DOI: 10.1007/978-3-319-42085-1_40

Modelling the MSSG in Terms of Cellular Automata

Sara D. Cardell[1(✉)] and Amparo Fúster-Sabater[2]

[1] Instituto de Matemática, Estatística e Computação Científica, UNICAMP, Campinas, Brazil
sdcardell@ime.unicamp.br
[2] Instituto de Tecnologías Físicas y de la Información (CSIC), 144, Serrano, 28006 Madrid, Spain
amparo@iec.csic.es

Abstract. The modified self-shrinking generator is a non-linear cryptographic sequence generator designed to be used in hardware implementations. In this work, the output sequence of such a generator is obtained as one of the output sequences of a linear model based on Cellular Automata. Although irregularly decimated generators have been conceived and designed as non-linear sequence generators, in practice they can be easy modelled in terms of simple linear structures.

Keywords: Modified self-shrinking generator · Cellular automata · rule 102 · rule 60 · Stream cipher · Cryptography

1 Introduction

Nowadays stream ciphers are the fastest among the encryption procedures. They are designed to generate, from a short key, a long sequence (*keystream sequence*) of seemingly random bits. Typically, a stream cipher consists of a keystream generator whose output sequence is bit-wise XORed with the plaintext (in emission) to obtain the ciphertext or with the ciphertext (in reception) to recover the original plaintext. Some well known designs in stream ciphers can be found in [4,13].

Most keystream generators are based on maximal-length Linear Feedback Shift Registers (LFSRs) [6]. Such registers are linear structures characterized by their length L, their characteristic polynomial $p(x)$ and their initial state IS (currently the key of the cryptosystem). Their output sequences, the so-called PN-sequences, are usually combined in a non-linear way in order to break their inherent linearity and produce sequences of cryptographic application. Combinational generators, nonlinear filters, irregularly decimated generators or LFSRs with dynamic feedback are just some of the most popular keystream generators that can be found in the literature [11,12].

Irregularly decimated generators produce good cryptographic sequences characterized by long periods, adequate correlation, excellent run distribution, balancedness, simplicity of implementation, etc. The underlying idea of this kind

O. Gervasi et al. (Eds.): ICCSA 2016, Part I, LNCS 9786, pp. 514–520, 2016.
DOI: 10.1007/978-3-319-42085-1_40

of generators is the irregular decimation of a PN-sequence according to the bits of another one. The result of this decimation is a binary sequence that will be used as keystream sequence in the cryptographic procedure of the stream cipher. Inside the family of irregularly decimated generators, the self-shrinking generator (SSG) was first introduced by Meier and Staffelbach in [10]. This keystream generator is a simplified version of the shrinking generator (SG) introduced by Coppersmith *et al.* in [2] that used two different LFSRs. In contrast to the SG, the SSG is based on a unique maximal-length LFSR that generates via a self-decimation process a pseudorandom binary keystream.

In [5] the authors modelled the output sequence of the SSG, the so-called self-shrunken sequence, by means of cellular automata (CA) that used rules 90 and 150. Later, in [1] a new family of cellular automata modelled this sequence through simpler and shorter cellular automata that used rule 102.

The modified self-shrinking generator (MSSG) was recently discover by Kanso [8]. It involves a unique maximal-length LFSR and uses an extended decimation rule based on the XORed value of a pair of bits. In this work, we model the modified self-shrunken sequence in terms of CA by using rule 102 too.

The paper is organized as follows: in Sect. 2 we provide some fundamentals and basic concepts. Section 3 shows how to model the modified self-shrunken sequence via cellular automata. In Sect. 4, we see how to recover the self-shrunken sequence by using a fixed number of intercepted bits. Finally, conclusions in Sect. 5 end the paper.

2 Preliminaries

Notation and basic concepts that will be used throughout the work are now introduced.

2.1 Modified Self-shrinking Generator

The **modified self-shrinking generator** (MSSG) was introduced by Kanso in 2010 for hardware implementations [8]. It is a special case of the self-shrinking generator [10], where the PN-sequence generated by a maximum-length LFSR is self-decimated. Here the decimation rule is very simple and can be described as follows: Given three consecutive bits $\{u_{2i}, u_{2i+1}, u_{2i+2}\}$, $i = 0, 1, 2, \ldots$ of a PN-sequence $\{u_i\}$, the output sequence $\{s_j\}$ is computed as

$$\begin{cases} \text{If } u_{2i} + u_{2i+1} = 1 \text{ then } s_j = u_{2i+2} \\ \text{If } u_{2i} + u_{2i+1} = 0 \text{ then } u_{2i+2} \text{ is discarded.} \end{cases}$$

We call the $\{s_j\}$ sequence as the **modified self-shrunken sequence**. If L is the length of the maximum-length LFSR that generates $\{u_i\}$, then the linear complexity LC of the corresponding modified self-shrunken sequence satisfies:

$$2^{\lfloor \frac{L}{3} \rfloor - 1} \leq LC \leq 2^{L-1} - (L-2),$$

and the period T of the sequence satisfies:

$$2^{\lfloor \frac{L}{3} \rfloor} \leq T \leq 2^{L-1}$$

as proved in [8]. As usual, the key of this generator is the initial state of the register that generates the PN-sequence $\{u_i\}$. Moreover, the characteristic polynomial of the register, $p(x)$, is also recommended to be part of the key.

Next a simple illustrative example is introduced.

Example 1. Consider the LFSR of length $L = 3$ with characteristic polynomial $p(x) = 1 + x^2 + x^3$ and initial state $IS = (1\ 0\ 0)$. The PN-sequence generated is $1001110\ldots$ with period $T = 2^3 - 1$.

Now the modified self-shrunken sequence can be computed as follows:

$$R: \underbrace{1\ 0}_{1}\ \textcircled{\textbf{0}}\ \underbrace{1\ 1}_{0}\ \cancel{\underbrace{0\ 1}_{1}}\ \textcircled{\textbf{0}}\ \underbrace{0\ 1}_{1}\ \textcircled{\textbf{1}}\ \underbrace{1\ 0}_{1}\ \textcircled{\textbf{1}}\ \underbrace{0\ 0}_{0}\ \cancel{\underbrace{1\ 1}_{0}}\ \cancel{0}\ldots$$

This sequence $0011\ldots$ has period $T = 4$ and its characteristic polynomial is $p_s(x) = (1 + x)^3$. Thus, the linear complexity of this modified self-shrunken sequence is $LC = 3$. ■

2.2 Cellular Automata

Cellular automata (CA) are discrete models where the contents of the cells (binary in our work) are updated following a function of k variables [14] called *rule*. The value of the cell in position i at time $t + 1$, notated x_i^{t+1}, depends on the value of the k neighbour cells at time t. If these rules used in the CA are composed exclusively by XOR operations, then the CA is said to be **linear**. In this work, the CA considered are **regular** (every cell follows the same rule) and **null** (null cells are considered adjacent to extreme cells). For $k = 3$, the rule 102 is given by:

$$\textbf{Rule 102: } x_i^{t+1} = x_i^t + x_{i+1}^t$$

111	110	101	100	011	010	001	000
0	1	1	0	0	1	1	0

According to Wolfram's terminology, the name rule 102 is due to the fact that 01100110 is the binary representation of the decimal number 102. In Table 1, we can find an example of a linear regular null 102-CA with initial state (0 0 1).

CA have been used for many cryptographic applications. In fact, many authors have proposed stream ciphers based on CA [3,7].

3 Modelling the Modified Self-shrunken Sequence in Terms of CA

The aim of this section is to construct a family of CA that generates the modified self-shrunken sequence as one of their output sequences.

Table 1. Example of linear regular null one-dimensional 102-CA of length 3

102	**102**	**102**
0	0	1
0	1	1
1	0	1
1	1	1
⋮	⋮	⋮

The characteristic polynomial of the modified self-shrunken sequence has the form $p_s(x) = (1+x)^{LC}$, where LC is the linear complexity of the sequence. Notice that the characteristic polynomial of the self-shrunken sequence has the same form [5].

Lemma 1. *Let $\{s_i\}$ be a binary sequence whose characteristic polynomial is $(1+x)^t$. Then, the characteristic polynomial of the sequence $\{u_i\}$, where $u_i = s_i + s_{i+1}$, is $(1+x)^{n-1}$.*

Now, we can introduce the next result.

Theorem 1. *Given a modified self-shrunken sequence, there exists a linear, regular, null 102-CA that generates such a sequence in its most left column. The length of such a CA is LC.*

Example 2. Consider the primitive polynomial $p(x) = 1 + x^3 + x^5$. Consider the LFSR with $p(x)$ as characteristic polynomial and initial state $IS = (1\ 0\ 0\ 0\ 0)$. The corresponding modified self-shrunken sequence is given by:

$$0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0$$

This sequence has period $T = 16$ and it is possible to check that its characteristic polynomial corresponds to $p_s(x) = (1+x)^{13}$. Therefore, the linear complexity of this sequence is $LC = 13$, and there exists a CA of length 13 that generates it (see Table 2). ■

The other sequences in the CA have a well determined structure:

- The sequence in the most right column is the identically 1 sequence.
- Then, there are 2^{i-1} sequences of period 2^i, for $1 \leq i \leq L-2$.
- Finally, there are $LC - 2^{L-2}$ sequences of period 2^{L-1} (including the modified self-shrunken sequence).

This is also due to the form of the characteristic polynomial, which is the same as that of the self-shrinking generator [1].

In Table 2, it is possible to check that the most right sequence is the identically 1 sequence. Next, there are one sequence of period 2, two sequences of period 4, four sequences of period 8 and five sequences of period 16.

On the other hand, it is worth noticing that rule 60, given by

Table 2. 102-CA of length 13 that produces the modified self-shrunken sequence generated in Example 2

102	**102**	**102**	**102**	**102**	**102**	**102**	**102**	**102**	**102**	**102**	**102**	**102**
0	0	1	0	0	0	0	0	1	1	1	1	1
0	1	1	0	0	0	0	1	0	0	0	0	1
1	0	1	0	0	0	1	1	0	0	0	1	1
1	1	1	0	0	1	0	1	0	0	1	0	1
0	0	1	0	1	1	1	1	0	1	1	1	1
0	1	1	1	0	0	0	1	1	0	0	0	1
1	0	0	1	0	0	1	0	1	0	0	1	1
1	0	1	1	0	1	1	1	1	0	1	0	1
1	1	0	1	1	0	0	0	1	1	1	1	1
0	1	1	0	1	0	0	1	0	0	0	0	1
1	0	1	1	1	0	1	1	0	0	0	1	1
1	1	0	0	1	1	0	1	0	0	1	0	1
0	1	0	1	0	1	1	1	0	1	1	1	1
1	1	1	1	1	0	0	1	1	0	0	0	1
0	0	0	0	1	0	1	0	1	0	0	1	1
0	0	0	1	1	1	1	1	1	0	1	0	1

Rule 60: $x_i^{t+1} = x_{i-1}^t + x_i^t$

111	110	101	100	011	010	001	000
0	0	1	1	1	1	0	0

generates exactly the same CA sequences but in reverse order. For example, in Table 3, we have a 60-CA that generates the same sequences as those of the 102-CA in Table 1. In brief, we have defined two different linear, regular, null 102-CA and 60-CA able to generate the modified self-shrunken sequence.

Table 3. Example of linear regular null one-dimensional 60-CA of length 3

60	**60**	**60**
1	0	0
1	1	0
1	0	1
1	1	1
⋮	⋮	⋮

4 Recovering the Modified Self-shrunken Sequence from Intercepted Bits

Assume LC is the linear complexity of the modified self-shrunken sequence. Given $2 \cdot LC$ intercepted bits, it is possible to determine the shortest LFSR that generates such a sequence by means of the Berlekamp-Massey algorithm [9].

We know that there exists a CA of length LC that generates the modified self-shunken sequence. Besides, we know that its most right sequence is always the identically 1 sequence. Therefore, it is enough to intercept $LC - 1$ bits of the modified self-shrunken sequence to recover the initial state of the CA and, consequently, to recover the complete sequence. Notice that this quantity is half the needed bits to apply the Berlekamp-Massey algorithm. Note that due to the recent design of this generator, no other approaches have been developed yet.

In Example 2, the modified self-shrunken sequence had period 16 and linear complexity 13. In Table 4, we can see that intercepting 12 bits of the self-shrunken sequence (bits in bold), we can recover the initial state of the CA (in grey) and, thus, the complete sequence.

Table 4. Bits needed to recover the initial state of the 102-CA given in Example 2

102	102	102	102	102	102	102	102	102	102	102	102	102
0	0	1	0	0	0	0	0	1	1	1	1	1
0	1	1	0	0	0	0	1	0	0	0		
1	0	1	0	0	0	1	1	0	0			
1	1	1	0	0	1	0	1	0				
0	0	1	0	1	1	1	1					
0	1	1	1	0	0	0						
1	0	0	1	0	0							
1	0	1	1	0								
1	1	0	1									
0	1	1										
1	0											
1												

5 Conclusions

Cryptographic generators based on irregular decimation were conceived as non-linear sequence generators. However, the sequences generated by these type of generators can be modelled as the output sequences of linear CA.

In this work, it is shown that the sequences generated by the modified self-shrinking generator are also output sequences of one-dimensional, linear, regular and null cellular automata based on rules 102 and 60. At the same time, the number of intercepted bits required by this 102/60 CA is half the number of bits needed by the Berlekamp-Massey algorithm to reconstruct the original sequence.

A natural extension of this work is the generalization of this procedure to many other cryptographic sequences, the so-called interleaved sequences, as they present similar structural properties to those of the sequences obtained from irregular decimation generators.

Acknowledgment. The work of the first author was supported by FAPESP with number of process 2015/07246-0. The work of the second author was supported by both Ministerio de Economía, Spain, under grant TIN2014-55325-C2-1-R (ProCriCiS), and Comunidad de Madrid, Spain, under grant S2013/ICE-3095-CM (CIBERDINE).

References

1. Cardell, S.D., Fúster-Sabater, A.: Linear models for the self-shrinking generator based on CA. J. Cell. Automata **11**(2–3), 195–211 (2016)
2. Coppersmith, D., Krawczyk, H., Mansour, Y.: The shrinking generator. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 22–39. Springer, Heidelberg (1994)
3. Das, S., RoyChowdhury, D.: Car30: a new scalable stream cipher with rule 30. Crypt. Commun. **5**(2), 137–162 (2013)
4. eSTREAM: the ECRYPT Stream Cipher Project, ECRYPT II, eSTREAM portfolio. http://www.ecrypt.eu.org/stream/
5. Fúster-Sabater, A., Pazo-Robles, M.E., Caballero-Gil, P.: A simple linearization of the self-shrinking generator by means of cellular automata. Neural Netw. **23**(3), 461–464 (2010)
6. Golomb, S.W.: Shift Register-Sequences. Aegean Park Press, Laguna Hill (1982)
7. Jose, J., Das, S., Chowdhury, D.R.: Inapplicability of fault attacks against trivium on a cellular automata based stream cipher. In: Was, J., Sirakoulis, G.C., Bandini, S. (eds.) ACRI 2014. LNCS, vol. 8751, pp. 427–436. Springer, Heidelberg (2014)
8. Kanso, A.: Modified self-shrinking generator. Comput. Electr. Eng. **36**(1), 993–1001 (2010)
9. Massey, J.L.: Shift-register synthesis and BCH decoding. IEEE Trans. Inf. Theory **15**(1), 122–127 (1969)
10. Meier, W., Staffelbach, O.: The self-shrinking generator. In: De Santis, A. (ed.) EUROCRYPT 1994. LNCS, vol. 950, pp. 205–214. Springer, Heidelberg (1995)
11. Menezes, A.J., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press, Boca Raton (1996)
12. Paar, C., Pelzl, J.: Understanding Cryptography. Springer, Heidelberg (2010)
13. Robshaw, M., Billet, O. (eds.): New Stream Cipher Designs. LNCS, vol. 4986. Springer, Heidelberg (2008)
14. Wolfram, S.: Cellular automata as models of complexity. Nature **311**(5985), 419–424 (1984)