The final publication is available at Springer via https://doi.org/10.1007/978-3-319-44365-2_4

Resilient Vectorial Functions and Cyclic Codes arising from Cellular Automata

Luca Mariot[1,2] and Alberto Leporati[1]

[1] Dipartimento di Informatica, Sistemistica e Comunicazione, Università degli Studi Milano-Bicocca, Viale Sarca 336, 20126 Milano, Italy
{luca.mariot,alberto.leporati}@unimib.it
[2] Laboratoire I3S, Université Nice-Sophia Antipolis, 2000 Route des Colles, 06903 Sophia Antipolis, France

Abstract. Most of the works concerning cryptographic applications of cellular automata (CA) focus on the analysis of the underlying local rules, interpreted as boolean functions. In this paper, we investigate the cryptographic criteria of CA global rules by considering them as vectorial boolean functions. In particular, we prove that the 1-resiliency property of CA with bipermutive local rules is preserved on the corresponding global rules. We then unfold an interesting connection between linear codes and cellular automata, observing that the generator and parity check matrices of cyclic codes correspond to the transition matrices of linear CA. Consequently, syndrome computation in cyclic codes can be performed in parallel by evolving a suitable linear CA, and the error-correction capability is determined by the resiliency of the global rule. As an example, we finally show how to implement the $(7, 4, 3)$ cyclic Hamming code using a CA of radius $r = 2$.

Keywords: cellular automata · boolean functions · S-boxes · resiliency · linear feedback shift registers · cyclic codes · Hamming codes

1 Introduction

Cellular Automata (CA) provide an interesting framework for developing cryptographic primitives such as *stream* and *block ciphers*. The reason is twofold: first, depending on the local rule, CA can exhibit chaotic and unpredictable dynamic behaviors, making them possibly useful for *pseudorandom number generation* (PRNG), one of the most important building blocks in cryptography. Second, being a massively parallel model, CA can be efficiently realized in hardware, and thus they are interesting for implementing cryptographic applications on devices with limited computational resources.

Wolfram [10] was the first to pioneer the use of CA for keystream generation, using the elementary rule 30. However, the design was discovered to be insecure against the Meier-Staffelbach [7] and Koc-Apohan attacks [3], due to the fact that rule 30, when considered as a boolean function $f : \mathbb{F}_2^3 \to \mathbb{F}_2$, is not 1-resilient and has a low nonlinearity. Since then, some researchers [4,2] focused on the search of CA local rules having good cryptographic profiles.

To the best of our knowledge, there are no works in the literature addressing the cryptographic properties of CA *global rules*. The aim of this paper is to begin filling this

gap by investigating CA as a particular kind of *vectorial boolean functions*. Specifically, we focus on the resiliency criterion, the reason being that resilient vectorial functions both have applications in stream ciphers and *error-correcting codes*.

We first show that the global rules of *bipermutive CA* are always at least 1-resilient, thus generalizing the result in [4] about bipermutive local rules. We then prove an equivalence between *linear CA* and linear *cyclic codes*. In particular, we show how the systematic encoding of cyclic codes actually corresponds to the preimage computation process of the all-zeros configuration in linear CA, while syndrome computation is equivalent to the application of the CA global rule, and can thus be performed in parallel. Leveraging on the theory of resilient vectorial functions, we remark that the resiliency order of a linear CA can be used to determine the minimum distance of its associated cyclic code. To sum up the results of the paper, we finally show how the $(7,4,3)$ cyclic Hamming code can be implemented using a CA of radius $r = 2$ and length $n = 7$.

The rest of the paper is organized as follows. Section 2 recalls some basic facts about cellular automata and vectorial boolean functions. Section 3 shows that the global rules of bipermutive CA are always at least 1-resilient. Section 4 recalls some key concepts about the theory of error-correcting codes, and presents the connection between linear cyclic codes and linear CA. Section 5 illustrates the results presented in the paper by showing how to simulate the $(7,4,3)$ cyclic Hamming codes using linear CA. Finally, Section 6 sums up the results presented in the paper and points out future directions of research on the subject.

2 Preliminaries on Cellular Automata and Boolean Functions

2.1 Cellular Automata

In what follows, we consider exclusively one-dimensional boolean cellular automata, formally defined below.

Definition 1. *A* one-dimensional boolean cellular automaton *(CA) is a triple $\langle C, \delta, f\rangle$, where C is a finite one-dimensional array of binary* cells*, $\delta \in \mathbb{N}$ is the* diameter *and $f : \mathbb{F}_2^\delta \to \mathbb{F}_2$ is the* local rule.

Given an array C of length $n \geq \delta$, the update of a CA is done as follows. If the diameter δ is odd with $\delta = 2r + 1$ for $r \in \mathbb{N}$, then each cell i in the range $\{r+1, \cdots, n-r\}$ synchronously updates its state by applying rule f to the neighborhood $\{i-r, \cdots, i, \cdots, i+r\}$. Otherwise, if δ is even and $r = \delta/2$, then each cell i in the range $\{r, \cdots, n-r\}$ synchronously updates its state by applying f to the neighborhood $\{i-r+1, \cdots, i+r\}$. In both cases, the parameter r is called the *radius* of the CA. The *global rule* of a CA $\langle C, \delta, f\rangle$ with array of length $n = m + \delta - 1$ is the function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ defined as

$$F(C) = F(c_1, \cdots, c_n) = (f(c_1, \cdots, c_\delta), \cdots, f(c_{n-\delta+1}, \cdots, c_n)) \ .$$

In what follows, we identify a CA $\langle C, \delta, f\rangle$ by its global rule $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$.

The most common way to represent a CA is by means of the *truth table* of its local rule f. Since f depends on δ variables, it means that there exist a total of 2^{2^δ} possible local rules. Another convenient way of representing a CA rule f is through its *Wolfram code*, which is basically the decimal encoding of the truth table of f.

2.2 Vectorial Boolean Functions

A *boolean function* is a mapping $f : \mathbb{F}_2^n \to \mathbb{F}_2$. The boolean functions adopted in cryptography for the design of stream and block ciphers must satisfy several criteria, among which one of the most important is *resiliency*:

Definition 2. *A boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ is said to be t-resilient if, by fixing at most t input coordinates, the resulting restriction of f is* balanced*, i.e. its truth table is composed of an equal number of zeros and ones.*

A very well-known secondary construction to obtain a $(t+1)$-resilient function of $n+1$ variables from a t-resilient function of n variables is to simply add (XOR) an additional variable, as shown in [9]. This method is formalized in the following result:

Proposition 1. *Let $I = \{i_1, \cdots, i_{t+1}\} \subseteq \{1, \cdots, n\}$ and $J = \{j_1, \cdots, j_{n-t-1}\} = \{1, \cdots, n\} \setminus I$ be complementary sets of indices. Additionally, let $f : \mathbb{F}_2^n \to \mathbb{F}_2$ be a boolean function of n variables defined as*

$$f(x_1, \cdots, x_n) = g(x_{j_1}, \cdots, x_{j_{n-t-1}}) \oplus x_{i_1} \oplus \cdots \oplus x_{i_{t+1}} \ ,$$

where $g : \mathbb{F}_2^{n-t-1} \to \mathbb{F}_2$ is a boolean function of $n-t-1$ variables. Then, f is t-resilient.

Let $n \geq m$. A *vectorial boolean function* (also called a *S-box*) is a mapping $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ with n input variables and m outputs. By $f_1, \cdots, f_m : \mathbb{F}_2^n \to \mathbb{F}_2$ we denote the *coordinate functions* of F, that is, the m boolean functions which specify the value of each output bit of F:

$$F(x_1, \cdots, x_n) = (f_1(x_1, \cdots, x_n), f_2(x_1, \cdots, x_n), \cdots, f_m(x_1, \cdots, x_n)) \ .$$

The *component functions* of F are defined as $v \cdot F$ for all $v \in \mathbb{F}_2^m \setminus \{\underline{0}\}$, where $\cdot$ denotes the scalar product modulo 2. Since

$$v \cdot F = v_1 f_1(x_1, \cdots, x_n) \oplus \cdots \oplus v_m f_m(x_1, \cdots, x_n) \ ,$$

it follows that the component functions are the linear combinations of the coordinate functions of F.

Remark 1. Let $F : \mathbb{F}_2^{m+\delta-1} \to \mathbb{F}_2^m$ be a one-dimensional boolean cellular automaton of length $n = m + \delta - 1$ defined by a local rule $f : \mathbb{F}_2^\delta \to \mathbb{F}_2$ of diameter δ. Since each output cell y_i depends only on the input cells $x_i, \cdots, x_{i+\delta-1}$ under application of the local rule, the coordinate functions of F are $f_i(x_1, \cdots, x_n) = f(x_i, \cdots, x_{i+\delta-1})$ for $i \in \{1, \cdots, m\}$.

We now define the resiliency property for vectorial functions.

Definition 3. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be a vectorial function, and $1 \leq t \leq n$. Function F is said t-*resilient *if, by fixing any t input variables $x_{i_1}, \cdots, x_{i_t}$, the resulting restriction $\tilde{F} : \mathbb{F}_2^{n-t} \to \mathbb{F}_2^m$ is balanced, i.e. for all $y \in \mathbb{F}_2^m$ it follows that $|\tilde{F}^{-1}(y)| = 2^m$.*

Note that for $m = 1$ Definition 3 is actually equivalent to Definition 2 for boolean functions. The resiliency of a vectorial function can be characterized by the resiliency of its component functions, as the next result proved in [1] shows:

Proposition 2. *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ be a vectorial boolean function in n variables and m outputs. Then, F is t-resilient if and only if for all $v \in \mathbb{F}_2^m \setminus \{\underline{0}\}$ the component function $v \cdot F$ is t-resilient.*

3 Resilient Vectorial Functions from Bipermutive CA

We now show that bipermutive cellular automata are always at least 1-resilient when considered as vectorial boolean functions. To this end, recall that a rule $f : \mathbb{F}_2^\delta \to \mathbb{F}_2$ of diameter $\delta \in \mathbb{N}$ is *bipermutive* if it is defined as

$$f(x_1, x_2, \cdots, x_{\delta-1}, x_\delta) = x_1 \oplus g(x_2, \cdots, x_{\delta-1}) \oplus x_\delta \tag{1}$$

for all $x = (x_1, x_2, \cdots, x_{\delta-1}, x_\delta) \in \mathbb{F}_2^\delta$, where $g : \mathbb{F}_2^{\delta-2} \to \mathbb{F}_2$. Clearly, by Proposition 1 any bipermutive local rule is also a 1-resilient boolean function. The following result characterizes the component functions of a cellular automaton based on a bipermutive rule:

Lemma 1. *Let $m, \delta \in \mathbb{N}$ and let $F : \mathbb{F}_2^{m+\delta-1} \to \mathbb{F}_2^m$ be a CA of length $n = m + \delta - 1$ defined by a bipermutive local rule $f : \mathbb{F}_2^\delta \to \mathbb{F}_2$. Then, for all $v \in \mathbb{F}_2^m \setminus \{\underline{0}\}$ the component function $v \cdot F$ is bipermutive as well.*

Proof. Let f be defined as in Equation (1)*. Given $v \in \mathbb{F}_2^m \setminus \{\underline{0}\}$, denote the* support *of v as follows:*

$$supp(v) = \{i_1, \cdots, i_k\} = \{i : v_i \neq 0\} \ . \tag{2}$$

Then, the component function $v \cdot F$ can be expressed as:

$$v \cdot F = x_{i_1} \oplus g(x_{i_1+1}, \cdots, x_{i_1+1+\delta-2}) \oplus x_{i_1+\delta-1} \oplus \cdots \oplus x_{i_k} \oplus g(x_{i_k+1}, \cdots, x_{i_k+\delta-2}) \oplus x_{i_k+\delta-1} \ . \tag{3}$$

Notice that the leftmost and rightmost variables x_{i_1} and $x_{i_k+\delta-1}$ appear exactly once in Equation (3)*, thus they are never canceled. Let G be the boolean function defined as:*

$$G(x_{i_1+1}, \cdots, x_{i_k+\delta-2}) = g(x_{i_1+1}, \cdots, x_{i_1+\delta-2}) \oplus x_{i_1+\delta-1} \oplus \cdots \oplus x_{i_k} \oplus g(x_{i_k+1}, \cdots, x_{i_k+\delta-2}) \ . \tag{4}$$

Hence, the component function $v \cdot F$ has the form:

$$v \cdot F = x_{i_1} \oplus G(x_{i_1+1}, \cdots, x_{i_k+\delta-2}) \oplus x_{i_k+\delta-1} \ . \tag{5}$$

As a consequence, $v \cdot F$ is bipermutive. □

Combining Lemma 1 and Proposition 1, we get the following result:

Theorem 1. *Let $m, \delta \in \mathbb{N}$ and let $F : \mathbb{F}_2^{m+\delta-1} \to \mathbb{F}_2^m$ be a CA of length $n = m + \delta - 1$ defined by a bipermutive local rule $f : \mathbb{F}_2^\delta \to \mathbb{F}_2$. Then, F is at least 1-resilient.*

4 Linear CA and Linear Codes

4.1 Basics on Linear Codes

We now restrict our attention to the class of *linear resilient CA*, i.e. resilient CA whose local rule is a linear combination of the neighborhood cells. In this case, an interesting connection with *linear codes* can be observed. We first recall some basic facts about linear codes; for a thorough treatment of the subject, the reader can refer to [6].

Definition 4. *Let $n,m,d \in \mathbb{N}$ such that $n \geq m$, and let $q = \rho^\alpha$ be the power of a prime number ρ. A (n,m,d)* linear code *C is a m-dimensional subspace of the vector space $\mathbb{F}_q^n$, such that the Hamming distance between any two vectors $c_1, c_2 \in C$ (called* codewords*) is at least d. The parameters n, m and d are respectively called the* length*, the* dimension *and the* minimum distance *of C.*

In what follows, we focus on *binary linear codes*, where $q = 2$.

Since a (n,m,d) linear code C is a subspace of dimension m of $\mathbb{F}_2^n$, it is possible to specify it using a $m \times n$ matrix G whose rows form a set of m linearly independent codewords of C. Such a matrix G is called a *generator matrix* for code C. The encoding process simply amounts to multiplying a *message vector* $\mu \in \mathbb{F}_2^m$ by matrix G, thus obtaining the codeword $c = \mu G$. Another matrix associated to a linear code is its *parity check matrix*, which is useful for error correction. The parity check matrix for C is a matrix H of dimensions $(n-m) \times n$ such that $Hx^\top = \underline{0}$ if and only if $x \in C$. In general, the vector $s = Hx^\top$ is called the *syndrome* of $x \in \mathbb{F}_2^n$.

The *dual code* of a (n,m,d) linear code C is the set $C^\perp = \{x \in \mathbb{F}_2^n : \ x \cdot y = 0, \forall y \in C\}$, that is, the set of all vectors in $\mathbb{F}_2^n$ which are orthogonal to the codewords in C. The parity check matrix H of C is a generator matrix for $C^\perp$, and vice versa the generator matrix G of C is a parity check matrix for $C^\perp$. This means that $C^\perp$ is a code of length n and dimension $n-m$.

Notice that each codeword $c \in C$ defines a *ball* B_c of radius $t = \lfloor (d-1)/2 \rfloor$, since the minimum distance is d. Suppose now that a codeword $c \in C$ is transmitted over a noisy channel, and at most t errors occur, i.e. at most t bits of c are flipped. The received word z will always be inside ball B_c, thus making it possible to retrieve the original codeword by determining the center of the ball to which z belongs to. This procedure can be carried out in linear codes through *syndrome decoding* as follows. Let $c \in C$ be a codeword and $e \in \mathbb{F}_2^n$ be an *error pattern* introduced by the channel, having Hamming weight at most t. A received word can thus be expressed as $z = c \oplus e$. Given a parity check matrix H of C, the syndrome of z is $s = Hz^\top = H(c \oplus e)^\top = Hc^\top \oplus He^\top = He^\top$. In order to retrieve c, it thus suffices to determine the error pattern e corresponding to s, and output $c = z \oplus e$. This task can be performed by storing in a table the set of all possible error patterns of weight at most t together with their syndromes.

We now introduce the class of linear *cyclic codes*.

Definition 5. *A (n,m,d) linear code $C \subseteq \mathbb{F}_2^n$ is called* cyclic *if it is closed under* cyclic shifts*, i.e. for all $c = (c_1, c_2 \cdots, c_n) \in C$, it holds that $c' = (c_2, \cdots, c_n, c_1) \in C$.*

A cyclic code is described by its *generator polynomial* $g(x) = g_0 + g_1 x + \cdots + g_{n-m} x^{n-m}$, where $g_i \in \mathbb{F}_2$ for all $i \in \{0, \cdots, n-m\}$. If the m-bit message $\mu = (\mu_0, \cdots, \mu_{m-1})$ is represented by the polynomial $\mu(x) = \mu_0 + \mu_1 x + \cdots + \mu_{m-1} x^{m-1}$, then the polynomial corresponding to the codeword c is $c(x) = \mu(x) g(x)$. There exists a one-to-one correspondence between cyclic codes of length n and divisors of $x^n - 1$. In particular, a (n,m,d) code C is cyclic if and only if its generator polynomial $g(x)$ divides $x^n - 1$.

Given a (n,m,d) cyclic code C with generator polynomial $g(x)$ of degree $n-m$, the polynomial $h(x) = (x^n - 1)/g(x)$ of degree m is called the *parity check polynomial* of C. Analogously to the parity check matrix, $h(x)$ satisfies the property that the codeword associated to a polynomial $d(x)$ belongs to C if and only if $d(x)h(x) = 0$. The

relationship between the generator/parity check polynomials of a cyclic code C and its generator/parity check matrices is given by the following result:

Theorem 2. *Let $C \subseteq \mathbb{F}_2^n$ be a (n,m,d) cyclic linear code with generator polynomial $g(x) = g_0 + g_1 x + \cdots + g_{n-m}x^{n-m}$ and parity check polynomial $h(x) = h_0 + h_1 x + \cdots + h_m x^m$. Then the following are respectively a generator and a parity check matrix for C:*

$$G = \begin{pmatrix} g_0 & \cdots & g_{n-m} & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & g_0 & \cdots & g_{n-m} & 0 & \cdots & \cdots & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & g_0 & \cdots & g_{n-m} \end{pmatrix}, \quad H = \begin{pmatrix} h_m & \cdots & h_0 & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & h_m & \cdots & h_0 & 0 & \cdots & \cdots & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & h_m & \cdots & h_0 \end{pmatrix}. \tag{6}$$

As a consequence of Theorem 2, the dual code $C^\top$ of a cyclic code is again a cyclic code of length n and dimension $n-m$.

One of the main advantages of cyclic codes is that they can be easily implemented using *Linear Feedback Shift Registers* (LFSR), as shown in [6, pp. 193–195]. In particular, if the parity check polynomial $h(x)$ of a (n,m,d) cyclic code is such that $h_0 \neq 0$, the codeword of a message $\mu \in \mathbb{F}_2^m$ can be generated by a LFSR of length m whose tap polynomial is the reciprocal $\tilde{h}(x) = h_m + h_{m-1}x + \cdots + x^m$ of $h(x)$. The registers are initialized to the values $\mu_0, \cdots, \mu_{m-1}$ of μ, and the LFSR is evolved for n steps. The output of length n produced by the LFSR is the codeword corresponding to μ. Notice that the first m output bits are exactly the original message μ, while the remaining $n-m$ are the parity check bits. This encoding procedure is called *systematic*, since the bits of the message appear unaltered in the corresponding codeword. If no errors are introduced by the channel, the decoding process is immediate since it just consists of truncating the codeword to its first m bits.

4.2 Linear CA and Cyclic Codes

A cellular automaton $F : \mathbb{F}_2^{m+\delta-1} \to \mathbb{F}_2^m$ is called *linear* if its local rule is defined as $f(x_1, \cdots x_\delta) = a_1 x_1 \oplus \cdots \oplus a_\delta x_\delta$, with $a_i \in \mathbb{F}_2$ for all $i \in \{1, \cdots, \delta\}$. The global rule of F is described by a $m \times (m+\delta-1)$ *transition matrix* M_F of the following form:

$$M_F = \begin{pmatrix} a_1 & \cdots & a_\delta & 0 & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & a_1 & \cdots & a_\delta & 0 & \cdots & \cdots & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & a_1 & \cdots & a_\delta \end{pmatrix}. \tag{7}$$

In particular, when the CA is bipermutive and linear we have $a_1 = a_\delta = 1$. The application of the CA global rule F to a configuration $x \in \mathbb{F}_2^{m+\delta-1}$ corresponds to the multiplication $y = M_F x^\top$.

One can notice that the generator and parity check matrices of Equation (6) in Theorem 2 have the same form of the linear CA matrix in Equation (7). In particular, the systematic encoding for cyclic codes described above can be simulated through cellular automata. As observed in [5], computing a preimage of a spatially periodic configuration in a linear bipermutive CA is equivalent to a *concatenation* of LFSR, where the

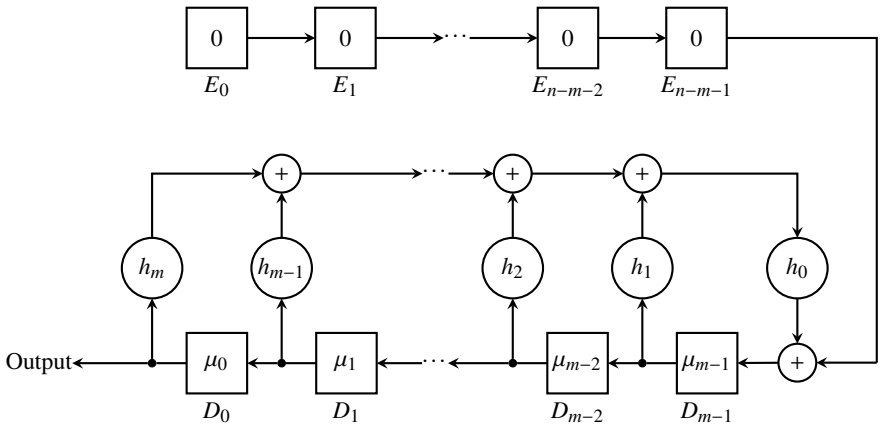


Fig. 1: Concatenation of a LFSR with a sequence of $n-m$ zeros, which computes a preimage $x \in F^{-1}(\underline{0})$.

LFSR associated to the local rule is disturbed by the LFSR which generates the spatially periodic configuration. In our case, we are only interested in a preimage of a finite configuration. Thus the general scheme consists of the LFSR associated to the rule where the feedback is additively disturbed by the bits of the configuration. If one takes the all-zeros configuration $\underline{0}$, it can be observed that the resulting concatenated LFSR of Figure 1 is equivalent to the LFSR used for the systematic encoding of a cyclic code. As a matter of fact, adding a sequence of zeros to the feedback of a LFSR does not change its dynamics. In the context of cellular automata, the system represented in Figure 1 is equivalent to the computation of a preimage of $\underline{0} \in \mathbb{F}_2^{n-m}$, in particular the preimage determined by the m-bit block μ.

To summarise the discussion above, we have thus proved the following result:

Theorem 3. *Let $F : \mathbb{F}_2^{m+\varrho} \to \mathbb{F}_2^m$ be a linear cellular automaton defined by a local rule $f(x) = a_1 x_1 \oplus \cdots \oplus a_\delta x_\delta$ of diameter $\delta = \varrho + 1$ with $\varrho \in \mathbb{N}$, and let $g(x) = a_1 + a_2 x + \cdots + a_\delta x^\varrho$ be the polynomial associated to f. If $g(x)$ divides $x^n - 1$ where $n = m + \varrho$, then F is equivalent to a cyclic code C of length n and dimension m. The generator matrix of C is the CA matrix M_F associated to F, while $g(x)$ is the generator polynomial of C. Additionally, let $\tilde{h}(x) = h_m + h_{m-1}x + \cdots + h_0 x^m$ be the reciprocal of the parity check polynomial $h(x) = (x^n - 1)/g(x)$, and let $\tilde{f}(x) = h_m x_1 \oplus \cdots \oplus h_0 x_{m+1}$ be the corresponding local rule. Then, the matrix $M_{\tilde{F}}$ associated to the linear CA $\tilde{F} : \mathbb{F}_2^{m+\varrho} \to \mathbb{F}_2^\varrho$ induced by rule $\tilde{f}$ is a parity check matrix for C, and $C = \tilde{F}^{-1}(\underline{0})$.*

In other words, by Theorem 3 we can implement a linear cyclic code of length n and dimension m with a cellular automaton as follows:

1. Given m and $n = m + \varrho$ with $\varrho \in \mathbb{N}$, determine a local rule f of diameter $\delta = \varrho + 1$ such that the associated polynomial $g(x)$ divides $x^n - 1$.
2. Compute the reciprocal $\tilde{h}(x)$ of the parity check polynomial $h(x) = (x^n - 1)/g(x)$, and determine the corresponding local rule $\tilde{f}$ of diameter $m + 1$.

3. *Systematic encoding*: Let $\tilde{F} : \mathbb{F}_2^{m+\varrho} \to \mathbb{F}_2^{\varrho}$ be the linear CA of length n induced by $\tilde{f}$. A message $\mu \in \mathbb{F}_2^m$ is encoded by computing the preimage $x \in \tilde{F}^{-1}(\underline{0})$ whose leftmost m-bit block equals μ. This preimage can be computed by the LFSR in Figure 1.
4. *Syndrome computation*: given $x \in \mathbb{F}_2^{m+\varrho}$, the syndrome of x is $s = \tilde{F}(x)$. If the syndrome s equals $\underline{0} \in \mathbb{F}_2^{\varrho}$ then x is a codeword of C. Otherwise, one can apply the syndrome decoding procedure to retrieve the original codeword.

The main advantage of the above procedure is that the computation of the syndrome can be performed in parallel, since it corresponds to the application of the CA global rule $\tilde{F}$ to the word $\tilde{x}$.

Notice that up to now we did not consider the minimum distance of the cyclic codes generated through linear CA, which is necessary in order to assess their error-correction capability. This is where the resiliency order of the CA comes into play. In particular, the connection between $(n, m, d-1)$ general linear resilient functions and linear codes is given by the following theorem [8]:

Theorem 4. *A $(n, m, d-1)$ resilient linear function $F : \mathbb{F}_2^n \to \mathbb{F}_2^m$ is equivalent to a (n, m, d) linear code C.*

We already know from the previous section that all bipermutive CA are always at least 1-resilient, thus a linear and bipermutive CA which satisfies the hypotheses of Theorem 3 is equivalent to a linear cyclic code with minimum distance at least 2. More in general, we can refine Theorem 3 on account of Theorem 4 as follows:

Theorem 5. *Let $F : \mathbb{F}_2^{m+\varrho} \to \mathbb{F}_2^m$ be a linear CA satisfying the hypotheses of Theorem 3. If F is $(d-1)$–resilient, then the cyclic code associated to F has minimum distance d.*

5 Cyclic Hamming Codes through Linear CA

To sum up the results presented in the previous section, we show an example of cyclic code generated by a linear CA. In particular we focus on *cyclic Hamming codes*, which are codes with minimum distance $d = 3$ and thus they can correct up to 1 error. The main reason for this choice is the simplicity of syndrome decoding in Hamming codes. As a matter of fact, the position of the column of the parity check matrix H containing the value of the syndrome is the position where the error occurred.

Example 1 (The $(7, 4, 3)$ cyclic Hamming code). Let $F : \mathbb{F}_2^7 \to \mathbb{F}_2^4$ be the linear CA induced by the local rule $f : \mathbb{F}_2^4 \to \mathbb{F}_2$ defined as $f(x) = x_1 \oplus x_2 \oplus x_4$. The associated polynomial is $g(x) = 1 + x + x^3$, while the CA matrix is:

$$M_F = \begin{pmatrix} 1\,1\,0\,1\,0\,0\,0 \\ 0\,1\,1\,0\,1\,0\,0 \\ 0\,0\,1\,1\,0\,1\,0 \\ 0\,0\,0\,1\,1\,0\,1 \end{pmatrix} . \tag{8}$$

The polynomial $g(x)$ divides $x^7 - 1$, and we have $h(x) = (x^7 - 1)/g(x) = 1 + x + x^2 + x^4$. Further, we can deduce from matrix M_F that F is 2-resilient. As a matter of fact, it

is not difficult to see by exhaustive enumeration that each nonzero vector v results in a sum of rows which always have at least 3 ones. Hence, by Theorem 5 the code C associated to F is the $(7,4,3)$ cyclic Hamming code. Remark that $\tilde{h}(x) = 1 + x^2 + x^3 + x^4$ is the reciprocal of the parity check polynomial $h(x)$. The local rule $\tilde{f}$ associated to the polynomial $\tilde{h}(x)$ is $\tilde{f}(x) = x_1 \oplus x_3 \oplus x_4 \oplus x_5$, and thus it has radius $r = 2$. In particular, the Wolfram code representing the truth table of $\tilde{f}$ is 1768527510. The transition matrix of the linear CA $\tilde{F} : \mathbb{F}_2^7 \to \mathbb{F}_2^3$ induced by rule $\tilde{f}$ is the following:

$$M_{\tilde{F}} = \begin{pmatrix} 1\,0\,1\,1\,1\,0\,0 \\ 0\,1\,0\,1\,1\,1\,0 \\ 0\,0\,1\,0\,1\,1\,1 \end{pmatrix} . \tag{9}$$

Let $\mu = (0,1,1,0) \in \mathbb{F}_2^4$ be a 4-bit message. The systematic encoding of μ under the Hamming code $(7,4,3)$ can be accomplished by computing the preimage x of $(0,0,0)$ under the action of $\tilde{F}$, with the leftmost 4 bits of x initialized to μ. This process is depicted in Figure 2. Hence, the codeword corresponding to μ is $x = (0,1,1,0,1,0,0)$.

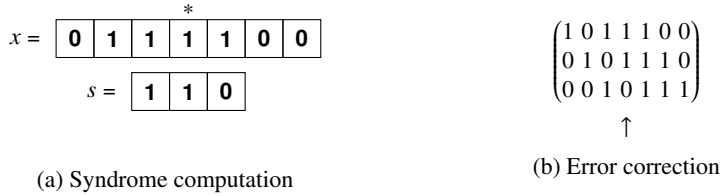
Let us now assume that x is transmitted through a noisy channel and the fourth bit of x is flipped, thus yielding the word $\tilde{x} = (0,1,1,1,1,0,0)$. The receiver applies to $\tilde{x}$ the CA $\tilde{F}$ defined by rule 1768527510, thus obtaining the syndrome $s = F(x) = (1,1,0)$, as shown in Figure 3(a). To correct the error, the receiver looks at the CA matrix $M_{\tilde{F}}$ and finds that the syndrome appears in the fourth column. Thus, the receiver knows that a transmission error has occurred in the fourth position of $\tilde{x}$, and the original codeword can be recovered as $\tilde{x} \oplus (0,0,0,1,0,0,0) = x$.

6 Conclusions and Future Directions

In this work, we began investigating the cryptographic properties of CA global rules, focusing on resiliency. In particular, we proved that the global rule of a bipermutive CA F is always at least 1-resilient, since each component of F is still a bipermutive boolean function. We then presented an equivalence between linear cyclic codes and linear CA, showing that syndrome computation in the former is equivalent to applying the global rule to the received word in the latter, and can thus be performed in parallel. Finally, the resiliency order of a linear and bipermutive CA can be used to determine the minimum distance of the corresponding cyclic code, and we applied these results by showing how the $(7,4,3)$ cyclic Hamming code can be implemented using a linear CA of radius $r = 2$.

(a) Initialization (b) Complete codeword

Fig. 2: Example of systematic encoding of $\mu = (0,1,1,0) \in \mathbb{F}_2^4$ using rule 1768527510, defined as $\tilde{f}(x) = x_1 \oplus x_3 \oplus x_4 \oplus x_5$.



(a) Syndrome computation

(b) Error correction

Fig. 3: Example of error correction using rule 1768527510. The cell marked by $*$ indicates where the error occurred.

There are several directions along which the present work can be extended, both on the cryptographic and on the coding-theoretic sides. For the cryptographic part, one could characterize the global rules of bipermutive CA in terms of other properties such as *nonlinearity* and *differential uniformity*. About the coding-theoretic part, cyclic codes form a broad class including for example *BCH* and *Reed-Solomon* codes. Hence, it could be interesting to investigate how to implement these codes through CA by elaborating on the method presented in this paper.

References

1. Carlet, C.: Boolean Functions for Cryptography and Error-Correcting Codes. In: Crama, Y., Hammer, P.L. (eds.) Boolean Models and Methods in Mathematics, Computer Science, and Engineering. Cambridge University Press, New York (2010)
2. Formenti, E., Imai, K., Martin, B., Yunès, J.-B.: Advances on Random Sequence Generation by Uniform Cellular Automata. In: Calude, C.S., Freivalds, R., Kazuo, I. (eds.) Computing with New Resources. LNCS vol. 8808, pp. 56–70. Springer, Heidelberg (2014)
3. Koc, C.K., Apohan, A.M.: Inversion of cellular automata iterations. In: IEE Proceedings – Computers and Digital Techniques 144(5):279–284. IET (1997)
4. Leporati, A., Mariot, L.: Cryptographic Properties of Bipermutive Cellular Automata Rules. J. Cell. Aut. 9(5–6):437–475 (2014)
5. Mariot, L., Leporati, A.: On the Periods of Spatially Periodic Preimages in Linear Bipermutive Cellular Automata. In: Kari, J. (ed.): AUTOMATA 2015. LNCS vol. 9099, pp. 181–195. Springer, Heidelberg (2015)
6. McEliece, R.J.: The Theory of Information and Coding. Cambridge University Press, New York (1985)
7. Meier, W., Staffelbach, O.: Analysis of Pseudo Random Sequences Generated by Cellular Automata. In: Davies, D.W. (ed.) EUROCRYPT '91. LNCS, vol. 547, pp. 186–200. Springer, Heidelberg (1991)
8. Stinson, D.R.: Combinatorial Designs: Constructions and Analysis. Springer, Heidelberg (2004)
9. Siegenthaler, T.: Decrypting a Class of Stream Ciphers Using Ciphertext Only. IEEE Trans. Comput. C-34(1), 81–85 (1985)
10. Wolfram, S.: Cryptography with Cellular Automata. In: Williams, H.C. (ed.) CRYPTO '85. LNCS, vol. 218, pp. 429–432. Springer, Heidelberg (1985)