Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 986174, 4 pages
http://dx.doi.org/10.1155/2014/986174

Research Article

A Novel Pseudorandom Bit Generator Based on Chirikov Standard Map Filtered with Shrinking Rule

Borislav Stoyanov and Krasimir Kordov

Faculty of Mathematics and Informatics, Konstantin Preslavski University of Shumen, 9712 Shumen, Bulgaria

Correspondence should be addressed to Borislav Stoyanov; borislav.stoyanov@shu-bg.net

Received 12 February 2014; Accepted 3 June 2014; Published 30 June 2014

Academic Editor: Jonathan N. Blakely

This communication proposes a simplified model of pseudorandom bit generator, based on two Chirikov standard maps filtered with shrinking rule. The study also demonstrates that the generated keystreams have excellent properties of randomness and resistance to statistical attacks, which is proven by using the NIST, ENT, and DIEHARD testing suites.

1. Introduction

The chaotic maps and the shrinking rules have been used widely in the fields of random simulations and secure communications. Patidar and Sud [1] introduced a pseudorandom bit generator with good cryptographic properties by using two Chirikov standard maps [2] combined with a threshold function. Lian et al. [3] and Fu et al. [4] proposed standard map-based pseudorandom confusion processes, which they used in chaotic image encryption schemes. Ye and Huang [5] presented two shuffle image encryption schemes, based on standard map orbit ergodicity. Coppersmith et al. [6] used two linear feedback shift registers, named shrinking generator, to create a third source of pseudorandom bits, which has better quality than the initial sources. Stoyanov [7] proposed new chaotic cryptographic scheme constructed from the Lorenz butterfly attractor and filtered by 32-bit bent Boolean function.

The aim of the paper is referred on the method of synthesis of a pseudorandom bit generation scheme based on two standard maps which are filtered by Jabri shrinking generator (JSG) [8]. The proposed combiner is tested by NIST [9], DIEHARD [10], and ENT [11] batteries of tests.

2. The Proposed Pseudorandom Bit Generator

The Chirikov standard map is an area-conserving chaotic map defined by a set of difference equations:

$$p_{t+1} = p_t + K \sin(x_t),$$
$$x_{t+1} = p_{t+1} + x_t, \tag{1}$$

where the quantities p and x (momentum and coordinate) are taken modulo 2π. The stochasticity parameter K controls the degree of chaos. The nonlinearity of the map grows with large K.

Jabri pointed out that using the classic shrinking function leads to statistical disadvantage and proposed a modified shrinking rule, which addresses the problem. If g_1 and g_2 are two bit generators, the sequences from these generators are denoted by $\mathbf{b} = \{b_0, b_1, \ldots, \}$ and $\mathbf{s} = \{s_0, s_1, \ldots, \}$, respectively. An output sequence, $\mathbf{z} = \{z_0, z_1, \ldots, \}$, corresponding to the Jabri search-based output was then built from these sequences by using the following rule: $z_k = b_{ik}$ for $k = 0, 1, \ldots$, where ik is the kth position for which b_i and s_i are different. That is, the sequence $\mathbf{z}$ will include only those bits b_i of the sequence $\mathbf{b}$, which are different from $\mathbf{s}$, while the other bits are ignored.

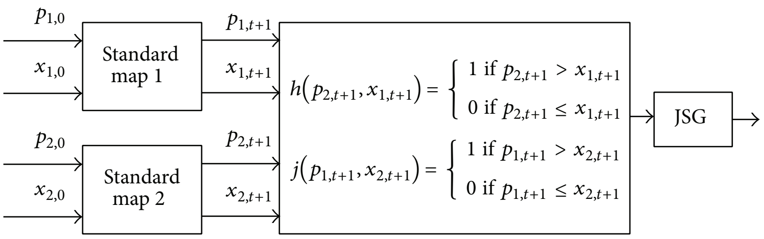

$p_{1,0}$
$x_{1,0}$
Standard map 1
$p_{1,t+1}$
$x_{1,t+1}$
$p_{2,0}$
$x_{2,0}$
Standard map 2
$p_{2,t+1}$
$x_{2,t+1}$
$h(p_{2,t+1}, x_{1,t+1}) = \begin{cases} 1 \text{ if } p_{2,t+1} > x_{1,t+1} \\ 0 \text{ if } p_{2,t+1} \leq x_{1,t+1} \end{cases}$
$j(p_{1,t+1}, x_{2,t+1}) = \begin{cases} 1 \text{ if } p_{1,t+1} > x_{2,t+1} \\ 0 \text{ if } p_{1,t+1} \leq x_{2,t+1} \end{cases}$
JSG

Figure 1: Schematic description of the proposed chaos based generator.

This study was inspired by the work of Patidar and Sud [1]. The original pseudorandom bit generator is based on the following two Chirikov standard maps:

$$\begin{aligned} p_{1,t+1} &= p_{1,t} + K_1 \sin(x_{1,t}), \\ x_{1,t+1} &= p_{1,t+1} + x_{1,t}, \\ p_{2,t+1} &= p_{2,t} + K_2 \sin(x_{2,t}), \\ x_{2,t+1} &= p_{2,t+1} + x_{2,t}, \end{aligned} \tag{2}$$

where the initial conditions $p_{1,t}$, $x_{1,t}$, $p_{2,t}$, and $x_{2,t}$ are taken modulo 2π. The maps are starting from six floating-value numbers: $(p_{1,0}, x_{1,0}, p_{2,0}, x_{2,0}) \in [0, 2\pi)$ and the control parameters K_1 and K_2 are real numbers greater than 18.9. The pseudorandom bits are generated by comparing two outputs of both maps in the following way:

$$h(p_{2,t+1}, x_{1,t+1}) = \begin{cases} 1 & \text{if } p_{2,t+1} > x_{1,t+1} \\ 0 & \text{if } p_{2,t+1} \leq x_{1,t+1}. \end{cases} \tag{3}$$

The keystream from the above scheme is produced by using two output values from the Chirikov standard maps. In order to use all computed values in the output stream calculation, we propose a novel pseudorandom bit generator by adding to the above generator a second threshold function:

$$j(p_{1,t+1}, x_{2,t+1}) = \begin{cases} 1 & \text{if } p_{1,t+1} > x_{2,t+1} \\ 0 & \text{if } p_{1,t+1} \leq x_{2,t+1}. \end{cases} \tag{4}$$

Then we shrink the constructed couple of bits from h and j with the Jabri shrinking rule. The schematic description of the proposed chaotic based generator is shown in Figure 1. The novel hybrid scheme is based on the combination of all four outputs of two Chirikov standard maps and it has the extra security features of the search-based rule.

3. Experimental Statistical Tests

The proposed pseudorandom bit generator is implemented softwarely in Dev-C++ 5.0 beta 9.2 (4.9.9.2) environments with Mingw/GCC 3.4.2. We produced a set of 1000 sequences of 1000000 bits each, using the following initial numbers: $p_{1,0} = 2.56$, $x_{1,0} = 3.05$, $p_{2,0} = 1.24$, $x_{2,0} = 3.27$, $K_1 = 1500$, and $K_2 = 2100.37$. In order to test the randomness of the novel scheme, we used the NIST, DIEHARD, and ENT statistical test packages.

The NIST suite [9, 12] includes 15 tests, which were developed to check the randomness of binary sequences produced by pseudorandom generators. These tests are as follows: frequency (monobit), block-frequency, cumulative sums (forward and reverse), runs, longest run of ones, rank, fast Fourier transform (spectral), nonoverlapping templates, overlapping templates, Maurers "universal statistical", approximate entropy, random excursion, random-excursion variant, serial, and linear complexity. The testing process consists of the following steps.

(1) State the null hypothesis. Assume that the zero/one sequence is random.

(2) Compute a sequence test statistic. Testing is carried out at the bit level.

(3) Compute the P value, P value $\in [0, 1]$.

(4) Fix α, where $\alpha \in [0.0001, 0.01]$. Compare the P value to α. Success is declared whenever P value $\geq \alpha$; otherwise, failure is declared.

The NIST suite calculates the proportion of sequences that pass the particular tests. The range of acceptable proportion is determined using the confidence interval defined as

$$\widehat{p} \pm 3\sqrt{\frac{\widehat{p}(1-\widehat{p})}{m}}, \tag{5}$$

where $\widehat{p} = 1 - \alpha$ and m is the number of binary tested sequences. NIST recommends that, for these tests, the user should have at least 1000 sequences of 1000000 bits each. In our setup $m = 1000$. Thus the confidence interval is

$$0.99 \pm 3\sqrt{\frac{0.99\,(0.01)}{1000}} = 0.99 \pm 0.0094392. \tag{6}$$

The proportion should lie above 0.9805607 with exception of random excursion and random excursion variant tests. These two tests only apply whenever the number of cycles in a sequence exceeds 500. Thus the sample size and minimum pass rate are dynamically reduced taking into account the tested sequences.

The distribution of P values is examined to ensure uniformity. The interval between 0 and 1 is divided into 10 subintervals. The P values that lie within each subinterval are counted. Uniformity may also be specified through an application of a χ^2 test and the determination of a P value

Table 1: NIST test results.

NIST	Proposed generator	
Statistical test	P value	Pass rate
Frequency (monobit)	0.228367	994/1000
Block frequency	0.186566	983/1000
Cumulative sums (forward)	0.759756	990/1000
Cumulative sums (reverse)	0.003224	991/1000
Runs	0.647530	993/1000
Longest run of ones	0.960198	990/1000
Rank	0.670396	992/1000
FFT	0.187581	988/1000
Nonoverlapping templates	0.482512	990/1000
Overlapping templates	0.166260	987/1000
Universal	0.281232	987/1000
Approximate entropy	0.903338	988/1000
Random-excursion	0.532463	590/599
Random-excursion variant	0.409049	591/599
Serial 1	0.066465	986/1000
Serial 2	0.442831	989/1000
Linear complexity	0.985788	984/1000

corresponding to the goodness-of-fit distributional test on the P values obtained for an arbitrary statistical test, P value of the P values. This is implemented by calculating

$$\chi^2 = \sum_{i=1}^{10} \frac{(F_i - s/10)^2}{s/10}, \tag{7}$$

where F_i is the number of P values in subinterval i and s is the sample size. A P value is computed such that P value$_T$ = IGAM $(9/2, \chi^2/2)$, where IGAMC is the complemented incomplete gamma statistical function. If P value$_T \geq 0.0001$, then the sequences can be considered to be uniformly distributed.

The empirical results we obtained are presented in Table 1. All the P values from all 1000 sequences are distributed uniformly and the pass rate is also in an acceptable range.

The minimum pass rate for the random excursion (variant) test is approximately 585 for a sample size of 599 binary sequences for the proposed pseudorandom algorithm.

The Marsaglias Diehard test package consists of 18 statistical tests: Birthday spacings, Overlapping 5-permutations, Binary rank (31 × 31), Binary rank (32 × 32), Binary rank (6 × 8), Bitstream, Overlapping-Pairs-Sparse-Occupancy, Overlapping-Quadruples-Sparse-Occupancy, DNA, Stream count-the-ones, Byte-count-the-ones, Parking lot, Minimum distance, 3D spheres, Squeeze, Overlapping sums, Runs (up and down), and Craps. The tests return P values, which should be uniform in $[0, 1)$, if the input file contains truly independent pseudorandom bits. The P values are obtained by $p = F(y)$, where F is the assumed distribution of the sample random variable y, often the normal distribution.

We will introduce the particular tests briefly [10, 13]: *Birthday spacings* chooses m random points (birthdays) in a year of n days. The spacings between the points should be asymptotically Poisson distributed. *Overlapping 5-permutations* looks at a sequence of one million 32-bit random integers where the 120 possible permutations of 5 consecutive random numbers occur with equal statistical probability. *Three Binary rank tests*, *(31 × 31)*, *(32 × 32)*, and *(6 × 8)* form a binary matrix and determines the rank of the matrix. *Bitstream* counts the number of missing 20-bit words in a string of 2^{21} overlapping 20-bit words. *OPSO*, *OQSO*, and *DNA* analyse overlapping 2-letter, 4-letter, and 10-letter words. The words which do not appear in the entire sequence should be very close to normally distributed. *Stream and Byte count-the-ones* uses the probabilities of the number of ones to determine different 4-letter and 5-letter words. *Parking lot* is an empty 100 by 100 matrix which is randomly filled with elements (cars). The number of successful attempts without crash with one already parked is very closely normally distributed. *Minimum distance* chooses 8,000 random points in a square of side 10,000. Measures the squared distance between random points. The square distance should be very close to exponentially distributed. *3D spheres* chooses 4,000 random points in a cube of side 1,000. Eachpoint centers a sphere large enough to reach the next closest point. The volume of the smallest such sphere should be exponentially distributed. *Squeeze* is where the test finds the number of iterations necessary to reduce the number $m = 2^{31}$ to 1, using the reduction $m = \lceil m * U \rceil$, where the function $\lceil x \rceil$ gives the smallest integer $\geq x$ and U is provided by floating integers from the input file. *Overlapping sums* forms sequences of overlapping sums of uniform variables. *Runs* counts runs up and runs down in a sequence of uniform $[0, 1)$ variables. *Craps* plays 200,000 games of craps. The number of wins should be a normally distributed.

Table 2: Diehard test results.

DIEHARD	Proposed generator
Statistical test	P value
Birthday spacings	0.576866
Overlapping 5-permutation	0.191766
Binary rank (31 × 31)	0.393875
Binary rank (32 × 32)	0.326959
Binary rank (6 × 8)	0.532371
Bitstream	0.489218
OPSO	0.462404
OQSO	0.462404
DNA	0.559898
Stream count-the-ones	0.521853
Byte count-the-ones	0.596708
Parking lot	0.861929
Minimum distance	0.765773
3D spheres	0.383131
Squeeze	0.496864
Overlapping sums	0.008502
Runs up	0.289339
Runs down	0.449145
Craps	0.497628

Table 3: ENT test results.

ENT	Proposed generator
Statistical test	Results
Entropy	7.997502 bits per byte
Optimum compression	OC would reduce the size of this 125000000 byte file by 0%
χ^2 distribution	For 125000000 samples is 438239.72 and randomly would exceed this value less than 0.01% of the times
Arithmetic mean value	127.5013
Monte Carlo π estim.	3.140569010 (error 0.03%)
Serial correl. coeff.	−0.000147 (totally uncorrelated = 0.0)

Table 2 shows results obtained from testing a single 80 million bits file used for experimental purposes. It is evident that all Diehard tests pass for our novel pseudorandom bit generator. The output streams did not exhibit a noticeable deviation from randomness.

The ENT suite performs 6 tests to sequences of bytes stored in files and outputs the results of those tests. We tested output stream of 125000000 bytes of the proposed scheme. The results are summarized in Table 3 and show that the novel pseudorandom binary generator passed all the tests of ENT.

4. Conclusions

In summary, we propose a novel chaos-based pseudorandom bit generator, which uses two Chirikov standard maps filtered by a search-based rule. We did detailed analysis by NIST, Diehard, and ENT statistical packages to show that the novel generator did not reveal a noticeable deviation from randomness.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors thank anonymous reviewers for their valuable comments and suggestions. This paper is supported by the Project BG051PO 00l-3.3.06-0003 "Building and steady development of PhD students, post-PhD and young scientists in the areas of the natural, technical, and mathematical sciences." The project is realized by the financial support of the Operative Program "Development of the human resources" of the European social fund of the European Union.

References

[1] V. R. Patidar and K. K. Sud, "A novel pseudo random bit generator based on chaotic standard map and its testing," *Electronic Journal of Theoretical Physics*, vol. 6, no. 20, pp. 327–344, 2009.

[2] B. V. Chirikov, "A universal instability of many-dimensional oscillator systems," *Physics Reports C*, vol. 52, no. 5, pp. 264–379, 1979.

[3] S. Lian, J. Sun, and Z. Wang, "A block cipher based on a suitable use of the chaotic standard map," *Chaos, Solitons and Fractals*, vol. 26, no. 1, pp. 117–129, 2005.

[4] C. Fu, J. Chen, H. Zou, W. Meng, Y. Zhan, and Y. Yu, "A chaos-based digital image encryption scheme with an improved diffusion strategy," *Optics Express*, vol. 20, no. 3, pp. 2363–2378, 2012.

[5] R. Ye and H. Huang, "Application of the chaotic ergodicity of standard map in image encryption and watermarking," *International Journal of Image, Graphics and Signal Processing*, vol. 2, no. 1, pp. 19–29, 2010.

[6] D. Coppersmith, H. Krawczyk, and Y. Mansour, "The shrinking generator," in *Advances in Cryptology-CRYPTO '93*, vol. 773 of *Lecture Notes in Computer Science*, pp. 22–39, Springer, Berlin, Germany, 1994.

[7] B. P. Stoyanov, "Chaotic cryptographic scheme and its randomness evaluation," *AIP Conference Proceedings*, vol. 1487, pp. 397–404, 2012.

[8] A. K. A. Jabri, "Shrinking generators and statistical leakage," *Computers and Mathematics with Applications*, vol. 32, no. 4, pp. 33–39, 1996.

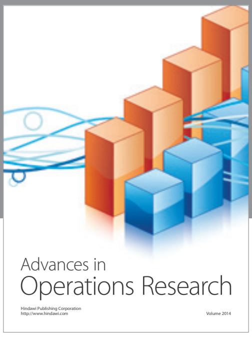
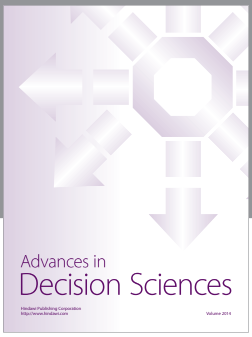
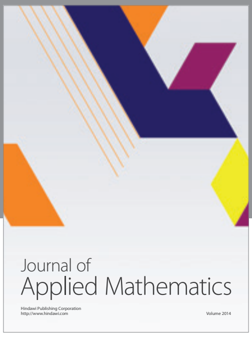
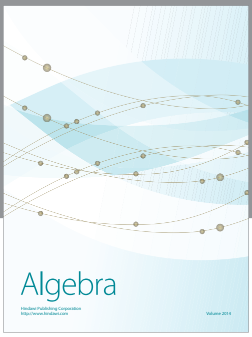
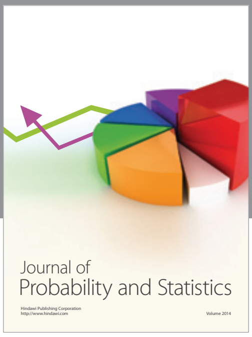
[9] A. Rukhin, J. Soto, J. Nechvatal et al., A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Application, NIST Special Publication 800-22, Revision 1a (Revised: April 2010), Lawrence E. Bassham III, 2010, http://csrc.nist.gov/groups/ST/toolkit/rng/index.html.

[10] G. Marsaglia, DIEHARD: a battery of tests of randomness, http://www.fsu.edu/pub/diehard.

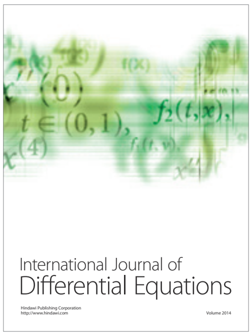
[11] J. Walker, "*ENT: A Pseudorandom Number Sequence Test Program*," http://www.fourmilab.ch/random/.

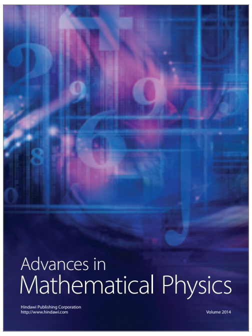
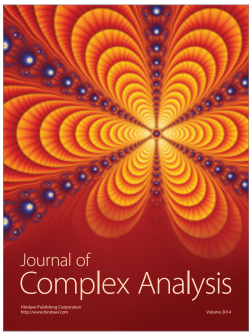
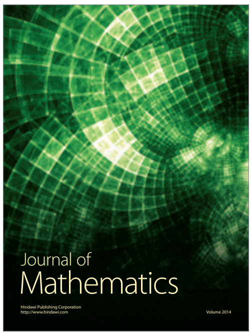
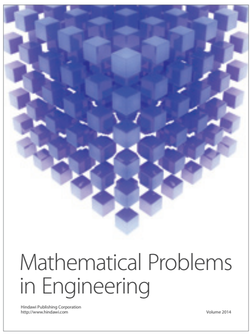
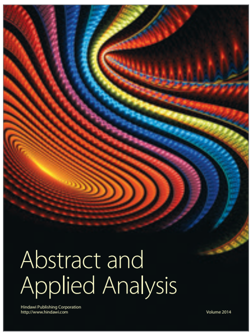
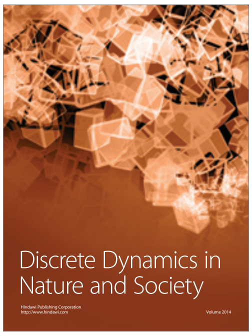
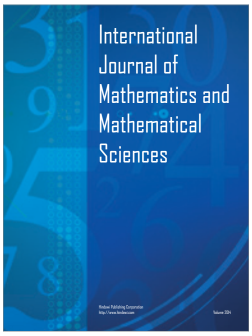
[12] J. Soto, "Randomness testing of the advanced encryption standard candidate algorithms," NIST Internal Reports 6390, 1999, http://csrc.nist.gov/publications/nistir/ir6390.pdf.

[13] W. Rotz, E. Falk, D. Wood, and J. Mulrow, "A comparison of random number generators used in business," in *Proceedings of the Annual Meeting of the American Statistical Association*, pp. 1–6, 2001.

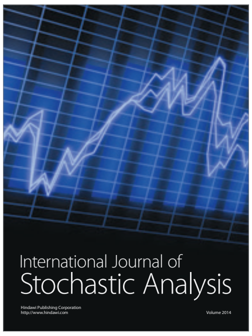

Advances in
Operations Research

Advances in
Decision Sciences

Journal of
Applied Mathematics

Algebra

Journal of
Probability and Statistics

The Scientific
World Journal


International Journal of
Differential Equations

International Journal of
Combinatorics

Advances in
Mathematical Physics

Journal of
Complex Analysis

Journal of
Mathematics

Mathematical Problems
in Engineering

Abstract and
Applied Analysis

Discrete Dynamics in
Nature and Society

International
Journal of
Mathematics and
Mathematical
Sciences

Journal of
Discrete Mathematics

Journal of
Function Spaces

International Journal of
Stochastic Analysis

Journal of
Optimization