 + X_7 \\
SX_7 &= X_6 + X_7 + X_8 \\
SX_8 &= X_7 + X_8 + X_9 \\
SX_9 &= X_8 + X_{10} \\
SX_{10} &= X_9
\end{align*}
\]

Expressing each \(X_i\) as a function of \(X_{10}\), we obtain the following system:

\[
\begin{align*}
X_1 &= (S^9 + S^4 + S^3 + S^2 + S + 1)X_{10} \\
X_2 &= (S^8 + S^6 + S^5 + S^4 + S^3 + S + 1)X_{10} \\
X_3 &= (S^7 + S^6 + S^5 + S^3 + 1)X_{10} \\
X_4 &= (S^6)X_{10} \\
X_5 &= (S^5 + S^3 + 1)X_{10} \\
X_6 &= (S^4 + S)X_{10} \\
X_7 &= (S^3 + S^2 + 1)X_{10} \\
X_8 &= (S^2 + 1)X_{10} \\
X_9 &= (S)X_{10}
\end{align*}
\]

Now taking logarithms in both sides,

\[
\begin{align*}
\log(X_1) &= \log(S^9 + S^4 + S^3 + S^2 + S + 1) + \log(X_{10}) \\
\log(X_2) &= \log(S^8 + S^6 + S^5 + S^4 + S^3 + S + 1) + \\ &\quad \log(X_{10}) \\
\log(X_3) &= \log(S^7 + S^6 + S^5 + S^3 + 1) + \log(X_{10}) \\
\log(X_4) &= \log(S^6) + \log(X_{10}) \\
\log(X_5) &= \log(S^5 + S^3 + 1) + \log(X_{10}) \\
\log(X_6) &= \log(S^4 + S) + \log(X_{10}) \\
\log(X_7) &= \log(S^3 + S^2 + 1) + \log(X_{10}) \\
\log(X_8) &= \log(S^2 + 1) + \log(X_{10}) \\
\log(X_9) &= \log(S) + \log(X_{10})
\end{align*}
\]

On the other hand, we have:

\[
D^{26} \mod P(D) = D^2 + 1. \tag{12}
\]
Next we define,

\[ \log(D) \equiv 1 \]  \hspace{1cm} (13)

According to the algorithm proposed by Bardell, we can identify the following equations:

\[
\begin{align*}
\log(X_9) - \log(X_{10}) &= 1 \\
\log(X_8) - \log(X_{10}) &= 26 \\
\log(X_4) - \log(X_{10}) &= 6 \\
\end{align*}
\]

and,

\[
\begin{align*}
\log(X_2) - \log(X_1) &= 1 \\
\log(X_3) - \log(X_1) &= 26 \\
\log(X_7) - \log(X_1) &= 6 \\
\end{align*}
\]

According to the previous results, the same binary sequence is generated in cells 1, 2, 3 and 7 as well as the same sequence is produced in cells 10, 9, 8 and 4. The phaseshifts of the outputs 2, 3 and 7 relative to cell 1 are 1, 26 and 6 respectively. Similar values are obtained in the other group of cells, that is cells 4, 8 and 9 relative to cell 10. The rest of cells generate different sequences.

Studying the distance among the shifted sequences and concatenating them, the original sequence can be reconstructed. Nevertheless, the shifts among the different shrunken sequences depend on the particular structure of the automaton considered. In fact, once the automaton is known the Bardell’s algorithm has to be applied.

6 Conclusions

In this paper, the relationship between LFSR-based structures and cellular automata have been stressed. More precisely, a particular family of LFSR-based keystream generators, the so-called Shrinking Generators, has been analyzed and identified with a subset of linear cellular automata. In fact, a linear model describing the behavior of the SG has been derived.

The algorithm to convert the SG into a CA-based linear model is very simple and can be applied to shrinking generators in a range of practical interest. The key idea of this algorithm is that the number of steps to be carried out is proportional to \( L_1 \) instead of \( 2^{L_1} \).

Once the linear equivalent model has been developed, the linearity of this cellular model can be advantageously used in the analysis and/or cryptanalysis of the SG. Besides the traditional cryptanalytic attacks (e.g. the linear complexity attack that here requires half the sequence needed by the Berlekamp-Massey algorithm and the LCT attack), an outline of a new attack that exploits the weaknesses inherent to these CA has been introduced too.
The proposed linear model is believed to be a very useful tool to analyze the strength of the sequence produced by a SG as a keystream generator in stream ciphers procedures.

References

[1] O. Martin, A.M. Odlyzko and S. Wolfram, "Algebraic properties of Cellular Automata", Commun. Math. Phys., Vol. 93, 1984, pp. 219-258.

[2] S. Nandi, B.K. Kar and P.P. Chaudhuri, "Theory and applications of cellular automata in Cryptography", IEEE Transactions on Computers, Vol. 43, 1994, pp. 1346-1357.

[3] A.K. Das, A. Ganguly, A. Dasgupta, S. Bhawmik and P.P. Chaudhuri, "Efficient characterisation of cellular automata", IEE Proc., Part E, Vol. 1, 1990, pp. 81-87.

[4] S. Golomb, Shift register sequences (revised edition), Aegean press, 1982.

[5] D. Coppersmith, H. Krawczyk and Y. Mansour, "The Shrinking Generator", Advances in Cryptology–CRYPTO'93, LNCS 773, 1994, pp. 22-39.

[6] K. Cattell and J. Muzio, "Analysis of One-Dimensional Linear Hybrid Cellular Automata over GF(q)", IEEE Transactions on Computers, Vol. 45, No. 7, 1996, pp. 782-792.

[7] A.A. Kanso, "Clock-Controlled Shrinking Generator", Proc. of the VII RECSI (Spanish Meeting on Cryptology and Information Security), September 5-7, Oviedo (Spain), Vol. II, 2002, pp. 447-460.

[8] J.L. Massey, "Shift-Register Synthesis and BCH Decoding", IEEE Trans. Inform. Theory, Vol. 15, 1969.

[9] K. Zeng, C. Hang and T. Rao, "On the linear consistency test (LCT) in cryptanalysis with applications", Advances in Cryptology–CRYPTO'89, LNCS 435, 1990, pp. 164-174.

[10] P.H. Bardell, "Analysis of Cellular Automata Used as Pseudorandom Pattern Generators", Proceedings of the IEEE International Test Conference, Paper 34.1, 1990, pp. 762-768.

[11] J. Golic, "Correlation Analysis of the Shrinking Generator", Advances in Cryptology–CRYPTO'2001, LNCS 2139, 2001, pp. 440-457.