A Walsh exploration of elementary CA rules

Bruno Martin

A Walsh exploration of elementary CA rules

Bruno Martin
Université de Nice Sophia-Antipolis,
Laboratoire I3S, Projet Recif, UMR CNRS 6070,
930 Route des Colles, BP 145,
F–06903 Sophia–Antipolis Cedex, France.
email : Bruno.Martin@unice.fr

April 12, 2007

Abstract

In this paper, we explore the 256 elementary cellular automata rules by a Walsh transform in order to find out correlation-immune rules for generating good pseudo-random sequences. We prove that, except the 8 linear rules, there is no correlation-immune rule among the 256 rules. Thus, Wolfram cellular automata cannot be used as a cryptographic pseudo-random generator since the generated pseudo-random sequences are susceptible of correlation attacks. We conclude with some remarks on cryptography with cellular automata.

Keywords
Cellular automata, elementary rules, secret key cryptography, pseudo-random generation, Walsh transform.

Introduction

In his seminal paper [15], Wolfram proposed to use cellular automata (CA for short) to build a pseudo-random generator. He suggested that pseudo-random sequences generated by rule 30 could be used for cryptographic purposes as keys for a Vernam-type cipher when run on an initial configuration of 127 cells arranged on a ring. More recently, B. Preneel in [9] quantifies the security of this pseudo-random generator and claims that it is insecure for an initial configuration of size smaller than 1000. Actually, the pseudo-random sequence generated passed all classical statistical tests, suggesting a good pseudo-random quality. Unfortunately, a few years later, Meier and Staffelbach [7] proved that the Wolfram pseudo-random generator is in fact very weak. The reason why their attack was successful comes from the fact that there are correlations between the inputs and the output of rule 30. This correlation is used in the attack to recover an initial configuration leading to the same pseudo-random sequence.

In this paper, we propose to use a discrete transform, the Walsh transform, to explore the set of all the 256 elementary CA rules. The Walsh transform is a well-known tool in the field of cryptology for studying the correlation-immunity of Boolean functions: Xiao and Massey [18] have characterized the notion of correlation-immunity with the Walsh transform. We apply this technique to the pseudo-random sequences generated by all of the 256 binary rules and we provide evidence that there does not exist a non-linear rule which generates a correlation-immune pseudo-random sequence.

The paper is organized as follows: in section 1, we recall the definition of CA and how they are used by Wolfram to generate pseudo-random sequences; we also define the Walsh transform and its relation with correlation-immunity. Section 2 applies the Walsh transform to the set of the 256 binary rules and shows firstly that the choice of rule 30 was the best choice Wolfram could make and, secondly, that this choice does not provide a correlation-immune function. Finally, in section 3, we make final remarks on the generation of pseudo-random sequences with cellular automata and their use in cryptography. We assume the reader familiar with basic concepts of cryptology for which a good introduction is [12, 6].

1 Definitions and notation

In this section, we briefly recall the definition of cellular automata and we focus on the so called elementary rules (or Wolfram rules). We also define a discrete transform, the Walsh-Hadamard transform which allows to quickly study the space of the rules in order to find rules with low correlation.

1.1 Cellular automata

A *cellular automaton* is generally a bi-infinite array of identical cells which evolve synchronously according to a local transition function and communicate with their nearest neighbors. Here, we consider a ring of N cells which are indexed by $\mathbb{Z}_N$. All the cells are identical finite state machines with a finite number of states and a transition function which gives the new state of a cell according to its current state and the current states of its nearest neighbors.

Definition 1 *A* cellular automaton *is a finite array of identical cells indexed by $\mathbb{Z}_N$. Each cell is a finite state machine $C = (Q, f)$ where Q is a finite set of states and f a mapping $f : Q \times Q \times Q \to Q$.*

The mapping f, called *local transition function*, has the following meaning: the state of cell i at time $t+1$ (denoted by x_i^{t+1}) depends upon the state of cells $i-1$, i and $i+1$ at time t. We have the following equality: $x_i^{t+1} = f(x_{i-1}^t, x_i^t, x_{i+1}^t)$. For a fixed t, the sequence of all the values x_i for $i \in \mathbb{Z}_N$, is the *configuration* at time t. It is a mapping $c : \mathbb{Z}_N \to Q$ which assigns a state of Q to each cell of the cellular automaton.

We will restrict ourselves to the case where the set of states $Q = \mathbb{F}_2$ and f is a Boolean predicate with 3 variables, also called an *elementary rule*. These cellular automata have been considered by Wolfram in [17]: he considers the 256 different binary cellular automata and associates a natural number to each rule as follows:

$(x_{i-1}^t x_i^t x_{i+1}^t)$	111	110	101	100	011	010	001	000
x_i^{t+1}	0	1	0	1	1	0	1	0

The top line gives all possible preimages for f while the bottom line gives the images by f of the three binary values. Thus, f is fully specified by the 8-bit number written on the bottom line (01011010 in our example) which can be translated in basis 10 (as rule number 90 in our example). Equivalently, this rule can be considered as a Boolean function with (at most) 3 variables. Taking rule 90 again, the corresponding Boolean function is $x_i^{t+1} = x_{i-1}^t \oplus x_{i+1}^t$ with $\oplus$ denoting the Boolean XOR function.

More specifically in [15, 16], Wolfram uses a one-dimensional cellular automaton for pseudo-random bit generation by selecting the values taken by a single cell when iterating the computation of rule 30 from an initial finite configuration where the cells are arranged on a ring of N cells. Mathematically, Wolfram claims the sequence $\{x_i^t\}_{t\geq 0}$ is pseudo-random for a given i. Wolfram studied this particular rule extensively, demonstrating its suitability as a high performance randomizer which can be efficiently implemented in parallel; indeed, this is one of the pseudo-random generators which was shipped with the connection machine CM2.

Since we are dealing with pseudo-random generators, some rules are equivalent from this point of view by three transformations. To define the transformations, just recall that a transition is $f(x) = y$ with $x \in \mathbb{F}_2^3$ and $y \in \mathbb{F}_2$. In the sequel, we denote by $\tilde{w}$ the mirror image of word $w = w_1 \dots w_n$, $\tilde{w} = w_n \dots w_1$ and by $\overline{w}$ the word obtained from w by exchanging the 0's by 1's (and conversely) $\overline{w} = \overline{w_1} \dots \overline{w_n}$. The first is the *conjunctive* transformation which takes as an input rule r written in binary and returns $\overline{\tilde{r}}$. For instance, the conjunctive transformation turns rule 30 into rule 135. The second transformation, called *reflexive* just changes the order of the x's by: $y_i = f(\tilde{x_i})$ for $i \in [\![0, 7]\!]$. As an example, with reflexive transformation, rule 30 is changed into rule 86. The last transformation combines both and is called *conjunctive-reflexive*: $y_i = \overline{f(\overline{\tilde{x_i}})}$ for $i \in [\![0, 7]\!]$ and it changes rule 30 into rule 149. All these transformations keep the spectral values of the cellular automata dynamics and are thus statistically equivalent.

1.2 Walsh transform

Walsh transform allows to compute the correlation between the inputs and the outputs of the iterations of a cellular automaton. One of the great advantages of the Walsh transform is that its computation is even faster than computing a FFT (see [3] for instance).

Let us denote by $f(\underline{x})$ the value of function f at $\underline{x} = (x_0, x_1, x_2, \ldots, x_{n-1}) \in \mathbb{F}_2^n$ or, equivalently, $f(x)$, the value of f at $x = \sum_{i=0}^{n-1} x_i.2^i$, the decimal value corresponding to $\underline{x}$. Analogously, let $\underline{\omega} = (\omega_0, \ldots, \omega_{n-1}) \in \mathbb{F}_2^n$ and ω its corresponding decimal value. The *Walsh transform* of f is defined by:

$$S_f(\omega) = \sum_{x=0}^{2^n-1} f(x) \times (-1)^{\underline{x}\cdot\underline{\omega}}$$

with $\underline{x} \cdot \underline{\omega} = \sum_{i=0}^{n-1} x_i.\omega_i$ denoting the Cartesian product of the two binary vectors. From the spectral values of the Walsh transform, one can recover the function f with the *inverse Walsh transform*:

$$f(x) = 2^{-n} \sum_{\omega=0}^{2^n-1} S_f(\omega) \times (-1)^{\underline{x}\cdot\underline{\omega}}$$

The Walsh transform has some interesting statistical properties. For instance, the value of the transform at point 0 equals the mean value of the function: $S_f(0) = E[f(x)] = 2^{n-1}$. This property permits to test whether f is *balanced* (the number of 0 equals the number of 1 in the image domain of f). In addition, Walsh transform is the main tool to study the *correlation-immunity* of a function.

Definition 2 *A function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ is k-th* order correlation-immune *if, given n independent binary random variables $X_0, X_1, \cdots, X_{n-1}$ such that $P[X_i = 0] = P[X_i = 1] = \frac{1}{2}$ for $i \in [\![0, n-1]\!]$, then the random variable $Z = f(X_0, X_1, \ldots, X_{n-1})$ is independent from any random vector $(X_{i_1}, X_{i_2}, \ldots, X_{i_k})$, $0 \leq i_1 < \cdots < i_k < n$.*

In [18], Xiao and Massey have characterized (k-th order) correlation-immunity with the Walsh transform. We recall this result in Theorem 1 in which the *Hamming weight* just counts the number of non-zero values in a vector.

Theorem 1 *A function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ is k-th order correlation-immune if and only if $S_f(\omega) = 0$ for all $\omega = (\omega_0, \omega_2, \cdots, \omega_{n-1}) \neq 0$ whose Hamming weight is at most k.*

For readers interested in a readable proof of theorem 1, one can refer to [20].

Actually, the idea of using Walsh transform to test pseudo-random generators comes from [19]. In this paper, Yuen observed that a truly random sequence has an asymptotically flat Walsh power spectrum. This observation was used to devise a new test for randomness of the output of pseudo-random generators. It was an improvement of the tests described in Knuth [5] who does not deal with *cryptographic* pseudo-random generation.

2 A correlation study of Wolfram rules

We use a Walsh transform to study all the 256 elementary CA rules in order to find the best (non-linear) rules for generating pseudo-random sequences. To

t	1		2		3		4		5		conj	refl	c.r.
rule	cfg	val	cfg	val	cfg	val	cfg	val	cfg	val			
30	4	2	16	4	64	16	256	40	1024	80	135	86	149
60	0	0	0	0	0	0	0	0	0	0	195	102	153
86	1	2	1	4	1	16	1	40	1	80	149	30	135
90	0	0	0	0	0	0	0	0	0	0	165	90	165
102	0	0	0	0	0	0	0	0	0	0	153	60	195
105	0	0	0	0	0	0	0	0	0	0	105	105	105
135	4	2	16	4	64	16	256	40	1024	80	30	149	86
149	1	2	1	4	1	16	1	40	1	80	86	135	30
150	0	0	0	0	0	0	0	0	0	0	150	150	150
153	0	0	0	0	0	0	0	0	0	0	102	195	60
165	0	0	0	0	0	0	0	0	0	0	90	165	90
195	0	0	0	0	0	0	0	0	0	0	60	153	102

Table 1: "Good" rules after selection and their equivalent rules.

check the rules, we use the fast Walsh transform algorithm proposed in [3] whose time complexity is $O(n \log n)$. We proceed step by step.

We first remove all non-balanced rules by computing their Walsh transform; we only select rules f for which $S_f(0) = 4$; there are only 70 remaining balanced rules.

Among the 70 balanced rules f_i, we compute the maximum absolute value of the Walsh transform of the t^{th}-iterate of f_i at all the points $\underline{\omega}$ of Hamming weight 1 and we select the rules with a minimum spectral value. That is, we select rules f_i such that:

$$\min_{f_i} \max_{\omega=2^\ell} |S_{f_i^{(t)}}(\omega)|$$

where t denotes the iteration number and ω is the binary expansion of 2^ℓ for $\ell \in [\![0, 2t+1]\!]$, all the t bit vectors of Hamming weight one. Equivalently, we just test if, among the 70 balanced rules, there are some which are first order correlation-immune.

After this, there are only the 12 remaining rules listed in table 1 still containing linear rules which we recall in table 2. In table 2, the binary value encoding Wolfram's rule is written with the most signifiant bit on the rightmost part.

Finally, if we remove the 8 linear rules, the best remaining rules are 30, 135, 86 and 149 which are all equivalent by conjunctive, reflexive and conjunctive-reflexive transformations. None of them is first order correlation-immune nor correlation-immune as well. Thus, we can state that:

Theorem 2 *There is no non-linear correlation-immune elementary cellular automaton.*

That is the reason why pseudo-random sequences generated in this way can be reversed by a correlation attack like the one proposed in [7], although pseudo-random sequences generated by rule 30 (or, equivalently by rules 86, 135 and

Boolean function	binary	decimal
$x_i \oplus x_{i+1}$	00111100	60
$x_{i-1} \oplus x_i$	01100110	102
$x_{i-1} \oplus x_{i+1}$	01011010	90
$x_{i-1} \oplus x_i \oplus x_{i+1}$	01101001	150
$\overline{x_{i-1} \oplus x_i}$	10011001	153
$\overline{x_i \oplus x_{i+1}}$	11000011	195
$\overline{x_{i-1} \oplus x_{i+1}}$	10100101	165
$\overline{x_{i-1} \oplus x_i \oplus x_{i+1}}$	10010110	105

Table 2: Linear rules and their corresponding Boolean functions.

149) pass classical statistical tests like the ones proposed in [5]. The attack proposed by Meier and Staffelbach in [7] is simple. It suffices to write the way to obtain the sequence generated by rule 30:

$$x_i^{t+1} = x_{i-1}^t \oplus (x_i^t \vee x_{i+1}^t) \tag{1}$$

and to use the partial linearity to rewrite equation 1 as:

$$x_{i-1}^t = x_i^{t+1} \oplus (x_i^t \vee x_{i+1}^t) \tag{2}$$

From equation 1, the site values corresponding to the pseudo-random sequence are build from a triangle in the time-space diagram. And, from equation 2, one can guess the values of the leftmost part of the triangle (or equivalent partial configurations) and then find an initial configuration for generating the pseudo-random sequence, exploiting the correlation.

3 Conclusion

So, does theorem 2 destroy any hope to design a good pseudo-random generator by the means of cellular automata? Not necessarily. For instance, in [11] it was proposed to use co-evolving non-uniform cellular automata for that purpose. In this model, each cell may contain a different rule obtained by an evolutionary approach (genetic algorithm), usually 2-3 rules for the cellular automaton. A series of tests (including χ^2 test, serial correlation coefficient, entropy and Monte Carlo, but no correlation-immunity analysis) was made with good results, showing that co-evolving generators are at least as good as the best available CA randomizer. The drawback here is that the authors also use Wolfram-like rules which we proved to be not correlation-immune. This approach was further investigated more recently in [10]. In this paper, the authors generalize the former approach by considering also rules of radius 1 and 2 for non-uniform cellular automata. They find a new set of rules that were tested by a number of statistical tests required by the FIPS 140-2 standard [13] and the Marsaglia tests implemented in the Diehard program but no correlation-immunity analysis was made. And, for both approaches, it is recalled in [20] that it can be dangerous to combine "bad rules".

Last but not least, there is currently a great challenge in the use of a good pseudo-random generators; not only for use as a key for a Vernam-type cipher but also for computing a cryptographic hash function. Actually, from the collision attacks on MD5 and other hash functions (see [4] and [14]), one could use the improvements proposed by [1] and later by [8] of the hash function based on cellular automata which was originally suggested by Damgård in [2]. And, in order to design a secure cryptographic hash function, we need to have a robust pseudo-random generator.

References

[1] Daemen, J., R. Govaerts, and J. Vandewalle: *A framework for the design of one-way hash functions including cryptanalysis of Damgård's one-way function based on a cellular automaton*. In *ASIACRYPT'91*, number 739 in *LNCS*, pages 82–96. Springer-Verlag, 1991.

[2] Damgård, I. B.: *A design principle for hash functions*. In *Crypto'89*, number 435 in *LNCS*, pages 416–427. Springer-Verlag, 1989.

[3] Elliott, D.F. and K.R. Rao: *Fast transforms, algorithms, analysis, applications*. Academic press, 1982.

[4] Klima, V.: *Finding MD5 collisions - a toy for a notebook*. http://cryptography.hyperlink.cz/md5/MD5_collisions.pdf, 2005.

[5] Knuth, D.E.: *Seminumerical Algorithms*. Addison Wesley, 1969.

[6] Martin, B.: *Codage, cryptologie et applications*. Collection technique et scientifique des télécommunications. Presses Polytechniques et Universitaires Romandes, 2004.

[7] Meier, W. and O. Staffelbach: *Analysis of pseudo random sequences generated by cellular automata*. In *EUROCRYPT '91*, number 547 in *LNCS*, pages 186–200. Springer Verlag, 1991.

[8] Mihaljevic, M., Y. Zheng, and H. Imai: *A cellular automaton based fast one-way hash function suitable for hardware implementation*. In *Public Key Cryptography*, volume 1431 of *LNCS*, pages 217–234. Springer-Verlag, 1998.

[9] Preneel, B.: *Analysis and Design of Cryptographic Hash Functions*. PhD thesis, Katholieke Universiteit Leuven, 1993.

[10] Seredynski, F., P. Bouvry, and A. Y. Zomaya: *Cellular automata computations and secret key cryptography*. Parallel Comput., 30(5-6):753–766, 2004, ISSN 0167-8191.

[11] Sipper, M. and M. Tomassini: *Co-evolving parallel random number generators.* In *Parallel Problem Solving from Nature – PPSN IV*, pages 950–959. Springer Verlag, 1996.

[12] Stallings, W.: *Cryptography and Network Security.* Prentice Hall, fourth edition, 2006.

[13] Standards Technology, National Institute of: *FIPS publication 140-2, Security requirements for cryptographic modules.* US Gov. Printing Office, Washington, 1997.

[14] Wang, X. and H. Yu: *How to break MD5 and other hash functions.* In *EUROCRYPT 2005*, number 3494 in *LNCS*, pages 19–35. Springer-Verlag, 2005.

[15] Wolfram, S.: *Cryptography with cellular automata.* In *CRYPTO'85*, number 218 in *LNCS*, pages 429–432. Springer Verlag, 1985.

[16] Wolfram, S.: *Random sequence generation by cellular automata.* Adv. in applied math., 7:123–169, 1986.

[17] Wolfram, S.: *Theory and applications of cellular automata.* World Scientific, Singapore, 1986.

[18] Xiao, G Z. and J. L. Massey: *A spectral characterization of correlation-immune combining functions.* IEEE Trans. on Information Theory, 34(3):569–576, 1988.

[19] Yuen, C K.: *Testing random number generators by Walsh transform.* IEEE Trans. Computers, 26(4):329–333, 1977.

[20] Zémor, G.: *Cours de cryptographie.* Cassini, 2000.