Chapter 3
Modelling Through Linear Cellular Automata

The irregular decimation was introduced to break the linearity of the PN-sequences. However, in this chapter we will see that there exist linear structures that describe the behaviour of the shrinking generators, designed as non-linear. The inherent linearity of these structures can be used to cryptanalyse such generators as described in Chap. 4.

3.1 The Concept of Cellular Automaton

Cellular automata (CAs) are particular forms of finite state machines defined as uniform arrays of identical cells in an n-dimensional state. A cellular automaton (CA) evolves in discrete time steps, within the content of one cell being affected by the contents of cells in its neighbourhood on the previous time step. That is, the value of the ith cell at time $t+1$, denoted by x_i^{t+i}, depends on the contents of the k neighbour cells at time t.

One-dimensional CAs with $k = 3$ and with contents in the binary field are called elementary CAs. There are 2^3 possible configurations for each cell and its two immediate neighbours. The rule defining the cellular automaton must specify the resulting state for each of these possibilities so there are 2^{2^3} possible rules for elementary CA evolution. These rules can be considered as Boolean functions.

Stephen Wolfram proposed a scheme, known as the Wolfram code, to assign each rule a number from 0 to 255 [103]. Each possible current configuration of three neighbour cells is written in the order, 111, 110, … , 001, 000, and the resulting state for each configuration is written in the same order and interpreted as the binary

S. Díaz Cardell, A. Fúster-Sabater, *Cryptography with Shrinking Generators*, SpringerBriefs in Mathematics, https://doi.org/10.1007/978-3-030-12850-0_3

representation of an integer. For example, one can find the four rules we will use in this work below:

Rule 150: $x_i^{t+1} = x_{i-1}^t + x_i^t + x_{i+1}^t$							
111	110	101	100	011	010	001	000
1	0	0	1	0	1	1	0

Rule 90: $x_i^{t+1} = x_{i-1}^t + x_{i+1}^t$							
111	110	101	100	011	010	001	000
0	1	0	1	1	0	1	0

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

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

Notice that 10010110, 01011010, 01100110 and 00111100 are the binary representations of 150, 90, 102 and 60, respectively.

Many of the rules seem to generate patterns with evident structures. For example, Fig. 3.1 shows the AC-images generated by these four rules after applying 15 iterations to the one-dimensional CA. One can notice the symmetry between rules 60 and 102 and that both rules generate a fractal structure. Rules 150 and 90 produce symmetric structures and are both additive rules. Every additive rule is able to emulate itself and produce nested patters [103].

Observe, for example, Rules 30 and 94, which are non-linear:

Rule 30: $x_i^{t+1} = x_{i-1}^t + x_i^t + x_{i+1}^t + x_i^t x_{i+1}^t$							
111	110	101	100	011	010	001	000
0	0	0	1	1	1	1	0

Rule 94: $x_i^{t+1} = x_{i-1}^t + x_i^t + x_{i+1}^t + +x_{i-1}^t x_i^t + x_i^t x_{i+1}^t + x_{i-1}^t x_i^t x_{i+1}^t$							
111	110	101	100	011	010	001	000
0	1	0	1	1	1	1	0

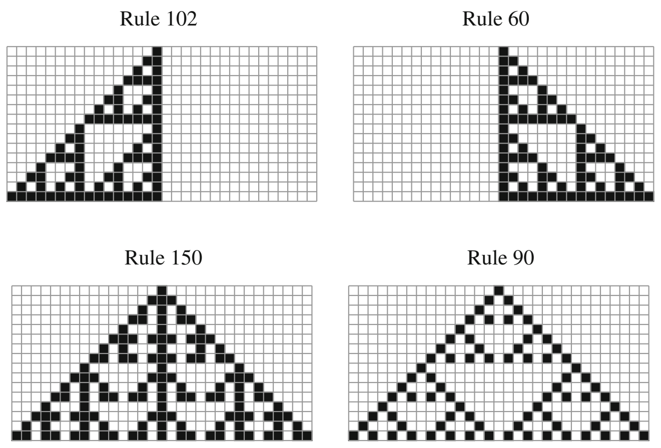

Fig. 3.1 AC-images generated with Rules 102, 60, 150 and 90

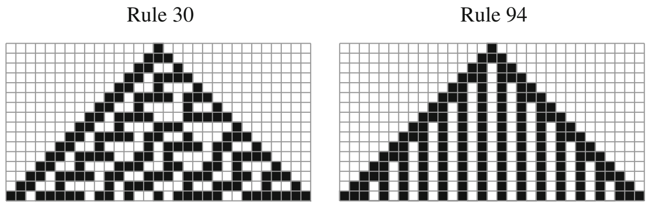

Fig. 3.2 AC-images generated by Rules 30 and 94

The AC-image generated by Rule 30 shows no recognizable pattern (see Fig. 3.2). On the other hand, Rule 94 is an example of simple CA whose evolution corresponds to computations that can be easily described in traditional mathematical terms. Patterns can show both for linear rules (e.g., Rules 60 and 102) and for non-linear rules (e.g., Rule 94).

When the rules involved in the CA use only XOR operations, the CA is said to be **linear**. Notice that Rules 60, 102, 150 and 90 use only XOR operations. This means that we will only consider linear CAs in this chapter.

Due to their capability to exhibit complex behaviours, CAs have applications in many different areas, for example, in modelling physical systems [58, 100] and non-linear chemical systems [102], studying problems of number theory [86, 102] or as pseudorandom number generators [97].

Furthermore, due to the speed and randomness in their sequences, CAs are a very good basis for stream ciphers. What is more, their hardware implementation is simple and their regular structure makes possible to find an efficient software implementation. The first cryptographic application of CAs was published in [101]. In this work, Wolfram used Rule 30 for building a stream cipher that was afterwards broken by Meier and Stafflebach [66]. Besides, other authors have proposed stream ciphers based on CAs along the years [20, 51, 73].

Next, we classify the elementary CAs.

Definition 3.1 An elementary CA is said to be:

- **Uniform** or **regular** if every cell is computed using the same rule.
- **Hybrid** if different rules are considered when computing the contents of the cells.
- **Null** if cells with null content are adjacent to the extreme cells when it is needed.
- **Periodic** or **cyclic** if extreme cells are adjacent.

In Table 3.1a we can find an example of a regular, cyclic 102-CA of length 3. Furthermore, since Rule 102 only operates the contents of a cell and its right neighbour cell, we consider cyclic boundary only on the right of the CA in order to compute the last vertical sequence. Given the initial state {1 1 0}, the CA generates as many new states of length 3 as we want in the following way:

$$\begin{tikzcd}
1 \arrow[d] & 1 \arrow[d] \arrow[dl] & 0 \arrow[d] \arrow[dl] & \mid \; 1 \arrow[dl] \\
\oplus \arrow[d] & \oplus \arrow[d] & \oplus \arrow[d] & \\
0 & 1 & 1 &
\end{tikzcd}$$

Table 3.1 Examples of elementary CAs

(a) Regular cyclic 102-CA

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

(b) Regular cyclic 60-CA

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

(c) Hybrid null 150/90-CA

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

However, at some point, these states start to recur; thus, the CA generates 3 (vertical) output sequences with period 3.

In Table 3.1b, we find a regular, cyclic 60-CA of length 3. Since Rule 60 only operates the contents of a cell and its left neighbour cell, in this case we consider cyclic boundary only on the left of the CA. Note that the (vertical) output sequences generated by this 60-CA are the same (vertical) sequences generated in the 102-CA in Table 3.1a (they appear in inverse order).

Finally, in Table 3.1c, we can find one example of hybrid null 150/90-CA of length 4. In this case, we have to consider null boundary in both sides of the CA. Besides, the CA generates four (vertical) output sequences with period 7.

In general, the (vertical) sequences generated by a 102-CA (60-CA) have different periods. In addition, due to the symmetry between rules 102 and 60, the sequences generated by a 102-CA of length L can be also generated by a 60-CA of length L.

3.2 Modelling a PN-Sequence

In this section we will see how to obtain PN-sequences by means of elementary linear CAs. We recall that a PN-sequence is a sequence generated by an LFSR whose characteristic polynomial is primitive.

3.2.1 Cattell–Muzio Algorithm

In [13], Cattell and Muzio presented a method for computing a 90/150-CA that generates the same sequences as those produced by a given irreducible characteristic polynomial. This approach is based on a correspondence between the characteristic polynomial calculations and GCD computations. In fact, they proved that each irreducible polynomial has exactly two CA realizations.

First of all we need to recall the definition of trace of a polynomial.

Definition 3.2 ([13]) The trace of a polynomial $q(x)$ with respect to an irreducible polynomial $p(x)$ of degree L is given by

$$\mathrm{Tr}(q(x)) = \left[q(x) + q(x)^2 + q(x)^4 + \cdots + q(x)^{2^{L-1}}\right] \bmod p(x).$$

It is worth noticing that the trace of a polynomial is always zero or one.

Example 3.1 Consider the polynomial $p(x) = x^2$ and the primitive polynomial $p(x) = 1 + x^2 + x^5$. First, we compute the powers of $q(x)$:

$$\begin{aligned} q(x)^2 &= x^4 \\ q(x)^4 &= x^8 \bmod p(x) = 1 + x^2 + x^3 \\ q(x)^8 &= \left(1 + x^4 + x^6\right) \bmod p(x) = 1 + x + x^3 + x^4 \\ q(x)^{16} &= \left(1 + x^2 + x^6 + x^8\right) \bmod p(x) = x. \end{aligned}$$

Summing these polynomials, we find the trace of $q(x)$:

$$\mathrm{Tr}(q(x)) = q(x) + q(x)^2 + q(x)^4 + q(x)^8 + q(x)^{16} = 0.$$

■

The method given in Algorithm 1 shows the necessary process to compute a CA for a given irreducible characteristic polynomial $p(x)$ of degree L. This algorithm is very easy to code in languages such as Maple, Python, etc. As a consequence of Algorithm 1, we can introduce the following result.

Theorem 3.1 ([28]) *For a PN-sequence generated by a primitive polynomial of degree L, there exists an hybrid, null 150/90-CA of length L that generates such PN-sequence.*

Example 3.2 Consider the primitive polynomial $p(x) = 1+x^2+x^5$. Since primitive implies irreducible, we can apply Algorithm 1.

Algorithm 1 Cattell–Muzio algorithm

```
Input:    An irreducible polynomial $p(x)$
01:  Compute $f(x) = (x^2 + x)\, p'(x)$
02:  Compute $g(x) = (1/f(x))^2$
03:  if $L$ is even
04:     Find $\theta(x)$ with trace 1
05:     Compute $\beta(x) = \sum_{i=1}^{L-1} \left(\sum_{j=0}^{i-1} g^{2^j}\right) \theta^{2^i}$
06:  else
07:     Compute $\beta(x) = \sum_{i=1}^{(L-1)/2} g^{2^{2i-1}}$
08:  endif
09:  $q(x) = \beta(x) f(x)$
10:  Compute gcd$(p(x), q(x))$, saving the quotients
11:  Construct the CA from the constant terms of the quotients
Output:
     A binary string of length $L$ codifying a CA corresponding to the PN-sequence generated
by $p(x)$
```

We compute the derivative of $p(x)$ modulo 2:

$$p'(x) \bmod 2 = \left(2x + 5x^4\right) \bmod 2 = x^4.$$

Now we compute $f(x)$ modulo $p(x)$:

$$f(x) = \left(x + x^2\right) p'(x) = \left(x^5 + x^6\right) \bmod p(x) = 1 + x + x^2 + x^3.$$

Next, we use the extended Euclidean GCD algorithm to compute the inverse of $f(x)$:

$$1/f(x) = 1 + x^2 + x^3.$$

We compute $g(x)$

$$g(x) = (1/f(x))^2 = \left(1 + x^4 + x^6\right) \bmod p(x) = 1 + x + x^3 + x^4$$

and the powers of $g(x)$:

$$\begin{aligned}
g^2(x) &= \left(1 + x^2 + x^6 + x^8\right) \bmod p(x) = x \\
g^4(x) &= x^2 \\
g^8(x) &= x^4 \\
g^{16}(x) &= x^8 \bmod p(x) = x^3 + x^2 + 1.
\end{aligned}$$

Summing $g(x)$ and its powers we get that the trace of $g(x)$ with respect to $p(x)$ is zero.

Since $L = 5$ odd, we compute $\beta(x)$ as follows:

$$\beta(x) = \sum_{i=1}^{2} g(x)^{2^{2i-1}} = g(x)^2 + g(x)^8 = x + x^4.$$

Finally,

$$\begin{aligned} q(x) &= \left(x + x^2\right) p'(x)\beta(x) \\ &= \left(x + x^2\right) x^4 \left(x + x^4\right) \\ &= 1 + x^2 + x^4. \end{aligned}$$

Now, we apply the Euclid's algorithm to search $\gcd(p(x), q(x))$:

$$\begin{aligned} 1 + x^2 + x^5 &= \left(1 + x^2 + x^4\right) \mathbf{x} + \left(1 + x + x^2 + x^3\right) \\ 1 + x^2 + x^4 &= (\mathbf{1} + \mathbf{x}) \left(1 + x + x^2 + x^3\right) + x^2 \\ 1 + x + x^2 + x^3 &= (\mathbf{1} + \mathbf{x})x^2 + (1 + x) \\ x^2 &= (\mathbf{1} + \mathbf{x})(1 + x) + 1 \\ 1 + x &= (\mathbf{1} + \mathbf{x})1 + 0. \end{aligned}$$

This process returns the quotients

$$[x, 1 + x, 1 + x, 1 + x, 1 + x]$$

and so the CA is constructed from the constant terms

$$[0, 1, 1, 1, 1]. \tag{3.1}$$

Now, we substitute 0 and 1 by 90 and 150, respectively. Thus the CA given by [90, 150, 150, 150, 150] generates the PN-sequences produced by $p(x)$.

Now, we consider the mirror image of (3.1)

$$[1, 1, 1, 1, 0]$$

that represents the CA

$$[150\ 150\ 150\ 150\ 90]$$

that also generates the PN-sequences produced by $p(x)$.

Table 3.2 Null 105/90-CA that generates the PN-sequence produced by $p(x) = 1 + x^2 + x^5$

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

For instance, consider the PN-sequence

$$\{1\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 0\ \ldots\}$$

generated by $p(x)$ with initial state $\{1\ 1\ 1\ 1\ 1\}$, this sequence can be generated as well by the 150/90-CA given in Table 3.2. ■

The algorithm is sufficiently fast for practical applications and the number of operations does not depend on the input polynomial, only on its degree.

3.2.2 Other CAs that Generate PN-Sequences

In this section, we show that for every PN-sequence there also exists a 102-CA that generates it.

We start the section with an important result about PN-sequences that will be needed afterwards.

Theorem 3.2 ([9, Theorem 3.6]) *For a PN-sequence $\{a_i\}$ generated by a primitive polynomial $p(x)$ of degree L, there exists a unique number $D \in \{2, 3, \ldots, 2^L - 2\}$ such that $a_i + a_{i+1} = a_{i+D}$. This number is $D = \mathscr{Z}_\alpha(1)$, with $\alpha \in \mathbb{F}_{2^L}$ a root of $p(x)$.*

It is worth recalling that $\mathscr{Z}_\alpha(1)$ is the Zech logarithm of 1 in basis α (see Definition 2.1).

According to Theorem 3.2 and the general form of a 102-CA (see Table 3.3), if the PN-sequence $\{a_i\}$ appears in the 0th column of a 102-CA, the other sequences are shifted versions of such PN-sequence. Furthermore, the sequence in the tth column it is a shifted version of $\{a_i\}$, that is, $\{a_{i+d}\}$, with $d = t \cdot D \bmod (2^L - 1)$. Eventually, the PN-sequence $\{a_i\}$ itself will appear again; thus, the 102-CA has finite length. The general form of the columns of a 102-CA can be found in Sect. 4.6.1 (Method 2).

Next result, whose proof is left as an exercise, claims that given a primitive polynomial there always exists a 102-CA that generates the PN-sequence produced by such polynomial.

Theorem 3.3 *There exists a regular, cyclic 102-CA of length $\frac{2^L-1}{\gcd(D,2^L-1)}$, with D as in Theorem 3.2 that generates the same PN-sequence as that produced by a primitive polynomial $p(x)$ of degree L.*

As an example, consider the PN-sequence generated by $p(x) = 1+x^2+x^5$ given in Example 3.2. There exists a regular, cyclic 102-CA of length 31 that generates such PN-sequence (see Table 3.4). What is more, all the sequences are shifted versions of the same PN-sequence. Since the characteristic polynomial of the PN-sequence is $p(x) = 1 + x^2 + x^5$, it is easy to check that $D = 18$. This means that the shift from one sequence to the following is 18. For example, the sequence in the first column is a shifted version of the PN-sequence in the 0th column, but starting

Table 3.3 General form of a 102-CA

102	**102**	**102**	**102**	**102**	...	**102**	...
a_0	$a_0 + a_1$	$a_0 + a_2$	$a_0 + a_1 + a_2 + a_3$	$a_0 + a_4$	...	$a_0 + a_8$	...
a_1	$a_1 + a_2$	$a_1 + a_3$	$a_1 + a_2 + a_3 + a_4$	$a_1 + a_5$	...	$a_1 + a_9$	...
a_2	$a_2 + a_3$	$a_2 + a_4$	$a_2 + a_3 + a_4 + a_5$	$a_2 + a_6$	...	$a_2 + a_{10}$	...
a_3	$a_3 + a_4$	$a_3 + a_5$	$a_3 + a_4 + a_5 + a_6$	$a_3 + a_7$	...	$a_3 + a_{11}$	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$	

Table 3.4 102-CA that generates the PN-sequence produced by $p(x) = 1 + x^2 + x^5$

102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102	102
1	$\textcircled{0}$	$\boxed{0}$	0	0	1	1	1	0	0	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	1	0	1	0	1
1	0	0	0	1	0	0	1	0	1	0	1	1	0	0	0	0	1	1	1	0	0	1	1	0	1	1	1	1	1	0
1	0	0	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	1	0	1	0	1	1	0	0	0	0	1	1
1	0	1	0	1	1	0	0	0	0	1	1	1	0	0	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0
1	1	1	1	0	1	0	0	0	1	0	0	1	0	1	0	1	1	0	0	0	0	1	1	1	0	0	1	1	0	1
$\boxed{0}$	0	0	1	1	1	0	0	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	1	0	1	0	1	1	0
0	0	1	0	0	1	0	1	0	1	1	0	0	0	0	1	1	1	0	0	1	1	0	1	1	1	1	1	0	1	0
0	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	1	0	1	0	1	1	0	0	0	0	1	1	1	0
1	0	1	1	0	0	0	0	1	1	1	0	0	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	1	0
1	1	0	1	0	0	0	1	0	0	1	0	1	0	1	1	0	0	0	0	1	1	1	0	0	1	1	0	1	1	1
0	1	1	1	0	0	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	1	0	1	0	1	1	0	0	0
1	0	0	1	0	1	0	1	1	0	0	0	0	1	1	1	0	0	1	1	0	1	1	1	1	1	0	1	0	0	0
1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	1	0	1	0	1	1	0	0	0	0	1	1	1	0	0	1
1	1	0	0	0	0	1	1	1	0	0	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	1	0	1	0
0	1	0	0	0	1	0	0	1	0	1	0	1	1	0	0	0	0	1	1	1	0	0	1	1	0	1	1	1	1	1
1	1	0	0	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	1	0	1	0	1	1	0	0	0	0	1
0	1	0	1	0	1	1	0	0	0	0	1	1	1	0	0	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0
1	1	1	1	1	0	1	0	0	0	1	0	0	1	0	1	0	1	1	0	0	0	0	1	1	1	0	0	1	1	0
$\textcircled{0}$	0	0	0	1	1	1	0	0	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	1	0	1	0	1	1
0	0	0	1	0	0	1	0	1	0	1	1	0	0	0	0	1	1	1	0	0	1	1	0	1	1	1	1	1	0	1
0	0	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	1	0	1	0	1	1	0	0	0	0	1	1	1
0	1	0	1	1	0	0	0	0	1	1	1	0	0	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	1
1	1	1	0	1	0	0	0	1	0	0	1	0	1	0	1	1	0	0	0	0	1	1	1	0	0	1	1	0	1	1
0	0	1	1	1	0	0	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	1	0	1	0	1	1	0	0
0	1	0	0	1	0	1	0	1	1	0	0	0	0	1	1	1	0	0	1	1	0	1	1	1	1	1	0	1	0	0
1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	1	0	1	0	1	1	0	0	0	0	1	1	1	0	0
0	1	1	0	0	0	0	1	1	1	0	0	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	1	0	1
1	0	1	0	0	0	1	0	0	1	0	1	0	1	1	0	0	0	0	1	1	1	0	0	1	1	0	1	1	1	1
1	1	1	0	0	1	1	0	1	1	1	1	1	0	1	0	0	0	1	0	0	1	0	1	0	1	1	0	0	0	0
0	0	1	0	1	0	1	1	0	0	0	0	1	1	1	0	0	1	1	0	1	1	1	1	1	0	1	0	0	0	1
0	1	1	1	1	1	0	1	0	0	0	1	0	0	1	0	1	0	1	1	0	0	0	0	1	1	1	0	0	1	1

in position 18 (see circled bits in Table 3.4). The sequence in the second column is a shifted version of the PN-sequence in the 0th column but starting in position $2 \cdot 18 \bmod 31$, that is, in position 5 (see squared bits in Table 3.4) and so on.

Note that when $\gcd(D, 2^L - 1) = 1$, the length of the 102-CAs mentioned in Theorem 3.3 is $2^L - 1$. The length of the 150/90-CAs proposed in Sect. 3.2.1 is much smaller. However, if we know $p(x)$ and the PN-sequence $\{a_i\}$, we can compute D as in Theorem 3.2 and we can complete the 102-CA with the corresponding shifted versions of $\{a_i\}$. In addition, since the 102-CA proposed in Theorem 3.3 is regular, every cell follows the same rule and the form of the CA is immediately obtained. On the other hand, in order to find the form of the 150/90-CA in Sect. 3.2.1, we need to apply the Cattell–Muzio Algorithm [13].

3.3 Modelling the Shrinking Generator

In this section, we present two different families of linear CAs that generate the shrunken sequence produced by two maximum-length LFSRs. From now on, we denote by $p_1(x)$ and $p_2(x)$ of degrees L_1 and L_2, the primitive characteristic polynomials of such registers, respectively.

3.3.1 The Fúster–Caballero Algorithm

In [28] the authors proposed an algorithm that provides a 150/90-CA that generates the shrunken sequence produced by two maximum-length LFSRs. This approach is based on the Cattell–Muzio Algorithm [13] seen in Sect. 3.2.1 and a CA-concatenation technique.

Algorithm 2 provides two hybrid, null 150/90-CAs that produce the shrunken sequence generated by $p_1(x)$ and $p_2(x)$. Actually, the algorithm is based on the concatenation of the CA produced applying the Cattell–Muzio algorithm for $p_2(x)$ [13].

Notice that $p_1(x)$ only contributes the number of concatenations according to its degree. This polynomial is no further implicated in the algorithm, this means that with $p_2(x)$ fixed, for different values of $p_1(x)$ with degree L_1 the algorithm would return the same result.

Example 3.3 Consider the primitive polynomial $p_2(x) = 1 + x + x^2 + x^4 + x^5$ and a primitive polynomial $p_1(x)$ of degree 3.

First, we compute $N = 1 + 2 + 4 = 7$ and the polynomial

$$p(x) = \left(x + \alpha^7\right)\left(x + \alpha^{14}\right)\left(x + \alpha^{28}\right)\left(x + \alpha^{56}\right)\left(x + \alpha^{112}\right) = 1 + x^2 + x^5,$$

where α is a primitive element of $\mathbb{F}_{2^5}$, root of $p_2(x)$.

Algorithm 2 Fúster–Caballero algorithm

```
Input:    $L_1$ and $p_2(x)$
01:  Compute $N = 2^0 + 2^1 + 2^2 + \cdots + 2^{L_1-1}$
02:  Compute $p(x) = \left(x + \alpha^N\right)\left(x + \alpha^{2N}\right)\cdots\left(x + \alpha^{2^{L_2-1}N}\right)$, with $\alpha$ root of $p_2(x)$
03:  Compute two linear 90/150 CA, denoted by $s_i$, $i = 1, 2$, for $p(x)$ using the Cattell-Muzio
     algorithm
04:   for $j = 1 : L_1 - 1$
05:      Complement the last bit of $s_i$ and denote the resultant string as $t_i$
06:      Compute de mirror image of $t_i$, denoted by $t_i^*$ and concatenate both strings: $s_i = t_i t_i^*$
07:   endfor
Output:
     Two binary strings of length $L_2 \cdot 2^{L_1-1}$ codifying two CAs corresponding to the shrinking
     generator
```

Now, applying the Cattell–Muzio algorithm to $p(x)$, we obtain two strings that represent two 150/90-CAs (see Example 3.2):

$$[01111] \rightarrow [90\ 150\ 150\ 150\ 150]$$

$$[11110] \rightarrow [150\ 150\ 150\ 150\ 90].$$

We choose, for example, the first CA and we perform the concatenation process $L_1 - 1 = 2$ times:

$$\begin{array}{l}[01111]\\ [0111001110]\\ [01110011111111001110]\,.\end{array}$$

For the second CA, we proceed in the same manner:

$$\begin{array}{l}[11110]\\ [1111111111]\\ [11111111100111111111]\,.\end{array}$$

Now, we substitute 0 and 1 by 90 and 150, respectively, and we obtain two CAs

$$[90\ 150\ 150\ 150\ 90\ 90\ 150\ 150\ 150\ 150\ 150\ 150\ 150\ 150\ 90\ 90\ 150\ 150\ 150\ 90]$$

$$[150\ 150\ 150\ 150\ 150\ 150\ 150\ 150\ 150\ 90\ 90\ 150\ 150\ 150\ 150\ 150\ 150\ 150\ 150\ 150]\,,$$

both of them capable of generating the shrunken sequence generated by $p_2(x)$ and $p_1(x)$. ■

3.3.2 Other CAs that Generate the Shrunken Sequence

In [9], the authors proposed a family of 102-CAs (60-CAs) that also generate the shrunken sequence.

Again, consider two primitive polynomials $p_1(x)$ and $p_2(x)$ of length L_1 and L_2, respectively. We can introduce the following result.

Theorem 3.4 ([9, Theorem 3.10]) *The shrunken sequence generated by $p_1(x)$ and $p_2(x)$ can be generated by a regular, cyclic 102-CA of length $\frac{T}{\gcd(2^{L_2}-1,D)}$, where $D = \mathscr{L}_\alpha(1)$, with $\alpha \in \mathbb{F}_{2^{L_2}}$ root of $p(x)$ (see Theorem 2.1) and $T = 2^{L_1-1}(2^{L_2}-1)$ is the period of the shrunken sequence.*

Apart from the shrunken sequence, other $2^{L_1-1} - 1$ sequences, *the companion sequences*, with the same period and characteristic polynomial as those of the shrunken sequence are generated by the 102-CA [9]. Furthermore, shifted versions of these sequences, including the shrunken sequence, appear along the 102-CA.

Notice that the sequences in columns $t \cdot 2^{L_1-1}$, with $t = 1, 2, \ldots, L/(2^{L_1-1}-1)$, are shifted versions of the shrunken sequence, with shift equal to $t \cdot D \cdot 2^{L_1-1}$, for $t = 1, 2, \ldots, L/(2^{L_1-1} - 1)$ [9, 11]. Moreover, the companion sequence in the column $t \cdot 2^{L_1-1} + m$, for $m = 1, 2, \ldots, 2^{L_1-1} - 1$ and $t = 0, 1, \ldots, L/(2^{L_1-1}) - 1$, is a shifted version of the companion sequence in the mth column starting in position $t \cdot D \cdot 2^{L_1-1}$[11].

Example 3.4 Consider the shrunken sequence generated by $p_1(x) = 1 + x + x^2$ and $p_2(x) = 1 + x + x^3$, in Example 2.2:

$$\{1\,0\,1\,1\,1\,0\,0\,0\,1\,1\,0\,1\,0\,1\ldots\}.$$

This sequence has characteristic polynomial $p(x)^2 = \left(1 + x^2 + x^3\right)^2$ and period $T = 14$. In Table 3.5 there is an example of a regular, cyclic 102-CA of length 14 that generates this sequence in the 0th column. This CA generates 2 different sequences, the shrunken sequence and one companion sequence, both with the same period and characteristic polynomial. Shifted versions of these two sequences appear 7 times along the 102-CA: the shrunken sequence appears in columns 0, 2, 4, 6, 8, 10 and 12, and the companion sequence appears in columns, 1, 3, 5, 7, 9, 11 and 13.

Now, we can compute $2^{L_1-1} = 2$ and $D = \mathscr{L}_\alpha(1) = 5$, with $\alpha \in \mathbb{F}_{2^3}$ root of $p(x)$. We consider, for example, the 2nd column of the 102-CA. In this case $t = 1$, therefore this sequence is a shifted version of the shrunken sequence with shift equal to $2^{L_1-1} \cdot t \cdot D \bmod 14 = 10$ (see the circled bit of the shrunken sequence in Table 3.5). Consider now, for example, the 9th column of the 102-CA. Since now $t = 4$, the considered sequence is a shifted version of the companion sequence with shift equal to $2^{L_1-1} \cdot t \cdot D \bmod 14 = 12$ (see the squared bit of the shrunken sequence in Table 3.5). ■

In Sect. 2.1.2, we saw that the shrunken sequence is composed of interleaving shifted versions of a PN-sequence generated by the primitive polynomial $p(x)$.

Table 3.5 CA that generates the shrunken sequence in Example 2.2

102	**102**	**102**	**102**	**102**	**102**	**102**	**102**	**102**	**102**	**102**	**102**	**102**	**102**
1	1	$\textcircled{0}$	1	0	0	1	0	0	$\boxed{1}$	1	0	1	1
0	1	1	1	0	1	1	0	1	0	1	1	0	0
1	0	0	1	1	0	1	1	1	1	0	1	0	0
1	0	1	0	1	1	0	0	0	1	1	1	0	1
1	1	1	1	0	1	0	0	1	0	0	1	1	0
0	0	0	1	1	1	0	1	1	0	1	0	1	1
0	0	1	0	0	1	1	0	1	1	1	1	0	1
0	1	1	0	1	0	1	1	0	0	0	1	1	1
$\boxed{\mathbf{1}}$	0	1	1	1	1	0	1	0	0	1	0	0	1
1	1	0	0	0	1	1	1	0	1	1	0	1	0
$\textcircled{\textbf{0}}$	1	0	0	1	0	0	1	1	0	1	1	1	1
1	1	0	1	1	0	1	0	1	1	0	0	0	1
0	1	1	0	1	1	1	1	0	1	0	0	1	0
1	0	1	1	0	0	0	1	1	1	0	1	1	0

As a consequence of the formation rule of the 102-CA and the fact that summing elements of a PN-sequence generates another PN-sequence [41], it is possible to check that the companion sequences are also composed of interleaving shifted versions of the same PN-sequence. We leave this claim as an exercise for the reader.

Let us denote the interleaved PN-sequences of the shrunken sequence by $\left\{v_{d_0^0+i}\right\}, \left\{v_{d_1^0+i}\right\}, \left\{v_{d_2^0+i}\right\}, \ldots, \left\{v_{d^0_{2^{L_1-1}-1}+i}\right\}$, $i = 0, 1, \ldots$, where $d_0^0 = 0$. Remember that the positions d_k^0 depend on the location of the 1s in the PN-sequence $\{a_i\}$ generated by the first register R_1 (see Sect. 2.1.5).

Now, for the first companion sequence, let us denote the interleaved PN-sequences by $\left\{v_{d_0^1+i}\right\}, \left\{v_{d_1^1+i}\right\}, \left\{v_{d_2^1+i}\right\}, \ldots, \left\{v_{d^1_{2^{L_1-1}-1}+i}\right\}$, $i = 0, 1, \ldots$,. We can compute these new positions using Rule 102 and the definition the Zech logarithm as follows:

$$d_k^1 = \mathscr{Z}_\alpha\left(d_k^0 - d_{k+1}^0\right) + d_{k+1}^0, k = 0, 1, \ldots, 2^{L_1-1} - 2,$$

$$d^1_{2^{L_1-1}-1} = \mathscr{Z}_\alpha\left(d^0_{2^{L_1-1}-1} - 1\right) + 1.$$

Similarly, we can compute the shift positions for the jth companion sequence, $j = 1, 2, \ldots, L - 1$ as

$$d_k^j = \mathscr{Z}_\alpha\left(d_k^{j-1} - d_{k+1}^{j-1}\right) + d_{k+1}^{j-1}, k = 0, 1, \ldots, 2^{L_1-1} - 2,$$

$$d^j_{2^{L_1-1}-1} = \mathscr{Z}_\alpha\left(d^{j-1}_{2^{L_1-1}-1} - 1\right) + 1.$$

Recall that the sequence in the column $t \cdot 2^{L_1-1} + m$, for $m = 0, 1, 2, \ldots, 2^{L_1-1} - 1$ and $t = 0, 1, \ldots, L/(2^{L_1-1}) - 1$, is a shifted version of the sequence in the mth column starting in position $t \cdot D \cdot 2^{L_1-1}$. Therefore, we have that:

$$d_k^{t \cdot 2^{L_1-1}+m} = d_k^m + t \cdot D \bmod (2^{L_2} - 1)$$

for $k = 0, 1, \ldots, 2^{L_1-1}-1, m = 0, 1, \ldots, 2^{L_1-1}-1$ and $t = 0, 1, \ldots, L/(2^{L_1-1})-1$.

This means that the positions d_k^j for the companion sequence in the jth column with $j \geq 2^{L_1-1}$ can be computed easily using the positions d_i^s, with $0 \leq s < 2^{L_1-1}$ and without using logarithms.

Example 3.5 Consider again Example 3.4. If we decimate the shrunken sequence and the companion sequence in the 102-CA by distance 2, we obtain that both sequences are composed of interleaving shifted versions of the PN-sequence $\{1\ 1\ 1\ 0\ 1\ 0\ 0\ \ldots\}$ generated by $p(x) = 1 + x^2 + x^3$ (see Table 3.6a and b).

What is more, the positions of both PN-sequences of the shrunken sequence with respect to its first interleaved PN-sequence are $d_0^0 = 0$ and $d_1^0 = 5$, respectively. The positions of the interleaved PN-sequences of the companion sequence with respect to the first PN-interleaved sequence of the shrunken sequence are $d_0^1 = 2$ and $d_1^1 = 4$.

Let us consider the sequence in the 2nd column of the 102-CA. We have seen that this sequence is a shifted version of the shrunken sequence with shift equal to 10. We know that $t = 1$ and $D = 5$, so the two interleaved PN-sequences of this sequence are shifted versions of the first interleaved PN-sequence of the shrunken sequence (Table 3.6c) starting in positions:

$$d_0^2 = d_0^0 + D \cdot 1 \bmod 7 = 5 \quad \text{and} \quad d_1^2 = d_1^0 + D \cdot 1 \bmod 7 = 1, \text{respectively.}$$

Consider again the sequence in the 9th column of the 102-CA. This sequence was a shifted version of the companion sequence, with shift equal to 12. We know that $t = 4$ and $D = 5$, so the two interleaved PN-sequences of this sequence are

Table 3.6 Interleaved PN-sequences of the shrunken sequence and the companion sequences of Example 2.2

(a)			(b)		(c)		(d)	
	1	**0**	$\underline{1}$	$\underline{\underline{1}}$	$\textcircled{0}$	$\boxed{1}$	**0**	$\textcircled{0}$
$d_1^2=1\leftarrow$	$\boxed{1}$	1	0	0	0	1	1	0
$d_0^1=2\leftarrow$	$\underline{1}$	0	1	0	1	0	0	1
$d_1^0=d_0^9=3\leftarrow$	**0**	0	0	1	1	1	0	1
$d_1^1=4\leftarrow$	$\underline{\underline{1}}$	1	0	1	1	0	1	1
$d_0^2=d_0^9=5\leftarrow$	$\textcircled{0}$	1	1	1	0	0	1	0
	0	1	1	0	1	1	1	1

the same as the first interleaved PN-sequence of the shrunken sequence (Table 3.6d) starting in positions:

$$d_0^9 = d_0^1 + D \cdot 3 \bmod 7 = 3 \quad \text{and} \quad d_1^9 = d_1^1 + D \cdot 3 \bmod 7 = 5, \text{respectively.}$$

■

3.3.3 *Comparison of both Families*

In Sect. 3.3.1, we showed that the Fúster–Caballero algorithm produces an hybrid, null 150/90-CA that generates the shrunken sequence. Given two maximal-length LFSRs, this algorithm performs first the Cattell–Muzio algorithm explained in Sect. 3.2.1 and carries out a concatenation procedure to find the CA that generates the shrunken sequence. This fact makes impossible to predict the form of the CA, which depends on L_1 and $p_2(x)$.

In Sect. 3.3.2 since the 102-CAs (60-CAs) are regular we do not need to perform any computations to find the form of the CA; we only need to find its length. On the other hand, according to Theorem 3.4, the length of the 102-CA is, at the most, $T = 2^{L_1-1}\left(2^{L_2}-1\right)$ $\left(\text{when } \gcd\left(2^{L_2}-1, T\right)=1\right)$, which is greater than $\left(2^{L_1}-1\right)L_2$ (the length of the 150-90-CA given in Sect. 3.3.1). However, the 102-CAs generate 2^{L_1-1} different sequences, the other sequences are shifted versions of these, which is an advantage compared to the 90/150-CA.

As a conclusion, we can say that the 102-CAs are longer but this disadvantage becomes less relevant when we notice the complex process developed in the Fúster–Caballero algorithm to obtain the 150/90-CAs. Besides, this length is reduced to 2^{L_1-1}, since the first 2^{L_1-1} sequences repeat along the 102-CA.

3.4 Modelling the Generalized Self-Shrinking Generator

Since we have seen that the sequences produced by the MSSG and the SSG are sequences produced by the GSSG (Sect. 2.4.3), in this section we only consider the families of CAs that generate the generalized self-shrunken sequences. We recall that the GSS-sequences are a family of sequences generated by a maximum-length LFSR of L stages. We also recall that the characteristic polynomial of the GSS-sequences is of the form $p_t(x) = (1+x)^t$, with $0 < t \leq 2^{L-1} - (L-2)$.

3.4.1 *Characterization of the 150/90-CA*

In this section we present a family of 150/90-CA that generates the family of GSS-sequences.

Table 3.7 GSS-sequences generated by $q(x) = 1 + x^3 + x^4$

	G				$S(G)$								$p_n(x)$
0	0	0	0	0	0	0	0	0	0	0	0	0	$p_1(x)$
1	0	0	0	1	**0**	**0**	**0**	**1**	**1**	**0**	**1**	**1**	$p_6(x)$
2	0	0	1	0	0	0	1	1	1	1	0	0	$p_5(x)$
3	0	0	1	1	0	0	1	0	0	1	1	1	$p_6(x)$
4	0	1	0	0	0	1	1	1	0	0	1	0	$p_6(x)$
5	0	1	0	1	0	1	1	0	1	0	0	1	$p_5(x)$
6	0	1	1	0	0	1	0	0	1	1	1	0	$p_6(x)$
7	0	1	1	1	0	1	0	1	0	1	0	1	$p_2(x)$
8	1	0	0	0	1	1	1	1	1	1	1	1	$p_1(x)$
9	1	0	0	1	1	1	1	0	0	1	0	0	$p_6(x)$
10	1	0	1	0	1	1	0	0	0	0	1	1	$p_5(x)$
11	1	0	1	1	1	1	0	1	1	0	0	0	$p_6(x)$
12	1	1	0	0	1	0	0	0	1	1	0	1	$p_6(x)$
13	1	1	0	1	1	0	0	1	0	1	1	0	$p_5(x)$
14	1	1	1	0	1	0	1	1	0	0	0	1	$p_6(x)$
15	1	1	1	1	1	0	1	0	1	0	1	0	$p_2(x)$

Theorem 3.5 ([30]) *Given a generalized self-shrunken sequence of period 2^t, $0 \leq t \leq 2^{L-1}$, there exists an hybrid, null 150/90-CA of length 2^t that generates such sequence. Furthermore, the CA will have the form*

$$[90\ \ 150\ \ 150\ \ldots\ 150\ \ 150\ \ 90].$$

Example 3.6 Given a primitive polynomial $p(x) = x^4 + x^3 + 1 \in \mathbb{F}_2[x]$ and an initial state {1 1 1 1}, the PN-sequence generated is {1 1 1 1 0 1 0 1 1 0 0 1 0 0 0 ...}. In Table 3.7, it is possible to see the 16 GSS-sequences generated by this PN-sequence. We choose, for example, the sequence number corresponding to $G = 1$, {0 0 0 1 1 0 1 1 ...}. This sequence has period 8; therefore, there exists a 105/90-CA with length 8 and form

$$[90\ \ 150\ \ 150\ \ 150\ \ 150\ \ 150\ \ 150\ \ 90]$$

that generates such a sequence (see Table 3.8a). ■

3.4.2 *Characterization of the 102-CA*

We start this section with two minor results, whose proofs can be found in [8].

Table 3.8 CAs that generate the GSS-sequence of Example 3.6

(a) 150/90-CA

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

(b) 102-CA

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

(c) 60-CA

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

Lemma 3.1 ([8, Lemma 2]) *Let $\{u_i\}$ be a binary sequence whose characteristic polynomial is $(x+1)q(x) \in \mathbb{F}_2[x]$. Then, $q(x)$ generates the sequence $\{v_i\}$, where $v_i = u_i + u_{i+1}$.*

Lemma 3.2 ([8, Theorem 1]) *Let $\{u_i\}$ be a binary sequence whose characteristic polynomial is $p_t(x)$. Then, the characteristic polynomial of the sequence $\{v_i\}$, where $v_i = u_i + u_{i+1}$, is $p_{t-1}(x)$.*

Due to the previous results, we can introduce the following theorem that gives us the length of the CAs that generate the GSS-sequences.

Theorem 3.6 ([10, Theorem 6]) *Given a GSS-sequence with characteristic polynomial $p_t(x)$, there exists a regular, null 102-CA of length t that generates such sequence.*

Recall that the previous results are similar for rule 60. In this case, the 60-CA provides the same sequences but obtained in reverse order. Let us see an illustrative example.

Example 3.7 Consider the GSS-sequence corresponding to $G = 1$ generated by an LFSR with characteristic polynomial $p(x) = 1 + x^3 + x^4$ in Example 3.6:

$$\{0\,0\,0\,1\,1\,0\,0\,1\,1\,\ldots\}.$$

According to Theorem 3.6, there exists a regular, null 102-CA of length 6 that generates this sequence as one of its output (vertical) sequences (see Table 3.8b). The characteristic polynomial of this sequence is $p_6(x)$ and thus its linear complexity is $LC = 6$. It is possible to check that the characteristic polynomials of the remaining sequences in the CA are $p_5(x)$, $p_4(x)$, $p_3(x)$, $p_2(x)$ and $p_1(x)$, respectively (consequence of Lemma 3.2). This means that the linear complexity of the (vertical) sequences generated by the null 102-CA is strictly decreasing.

Recall that there exists a 60-CA of length 6 that generates the same exact sequences in inverse order (see Table 3.8c). Therefore, all the results here obtained can be applied to the 60-CA model. ■

Additionally, as a consequence of Lemmas 3.1 and 3.2, these CAs have a well-defined structure which is given in the following result.

Theorem 3.7 ([10, Theorem 7]) *Consider a GSS-sequence with linear complexity LC. The regular, null* 102*-CA that generates such a sequence also produces:*

- *The identically 1 sequence (with period 1) in the rightmost column,*
- 2^{i-1} *sequences of period* 2^i*, for* $1 \leq i \leq L-2$ *and*
- $LC - 2^{L-2}$ *sequences of period* 2^{L-1} *(including the considered GSS-sequence).*

For example, in Table 3.8b we have a 102-CA of length 6 that generates six (vertical) sequences: the identically 1 sequence, one sequence with period 2, two sequences with period 4 and finally, two sequences with period 8 (including the given GSS-sequence).

Comparing the 90/150-CAs given in Sect. 3.4.1 with the 102-CAs (60-CAs) proposed in this section, it can be stated that:

1. Both proposals provide CAs with a defined structure. For the 90/150-CAs, Rule 90 is applied to the extreme cells, while Rule 150 is applied to the remaining cells:

$$[90\ 150\ 150\ \ldots\ 150\ 150\ 90].$$

   The 102-CAs are regular; therefore, the same rule is applied for all the cells and the form of the CAs is very simple:

$$[102\ 102\ \ldots\ 102\ 102].$$

2. The length of the proposed 90/150-CAs is 2^{L-1}. On the other hand, the length of the 102-CAs (60-CAs) is the linear complexity of the GSS-sequence considered. We claimed, without proving, that $2^{L-2} < LC \leq 2^{L-1} - (L-2)$. Therefore, the improvement on the length is not much significant.
3. Finally, in the 90/150-CA model, all the cells (except extreme cells) use Rule 150, which involves the addition of three bits, while the 102-CA (60-CA) model involves the addition of only two bits. Consequently, the number of logic operations to compute the GSS-sequence is much smaller. Furthermore, the periods of the (vertical) sequences of the 102-CA are well known (Theorem 3.7). Therefore, we do not need to compute the whole sequences to complete the CA.