Cryptographic Properties and Applications of Bipermutive Cellular Automata

Luca Mariot

Dipartimento di Informatica, Sistemistica e Comunicazione,
Università degli Studi Milano - Bicocca,
l.mariot@campus.unimib.it

Nice, April 16, 2014

Outline

Outline

One-Dimensional Cellular Automata

Definition

A finite boolean one-dimensional cellular automaton (CA) is a triple $\langle n, r, f \rangle$ where $n \in \mathbb{N}$ is the number of cells, $r \in \mathbb{N}$ is the radius and $f : \mathbb{F}_2^{2r+1} \to \mathbb{F}_2$ is a boolean function specifying the CA local rule.

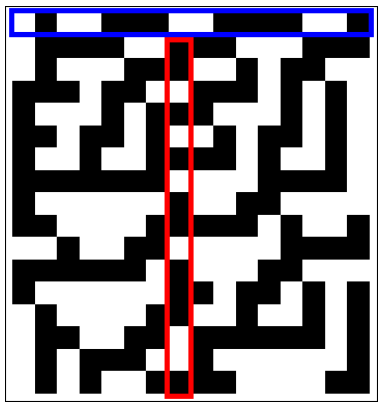
- During a single time step, a cell i updates its boolean state c_i in parallel by computing $f(c_{i-r}, \cdots, c_i, \cdots, c_{i+r})$
- Periodic CA: Each cell updates its state, and the array of n cells is seen as a ring, with the first cell following the last one
- No Boundary CA: only the central cells $i \in \{r+1, \cdots, n-r\}$ update their states; the array shrinks by $2r$ cells at each time step

Cryptographic Pseudorandom Numbers Generators

- Cryptography heavily relies upon the use of pseudorandom numbers, especially in the context of Vernam-like stream ciphers
- Cellular Automata provide an interesting framework to design Cryptographic PRNGs, for two reasons:
  - Some CAs show a chaotic dynamic behaviour, which can be exploited to make cryptanalysis harder.
  - CAs are massively parallel systems, and can be efficiently implemented in hardware (FPGA, etc.)
- The first CA-based cryptographic PRNG dates back to [Wolfram, 1986]

Wolfram's PRNG

- Main idea: sample the trace of a particular cell in a CA equipped with the elementary rule 30 (radius $r = 1$) as a pseudorandom sequence, using a random initial configuration as seed



Example with 16 cells CA, 8^{th} cell sampled. Wolfram suggested to use a CA having at least $n = 127$ cells

- Pseudorandom quality of the generated sequences assessed only by means of statistical tests

Statistical Tests and Cryptographic Properties

- Statistical testing is a necessary but not sufficient condition to verify the cryptographic robustness of a PRNG
- A failed test can be used to discard a bad generator: the null hypothesis H_0 "The generated numbers are random" is rejected
- On the other hand, a passed test cannot be used to prove the security of a generator
- There are several properties that a boolean function used in a cryptographic PRNG should satisfy, in order to resist to specific attacks

Mathematical Transforms of Boolean Functions

Some cryptographic properties of a boolean function $f : \mathbb{F}_2^m \to \mathbb{F}_2$ can be characterized through the following discrete transforms:

- Walsh transform:

$$\hat{F}(\omega) = \sum_{x \in \mathbb{F}_2^m} \hat{f}(x) \cdot (-1)^{\omega \cdot x} \quad , \forall \omega \in \mathbb{F}_2^m$$

- Autocorrelation function:

$$\hat{r}(s) = \sum_{x \in \mathbb{F}_2^m} \hat{f}(x) \cdot \hat{f}(x \oplus s) \quad , \forall s \in \mathbb{F}_2^m$$

where $\hat{f}(x) = (-1)^{f(x)}$, $\hat{f}(x \oplus s) = (-1)^{f(x \oplus s)}$ and $\omega \cdot x$ denotes the usual dot product on $\mathbb{F}_2^m$ between ω and x

Cryptographic Properties of Boolean Functions (1/2)

Some important cryptographic properties for a boolean function f:

- Balancedness: The counterimages $f^{-1}(0)$ and $f^{-1}(1)$ have the same cardinality, 2^{m-1}. This is verified if and only if $\hat{F}(0) = 0$
- Algebraic Degree: The degree of the Algebraic Normal Form of f should be as high as possible. A boolean function with degree 1 is called affine or linear
- Nonlinearity: The Hamming distance of f from the set of affine functions should be as high as possible. It is computed as $Nl(f) = 2^{-1}(2^m - W_{\max}(f))$, where $W_{\max}(f)$ is the maximum absolute value of $\hat{F}(\omega)$ for all $\omega \in \mathbb{F}_2^m$

Cryptographic Properties of Boolean Functions (2/2)

- Resiliency: f is k-resilient if by fixing at most k variables the resulting restrictions are all balanced. This is verified if and only if $\hat{F}(\omega) = 0$ for all ω having Hamming weight at most k.
- Strict Avalanche Criterion: f satisfies the SAC if, by complementing a single input variable, the probability that the output changes is $1/2$.
- Propagation Criterion: f satisfies $PC(l)$ if for all vectors $s \in \mathbb{F}_2^m$ having Hamming weight at most l it results that $\hat{r}(s) = 0$. The Strict Avalanche Criterion corresponds to $PC(1)$
- Absence of Linear Structures: there should be no nonzero vector $s \in \mathbb{F}_2^m$ such that $f(x)f(x \oplus s)$ is constant. This condition is verified if and only if $|\hat{r}(s)| \neq 2^m$

Cryptographic Properties of Elementary CA Rules

- The elementary rule 30 used by Wolfram is both balanced and nonlinear, but it is not 1-resilient.
- More generally, [Martin, 2008] showed that there are no elementary rules which are both nonlinear and 1-resilient
- CA-based PRNGs using nonlinear elementary rules are thus vulnerable to correlation attacks
- Consequence: necessity to explore the sets of rules having radii $r > 1$ to find good trade-offs between cryptographic properties and pseudorandom quality of the generated sequences

Outline

Permutive and Bipermutive Functions

Notation: by $(x, \tilde{x}_{\{i\}})$ we denote the vector

$$(x, \tilde{x}_{\{i\}}) = (x_1, \ldots, x_{i-1}, \tilde{x}, x_i, \ldots, x_{m-1}) \in \mathbb{F}_2^m \ ,$$

where $x \in \mathbb{F}_2^{m-1}$ and $\tilde{x} \in \mathbb{F}_2$.

Definition

A boolean function $f : \mathbb{F}_2^m \to \mathbb{F}_2$ is i-permutive if, for all $x \in \mathbb{F}_2^{m-1}$, it results that $f(x, 0_{\{i\}}) \neq f(x, 1_{\{i\}})$.
Function f is called:

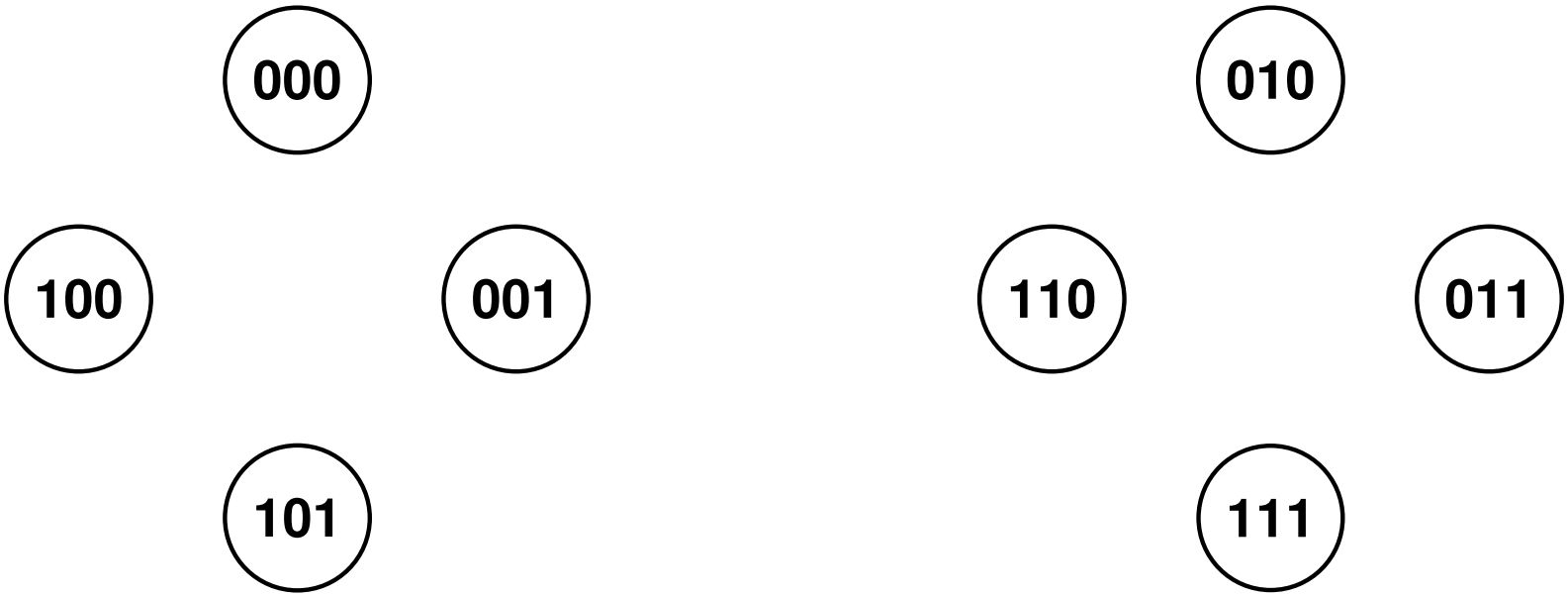
- leftmost (rightmost) permutive if it is 1-permutive (m-permutive)
- bipermutive if it is both leftmost and rightmost permutive

Chaotic CAs Induced by Bipermutive Rules

- Bipermutive rules are known to induce strongly chaotic CAs, when the latter are considered as discrete time dynamical systems on the set of biinfinite configurations $A^{\mathbb{Z}}$
- In particular, the two following results hold:
  - A CA based on a rule which is either leftmost or rightmost permutive is mixing chaotic
  - A CA based on a bipermutive rule is expansively chaotic
- Hence, bipermutive rules seem to be good candidates to design a CA-based PRNG

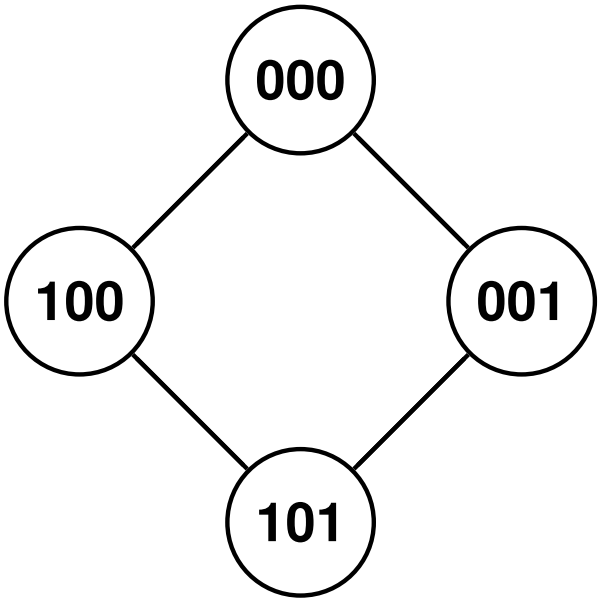
Graph-Based Enumerative Encoding for Bipermutive Rules (1/4)

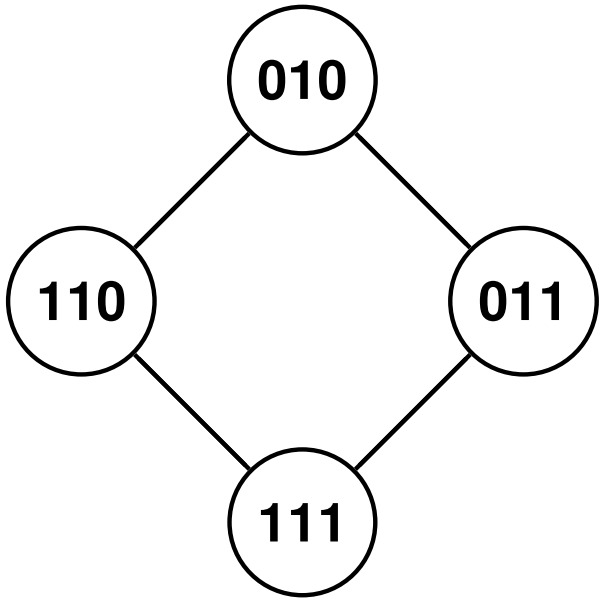
- Idea: represent the input vectors $x \in \mathbb{F}_2^m$ as vertices of an undirected graph $G = (V, E)$



Graph-Based Enumerative Encoding for Bipermutive Rules (2/4)

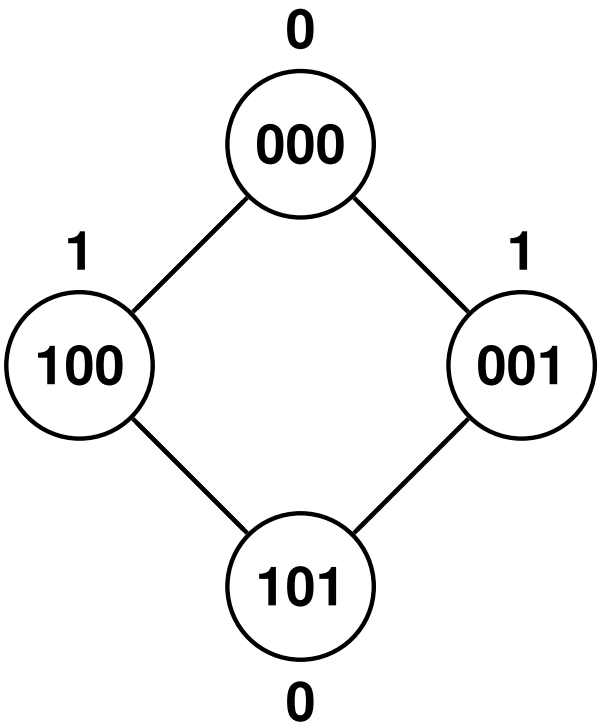
- Only those inputs which differ either in the leftmost or rightmost variable and agree on the remaining coordinates are connected

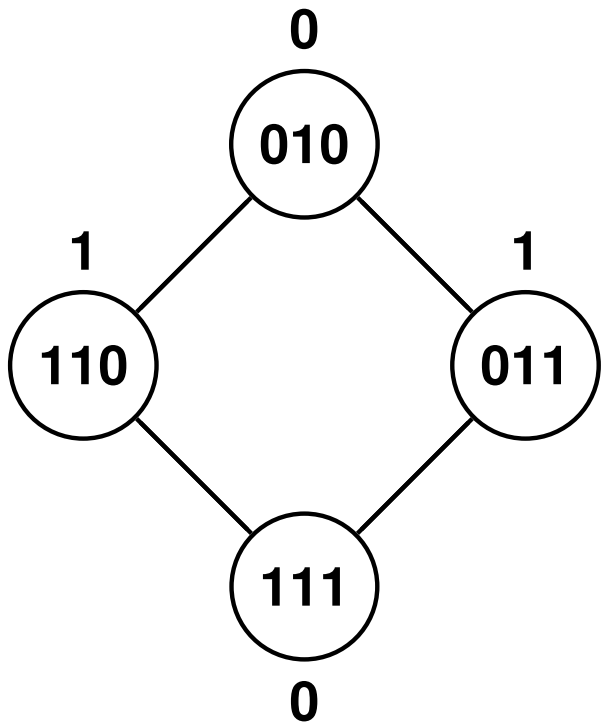




Graph-Based Enumerative Encoding for Bipermutive Rules (3/4)

- A bipermutive rule is represented as a label function $f : V \to \mathbb{F}_2$, where the values of adjacent labels differ





Graph-Based Enumerative Encoding for Bipermutive Rules (4/4)

- f is indexed by a binary string of length 2^{m-2}, which specifies the configuration of its representatives (shaded in gray)

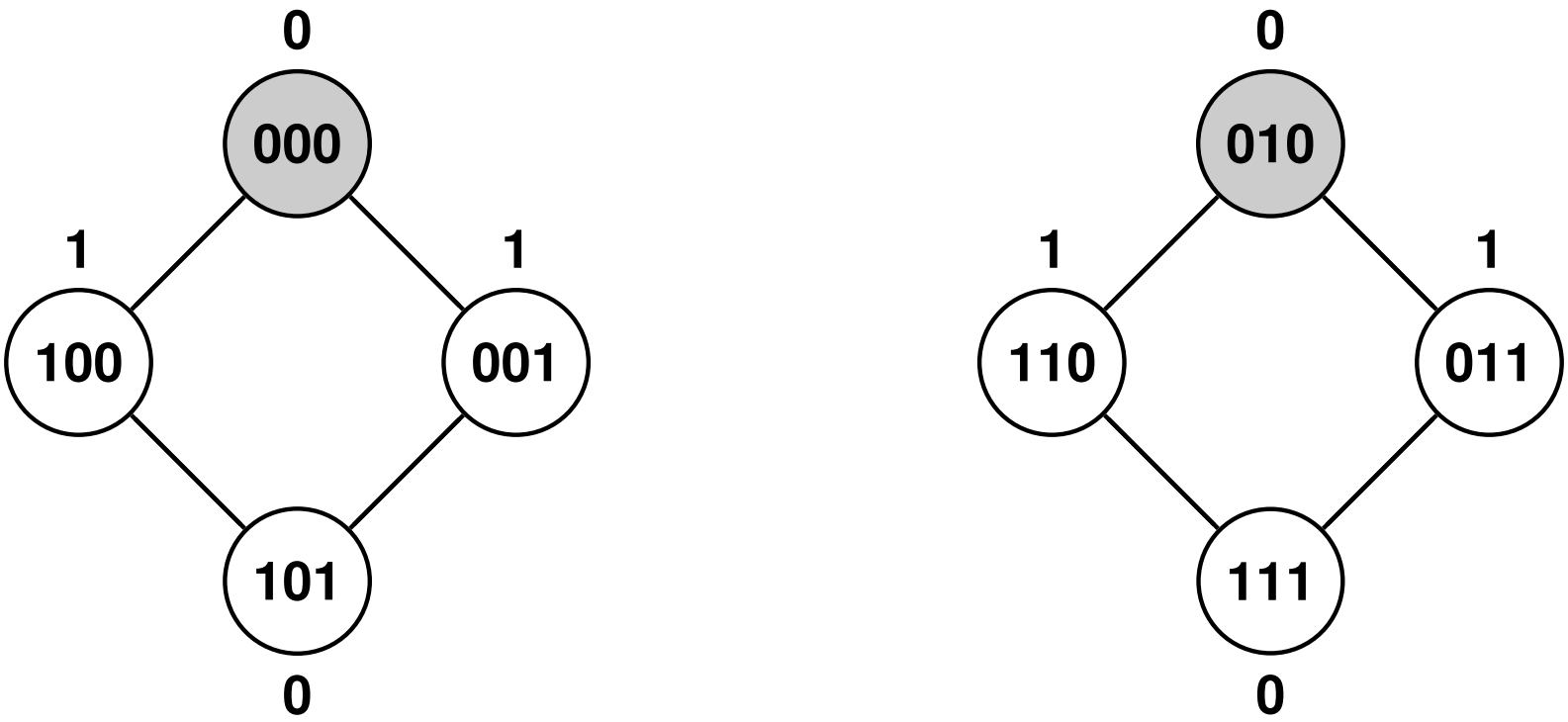
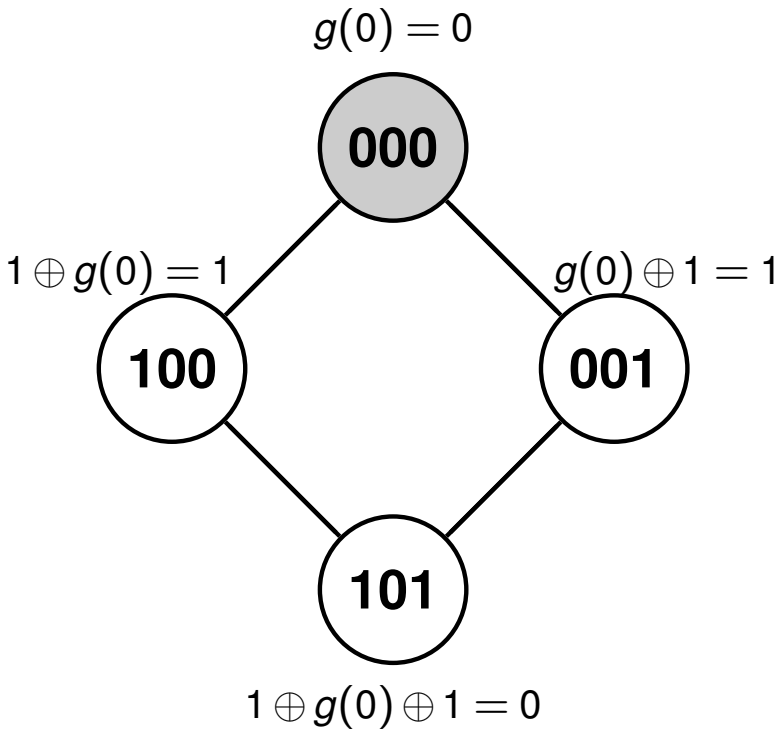
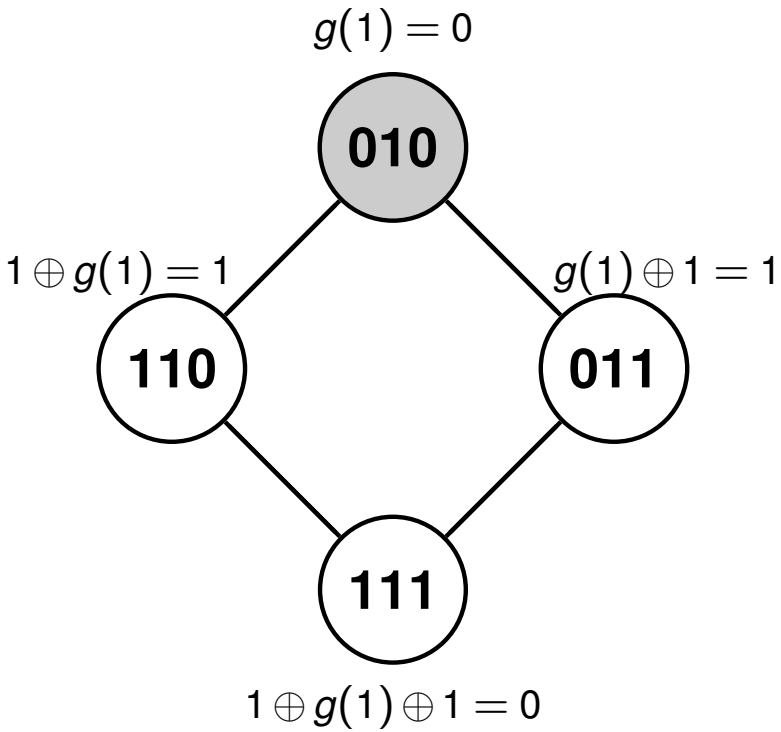


Figure: Representation of rule 90, corresponding to configuration string $c = 00$

Generating Function of a Bipermutive Rule (1/2)

- Let us consider the configuration string c of f as the truth table of a boolean function $g : \mathbb{F}_2^{m-2} \rightarrow \mathbb{F}_2$





Generating Function of a Bipermutive Rule (2/2)

- A bipermutive rule $f : \mathbb{F}_2^m \to \mathbb{F}_2$ (where $m = 2r + 1$) can thus be expressed in terms of its generating function g as follows:

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

- We can immediately deduce the following facts:
  - Every bipermutive rule f is *balanced* (just substitute (1) in the computation of $\hat{F}(0)$).
  - Let f be a bipermutive rule generated by a function g with degree $deg(g) \geq 1$. Then, the algebraic degree of f equals $deg(g)$ (otherwise, $deg(f) = 1$).

Walsh Spectrum of Bipermutive Rules

The Walsh Transform of a bipermutive rule can be efficiently computed using the following result:

Lemma

Let $f : \mathbb{F}_2^m \to \mathbb{F}_2$ *be a bipermutive rule with generating function* g.
If $\omega \in \mathbb{F}_2^m$ *is such that* $\omega_1 = 0$ *or* $\omega_m = 0$, *then*

$$\hat{F}(\omega) = 0 \ .$$

Otherwise, if both $\omega_1 = 1$ *and* $\omega_m = 1$, *then*

$$\hat{F}(\omega) = 4 \cdot \hat{G}(\omega_2, \cdots, \omega_{m-1}) \ ,$$

where $\hat{G}(\cdot)$ *is the Walsh transform of the generating function* g.

Nonlinearity and Resiliency

Consequences of the previous Lemma:

Theorem

Given a bipermutive function $f : \mathbb{F}_2^m \to \mathbb{F}_2$ and its generating function $g : \mathbb{F}_2^{m-2} \to \mathbb{F}_2$, the nonlinearity of f is equal to

$$Nl(f) = 4 \cdot Nl(g)$$

Theorem

Let $f : \mathbb{F}_2^m \to \mathbb{F}_2$ be a bipermutive function having generating function $g : \mathbb{F}_2^{m-2} \to \mathbb{F}_2$. Then, f is k-resilient if and only if g is $(k-2)$-resilient.

Autocorrelation Function and SAC

Lemma

Let $f : \mathbb{F}_2^m \to \mathbb{F}_2$ be a bipermutive rule. If $s \in \mathbb{F}_2^m$ is null in all coordinates except in the leftmost or in the rightmost one, then $|\hat{r}(s)| = 2^m$.

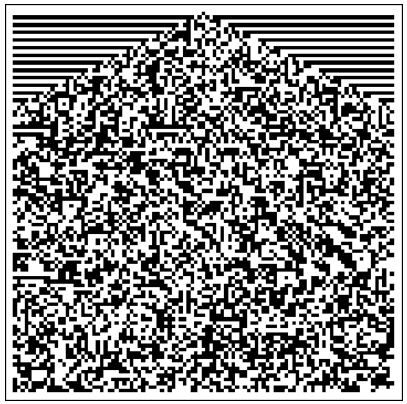
- The following facts follow from the previous lemma:
  - Every bipermutive rule has at least 3 linear structures, corresponding to the vectors $(1,0,\cdots,0)$, $(0,0,\cdots,1)$ and $(1,0,\cdots,1)$
  - A bipermutive rule never satifies the SAC, since the condition always fails in the leftmost and rightmost variables

Cryptographic Properties of Bipermutive Rules: Recap

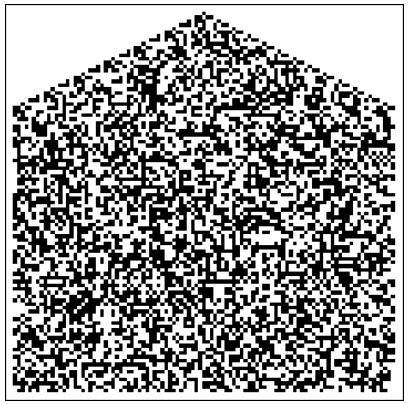
- Given a bipermutive boolean function $f : \mathbb{F}_2^m \to \mathbb{F}_2$ and its generating function $g : \mathbb{F}_2^{m-2} \to \mathbb{F}_2$ on the graph encoding:
  - The algebraic degree of f equals the degree of g
  - The nonlinearity of f is 4 times the nonlinearity of g
  - f is k-resilient if and only if g is $(k-2)$-resilient (in particular: every bipermutive rule is 1-resilient and a bipermutive rule based on a balanced generating function is 2-resilient)
  - f has at least three linear structures, and it never satisfies the SAC ($\Rightarrow$ fall back to *RSAC*: SAC on the generating function)
- Hence, the problem of finding good bipermutive rules can be reduced to the optimization of the cryptographic properties of their generating functions

Application to the Case $r = 2$

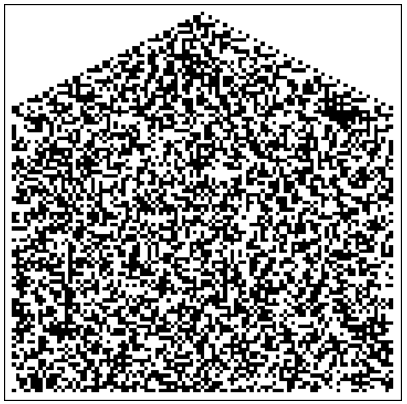
- In [Leporati and Mariot, 2013] the space of $2^{2^{5-2}} = 256$ bipermutive rules of radius $r = 2$ has been exhaustively explored
- The 56 rules being 2-resilient and having nonlinearity 8, algebraic degree 2 and 3 linear structures were subjected to the ENT and NIST test suites using a periodic CA of $n = 64$ cells. Three of them passed all the tests



(a) Rule 1452976485



(b) Rule 1520018790



(c) Rule 2778290790

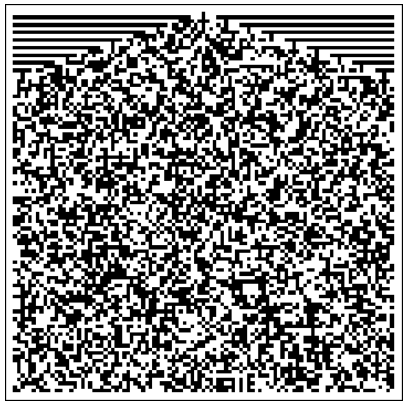
Application to the case $r = 3$ (1/2)

- A combinatorial algorithm has been used in [Mariot, 2013] to span the set of generating balanced functions in 5 variables, in order to get bipermutive rule which were at least 2-resilient
- Three groups of rules were subjected to the ENT and NIST suites, resulting in 5 rules passing all the tests

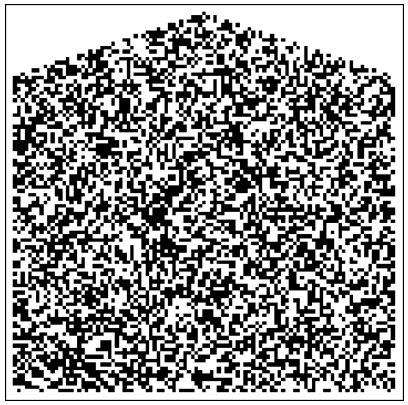
Table: *RES*: Resiliency, *NL*: Nonlinearity, *AD*: Degree, *LS*: Linear Structures, *RSAC*: Restricted SAC, $\#CARD$: Cardinality

Set ID	*RES*	*NL*	*AD*	*LS*	*RSAC*	$\#CARD$
$SET1_B$	3	48	3	3	No	96768
$SET2_B$	3	32	3	3	Yes	3840
$SET3_B$	4	32	2	7	No	520

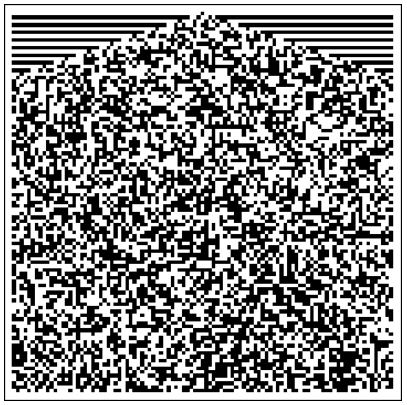
Application to the case $r = 3$ (2/2)



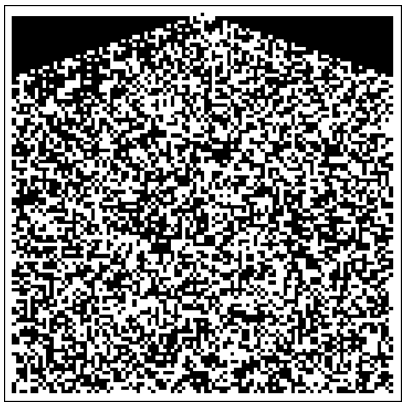
(d) R17 $\in SET1_B$



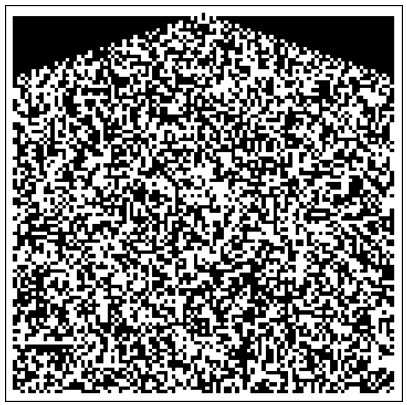
(e) R20 $\in SET1_B$



(f) R28 $\in SET1_B$



(g) R30 $\in SET2_B$



(h) R40 $\in SET2_B$

Heuristic Search (1/2)

- For all radii $r > 3$, the resulting space of bipermutive rules is too large for exhaustive search
- Example: bipermutive rules of radius $r = 4$ correspond to the space of generating functions in 7 variables, which has cardinality $2^{128} \approx 10^{34}$
- In [Mariot, 2013], the sets of balanced generating functions for bipermutive rules having radii $r = 4$, 5 and 6 have been explored using three soft computing techniques :
  - Genetic Algorithms (GA)
  - Discrete Particle Swarm Optimization (PSO)
  - Ant Colony Optimization (ACO)

Heuristic Search (2/2)

- Fitness Function maximised by the three heuristic algorithms:

$$fit(g) = NI(g) - RES(1)_g - PC(1)_g$$

where $RES(1)_g$ and $PC(1)$ are the deviations from 1-resiliency and SAC defined as:

$$RES(1) = \max\{|\hat{G}(\omega)| : hwt(\omega) = 1\}$$

$$PC(1) = \max\{|\hat{r}(s)| : hwt(s) = 1\}$$

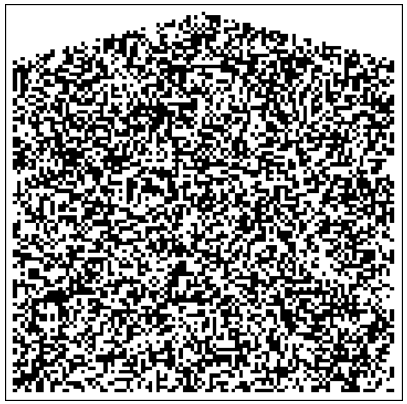
- Each algorithm managed to find generating functions which satisfied 1-resiliency and 10 rules passed all the ENT and NIST tests

Sets of Rules Generated by Heuristic Search

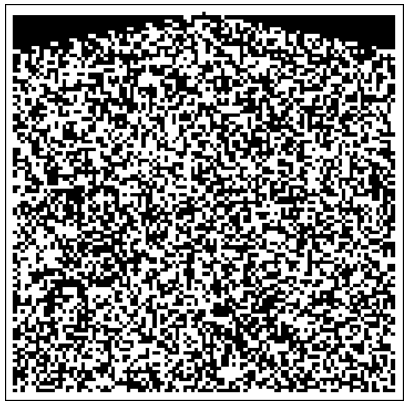
Sets generated by the three heuristic techniques which contained the 10 rules passing all the tests:

Set ID	Radius	Found By	*RES*	*NL*	*AD*	*LS*	*RSAC*
$SET7_B$	4	GA	3	224	4	3	No
$SET13_B$	5	GA	3	944	7	3	No
$SET20_B$	6	GA	3	3888	9	3	Yes
$SET21_B$	6	GA	3	3888	9	3	No
$SET23_B$	4	PSO	3	224	5	3	No
$SET34_B$	5	PSO	3	928	7	3	No
$SET49_B$	4	ACO	3	208	5	3	Yes

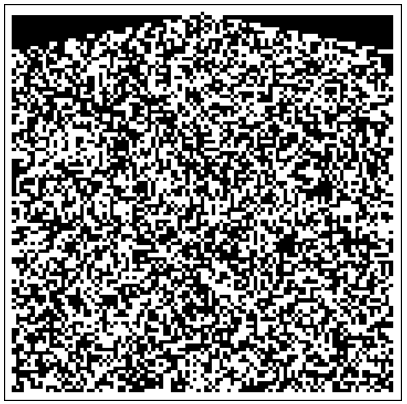
Final Rules Found by Heuristic Search (1/2)



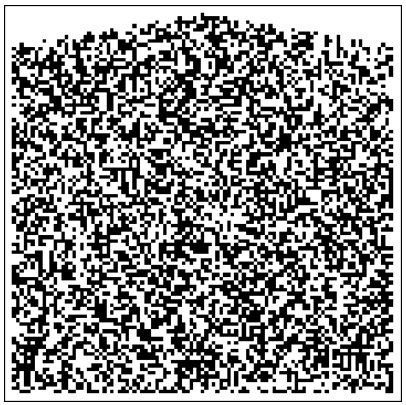
(i) R49 $\in SET7_B$



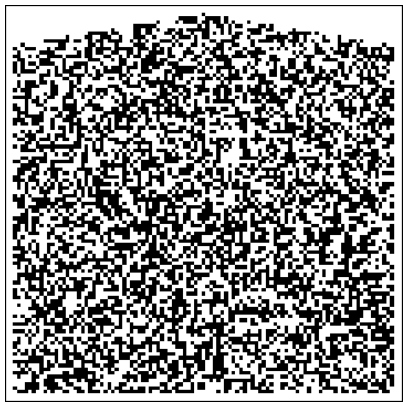
(j) R57 $\in SET13_B$



(k) R58 $\in SET13_B$

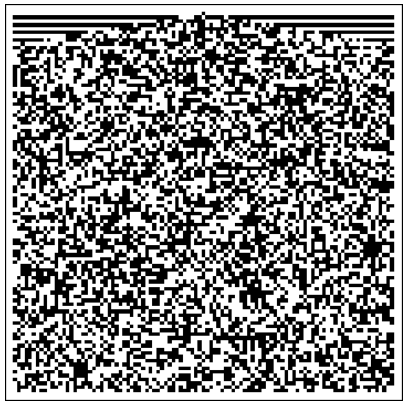


(l) R59 $\in SET20_B$

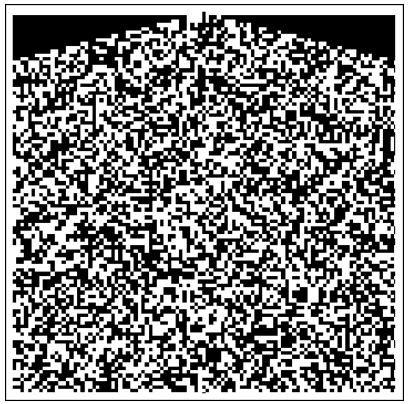


(m) R60 $\in SET20_B$

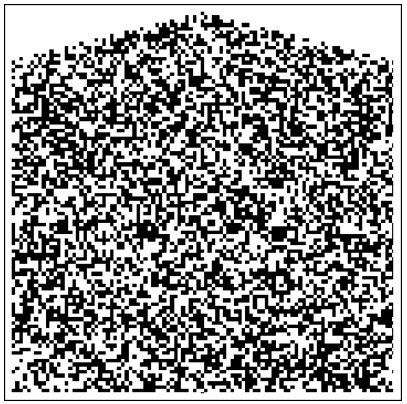
Final Rules Found by Heuristic Search (2/2)



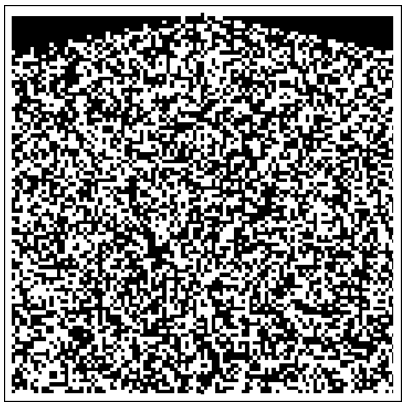
(n) R61 $\in SET21_B$



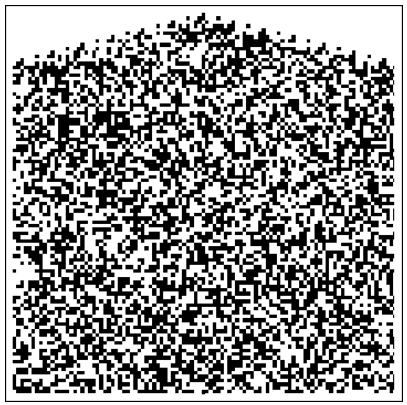
(o) R65 $\in SET23_B$



(p) R66 $\in SET23_B$



(q) R71 $\in SET34_B$



(r) R79 $\in SET49_B$

Outline

Secret Sharing Schemes (1/2)

- A secret sharing scheme is a procedure which enables a dealer to share a secret S among a set $\mathcal{P}$ of players, in such a way that only some authorized subsets can recover S.
- The authorized subsets are specified by an access structure $\Gamma \subseteq 2^{\mathcal{P}}$
- The access structure can be defined by its basis Γ_0 which contains the minimal authorized subsets. All the other subsets $A \in \Gamma$ are obtained as unions of elements from Γ_0
- In a (k, n)-threshold scheme (such as Shamir's scheme) the minimal authorized subsets are all those subsets of cardinality k

Secret Sharing Schemes (2/2)

- Let us assume that a probability distribution $Pr(S)$ is defined on the space of the secrets, and that δ_U represents a shares distribution to an unauthorized subset $U \notin \Gamma$
- A secret sharing scheme is perfect if for all unauthorized subsets $U \notin \Gamma$ and for all shares distributions δ_U it results that

$$Pr(S|\delta_U) = Pr(S)$$

- Thus, in perfect schemes an attacker which knows the shares of an unauthorized subset does not gain any information on the secret
- A secret sharing scheme is called ideal if the size of each share equals the size of the secret

Building Preimages of Permutive CAs (1/6)

Given a rightmost permutive rule $f : \mathbb{F}_2^{2r+1} \to \mathbb{F}_2$ and a configuration $c \in \mathbb{F}_2^m$, a preimage $p \in \mathbb{F}_2^{m+2r}$ of c can be computed as follows:

1. Set the leftmost $2r$ cells $p_1, \cdots, p_{2r}$ of the preimage p to random values

$p =$	**0**	**1**	**?**	**?**	**?**	**?**	**?**	**?**
$c =$		**1**	**0**	**0**	**1**	**1**	**0**	

Figure: Example of preimage construction under rule 30 (R-permutive)

Building Preimages of Permutive CAs (2/6)

Given a rightmost permutive rule $f : \mathbb{F}_2^{2r+1} \to \mathbb{F}_2$ and a configuration $c \in \mathbb{F}_2^m$, a preimage $p \in \mathbb{F}_2^{m+2r}$ of c can be computed as follows:

2. By right permutivity, the value of p_{2r+1} can be determined by computing $f(p_1, \cdots, p_{2r}, c_1)$

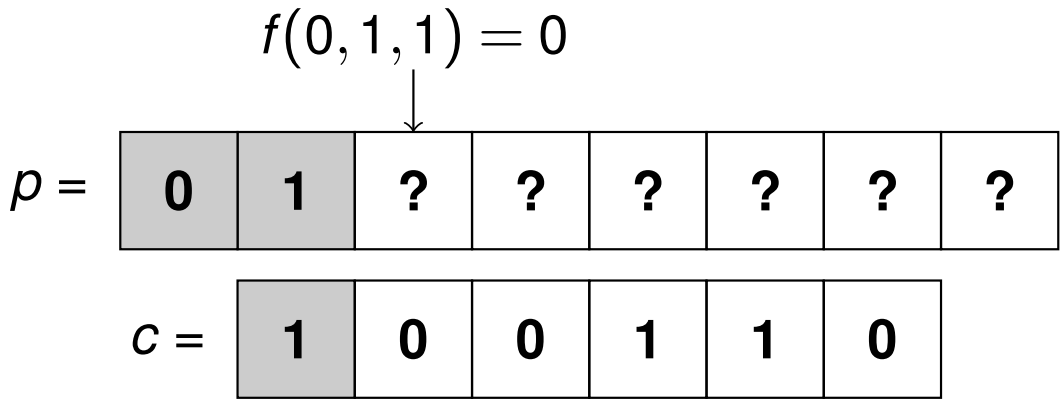


Figure: Example of preimage construction under rule 30 (R-permutive)

Building Preimages of Permutive CAs (3/6)

Given a rightmost permutive rule $f : \mathbb{F}_2^{2r+1} \to \mathbb{F}_2$ and a configuration $c \in \mathbb{F}_2^m$, a preimage $p \in \mathbb{F}_2^{m+2r}$ of c can be computed as follows:

3. Shift the $2r$-bit window one place to the right and compute $p_{2r+2} = f(p_2, \cdots, p_{2r+1}, c_2)$

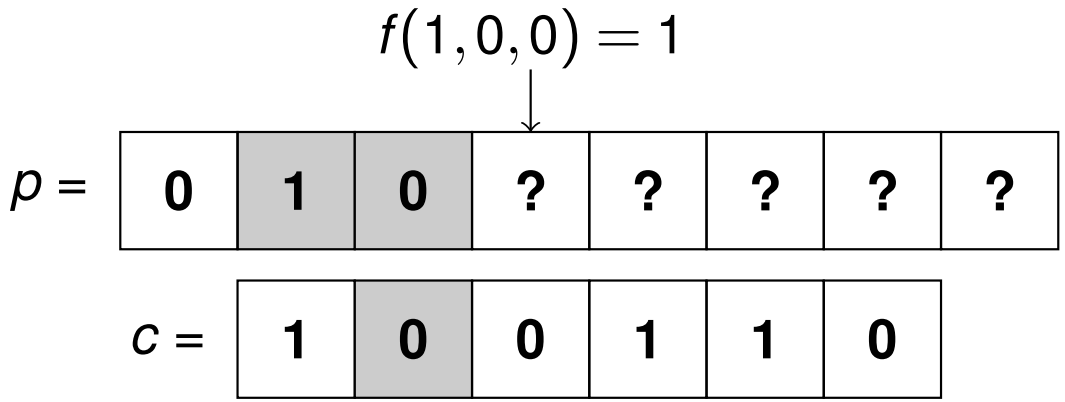


Figure: Example of preimage construction under rule 30 (R-permutive)

Building Preimages of Permutive CAs (4/6)

Given a rightmost permutive rule $f : \mathbb{F}_2^{2r+1} \to \mathbb{F}_2$ and a configuration $c \in \mathbb{F}_2^m$, a preimage $p \in \mathbb{F}_2^{m+2r}$ of c can be computed as follows:

4. Continue to apply Step 3 until the rightmost bit in the preimage has been computed

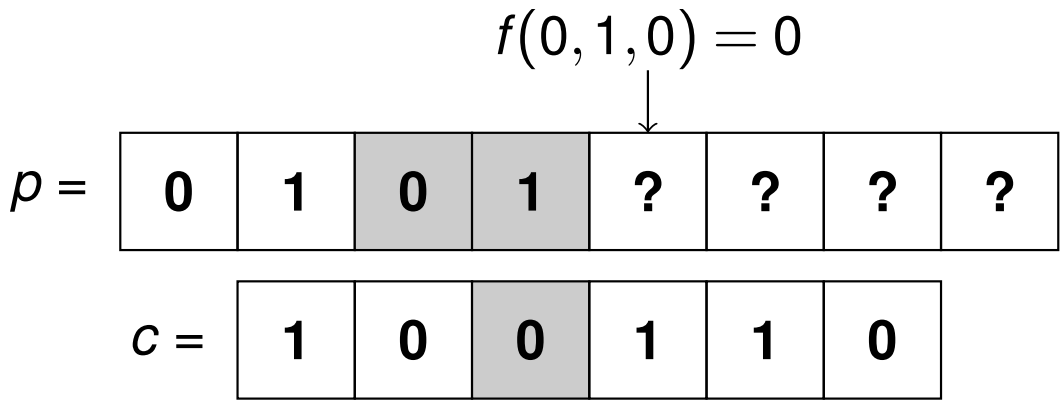


Figure: Example of preimage construction under rule 30 (R-permutive)

Building Preimages of Permutive CAs (5/6)

Given a rightmost permutive rule $f : \mathbb{F}_2^{2r+1} \to \mathbb{F}_2$ and a configuration $c \in \mathbb{F}_2^m$, a preimage $p \in \mathbb{F}_2^{m+2r}$ of c can be computed as follows:

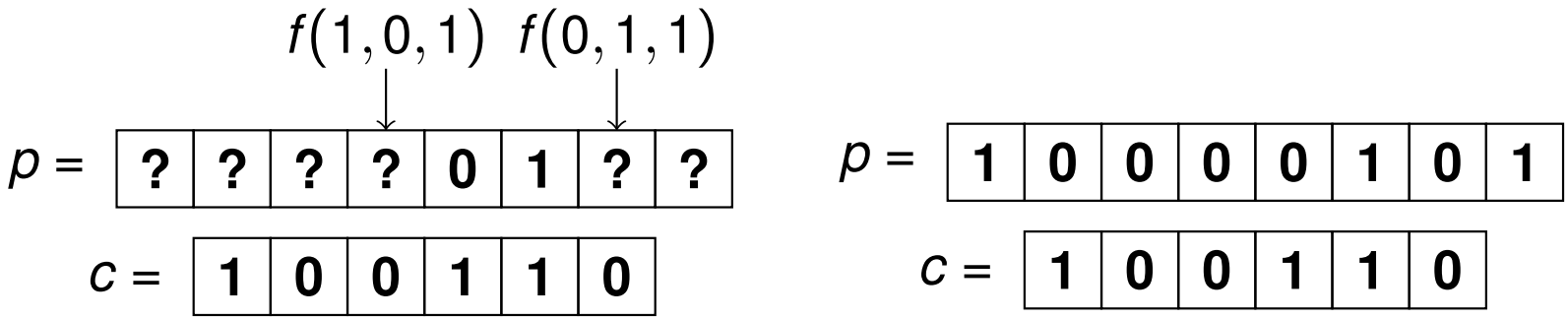
4. Continue to apply Step 3 until the rightmost bit in the preimage has been computed

$p =$	**0**	**1**	**0**	**1**	**1**	**0**	**0**	**0**
$c =$		**1**	**0**	**0**	**1**	**1**	**0**	

Figure: Example of preimage construction under rule 30 (R-permutive)

Building Preimages of Permutive CAs (6/6)

- For leftmost permutive rules, a symmetrical result holds by starting from the right and completing leftwards
- Each image in a rightmost (leftmost) permutive CA has thus 2^{2r} preimages
- If f is bipermutive, the initial block can be set in any position. This possibility does not increase the number of preimages



(a) Initialization (b) Complete preimage

Figure: Example with bipermutive rule 150

Observations on Preimage Computation

- By iterating the procedure of preimage computation, at each step the size of the preimage grows by $2r$ cells
- In particular, starting from a CA configuration c of length m, after t steps the resulting preimage will have length $L(t) = 2rt + m$
- Hence, given $k \in \mathbb{N}$, the number of iterations t necessary to get a preimage of length $k \cdot m$ is:

$$t = \frac{m(k-1)}{2r}$$

- Since t is integer, it means that $2r$ must divide $m(k-1)$
- Additional security requirement: $2r \mid m$

Basic (k, k) Secret Sharing Scheme (1/3)

Setup Phase

1. Assuming that there are k players, the *dealer* D sets the secret S as an m-bit configuration of a CA, and randomly selects a bipermutive rule of radius r, where r is such that $2r \mid m$
2. D evolves the CA backwards for $T = m(k-1)/2r$ iterations, randomly choosing at each step the value and the position of the initial $2r$-bit block
3. After T iterations, the dealer splits the resulting preimage in k blocks of m bits, and securely sends one block to each player
4. Finally, D publishes the bipermutive rule used to evolve the CA backwards

Basic (k, k) Secret Sharing Scheme (2/3)

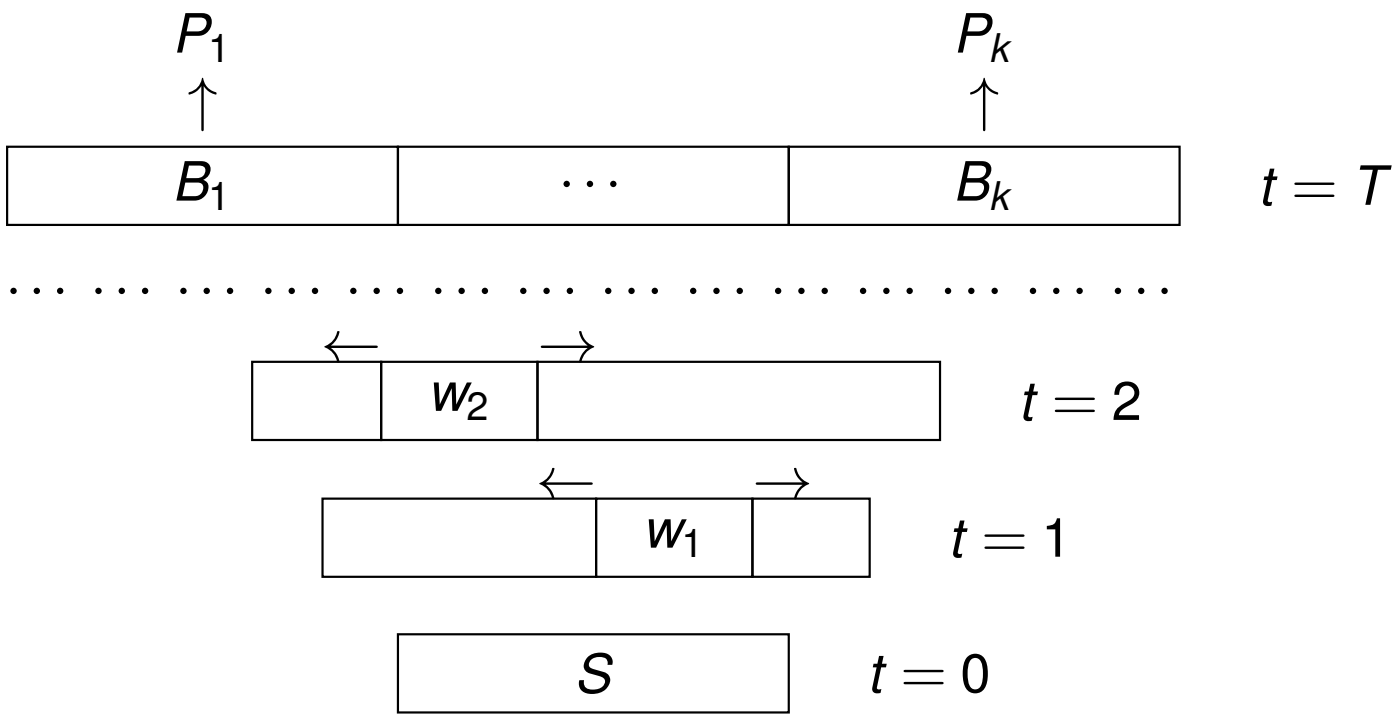


Figure: Setup phase of the (k, k) secret sharing scheme. The randomly placed blocks w_i represent the initial $2r$ random adjacent bits used to reconstruct each preimage.

Basic (k, k) Secret Sharing Scheme (3/3)

Recovery Phase

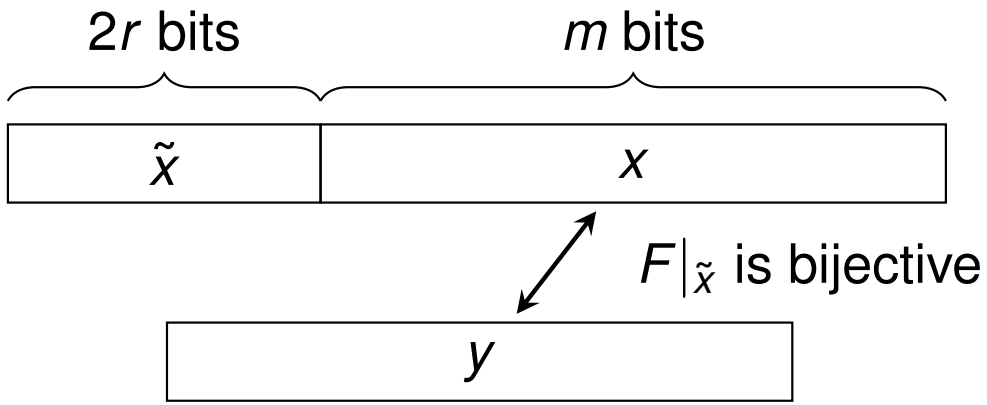
1. All the k players pool their shares in the correct order to get the complete preimage of the CA.
2. After having determined the preimage, the players evolve the CA forward for $T = m(k-1)/2r$ iterations, using the local rule published by the dealer.
3. The configuration obtained after T iterations is the secret S.

Notice that the players can compute by themselves T, since they know m (the size of a share), k (the number of players) and r (the radius of the public rule).

Security Properties of the Basic Scheme (1/2)

Lemma

Let $F : \mathbb{F}_2^{m+2r} \to \mathbb{F}_2^m$ be the global rule of a CA defined by a bipermutive local rule $f : \mathbb{F}_2^{2r+1} \to \mathbb{F}_2$. Then, by fixing the leftmost or the rightmost $2r$ cells to a value $\tilde{x} \in \mathbb{F}_2^{2r}$, the resulting restriction $F|_{\tilde{x}} : \mathbb{F}_2^m \to \mathbb{F}_2^m$ is a permutation on $\mathbb{F}_2^m$.



Security Properties of the Basic Scheme (2/2)

Lemma

Let B_l, with $1 \leq l \leq k$, be the only unknown share among $B_1, \cdots, B_k$. Then, under the condition that $2r \mid m$, there exists a permutation $\Pi : \mathbb{F}_2^m \to \mathbb{F}_2^m$ between B_l and the secret S.

From the previous Lemma, the following result holds:

Theorem

Suppose that the secret S and the $2r$-bit blocks in the setup phase are chosen uniformly at random. Then, the basic (k, k) scheme is perfect

Moreover, the basic scheme is also ideal, since each share is a block of m bits, as the secret

Extended Scheme (1/3)

- The basic scheme is impractical for more flexible access structures ($\Rightarrow$ the dealer has to re-run the setup phase for each authorized subset)
- Necessity to find an extended scheme which allows one to reuse the same shares
- Suppose that a set of k shares has been distributed to k players using the basic setup phase. In order to add an additional player, use the following procedure:
  1. Append a copy of the secret S to the right of the final CA image
  2. Update the preimages by completing them rightwards (note that it is not necessary to pick extra random bits)
  3. The last preimage will contain an additional block for the new player.

Extended Scheme (2/3)

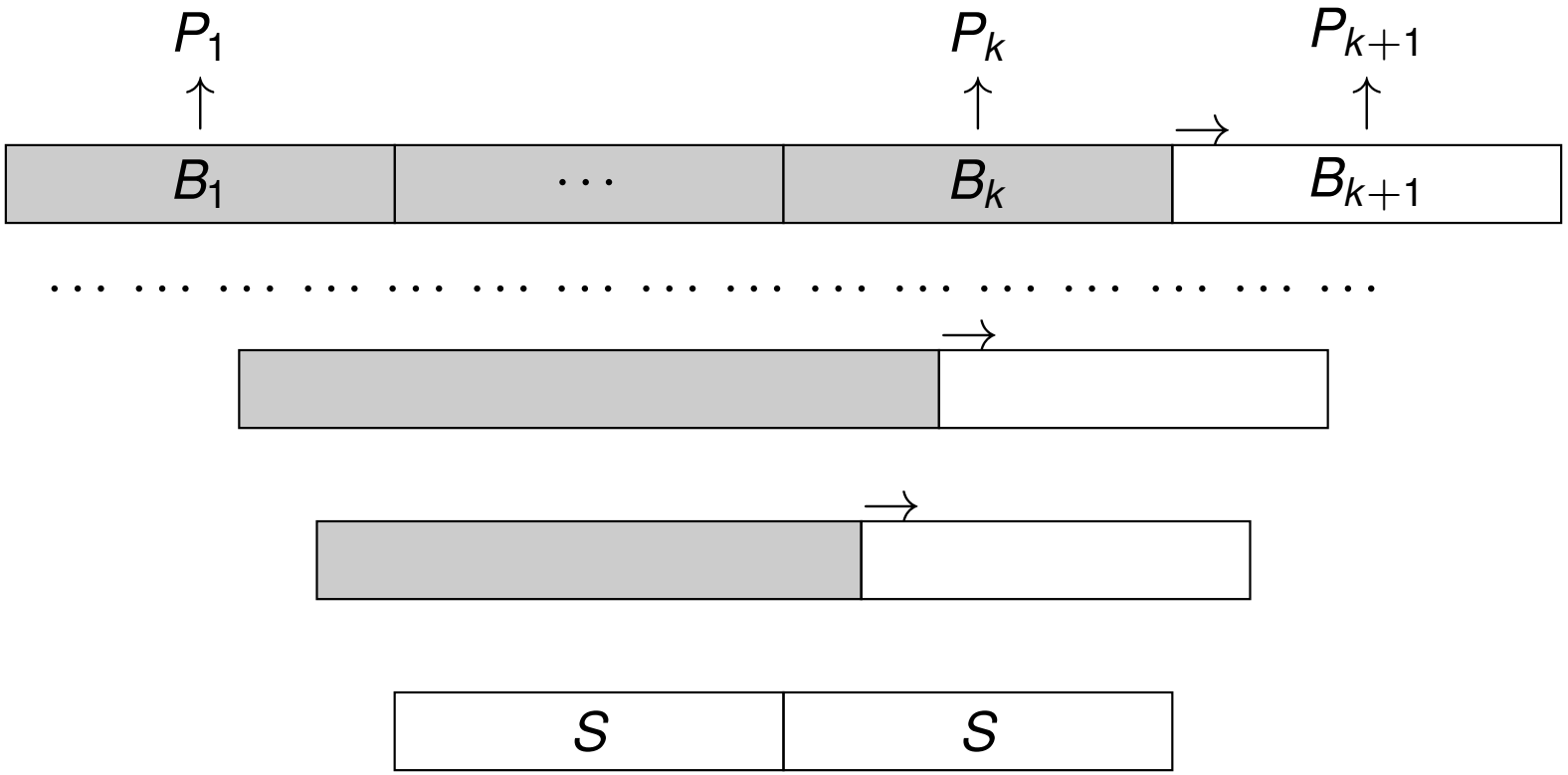


Figure: Extended scheme with $k+1$ players. The greyed out blocks are the known pieces of CA preimages after the first setup phase. In this case, the two sets $\{P_1, \cdots, P_k\}$ and $\{P_2, \cdots, P_{k+1}\}$ can recover the secret

Extended Scheme (3/3)

- The extended scheme implements a (k, n)-sequential threshold access structure: at least k consecutive shares are necessary to recover the secret
- In particular, if we continue to append copies of the secret, the final shares will eventually repeat. Thus, the access structure becomes cyclic

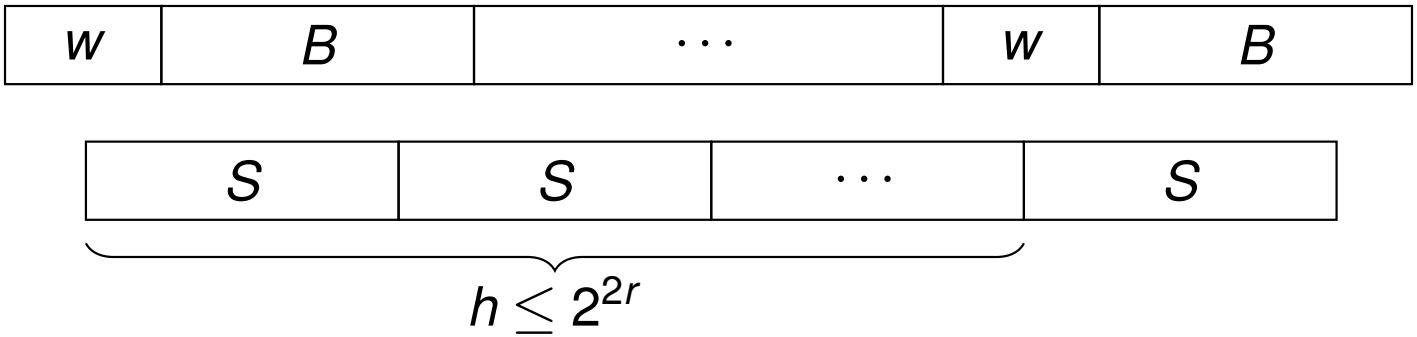


Figure: After at most $h \leq 2^{2r}$ juxtaposed copies of S, by completing rightwards th the $2r$-bit block w will be repeated at the end of the preimage.

Outline

Conclusions

- Bipermutive CAs are interesting for cryptographic CA-based PRNGs design, since they are strongly chaotic and 1-resilient
- The remaining cryptographic properties of a bipermutive rule can be computed considering its generating function
- Combinatorial and heuristic techniques can be applied to explore the spaces of bipermutive rules of high radii
- Besides PRNGs, the surjectivity of bipermutive CAs can be employed to design a perfect and ideal secret sharing scheme with cyclic access structure

Future Developments (CA-based PRNGs)

Some possible future directions of research about CA-based PRNG design include:

- Study the cryptographic properties of other classes of local rules which generate chaotic CAs (e.g., shifted rules with blocking words)
- Devise more sophisticated combinatorial techniques, in order to enumerate generating functions which satisfy stricter cryptographic properties (e.g., higher orders of resiliency)
- Cryptanalyse Wolfram's generator based on the new local rules

Future Developments (Secret Sharing Schemes)

Further improvements about the secret sharing scheme:

- Find a general method to compute after how many juxtapositions of the secret the shares begin to repeat themselves. This is equivalent to the following open problem:

Open Problem

Given a bipermutive CA and a spatially periodic configuration $c \in A^{\mathbb{Z}}$ with period m, find the periods of its preimages

- Other improvements: investigate possible applications of the scheme to secure multiparty computation protocols, and extend the scheme to d-dimensional CA with $d > 1$

Thanks for your attention!

References


Leporati, A., Mariot, L.: 1-Resiliency of Bipermutive Cellular Automata Rules. In: Kari, J., Kutrib, M., Malcher, A. (eds.) AUTOMATA 2013. LNCS, vol. 8155, pp. 110-123. Springer, Heidelberg (2013)


Mariot, L.: Cryptographic Pseudorandom Number Generators Based on Chaotic Cellular Automata. M.Sc. thesis, Università Milano-Bicocca (2013)


Martin, B.: A Walsh Exploration of Elementary CA Rules. J. Cell. Aut. 3(2), 145-156 (2008)


Millan, W., Clark, A., Dawson, E.: Heuristic Design of Cryptographically Strong Balanced Boolean Functions. In: Nyberg, K. (ed.) EUROCRYPT '98. LNCS, vol. 1403, pp. 489-499. Springer, Heidelberg (1998)


Wolfram, S.: Random Sequence Generation by Cellular Automata. Adv. Appl. Math. 7(2), 123-169 (1986)