Application of Cellular Automata for Cryptography.

Conference Paper · January 2003

Source: DBLP

CITATIONS
6

READS
1,778

3 authors:

Pascal Bouvry
University of Luxembourg
627 PUBLICATIONS **9,679** CITATIONS
SEE PROFILE

Franciszek Seredyński
Cardinal Stefan Wyszynski University in Warsaw
175 PUBLICATIONS **1,258** CITATIONS
SEE PROFILE

Albert Zomaya
The University of Sydney
1,156 PUBLICATIONS **34,760** CITATIONS
SEE PROFILE

Application of Cellular Automata for Cryptography

Pascal Bouvry[1], Franciszek Seredyński[2,3], and Albert Y. Zomaya[4]

[1] Faculty of Sciences, Technology and Communication
Luxembourg University
6, rue Coudenhove Kalergi, L-1359 Luxembourg-Kirchberg, Luxembourg
pascal.bouvry@ist.lu http://www.ist.lu/users/pascal.bouvry
[2] Polish-Japanese Institute of Information Technologies
Koszykowa 86, 02-008 Warsaw, Poland
[3] Institute of Computer Science of Polish Academy of Sciences
Ordona 21, 01-237 Warsaw, Poland
sered@ipipan.waw.pl http://www.ipipan.waw.pl/∼sered
[4] School of Information Technologies, University of Sydney
Sydney, NSW 2006 Australia
zomaya@it.usyd.edu.au http://www.cs.usyd.edu.au/∼zomaya

Abstract. New results concerning application of cellular automata (CAs) to secret key cryptography is described in this paper. One dimensional nonuniform CAs are considered for generating pseudo-random number sequences (PNSs) used in a secret key cryptographic system. The quality of PNSs highly depends on the set of applied CA rules. The search of rules relies on an evolutionary technique called cellular programming. Different rule sizes are considered. As the result of collective behavior of discovered set of CA rules very high quality PNSs are generated. Indeed the quality of PNSs outperforms the quality of known one dimensional CA-based PNS generators used for secret key cryptography. The extended set of CA rules proposed in this article makes the cryptography system much more resistant on attacks.

1 Introduction

Today there is no need for describing the increasing needs in terms of security. The emergence of ad-hoc and ubiquitous networking requires new generations of lightweight security solutions. Cryptography techniques are essential component of any secure communication. Two main cryptography systems are used today: secret and public-key systems. An extensive overview of currently known or emerging cryptography techniques used in both type of systems can be found in [13]. One of such promising cryptography techniques consists of applying cellular automata (CAs). Let's also highlight the massively parallel characteristic of CA-based solutions and their efficiency.

CAs were proposed for public-key cryptosystems by Guan [1] and Kari [5]. In such systems two keys are required: one key for encryption and the other for

R. Wyrzykowski et al. (Eds.): PPAM 2003, LNCS 3019, pp. 447–454, 2004.

decryption, and one of them is held in private, the other rendered public. The main concern of this paper are however cryptosystems with a secret key. In such systems the encryption key and the decryption key are the same. The encryption process is based on generation of pseudorandom bit sequences, and CAs can be effectively used for this purpose. CAs for systems with a secrete key were first studied by Wolfram [17], and later by Habutsu *et al.* [3], Nandi *et al.* [10] and Gutowitz [2]. Recently they were a subject of study by Tomassini & Perrenoud [15], and Tomassini & Sipper [16], who considered one and two dimensional (2D) CAs for encryption scheme. This paper is an extension of these recent studies and concerns of application of one dimensional (1D) CAs for the secret key cryptography.

The paper is organized as follows. The next section presents the idea of an encryption process based on Vernam cipher that is used for CA-based secret key cryptosystem. Section 3 outlines the main concepts of CAs, overviews current state of applications of CAs in secret key cryptography and states the problem considered in the paper. Section 4 outlines evolutionary technique called cellular programming and section 5 shows how this technique is used to discover new CA rules suitable for the encryption process. Section 6 contains the analysis of results and the last section concludes the paper.

2 Vernam Cipher and Secret Key Cryptography

Let P be a plain-text message consisting of m bits $p_1p_2...p_m$, and $k_1k_2...k_m$ be a bit stream of a key k. Let c_i be the $i-th$ bit of a cipher-text obtained by applying XOR (exclusive-or) enciphering operation:

$$c_i = p_i \; XOR \; k_i.$$

The original bit p_i of a message can be recovered by applying the same operation XOR on c_i using the same bit stream key k:

$$p_i = c_i \; XOR \; k_i.$$

The enciphering algorithm called Vernam cipher is known to be [8,13] perfectly safe if the key stream is truly unpredictable and used only one time. From practical point of view it means that one must find answers on the following questions: (a) how to provide a pure randomness of a key bit stream and unpredictability of random bits, (b) how to obtain such a key with a length large enough to encrypt practical amounts of data, and (c) how to pass safely the key from the sender to receiver and protect the key.

In this paper we address questions (a) and (b). We will apply CAs to generate high quality pseudorandom number sequences (PNSs) and a safe secret key. CAs have been used successfully to generate PNSs. We will show that by using 1D CAs, the quality of PNSs for secret key cryptography and a safety of the key can be increased.

3 Cellular Automata and Cryptography

One dimensional CA is in the simplest case a collection of two-state elementary automata arranged in a lattice of the length N, and locally interacted in a discrete time t. For each cell i called a central cell, a neighborhood of a radius r is defined, consisting of $n_i = 2r + 1$ cells, including the cell i. When considering a finite size of CAs a cyclic boundary condition is applied, resulting in a circle grid.

It is assumed that a state q_i^{t+1} of a cell i at the time $t + 1$ depends only on states of its neighborhood at the time t, i.e. $q_i^{t+1} = f(q_i^t, q_{i1}^t, q_{i2}^t, ..., q_{ni}^t)$, and a transition function f, called a *rule*, which defines a rule of updating a cell i. A length L of a rule and a number of neighborhood states for a binary uniform CAs is $L = 2^n$, where $n = n_i$ is a number of cells of a given neighborhood, and a number of such rules can be expressed as 2^L. For CAs with e.g. $r = 2$ the length of a rule is equal to $L = 32$, and a number of such rules is 2^{32} and grows very fast with L. When the same rule is applied to update cells of CAs, such CAs are called uniform CAs, in contrast with nonuniform CAs when different rules are assigned to cells and used to update them.

The first who applied CAs to generate PNSs was S. Wolfram [17]. He used uniform, 1D CAs with $r = 1$, and rule 30. Hortensius *et al.* [4] and Nandi *et al.* [10] used nonuniform CAs with two rules 90 and 150, and it was found that the quality of generated PNSs was better that the quality of the Wolfram system. Recently Tomassini and Perrenoud [15] proposed to use nonuniform, 1D CAs with $r = 1$ and four rules 90, 105, 150 and 165, which provide high quality PNSs and a huge space of possible secret keys which is difficult for cryptanalysis. Instead to design rules for CAs they used evolutionary technique called cellular programming (CP) to search for them.

In this study we continue this line of research. We will use finite, 1D, nonuniform CAs. However, we extend the potential space of rules by considering two sizes of rule neighborhoods, namely neighborhood of radius $r = 1$ and $r = 2$. To discover appropriate rules in this huge space of rules we will use CP.

4 Cellular Programming Environment

4.1 Cellular Programming

CP [9] is an evolutionary computation technique similar to the diffusion model of parallel genetic algorithms and is introduced [14] to discover rules for nonuniform CAs. In contrast with the CP used in [15] the system has the possibility to evaluate nonuniform rules of two types. The system consists of a population of N rules (left) and each rule is assigned to a single cell of CAs (right). After initiating states of each cell, i.e. setting an initial configuration, the CAs start to evolve according to assigned rules during a predefined number of time steps. Each cell produces a stream of bits, creating this way a PNS.

After stopping CAs evolution, all PNSs are evaluated. The entropy E_h is used to evaluate the statistical quality of each PNS. To calculate a value of the entropy each PNS is divided into subsequences of a size h. In all experiments

the value $h = 4$ was used. Let k be the number of values which can take each element of a sequence (in our case of binary values of all elements $k = 2$) and k^h a number of possible states of of each sequence ($k^h = 16$). E_h can be calculated in the following way:

$$E_h = -\sum_{j=1}^{k^h} p_{h_j} log_2\ p_{h_j},$$

where p_{h_j} is a measured probability of occurrence of a sequence h_j in a PNS. The entropy achieves its maximal value $E_h = h$ when the probabilities of the k_h possible sequences of the length h are equal to $1/k^h$. It is worth to mention that the entropy is only one of possible statistical measures of PNSs. It will be used as a fitness function of CP. To decide about final statistical quality of PNSs and suitability of the discovered rules for cryptography purposes some additional tests must be conducted.

A single PNS is produced by a CA cell according to assigned rules and depends on a configuration c_i of states of CAs. To evaluate statistically reliable value of the entropy, CAs run with the same set of rules C times for different configurations c_i, and finally the average value of entropy is calculated and serves as a fitness function of each rule from the population of rules.

After evaluation of a fitness function of all rules of the population genetic operators of selection, crossover and mutation are locally performed on the rules. The evolutionary algorithm stops after some predefined number of generations.

4.2 Genetic Operators

In contrast with standard genetic algorithm population, rules - individuals of CP population occupy specific place in the population and have strictly defined neighborhood. We assume that rules are either of type 1 ($r = 1$, short rules) or of type 2 ($r = 2$, long rules).

Additionally to a neighborhood associated with two types of rules we introduce for rules an evolutionary neighborhood, i.e. the neighborhood of rules which are considered for mating when genetic operators are locally applied to a given rule. The size and pattern of this neighborhood may differ from the neighborhood associated with types of rules.

A sequence of genetic operators performed locally on a given rule depends on values of fitness function of rules from the evolutionary neighborhood of this rule. Genetic operators are applied in the following way:

1. if the $k - th$ rule is the best (the highest value of the fitness function) in its evolutionary neighborhood then the rule survives (selection) and remains unchanged for the next generation; no other genetic operators are performed
2. if in the evolutionary neighborhood of the rule k only one rule exists which is better than considered rule then the rule k is replaced by better rule (selection) only if both rules are of the same type, and next mutation on this rule is performed; the rule remains unchanged if the better rule is of the other type

3. if two rules that are better than the rule k exist in the neighborhood then a crossover on the pair of better rules is performed; a randomly selected child from a pair of children replaces rule k, and additionally the mutation operator is performed
4. if more than two rules better than the rule k exist in the neighborhood then two randomly selected better rules create (crossover) a pair of children; on a randomly selected child a mutation is performed, and the child replaces the rule k.

Two types of rules existing in a CP population can be considered as two species of a coevolutionary algorithm. Therefore to perform a crossover between rules special regulations are required. It is assumed that two parental rules of the same species create a single child rule of the same species, which can replace either the first type of a rule or the second type of the rule. If rules of different types take part in the mating then a species of a child depends on species of a replaced rule, and is the same as a species of a rule to be replaced.

The short rule $P1$ taking part in crossover consists of 8 genes ($n = 0, ..., 7$) which values correspond to values of transition function defined on 8 neighborhood states $\{000, 001, ..., 111\}$ existing for $r = 1$. The long rule $P2$ consists of 32 genes, each corresponding to values of transition function defined on 32 neighborhood states existing for $r = 2$. The long rule is folded because there is a strict relation between a state order number which corresponds to $j - th$ gene of $P1$ and states' order numbers corresponding to genes $2j, 2j+1$ and $2j+16, 2j+17$ of $P2$. These order numbers of states of $P2$ are just an extension of corresponding order number of a gene from $P1$. For example, the gene $n = 7$ of $P1$ corresponds to the neighborhood state $\{\mathbf{111}\}$, and genes 15, 14 and 31, 30 of $P2$ correspond to states respectively $\{0\mathbf{111}1, 0\mathbf{111}0\}$ and $\{1\mathbf{111}1, 1\mathbf{111}0\}$ containing the state of $P1$ (marked in bold).

Last genetic operator is a flip-bit mutation performed with the probability $p_m - 0.001$.

5 Discovery of Rules in 1D, Nonuniform CAs by Using CP

In all conducted experiments a population of CP and the size of nonuniform CAs were equal to 50 and the population was processed during 50 generations. The CAs using an initial random configuration of states and a set of assigned rules evolved during $M = 4096$ time steps. Running CAs with a given set of rules was repeated for $C = 300$ initial configurations.

A typical result of a single run of an evolutionary process starting with a random rules assigned to cells of CAs is discovering by CP a small set of good rules which divide the cellular space of CAs into domains - areas where the same rules live together. Evolutionary process is continued on borders of domains where different rules live. This process may result in increasing domains of rules which are only slightly better than neighboring rules, which domains will

decrease and finally disappear. This happens in particular when two neighboring domains are occupied respectively by the same short rules and the same long rules. The search space of short rules ($r = 1$) is much smaller than the search space of the long rules ($r = 2$). Therefore better short rules are discovered faster than better long rules, and for this reason long rules are gradually replaced by short rules. To limit this premature convergence of short rules, the short and long rules are initially randomly assigned to cells in the proportion of 1:3 in all subsequent experiments.

To find out what is the influence of a shape of the evolutionary neighborhood on the quality of PNSs generated by CAs, each experiment with a given shape of the neighborhood was repeated 10 times, and the average value of the entropy over each set of experiments was considered. The experiments have shown that while for each shape of the neighborhood very good rules with the entropy equal or close to 3,989 were observed, the average value of the entropy over sets of experiments ranged from 3,946 to 3,956 for neighborhoods 111 and 1__1__1, and from 3.960 to 3.975 for the remaining neighborhoods. For this reason only neighborhoods 11111, 1111111, and 11_1_11 were considered in next experiments.

The purpose of the experiments which followed was to discover an enlarged set of rules (to enlarge the key space of cryptography system) that working collectively would produce very high quality PNSs. It was noticed that in a single run of CP the evolutionary algorithm produces typically a set of four rules with a very high value of the entropy, but the quality of a rule depends on a neighborhood of the rule. As the result of experiments 8 short rules ($r = 1$) was selected: the rules 30, 90, 105, 150 and 165 discovered previously by [15] and additionally new rules 86, 101 and 153, and also 39 long rules ($r = 2$) were discovered.

6 Analysis and Comparison of Results

The entropy used as a fitness function for evolution of high quality CA rules is only one of existing statistical tests of PNSs. None of them is enough strong to claim statistical randomness of a PNS in the case of passing a given test. Passing by a PNS of n statistical tests increases certainty about degree of its randomness but there is not any guarantee that the PNS will not fail on the next test. For this reason discovered sets of rules need to be verified by additional number of statistical sets. Even passing all statistical tests does not exclude a possibility that the PNS is not suitable for cryptographic purposes. Before a PNS is accepted it should pass special cryptographic tests.

To check statistical quality of discovered rules and their cryptographic suitability some additional testing of rules has been performed. For this purpose uniform CAs consisting of 50 cells evolved during 65536 time steps with each single rule. Each of the 50 PNSs was divided into 4-bit words and tested on general statistical tests such as the entropy, chi-square test, serial correlation test [6], and on a number of statistical tests required by the FIPS 140-2 standard [11], such as monobit test, poker test, runs test, and long runs test. The best results

were achieved by rules 30, 86, 101, 153 ($r = 1$) and 8 long rules. Rules 90, 105, 150 and 65 [15] working separately in uniform CAs obtained good results in test of entropy and long runs test, quite good results in serial correlation test and monobit test but were weak in chi-square test, poker test and runs test. However this set of rules working together in nonuniform CAs achieves good results. For this reason only 10 rules were removed from discovered set of rules, which were worse than Tomassini & Perrenoud rules.

Rules which passed tests were next expressed to a set of Marsaglia tests [7] - a set of 23 very strong tests of randomness implemented in the Diehard program. Only 11 tests passed all 23 Marsaglia tests. These are short rules 30, 86, 101, and long rules 869020563, 1047380370, 1436194405, 1436965290, 1705400746, 1815843780, 2084275140 and 2592765285.

The purpose of the last set of experiments was a selection of a small set of short and long rules for nonuniform CAs to provide a generation of very high quality RNSs suitable for the secret key cryptography. Simple combinations of different rules which passed all Marsaglia tests in nonuniform CAs have shown that resulting PNSs may have worse statistical characteristic than PNSs obtained with use of rules in uniform CAs. On the other hand, experiments with Tomassini & Perrenoud rules show that rules working separately worse can provide better quality working collectively. For these reasons rules 153 and some long rules which obtained very good results in general tests but not passed all Marsaglia tests were also accepted for the set of rules to search a final set of rules.

In the result of combining rules into sets of rules and testing collective behavior of these sets working in nonuniform CAs the following set of rules has been selected: 86, 90, 101, 105, 150, 153, 165 ($r = 1$), and 1436194405 ($r = 2$). The proposed set of rules give similar results in terms of general and FIPS140-2 tests with the ones proposed by Tomassini & Perrenoud [12]. However, the main difference between these results can be observed at the level of Marsaglia tests: while the new discovered set of rules passes all 23 Marsaglia tests, the Tomassini & Perrenoud set of rules passes only 11 tests.

The secret key K which should be exchanged between two users of considered CA-based cryptosystem consists of a pair of randomly created vectors: the vector R_i informing about assigning 8 rules to N cells of CAs and the vector $C(0)$ describing an initial binary state of CA cells. The whole key space has therefore the size $8^N * 2^N$. The key space is much larger than the key space of 1D CA-based system [15] ($4^N * 2^N$) and and slightly greater than 2D CA-based system [16]. Therefore the proposed system is much more resistant for cryptographic attacks.

7 Conclusions

In the paper we have reported results of the study on applying CAs to the secret key cryptography. The purpose of the study was to discover a set of CA rules which produce PNSs of a very high statistical quality for a CA-based cryptosystem which is resistant on attempts of attacks. The main assumption of our approach was to consider nonuniform 1D CAs operating with two types of

rules. An evolutionary approach called CP was used to discover suitable rules. After discovery of a set of rules they were carefully selected using a number of strong statistical and cryptographic tests. Finally, the set consisting of 8 rules has been selected. Results of experiments have shown that discovered rules working collectively are able to produce PNSs of a very high quality outperforming the quality of known 1D CA-based secret key cryptosystems, which also are much more resistant for breaking cryptography keys that known systems.

References

1. P. Guan, Cellular Automaton Public-Key Cryptosystem, *Complex Systems* 1, 1987, pp. 51-56
2. H. Gutowitz, Cryptography with Dynamical Systems, in E. Goles and N. Boccara (Eds.) *Cellular Automata and Cooperative Phenomena*, Kluwer Academic Press, 1993
3. T. Habutsu, Y. Nishio, I. Sasae, and S. Mori, A Secret Key Cryptosystem by Iterating a Chaotic Map, *Proc. of Eurocrypt'91*, 1991, pp. 127-140
4. P. D. Hortensius, R. D. McLeod, and H. C. Card, Parallel random number generation for VLSI systems using cellular automata, *IEEE Trans. on Computers* 38, October 1989, pp. 1466-1473
5. J. Kari, Cryptosystems based on reversible cellular automata, personal communication, 1992
6. D. E. Knuth, *The Art of Computer Programming*, vol. 1 & 2, *Seminumerical Algorithms*, Addison-Wesley, 1981
7. G. Marsaglia, Diehard, *http://stat.fsu.edu/∼geo/diehard.html*, 1998
8. A. Menezes, P. van Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996
9. A. Mroczkowski, Application of Cellular Automata in Cryptography, Master Thesis (in Polish), Warsaw University of Technology, 2002
10. S. Nandi, B. K. Kar, and P. P. Chaudhuri, Theory and Applications of Cellular Automata in Cryptography, *IEEE Trans. on Computers*, v. 43, December 1994, pp. 1346-1357
11. National Institute of Standards and Technology, Federal Information Processing Standards Publication 140-2: *Security Requirements for Cryptographic Modules*, U.S. Government Printing Office, Washington 1999
12. F. Seredynski, P. Bouvry, A. Y. Zomaya, Cellular Programming and Symmetric Key Cryptography Systems, in E. Cantu-Paz et al. (Eds.), *Genetic and Evolutionary Computation-GECCO 2003*, LNCS 2724, Springer, pp. 1369-1381
13. B. Schneier, *Applied Cryptography*, Wiley, New York, 1996
14. M. Sipper and M. Tomassini, Generating parallel random number generators by cellular programming, *Int. Journal of Modern Physics C*, 7(2), 1996, pp. 181-190
15. M. Tomassini and M. Perrenoud, Stream Ciphers with One- and Two-Dimensional Cellular Automata, in M. Schoenauer at al. (Eds.) *Parallel Problem Solving from Nature - PPSN VI*, LNCS 1917, Springer, 2000, pp. 722-731
16. M. Tomassini and M. Sipper, On the Generation of High-Quality Random Numbers by Two-Dimensional Cellular Automata, *IEEE Trans. on Computers*, v. 49, No. 10, October 2000, pp. 1140-1151
17. S. Wolfram, Cryptography with Cellular Automata, in *Advances in Cryptology: Crypto '85 Proceedings*, LNCS 218, Springer, 1986, pp. 429-432