International Journal of Security and Its Applications
Vol. 10, No. 11 (2016), pp.109-118
http://dx.doi.org/10.14257/ijsia.2016.10.11.10

Design of New Pseudo-Random Number Generator Based on Non-Uniform Cellular Automata

Charifa Hanin[*1], Fouzia Omary[1], Souad Elbernoussi[2] and Bouchra Boulahiat[1]

[1]*Laboratory of computer science research*
[2]*Laboratory of Mathematics, Computing and Applications*
[1,2]*Mohammed V University in Rabat, Faculty of sciences BP1014 RP, Rabat, Morocco*
[*]*charifa.hanin@gmail.com, omary@fsr,ac,ma, s.elbernoussi@fsr.ac.ma, boulahiat.bouchra@gmail.com*

Abstract

The random numbers are needed in variety fields of applications in particular cryptography. They can be constructed by using several methods such as cellular Automata (CA). This later cannot produce a long random numbers sequence, supplementary the quality of random numbers depends especially on applied CA rules. For this reason, the modified binary particle swarm optimization (MBPSO) is used to discover a new rules set for CA. Rules of radius r=1 and periodic boundary conditions are considered for a non-uniform one-dimensional CA. Our contribution consists to combine this CA with MBPSO for satisfying the pseudo random number generator (PRNG) characteristics, called "PSOCA". Thus, different tests were applied to our PSOCA algorithm to prove its generated sequences quality, such as Diehard, Nist and other statistical tests, which have been successfully passed. Moreover, the comparison with other systems ensure the highlight randomness quality of our proposal system.

Keywords: *cryptography; pseudo-random number generator; cellular automata; binary particle swarm optimization; statistical tests.*

1. Introduction

A PRNG uses one or more inputs and generates multiple "pseudorandom" numbers. Inputs to PRNGs called seeds. In contexts in which unpredictability is needed, the seed itself must be random and unpredictable [1]. The PRNG, play an important role in various fields of applications, as well as Monte Carlo techniques [2], Brownian dynamics [3], stochastic optimization methods [4] and key based cryptography [5]. There is several methods for constructing the class of pseudo-random generator. These methods are able to generate sequences having the following three characteristics; Long Period, good statistical properties and large linear complexity. These three properties of PRNG determined by statistical analyses such as Diehard Test, Nist Test suites and other statistical tests. However, finding good generator is a difficult task. CA is a good candidate to generate random numbers. However, the CA's self-organization property limits random number [6] and the quality of random numbers depends especially on applied CA rules, to solve this problem metaheuristic algorithms are used especially for explore and exploit the search space. In this paper, MBPSO used to discover the relevant rules for non-uniform CA (i.e rules that used to generate pseudo-random number of high quality). The paper structured as follows. Section 2 covers the related works, the basic notions, CA terminologies, and the binary particle swarm optimization. Section 3 shows the PSOCA algorithm and the proposed MBPSO. Our test results showed in Section 4. Finally, in Section 5, the conclusion and future works will be presented.

[*]Corresponding Author (Email: charifa.hanin@gmail.com)

ISSN: 1738-9976 IJSIA
Copyright ⓒ 2016 SERSC

2. Background

In this section, we firstly give a brief presentation of CA concepts and CA based PRNG. Thereafter, we introduce BPSO principle.

2.1 One-dimensional Cellular Automata and Related Works

Cellular Automata is a dynamical system consists of an array of cells where, each of which takes a value in a finite number of states S, called the State set. Every cell can change its state according to a transition rule, based on the old states of its neighbor's cells including cell itself. Mathematically, A CA A od dimension d is defined by $A = (\mathbb{Z}^d, S, f, v)$ where, $\mathbb{Z}^d$ is the space of A, S is the finite set of states, called alphabet, and $f: S^n \to S$ is the transition rule; n is the number of neighbors where $v = \{(\mathrm{i} - \mathrm{r}), .., (\mathrm{i}), .., (\mathrm{i} + \mathrm{r})\} \in (\mathbb{Z}^d)^n$ r is the radius. For more particulars on CA can be found in [7,8].

To represent a rule of radius r for CA where $S = \{0,1\}$, a binary string of length L is used, where $L = 2^{2r+1}$, (see Table1).

In this paper, we used a non-uniform (the CA cells obey to different rule) one-dimensional CA with periodic boundary condition and $S = \{0,1\}$ as its alphabet and $v = \{(\mathrm{i} - \mathrm{r}), (\mathrm{i}), (\mathrm{i} + \mathrm{r})\}$ the neighbors of each cell.

Table 1. Representation of Rule 30 with r = 1

Neighborhood number	7	6	5	4	3	2	1	0
v_i^t	111	110	101	100	011	010	001	000
$f(v_i^t)$	0	0	0	1	1	1	1	0

In cryptography field, CA allows generating pseudo random sequences. Wolfram was the first to apply CA to generate pseudo random number generator [9]. He used a uniform one-dimensional CA to construct a stream cipher with r =1 and Rule 30. Hotensius et al. [10] and Nandi et al. [11] used non-uniform CA with two rules 90 and 150, and it was found that the quality of generated pseudo numbers sequence was better than the quality of the Wolfram system. Recently, in the previous works [12] and [6] new design with combination of rules and complex behaviors has introduced, several works has done for generating random numbers by using CA and evolutionary algorithms. For example, genetic algorithm, which were evoked in these, researches [13] [14] and [15] and particle swarm optimization (PSO) algorithm have been used in [16]. Our contribution used a MBPSO algorithm with a new parameter to update velocity value described below (section 3.1).

2.2 Binary Particle Swarm Optimization

The PSO is a good stochastic optimization method based on the movement and intelligence of swarms designed by Eberhart and Kennedy in 1995 [17]. It uses a number of particles that constitute a swarm moving around in the search space looking for the best solution. Each particle is defined on time t by its position X_{it} and its velocity V_{it}. Each particle i can communicate with a subset of particles called neighborhood. It was supposed that a d-dimensional is our search space; the i^{th} particle of the swarm can be represented by position vector $X_i = (\mathrm{x_{i1}}, \mathrm{x_{i2}}, \ldots, \mathrm{x_{id}})$. The velocity of the particle is denoted by $V_{it} = (\mathrm{v_{i1}}, \mathrm{v_{i2}}, \ldots, \mathrm{v_{id}})$. In addition, it considered personnel best position for the particle is $P_{ibest} = (\mathrm{p_{i1}}, \mathrm{p_{i2}}, \ldots, \mathrm{p_{id}})$ and the global best position is $P_{gbest} =$

 Copyright ⓒ 2016 SERSC

$(p_{g1}, p_{g2}, \ldots, p_{gd})$. So in classic PSO, the equations (eq. 1, eq. 2) used to update position of the particle and its velocity.

$$v_i^{t+1} = \omega v_i^t + c_1 r_1 (p_i - x_i^t) + c_2 r_2 (p_g - x_i^t) \quad (1)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (2)$$

Where c_1 and c_2 are two fixed variables which are determined by user, $r_1, r_2 \in [0,1]$, and ω is the inertia weight.

Kennedy and Eberhart introduced the BPSO algorithm to solve binary problems cases [18]. It uses the concept that a position takes on one or zero as a probability to modified the velocity value. In the BPSO, Eq.(1) for updating the velocity still unchanged, but Eq.(2) is re-defined by the equation (3) for updating the position

$$X_{ij}^{t+1} = \begin{cases} 0 \text{ if rand}(\quad) \geq S\left(v_{ij}(t+1)\right) \\ 1 \text{ if rand}(\quad) \leq S\left(v_{ij}(t+1)\right) \end{cases} \quad (3)$$

Where $S(\cdot)$ is the sigmoid function for transforming the velocity to the probability as the expression (eq. 4), and $rand()$ is the pseudo-random number selected from a uniform distribution in the range [0,1].

$$S(v_{ij}\ (t+1)) = \frac{1}{1+e^{-v_{ij}\ (t+1)}} \quad (4)$$

3. Proposed PRNG based on Combination of the MBPSO and CA

This section presents the MBPSO used in the PSOCA algorithm and we describe the details of the proposed PRNG.

3.1 Modified Binary Particle Swarm Optimization (MBPSO)

In our study, we are inspired by [19]. Our proposed MBPSO is described using the XoR neighbor of CA cell noted by P_v. for example: $\{i-1, i, i+1\} = \{0,1,1\}$ so in this case $P_{iv} = 0$.

To calculate $c_3 r_3 \left(P_{iv}^j \ominus x_i^j(t)\right)$expression, we are used the operator $\ominus$ described in equation (5). The equation (5) represent the result of $c_3 r_3 \left(P_{iv}^j \ominus x_i^j(t)\right)$ expression. The velocity is updated by equation (6) where c_3, and $r_3$is a random variable in the range [0,1].

$$a \ominus b = \begin{cases} 1 \text{ if } a = b \\ 0 \text{ if } a \neq b \end{cases} \quad (5)$$

The velocity is interpreted as a probability to change a bit from 0 to 1 or from 1 to 0.

$$v_i^{t+1} = \omega v_i^j(t) + c_1 r_1 \left(p_i^j - x_i^j(t)\right) + c_2 r_2 \left(p_g^j - x_i^j(t) +\right) + c_3 r_3 \left(P_{iv}^j \ominus x_i^j(t)\right) \quad (6)$$

Then, each particle move to a new position using the following equation:

$$X_i^{t+1} = \begin{cases} \overline{X}_i(t) \text{ if } r_i, < v_i \\ X_i(t) \text{if } r_i > v_i \end{cases}$$ Where $r \in [0,1]$ and $\bar{X}_i$is the complement of X_i

Copyright ⓒ 2016 SERSC

3.2 Description of the Proposed PRNG

In this paper, we present a new approach to generate a pseudo-random numbers based on CA and a MBPSO described in the previous section. In our generator (PSOCA) shown in "Algorithm1". A one-dimensional, non-uniform of n cells in CA, with a periodic boundary condition and (r=1), is combined with (MBPSO) algorithm. Such as a rule, represents a particle and the entropy h as the fitness. Let "k" be the number of CA states and "h" a subsequence length. h (Measured in bits) for the set of probabilities of the possible subsequences is assumed by:

$$E_h = -\sum_{j=0}^{k^h} P_{hj} \log_2(P_{hj}) \quad (7)$$

The entropy attains its maximal value $E_h = h$ when the probabilities of all possible subsequences of length h are equal to $\frac{1}{k^h}$.

In the proposed generator, one rule for each cell is selected randomly in CA. Each cell has exactly two fixed neighbors; the neighbors of the cell i are cell $i-1$ and $i+1$. Moreover, as shown in figure1, the neighbors of the cell1 are the cell2 and n^{th} cell.

To generate random numbers, we assign the initial values of CA cells randomly. We use a population of P_{max} rules. After, the initiation values of each particle, the CA starts to evolve according to each particle (rule) to find a new set of appropriate rules. We make the entropy (eq.(7)) of each CA as the fitness of the MBPSO (section 3.1), in order to select the basic principles of CA rules. Finally, we make the CA evolve M time steps in accordance with selected rules. After stopping running, CA all pseudo random numbers sequence obtained will be evaluated. If the conditions of tests are not satisfied, we applying again the selected rules on CA cells.

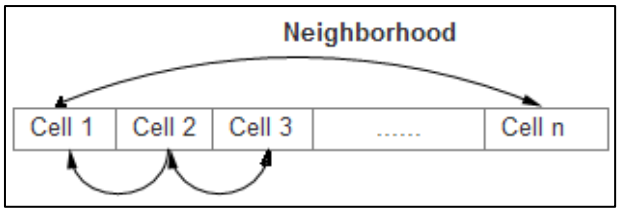

Figure 1. Selection of Cells Neighborhood

Algorithm 1: The pseudo code of the proposed algorithm

Data: CA of n cells
Result: pseudo random sequences

1. Initiate the population of N rules randomly with r =1;
2. Initiate the values of CA cells randomly;
3. Selecting rules by using MBPSO;
4. **while** not done **do**
   for each cell in CA **do**
   affect a rule randomly
   end
   Applying the selected rules on CA cells
5. **end**
6. **end**

 Copyright ⓒ 2016 SERSC

4. Experiments Results

In this section, we specify the parameters selection by using the MBPSO, and then we discuss the achieved rules with the best entropy results. We then present the results of Diehard and Nist test suite. The algorithm was implemented in java, All the runs were carried out on a laptop at Intel(R) Core(TM) i7 CPU processor with 2.70 GHz processor speed and 4.096 Go usable RAM.

4.1. MBPSO Parameters Selection

Before running the MBPSO, we need to specify some parameters. The most important parameters are: number of particles P_max, Iteration number $N\_\max$ and velocity parameters which are: maximum velocity $V\_\max$, Acceleration coefficients (c_1, c_2, c_3) and inertia weight (w). In our experiment, each parameter runs 10 times in order to study its influence on the solution quality given by the MBPSO. The value of c_1 and c_2 are fixed to two in this application.

The figure 2 shows the fitness changes with respect to the number of iterations. Horizontal axis shows the number of iterations, and vertical axis shows its best, average and worst fitness.

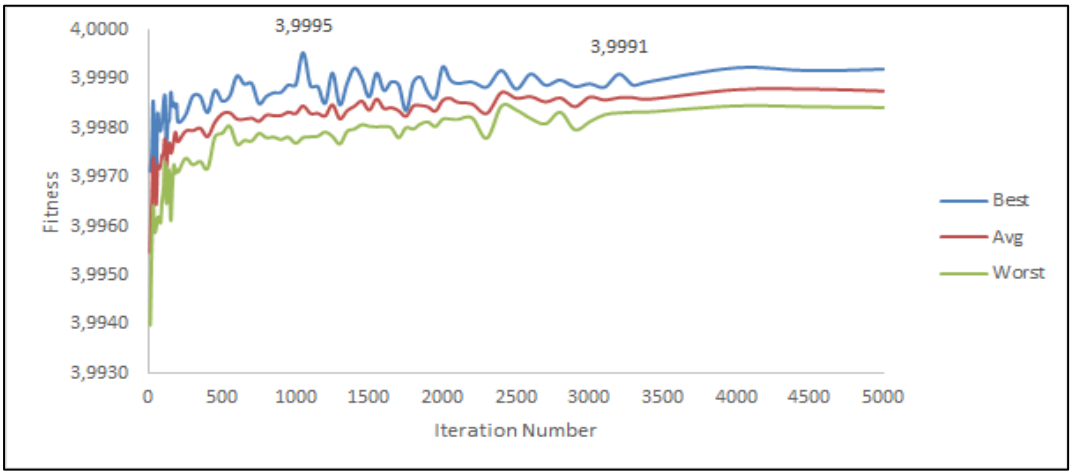

Figure 2. Selection of the Number Iteration

The figure 3 represents the variation of $E_{avg} - E$ in function of particles number, where E_{avg} and E are respectively the maximum of all average and average value. The table 2 introduce the suitable c_3 parameter.

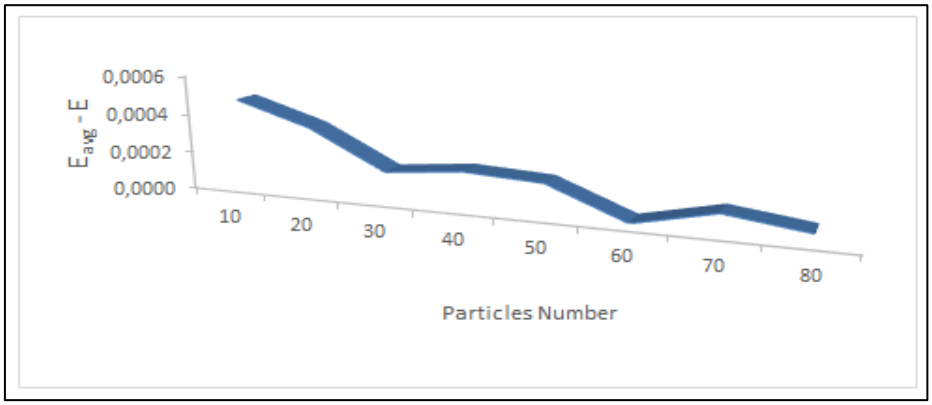

Figure 3. Selection of the Particle Number

Copyright ⓒ 2016 SERSC

C_3	0.0559	0.5352	0.7090	0.8832
Fitness value	3.9991	3.9995	3.9992	3.9994

Table 2. The C_3 Parameter

As shown in the figure 2, the fitness converge to 4 after the 4000^{th} iteration. The selection of the maximum particle that can be used which is Pmax=60. From the table 2 we can notice that the better value of c_3 parameter is0.5352.

4.2 Entropy Test and Rules Selection

We used a population of $P_{\max} = 60$(see fig (3)) particles and a uniform CA of n=4096 cells with 4000 iterations. the Table 3 represents a typical result of a one run of PSOCA algorithm beginning with a random rules assigned to cells of CA is discovering by MPSO a small set of good rules.

Table 3. Rule and Fitness with r = 1

Rule number	Rule name	Fitness value
10110100	45	3.99929
11101001	105	3.99912
10110010	89	3.99909
10100101	165	3.99920
10101100	86	3.99901
11010010	210	3.99922
10011010	154	3.99953
10110100	90	3.99922
11100001	225	3.99924
10101001	169	3.99910
11000011	195	3.99910
11001010	101	3.99917
10101010	170	3.99923
11110000	120	3.99916

4.3 Diehard and NIST Test Suite

4.3.1 Diehard Test Suite

In this section, we describe results of applying the PSOCA algorithm of random number generators by Diehard test suite [20]. It is a one of the standard random number test suites and has been widely used in the fields of testing random numbers recently. Table 4 shows the result of this test for the proposed generator. The evolution of CA by using the combination of rules found by MBPSO algorithm 1, taking into account all the diehard tests, we can conclude that these rules provides a good generator better than generator achieved by Q.Wang et all, also the bitstream, OPSO, OQSo and DNA are passed successfully.

Copyright ⓒ 2016 SERSC

Table 4. Diehard Tests

Test Number	Test Name	Number of P-values	Q.Wang
1	Birthday spacing	0.584	0.705
2	Overlapping 5-permutations	0.701	0.831
3	Binary rank 31 x 31 32 x 32	0.158	0.021
4	Binary rank 6 x 8	0.164	0.747
5	Bitstream	0.432	Fail
6	OPSO	0.896	Fail
7	OQSO	0.651	Fail
8	DNA	0.467	Fail
9	Count the ones - stream	0.208	0.612
10	Count the ones - specific	0.385	0.612
11	Parking lot	0.811	0.351
12	Minimum distance	0.461	0.395
13	3D sphere	0.729	0.773
14	Squeeze	0.542	0.473
15	Overlapping sum	0.957	0.351
16	Runs	0.957	0.942
17	Craps	0.345	0.556

4.3.2 NIST Test Suite

The NIST Statistical Test Suite is a set of procedures tests which try to identify sequences of binary numbers which do not act in a truly random manner. To make this, these tests provide a p-value for every sequence of bits. Each test passes if the p-value is greater than some fixed significance level. A full explanation of these testing algorithms can be found in a NIST Special Publication [1]. The statistical test suite also separates a given input sequence into several reduced subsequence and achieve the tests on each of these strings. We are used 100 random number sequence as an input to the NIST test suite with respecting the recommendations of NIST where the proportion of passing sequences is 0.01 for each test. All results are shown in table 5.

Table 5. NIST Tests

Test name	P-value
Frequency (Monobit)	0.702461
Block-Frequency	0.572546
Cumulative Sums	
Forward	0.237514
Reverse	0.476775
Runs	0.515598
Longest Run	0.068569
Rank	0.954371
FFT	0.073544
Overlapping Templates	0.783949
Non-overlapping Templates	0.739918
Universal	0.748171
Serial	
P-value1	0.142718

Copyright ⓒ 2016 SERSC

P-value2	0.060400
Linear Complexity	0.255157
Approximate Entropy (m = 10)	0.888320

5. Conclusion and Future Works

This paper presents a new pseudo-random number generator, using CA combined with a modified binary particle swarm optimization. Applied to a non-uniform one-dimensional CA with boundary conditions and two neighbors, its purpose is to discover a new set of rules. The experimental results show that our PSOCA generator, applied on generated sequences, is better than Q.Wang et al. It passed successfully Nist tests and the Diehard's Bitstream, OPSO, OQSO and DNA. Moreover, it is able to generate long sequence of random bits with uniform distribution. In conclusion, our PSOCA algorithm could be used to generate a secret key cryptography. Furthermore, it is easy to be implemented efficiently in hardware. In the future works, we will apply our PRNG especially in stream cipher system and in other cryptographic applications.

References

[1] A. Rukhin, J. Soto, J. Nechvatal, M. Smid, E. Barker, S. Leigh, M. Levenson, M. Vangel, D. Banks, A. Heckert, J. Dray and San Vo. "A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications", NIST Special Publication 800-22, **(2001).**
[2] J. Gentle, "Random Number Generation and Monte Carlo Methods", Springer, **(2003).**
[3] A. Reese, "Random Number Generators in Genetic Algorithms for Unconstrained and Constrained Optimization", Nonlinear Analysis: Theory, Methods and Applications. 71, **(2009),** pp. 679-692.
[4] P.L. Ecuyer, "Random Numbers for Simulation", Communications of the ACM 33, **(1990)**, pp. 85-97.
[5] M. Tomassini, M. Sipper, M. Perrenoud, "On the generation of high quality random numbers by two-dimensional cellular automata", IEEE Transactions on Computers 49 **(2000)** 1146–1151.
[6] H. Karimi, S. M. Hosseini and M. V. Jahan, "On the combination of self-organized systems to generate pseudo-random numbers", Information Sciences, **(2013).**
[7] S. Wolfram, "A New Kind of Science", Wolfram media Champaign, **(2002).**
[8] P. Sarkar, "A brief history of cellular automata", ACM Computing Surveys. vol. 32, no. 1, **(2000)**, pp. 80–107.
[9] S.Wolfram, "Cryptography with cellular automata", The Institute for Advanced Study, Princeton NJ 08540, **(1986).**
[10] P.D. Hortensius, R.D. McLeod and H.C. Card, "Parallel random number generation for VLSI systems using cellular automata", IEEE Transactions on Computers 38, **(1989).**
[11] S. Nandi, B. K. Kar, and P. Pal Chaudhuri,"Theory and applications of cellular automata in cryptography", IEE Transactions on Computers. vol.43, no.12, **(1994).**
[12] Seyed Morteza Hosseini, Hossein Karimi and Majid Vafaei Jahan, "Generating pseudo-random numbers by combining two systems with complex behaviors", Information Security and Applications, **(2014).**
[13] F. Seredynski, P. Bouvry and A. Zomaya, "Cellular automata computations and secret key cryptography", Parallel Computing Elsevier, **(2003).**
[14] M. Tomassini, M. Sipper, M. Zolla and M. Perrenoud, "Generating highquality random numbers in parallel by cellular automata", Future Generation Computer Systems 16 **(1999)**, pp. 291–305.
[15] Miroslaw Szaban, Franciszek Seredynski and Pascal Bouvry, "Evolving collective behavior of cellular automata for cryptography", IEEE proceeding of MELECON, **(2006)**.
[16] Q.Wang, S. Yu, W. Ding and M. Leng, "Generating high-quality random numbers by cellular automata with PSO", Fourth International Conference on Natural Computation, **(2008).**
[17] R. Eberhart and J. Kennedy, "A New Optimizer Using Particles Swarm Theory", Proc. Sixth International Symposium on Micro Machine and Human Science (Nagoya, Japan), IEEE8, **(1995)**
[18] Y. Shi and R.C. Eberhart, "Empirical study of particle swarm optimization", Proceedings of the 1999 Congress on Evolutionary Computation, **(1999)**
[19] M. A. Khanesar, Member, IEEE, M. Teshnehlab and M. A. Shoorehdeli, "A Novel Binary Particle Swarm Optimization", Proceedings of the 15th Mediterranean Conference on Control and Automation, (2007)
[20] G. Marsaglia, "DIEHARD Test", **(1998)**. http://stat.fsu.edu/pub/diehard/

 Copyright ⓒ 2016 SERSC

Authors

Charifa Hanin, She received her Master's degree in Codes, Cryptography and Information Security from Mohammed-V University in Rabat, Morocco. She is actually a PhD student in the Laboratory of computer science research (LRI) in Rabat-Morocco. Her major research interests include cryptography, information security and cellular automata.

Fouzia Omary, she is a full professor in the department of computer science at Mohammed V University in Rabat, Morocco, and a director of the Laboratory of computer science research (LRI). Her research interests include cryptology, information security and cellular automata.

Souad Elbernoussi, she is a full professor at the Mohammed-V University in Rabat, Morocco, and a member of the laboratory of Mathematics, Computing and Applications (LabMIA). His research interests include optimization problems, metaheuristics and complexity.

Bouchra Boulahiat, She received her Master's degree in in Applied Computer Science Offshoring from Mohammed-V University in Rabat, Morocco. She is actually a PhD student in the Laboratory of computer science research (LRI) in Rabat-Morocco. Her research interests include encryption systems and pseudo random numbers generator.

Copyright ⓒ 2016 SERSC

Copyright ⓒ 2016 SERSC