Entropy and Security of Pseudorandom Number Generators based on Chaotic Iterations

Keywords: Pseudorandom Number Generators, Dynamical Systems, Security

Abstract: In the domain of cryptography, an important role is played by PseudoRandom Number Generators (PRNGs). Designing such generators might be complicated for different reasons: an appropriate formal abstract notion of *randomness* should be formulate, and after that, it may be hard to design an algorithm that produces such *random* numbers on a finite state machine. A possible approach to this problem has been proposed and studied in recent works (for instance (Guyeux and Bahi, 2012)), where the authors considered to post-operate on existing PRNGs, using the so-called *chaotic iterations*, i.e., specific iterations of a boolean function and a shift operator that use the inputted generator. This process has at least two positive aspects : boolean functions avoid the problem of numbers representation (e.g. floating point arithmetic), and it is possible to describe the PRNGs based on chaotic iterations as dynamical systems, with a formal mathematical description. This class of PRNGs has been proven to be useful also for cryptographical applications, after a suitable redefinition of the generators in the cryptographical domain. In this article we propose a Markov chain model of the PRNGs based on chaotic iterations and we will use it to compute the entropy of the proposed generators. Moreover we will prove that the security property is preserved when a cryptographic PRNG is post processed with iterations of a suitable boolean functions.

1 INTRODUCTION

To design a PseudoRandom Number Generator (PRNG) is a delicate task which has a lot of application in numerical simulation, information security etc. It is logical to think that introducing elements of chaos, such as well-known chaotic real functions (for instance the logistic map), might improve the random-like quality of the output of these algorithms. As far as we know, no result states that the chaotic properties of a real function are preserved in floating point arithmetic. Moreover, in the opposite direction, results indicates that numerical truncation may change drastically the statistically property of a system, see, *e.g.*, (Galatolo et al., 2014; Guiheneuf, 2015).

In (Galatolo et al., 2007; Galatolo et al., 2012), it has been established that every computable dynamical system has computable initial conditions which are statistically typical for the dynamics, i.e. initial conditions which are computable by an algorithm but their statistical behaviour is the same as typical "random" initial conditions. Such a theoretical result motivates the dynamical system approach for pseudorandom number generation, since we can think at these computable typical conditions as pseudorandom initial conditions. However in practice, finding these computable initial conditions might be computationally hard.

A possible solution to these problems, is to consider an asynchronous iteration scheme which includes a boolean function coupled with a shift as core of a PRNG (which is referred as chaotic iterations). Indeed, the use of a boolean function avoids the issues arising with floating point arithmetic, since only integers are involved in the process and the whole iteration scheme can be represented as a discrete dynamical system, which can be studied with many mathematical theoretical tools. For instance, in (Guyeux and Bahi, 2012), the authors proved that these iterations, viewed as a class of operators on a suitable discrete space, satisfy various topological properties of chaos like topological transitivity.

This process can be adapted in the cryptographical framework, in which the *randomness* quality of a generator is of fundamental importance. It is reasonable to think that there is some link between the notion of chaos and security, and that a "chaotic" PRNG is also, in a certain extent, a "secure" PRNG. In (Bahi et al., 2015), authors presented a secure cryptographical PRNG (cPRNG) based on this asynchronous iteration scheme, and in (Marangio et al., 2018) a first attempt to study the collision-free property of this kind of PRNG was made. In this article, we recall how to design a PRNG based on chaotic iterations, how to describe it as a dynamical system and, for the sake of completeness, we recall some of the topological properties of chaos previously obtained. There are two main contributions in this paper: we first propose a Markov Chain model of the involved generators, which is slightly different from the topological models already presented but allow us to explicitly compute the entropy of the PRNG, under some suitable assumption. On the other side, we prove that the security property for cPRNG is preserved under chaotic iterations.

This research work is organized as follows. In Section 2, we introduce chaotic iterations and the PRNGs based on it, with an explicit example. In Section 3 we recall how to model these PRNGs as discrete dynamical systems and we present some of the results about their randomness, by using various notions taken form the field of mathematical topology. In Section 4, we introduce a Markov chain model of the chaotic iterations, we translate it in the measure preserving system language, and we compute its entropy. Then, in Section 5, we change the framework and we introduce the cPRNG based on chaotic iterations. We then provide a security analysis for such generators. This research work ends by a conclusion section, in which our contributions are summarized and intended future work is outlined.

2 PRNGS BASED ON CHAOTIC ITERATIONS

In this section we will show how to use an iteration of a boolean function to post process a given PRNG. The output of this procedure will be a new PRNG, which may be modeled as a dynamical system. Let us first introduce some notations.

Fix $N \in \mathbb{N}^*$, which is the length of the string of bits which we will consider, and let 2^N be the space of all the strings of length N.

Definition 2.1. *Let $f_0 : 2^N \to 2^N$ be the boolean negation, i.e. the function such that*

$$f_0(x_0, \dots, x_{n-1}) = (\bar{x}_0, \dots, \bar{x}_{n-1}),$$

where $\bar{x} = x+1$ mod 2.

We will use the following function:

Definition 2.2. *Let $\sigma : [1,N]^{\mathbb{N}} \to [1,N]^{\mathbb{N}}$ be the shift function defined as*

$$\sigma : (s_0, s_1, s_2, \dots) \mapsto (s_1, s_2, s_3, \dots).$$

If x is a sequence of 0 and 1 of length N, produced for instance by a PRNG, we may post process x with an iterative application of a boolean function to some of the digits of x, chosen accordingly to a *strategy s*, which is a sequence of integers between 1 and N. The following definition formalized this procedure:

Definition 2.3. *Let $N \in \mathbb{N}^*$, $f : 2^N \longrightarrow 2^N$ be a function, and $s \in [1,N]^{\mathbb{N}}$ be a sequence of integers between* 1 *and N called the chaotic strategy. The so-called* chaotic iterations *are defined by $x^0 \in 2^N$ and*

$$\forall n \in \mathbb{N}^*, \forall i \in [1,N], x_i^n = \begin{cases} x_i^{n-1} & \text{if } s_n \neq i \\ \left(f(x^{n-1})\right)_{s_n} & \text{if } s_n = i. \end{cases} \tag{1}$$

Let us explain it with an example:

Example. For the sake of concreteness, let us consider that $N = 3$. Let s be the sequence $s = (123123123123...)$ and let f_0 be the Boolean negation. Start with input (000), then the chaotic iterations will produce the following output

$$(000) \to (100) \to (110) \to (111) \to (011) \to \dots$$

At each iteration step, look at the correspondent element of the sequence s (at the n-th step, look at the n-th element of s), and then apply f_0 to the element of the input sequence suggested by the element of s that we are considering. In the example, for instance, at step number 5, the second bit of the input will change since the fifth element of s is 2.

We can thus define the algorithm CI_1, which takes as input a PRNG P and it post processes the output of P with chaotic iterations:

```
Data: A seed $x_0$; a PRNG $s$ acting as a chaotic
      strategy; a boolean function $f$; execution
      time $n$
Result: A pseudo random sequence $x_n$
for $i = 0, \dots, n$ do
    apply chaotic iterations $(f, s)$ on $x_i$ to
    obtain $x_{i+1}$
end
return $x_n$
```

Algorithm 1: A pseudo code for PRNG based on chaotic iterations, the algorithm CI_1.

3 A MATHEMATICAL MODEL FOR CHAOTIC ITERATIONS

In this section we present a mathematical formalization of the algorithm CI_1 and we will recall the most relevant results about it.

Definition 3.1. *Given a boolean function $f : 2^N \to 2^N$, let us consider the function $F_f : 2^N \times [1,N] \to 2^N$, defined as*

$$F_f(x,i) = (x_1, \dots, x_{i-1}, f_i(x), x_{i+1}, \dots, x_N).$$

Consider the phase space $\mathcal{X} = 2^N \times [1,N]^{\mathbb{N}}$ and a boolean function $f : 2^N \to 2^N$.

Definition 3.2. *The map $H_f : \mathcal{X} \to \mathcal{X}$ is defined as*

$$H_f : (x,s) \mapsto (F_f(x,s_0), \sigma(s)).$$

The algorithm CI_1 can be model with the following dynamical system $(\mathcal{X}, H_f)$:

$$\begin{cases} x^0 \in \mathcal{X} \\ x^{k+1} = H_f(x^k). \end{cases} \tag{2}$$

We can define a distance d on $\mathcal{X}$; let $x = (e,s), y = (\check{e},\check{s}) \in \mathcal{X}$, let δ be the *discrete Boolean metric*, i.e. $\delta(x,y) = 0 \Leftrightarrow x = y$. Let d be the sum of two distances d_1 and d_2,

$$d(x,y) = d_1(e,\check{e}) + d_2(s,\check{s}),$$

defined as follows:

$$\begin{cases} d_1(e,\check{e}) & = & \displaystyle\sum_{k=1}^{N} \delta(e_k,\check{e}_k), \\ d_2(s,\check{s}) & = & \displaystyle\frac{9}{N}\sum_{k=1}^{\infty} \frac{|s_k - \check{s}_k|}{10^k}. \end{cases} \tag{3}$$

The distance d has been introduced in order to satisfy the following requirements.

- When the number of different cells between two systems is increasing, then their distance should increase too.
- In addition, if two systems present the same cells and their respective strategies start with the same terms, then the distance between these two points must be small because the evolution of the two systems will be the same for a while. Indeed, the two dynamical systems start with the same initial condition, use the same update function, and as strategies are the same for a while, then components that are updated are the same too.

We have an explicit description for the distance d:

- if the floor value $\lfloor d(x,y) \rfloor$ is equal to n, then e and $\check{e}$ differ in n cells;
- $d(x,y) - \lfloor d(x,y) \rfloor$ is a measure of the difference between s and $\check{s}$: its floating part is less than 10^{-k} if and only if the first k terms of s and $\check{s}$ are equal. If the k^{th} digit (of $d(x,y)$) is non zero, then the first k^{th} terms of the two strategies are different.

The distance d has been proven to be well defined on $\mathcal{X}$ and the map H_f is a continuous function with respect to this metric. In this framework we can define and possibly prove different notions of chaos:

Definition 3.3. *A continuous function f is* topologically transitive *if, for any pair of open sets $U,V \subset \mathcal{X}$, there exists an integer $k > 0$ such that $f^k(U) \cap V \neq \emptyset$.*

Definition 3.4. *A function f is* regular *on $(\mathcal{X},\tau)$ if the set of periodic points for f is dense in $\mathcal{X}$: for any point x in $\mathcal{X}$, any neighborhood of x contains at least one periodic point.*

Definition 3.5. *A continuous function f is* topologically mixing *if, for any pair of open sets $U,V \subset \mathcal{X}$, there exists an integer $m > 0$, such that for every $n > m$ $f^k(U) \cap V \neq \emptyset$.*

In (Guyeux and Bahi, 2012) it has been shown:

Theorem 3.1. *$(\mathcal{X}, d, H_{f_0})$, where f_0 is the boolean negation, is regular and topological transitive.*

Theorem 3.2. *$(\mathcal{X}, d, H_{f_0})$, where f_0 is the boolean negation, is topological mixing.*

Thus in the particular case of the boolean negation, the dynamical system associated to CI_1 is chaotic according to the previous definitions. We can characterize all the boolean functions which induce a topological transitive dynamical systems:

Definition 3.6. *Let f be a map from 2^N to itself. The asynchronous iteration graph associated with f (shortly, the iteration graph of f) is the directed graph $\Gamma(f)$ defined by:*

- *The set of vertices is 2^N;*
- *$\forall x \in 2^N, \forall i \in [1,N]$ the graph $\Gamma(f)$ contains an arc from x to $F_f(i,x)$.*

It has been proved in (Bahi et al., 2011):

Theorem 3.3. *$(\mathcal{X}, d, H_f)$ is regular and topological transitive if and only if $\Gamma(f)$ is strongly connected.*

This characterization theorem establishes the link between two mathematical models of the CI_1 generator: from one side, we have a topological dynamical system, meanwhile on the other side we have the Markov chain associated to the boolean function f, which is represented by the iteration graph $\Gamma(f)$. In the next section we will use the Markov chain model to compute, under suitable hypothesis, the entropy of the proposed algorithm.

4 ENTROPY

In this section we will show how to build a Markov chain model of the generator CI_1 and, to do so, we will recall some basic result on Markov chains. Finally we will apply standard ergodic theory to compute the entropy of the system.

Let $f : 2^N \to 2^N$ be a boolean function with $\Gamma(f)$ strongly connected. Let $\check{M}$ be the adjacency matrix associated to the graph $\Gamma(f)$.

Definition 4.1. *The Markov matrix associated to the boolean function f, is the $n \times n$ matrix M defined by $M_{ij} = \frac{1}{n}\check{M}_{ij}$ if $i \neq j$ and $M_{ii} = 1 - \frac{1}{n} \sum\limits_{j=1, j\neq i}^{n} \check{M}_{ij}$ otherwise.*

The matrix M is a double stochastic matrix, i.e. this is a finite Markov chain, which models the generator CI_1. In this framework, it is possible to introduce (at least) two notions of chaos, namely ergodicity and mixing. If a finite Markov chain is mixing, then we have an explicit formula to compute its entropy, and this is the path that we will follow.

Let us introduce the notion of ergodicity and mixing for a measure preserving system; we stress on the fact that a Markov chain can be always viewed as a measure preserving system in a standard way.

Definition 4.2. *Consider a probability space $(\mathcal{X}, \mathcal{E}, \mu)$ and a measurable transformation $T : \mathcal{X} \to \mathcal{X}$ which preserves the measure μ, i.e. $\forall A \in \mathcal{E}$, $\mu(A) = \mu(T^{-1}(A))$. In this case $(\mathcal{X}, \mathcal{E}, \mu, T)$ is said to be a measure preserving system.*

- *T is said to be* ergodic, *if for every $A, B \in \mathcal{E}$ of positive measure, there exists $n > 0$ such that*
$$\mu(T^{-n}(A) \cap B) > 0;$$
- *T is said to be* mixing *if, for any $A, B \in \mathcal{E}$,*
$$\lim_{n} \mu(T^{-n}(A) \cap B) = \mu(A)\mu(B).$$

Since M is a stochastic matrix, from Perron-Frobenius's theorem, it follows that there exists a probability distribution $\pi = (\pi_1, \cdots, \pi_N)$ on $\mathcal{S}$ invariant for M, i.e. $\pi = \pi M$, where $\mathcal{S} = 2^N$. Moreover, if there exists some k such that all the elements of M^k are positive, then this vector is unique. To transform a Markov chain in a measure preserving system, there is the following standard method; consider $\Omega = \mathcal{S}^{\mathbb{N}}$ and the σ-algebra of the cylinders:

Definition 4.3. *For every $k \in \mathbb{N}$ and for every $\omega \in \Omega$, a* standard cylinder *is*
$$C_k(\omega) = \{z \in \Omega : z_i = \omega_i,\ \forall i = 1, \cdots, k\};$$

Thus we can consider the σ-algebra generated by the standard cylinders, denoted by $\mathcal{C}$. Observe that if C is a *generic* cylinder, i.e. is an element of the σ-algebra $\mathcal{C}$, then there are $a_1, \cdots, a_s \in [1, N]$ such that
$$C = \{\omega : \omega_{i_1} = a_1, \cdots, \omega_{i_s} = a_s\}.$$

Now consider the following measure m on $\mathcal{C}$:
$$m(\{\omega : \omega_1 = e_1, \ldots, \omega_r = e_r\}) = \pi_{e_1} m_{e_1 e_2} \ldots m_{e_{r-1} e_r}.$$

In such a way the space $(\Omega, \mathcal{C}, m, \sigma)$, where $\sigma : \Omega \to \Omega$ is the shift function, is a measure preserving system. Before proving it, let us remark that the shift "here" and the shift "inside" the H_f are not linked at all between them. Here, the shift function allows us to visualize at each step an element of S (by shifting a sequence in Ω), but the output depends on the probability of the cylinders, and they depend on the matrix M that describes our dynamical system $(\mathcal{X}, H_f)$.

Theorem 4.1. $(\Omega, \mathcal{C}, m, \sigma)$ *is a measure preserving system.*

Proof. Let $E = \{\omega : \omega_1 = e_1, \ldots, \omega_r = e_r\}$ be a cylinder. Then, we can describe $\sigma^{-1}(E)$ as a finite union on $\mathcal{S}$:
$$\sigma^{-1}(E) = \bigcup_{i \in S} \sigma_i^{-1}(E),$$
where $\sigma_i^{-1}(E) = \{\omega : \omega_1 = i, \ldots, \omega_2 = e_1, \ldots, \omega_{r+1} = e_r\}$. Thus $\sigma^{-1}(E)$ is measurable since it is a finite union of measurable sets. Let us prove the invariance of H_f. We have
$$m(E) = \pi_{e_1} m_{e_1 e_2} m_{e_2 e_3} \ldots m_{e_{r-1} e_r}$$

$$m(\sigma_j^{-1}(E)) = \pi_j m_{j e_1} m_{e_1 e_2} \ldots m_{e_{r-1} e_r}$$
thus
$$m(\sigma^{-1}(E)) = m(\bigcup_{i \in S} \sigma_i^{-1}(E)) = \sum_{i \in S} m(\sigma_i^{-1}(E)) =$$
$$= m_{e_1 e_2} \ldots m_{e_{r-1} e_r} \sum_{i \in S} \pi_i m_{i e_1}$$
since $\pi P = \pi$ we have that $\sum_{i \in S} \pi_i m_{i e_1} = \pi_{e_1}$, and thus
$$m(\sigma^{-1}(E)) = m_{e_1 e_2} \ldots m_{e_{r-1} e_r} \cdot \pi_{e_1} = m(E).$$

□

For a finite state Markov chain the notions of ergodicity and mixing can be characterized in the following way:

Definition 4.4. *A matrix M is said to be*

- irreducible *if for every i, j, there exists $k \in \mathbb{N}^*$ such that $M_{ij}^k > 0$;*
- regular *if there exists $k \in \mathbb{N}^*$ such that for every i, j, $M_{ij}^k > 0$.*

It is well known that an irreducible Markov chain is ergodic, meanwhile a regular Markov chain is mixing. Thus we can deduce the following theorem on the generator CI_1

Theorem 4.2. *Let $f : 2^N \to 2^N$ be a boolean function, and let M be the adjacency matrix of Γ_f.*

1. *If M is an irreducible stochastic matrix, then CI_1 is ergodic;*

2. *if M is a regular stochastic matrix, then CI_1 is mixing.*

Under the mixing hypotheses, we can compute the entropy of a Markov chain, and thus of our generators. Let us recall briefly what entropy is, before to state the result. Let $(X,\mathcal{E},\mu,T)$ be a measure preserving system; we say that $\mathcal{P}$ is a *partition* of X, if $\mathcal{P}$ is a family of measurable and mutually disjoint subsets of X. We will always assume that $\mathcal{P}$ is finite or countable and we will write $\mathcal{P} = \{P_0,\cdots,P_k\}$ with $2 \leq k \leq \infty$.

We know how to calculate the entropy of a partition of the system, and in order to link this entropy to the entropy of the system, we would like to have a "special" partition, the so-called *generating* partition.

Definition 4.5. *Let $\mathcal{A} = \{A_1,\dots,A_r\}$ and $\mathcal{B} = \{B_1,\dots,B_s\}$ be two partitions of a measure preserving system $(X,\mathcal{E},\mu,T)$. We define:*

- *for $k \in \mathbb{Z}$, $T^k\mathcal{A} := \{T^kA_1,\dots,T^kA_r\}$*
- *$\mathcal{A}\vee\mathcal{B}$ is the* join *of the two partitions defined by*
$$\mathcal{A}\vee\mathcal{B} := \{A_i\cap B_j | i = 1,\dots,r,\ j = 1,\dots,s\}$$
- *we say that $\mathcal{B}$ is a* refinement *of $\mathcal{A}$, and we write $\mathcal{B}\geq\mathcal{A}$, if every $B\in\mathcal{B}$ is a subset of some set $A\in\mathcal{A}$ such that $\mu(B\setminus(B\cap A)) = 0$. Observe that, in this case, $\mathcal{A}\vee\mathcal{B} = \mathcal{B}$.*
- *$\mathcal{A}^n_m := \vee^n_{k=m}T^k\mathcal{A}$.*
- *A partition $\mathcal{P}$ is a* generating *partition if $\mathcal{P}^{+\infty}_{-\infty} = \mathcal{E}$, up to null measure sets.*

We can now define the entropy of a partition $\mathcal{P} = \{P_0,\dots,P_k\}$ as

$$H(\mathcal{P}) := \sum_{i=0}^{k}\phi(\mu(P_i)),$$

where ϕ is the function defined on $[0,1]$ by the formula

$$\phi(x) = \begin{cases} -x\log(x) & \text{if} \quad 0 \leq x \leq 1 \\ 0 & \text{if} \quad x = 0. \end{cases}$$

Starting from a partition $\mathcal{P}$ one can prove that the following limit is well-defined

$$h_\mu(T,\mathcal{P}) := \lim_{n\to\infty}\frac{H(\mathcal{P}^{n-1}_0)}{n},$$

and we shall call this quantity, the *entropy* of $\mathcal{P}$ relative to T.

We can also define the *metric entropy* as

$$h_\mu(T) = sup_{\mathcal{P}}h_\mu(T,\mathcal{P}).$$

Finally the Kolmogorov-Sinai theorem tell us that if $\mathcal{P}$ is a generating partition, then

$$h_\mu(T) = h_\mu(T,\mathcal{P}).$$

So far from now we saw that we can calculate the entropy of CI_1 by calculating the entropy of a generating partition for this system. We are ready to prove the following:

Theorem 4.3. *Let $f : 2^N \to 2^N$ be a boolean function with M, the Markov matrix of Γ_f, irreducible and aperiodic. Then the entropy of the measure preserving system $(\Omega,\mathcal{C},m,\sigma)$ is*

$$-\sum_{j=0}^{N-1}\sum_{i=0}^{N-1}\pi_i m_{ij}\log m_{ij}.$$

Proof. Let $(\Omega,\mathcal{C},m,\sigma)$ be the measure preserving system previously defined which describes the CI_1 algorithm. Let us define

$$P^0_i := \{\omega : \omega_0 = i\},$$

and thus the partition

$$\mathcal{P} = \{P^0_0,\dots,P^0_{N-1}\}.$$

Recalling that $\mathcal{P}^0_{-n+1}$ is by definition $\vee^0_{k=-n+1}\sigma^k\mathcal{P}$, observe that any element of $\mathcal{P}^{n-1}_0$ has the form

$$P^0_{i_0}\cap\sigma^{-1}P^1_{i_1}\cap\dots\sigma^{-n+1}P^{n-1}_{i_{n-1}},$$

for some $i_0,\dots,i_{n-1}$, and the measure of such an element is

$$\pi_{i_0}m_{i_0i_1}\dots m_{i_{n-2}i_{n-1}}.$$

Thus

$$H(\mathcal{P}^0_{-n+1}) =$$
$$= -\sum_{i_0,\dots,i_{n-1}}\pi_{i_0}m_{i_0i_1}\dots m_{i_{n-2}i_{n-1}}\log(\pi_{i_0}m_{i_0i_1}\dots m_{i_{n-2}i_{n-1}}) =$$
$$= -\sum_i\pi_i\log(\pi_i) -$$
$$\sum_{i_0,\dots,i_{n-1}}\pi_{i_0}m_{i_0i_1}\dots m_{i_{n-2}i_{n-1}}\log(\pi_{i_0}m_{i_0i_1}\dots m_{i_{n-2}i_{n-1}}) =$$
$$= -\sum_i\pi_i\log(\pi_i) - \sum_{ij}\pi_i m_{ij}\log(m_{ij}) -$$
$$\sum_{i_1,\dots,i_{n-1}}\pi_{i_1}m_{i_1i_2}\dots m_{i_{n-2}i_{n-1}}\log(m_{i_1i_2}\dots m_{i_{n-2}i_{n-1}}) =$$
$$= \dots = -\sum_i\pi_i\log(\pi_i) + n\big(-\sum_{ij}\pi_i m_{ij}log(m_{ij})\big).$$

where we applied over and over again the properties $\sum_j m_{ij} = 1$ and $\sum_i\pi_i m_{ij} = \pi_j$.

But now $\mathcal{P}$ is a generating partition since it is build up from cylinders and $\mathcal{C}$ is actually the σ-algebra generated by the cylinders. By taking the limit on n of the last quantity divided by n and applying the Kolmogorov-Sinai theorem we obtain the thesis, i.e. the entropy of CI_1 is

$$-\sum_{j=0}^{N-1}\sum_{i=0}^{N-1}\pi_i m_{ij}\log m_{ij}.$$

□

5 SECURITY ANALYSIS

The definition of PRNG should be reviewed in the cryptographic applications, see for instance (Schindler, 2013). The goal of this section is to show that chaotic iterations not only improve the *randomness* of the inputted generator, but they also preserve the security property, in a cryptographic framework. In this section, if u, v are two strings, then uv denotes the concatenation of the two strings (when it is necessary, we will write $(u)(v)$).

Definition 5.1. *Let $0 < l_1 \leq l_2$ two integers; a cryptographic PRNG (cPRNG) is an algorithm G that takes in input a seed $s \in 2^{l_1}$ and produce an output $t \in 2^{l_2}$ such that:*

A1 there should be a high probability that for different seeds the generated sequences of random numbers are different from each other;

A2 the output should be indistinguishable from a "true" random sequence, according to specified statistical tests (NIST, TestU01...);

A3 it should be impossible for any attacker (for all practical purposes) to calculate, or otherwise guess, from any given sub sequence, any previous or future values in the sequence, nor any inner state of the generator;

A4 it should be impossible, for all practical purposes, for an attacker to calculate, or guess from an inner state of the generator, any previous numbers in the sequence or any previous inner generator states

Those properties can be summarized, for instance, in the following definitions (see (Bahi et al., 2015))

Definition 5.2. *A cryptographic PRNG (cPRNG) is a deterministic algorithm G transforming strings into strings and such that, for any seed s of length m, $G(s)$ (the output of G on the input s) has size $l_G(m)$ with $l_G(m) > m$.*

Definition 5.3. *A cPRNG G is* secure *if for any probabilistic polynomial time algorithm D, for any positive polynomial p, and for all sufficiently large m s,*

$$|\mathbb{P}[D(G(\delta_m)) = 1] - \mathbb{P}[D(\delta_{l_G(m)}) = 1]| < \frac{1}{p(m)},$$

where δ_m is the uniform distribution on $\mathbb{B}^m$ and the probabilities $\mathbb{P}$ are taken over δ_m, $\delta_{l_{G(m)}}$ as well as over the internal coin tosses of D.

Intuitively, it means that there is no polynomial time algorithm that can distinguish a perfect uniform random generator from G with a non negligible probability.

A secure cPRNG based on CI_1 was presented for instance in (Bahi et al., 2015); let us recall, given a cPRNG, how to post process it with chaotic iterations:

Definition 5.4. *Let K be a cPRNG, and let $f : 2^N \to 2^N$ be a boolean function. Let us define a new cPRNG K_f, in the following way:*

- *K takes as input any string x_0S_0 of length $2N$, with $|x_0| = |S_0| = N$;*
- *assume, without loss of generality, that for any string S_0 of length N, the length of $K(S_0) = kN$, with $k \geq 2$;*
- *let $S_1, \ldots, S_k$ be the strings of length N, such that $K(S_0) = S_1 \ldots S_k$;*
- *K_f returns as output the following string of length kN*
$$(f_1^S(x_0))(f_2^S(x_0)) \ldots (f_k^S(x_0)),$$
where, for instance,
$$f_3^S(x_0) = F_f(S_3, F_f(S_2, F_f(S_1, x_0))).$$

Now we claim that if K is a secure cPRNG, then K_f defined above, is secure too.

Theorem 5.1. *Let K be a cPRNG, with $l_K(N) = kN$, $k \geq 2$. For any $y \in 2^{kN}$ let $\Phi_y : 2^{kN} \to 2^{kN}$ be the function mapping $\omega = \omega_1 \ldots \omega_k$ into $(f_1^\omega(y))(f_2^\omega(y)) \ldots (f_k^\omega(y))$. If K is secure and for every y, Φ_y is bijective, then K_f is secure too.*

Proof. Assume that K_f is not secure. Then by definition there exists a polynomial time probabilistic algorithm D and a positive polynomial p, such that for every $\varepsilon > 0$ there exists $L \geq \varepsilon/2$ satisfying:

$$|\mathbb{P}[D(K_f(\delta_{2L})) = 1] - \mathbb{P}[D(\delta_{kL}) = 1]| \geq \frac{1}{p(2L)}.$$

Let us define a new probabilistic polynomial time algorithm, say D':

- takes as input a string $\omega = \omega_1 \ldots \omega_k$ of length kL, where for all i, $|\omega_i| = L$;
- randomly choose a string y of length L;
- compute $z = (f_1^\omega(y))(f_2^\omega(y)) \ldots (f_k^\omega(y)$;
- return $D(z)$ as output.

Observe that by definition of Φ_y

$$D'(\omega) = D(\Phi_y(\omega))$$

and also

$$\Phi_y(K(x)) = K_f(yx).$$

Using these expressions we obtain

$$D'(K(x)) = D(\Phi_y(K(x))) = D(H_f(yx)),$$

thus

$$\mathbb{P}[D'(K(\delta_L))) = 1] = \mathbb{P}[D(K_f(\delta_{2L})) = 1].$$

Furthermore, since Φ_y is bijective by hypothesis, we have

$$\mathbb{P}[D'(\delta_{kL}) = 1] = \mathbb{P}[D(\Phi_y(\delta_{kL})) = 1] = \mathbb{P}[D(\delta_{kL}) = 1].$$

Putting everything together we are able to conclude that K is not secure, reaching a contradiction and thus a proof of this theorem. In fact we have found a polynomial time probabilistic algorithm D', and a positive polynomial p, such that for all $\varepsilon > 0$ there exists $L \geq \varepsilon/2$ such that

$$|\mathbb{P}[D'(K(\delta_L))) = 1] - \mathbb{P}[D'(\delta_{kL}) = 1]| =$$

$$|\mathbb{P}[D(K_f(\delta_{2L})) = 1] - \mathbb{P}[D(\delta_{kL}) = 1]| \geq \frac{1}{p(2L)}.$$

□

6 CONCLUSIONS

Designing pseudorandom number generators is an interesting and challenging task. In this article we presented a class of generators, we modeled it as a class of discrete dynamical systems or a class of Markov chains, and we used theoretical tools from topology and ergodic theory to deduce chaotic properties of these systems and in particular to compute their entropy, under the mixing assumption. A further step in this direction will be the study of the speed of mixing of these PRNG, which is interesting from both practical and theoretical points of view.

In the field of security, we proved that cPRNG based on chaotic iterations are suitable for cryptographical applications, in the sense that they preserved the security property of an inputted generator. Many other properties could be analyzed in this framework, such as a more extended study of the collision-free property, or the study of the avalanche effect. Another research direction will be to exploit the connection between the secure boolean functions (boolean function satisfying the hypothesis of Theorem 5) and the chaotic boolean functions (the ones with a regular associated Markov matrix).

REFERENCES

Bahi, J. M., Couchot, J.-F., Guyeux, C., and Richard, A. (2011). On the link between strongly connected iteration graphs and chaotic boolean discrete-time dynamical systems. In *18th International Symposium on Fundamentals of Computation Theory*. Lecture Notes in Computer Science.

Bahi, J. M., Couturier, R., C.Guyeux, and Cyrille, P. (2015). Efficient and cryptographically secure generation of chaotic pseudorandom numbers on gpu. In *The journal of Supercomputing*.

Galatolo, S., Hoyrup, M., and Rojas, C. (2007). A constructive borel-cantelli lemma. constructing orbits with required statistical properties. In *arXiv preprint arXiv:0711.1478*.

Galatolo, S., Hoyrup, M., and Rojas, C. (2012). Dynamical systems, simulation, abstract computation. In *Chaos Solitons and Fractals*.

Galatolo, S., Nisoli, I., and Rojas, C. (2014). Probability, statistics and computation in dynamical systems. In *Mathematical Structures in Computer Science*. Cambridge University Press.

Guiheneuf, P.-A. (2015). Dynamical properties of spatial discretizations of a generic homeomorphism. In *Ergodic Theory and Dynamical Systems*. Cambridge University Press.

Guyeux, C. and Bahi, J. (2012). A topological study of chaotic iterations. application to hash functions. In *Studies in Computational Intelligence*. Springer.

Marangio, L., Guyeux, C., and Bahi, J. M. (2018). On the collision property of chaotic iterations based post-treatments over cryptographic pseudorandom number generators. In *2018 IEEE Middle East and North Africa Communications COnference*. IEEE.

Schindler, W. (2013). Functionality classes and evaluation methodology for deterministic random number generators. In *Anwendungshinweise und Interpretationen (AIS). Bundesamt fr Sicherheit in der Informationstechnik.*