International Journal of Applied Mathematics

Volume 34 No. 4 2021, 705-719
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)
doi: http://dx.doi.org/10.12732/ijam.v34i4.9

TEXT ENCRYPTION ALGORITHM FOR SECURE COMMUNICATION

Krasimir Kordov

Department of Computer Informatics
Faculty of Mathematics and Computer Science
Konstantin Preslavsky University of Shumen
115 Universitetska str., 9700 Shumen, BULGARIA

Abstract: In this paper, a new encryption algorithm is presented, designed for secure text message communication. The proposed cryptographic system is based on pseudorandom generator, constructed with two chaotic maps. For security level determination, extensive cryptographic analysis is performed. The evaluation of the presented cryptographic scheme includes the results of statistical testing, key space analysis, frequency analysis, common correlation analysis, entropy analysis, key sensitivity analysis and speed performance. The results are presented and explained in this paper.

AMS Subject Classification: 68P25, 94A60, 68M25
Key Words: text encryption; cryptography; pseudorandom generator; cryptographic analysis

1. Introduction

Cryptography is used since ancient times for secure communication. The main purpose of cryptography is to modify messages into encrypted form, so only the sender and the receiver of the information to be able to read the messages with the right decryption process.

Nowadays communication process worldwide is realized mostly by using computer systems (personal computers, laptops, tablets, smartphones etc.). This means the transferred information is digital and the cryptographic algo-

Received: February 21, 2021

rithms need to be implemented for digital information encryption and decryption.

Other important aspect of the modern communications is that most of the transferred information is text in the form of short messages, emails, documents etc. That is the reason the paper to be focused on encryption of digital text information for secure communication.

Text encryption algorithms are presented in [1, 2, 3] and [4], where the authors are using chaotic maps for message cryptography. That approach provides good results in the presented cryptographic analysis because of the chaotic behavior of the used maps.

In [5] and [6], the authors are combining two chaotic maps for constructing cryptographic system for text encryption. The advantage in this approach is the extended secret key, which increases the level of security of the proposed method.

For a new encryption algorithm, we decided to use approach with combination of two chaotic maps to determine if the model is successful and secure enough for text communication. The evaluation is performed by extensive cryptographic analysis, which is described in this paper.

2. Pseudorandom Generator Model

The Pseudorandom Generators (PRG) are used for constructing cryptographic systems. The main difference with true random generators (TRG) is that PRG are software realized and have significantly less cost for implementation. The examples for PRG are presented in [7, 8, 9] and [10]. The examples demonstrate that using chaotic maps is a successful model in constructing pseudorandom generators.

2.1. Kaplan-Yorke Map

In [11] James L. Kaplan and James A. Yorke examine the chaotic behavior of the following equation:

$$\begin{aligned} x_{i+1} &= ax_i \bmod 1, \\ y_{i+1} &= by_i + \cos(4\pi x_i), \end{aligned} \tag{1}$$

where the parameters are bounded for $0 \leq a \leq 2$ and $0 \leq b \leq 1$.

Eq. 1 is used as part of the proposed pseudorandom generator with initial values $x_0 = 0.275968043$, $y_0 = 0.356324678$ and control parameters $a = 1.25$ and $b = 0.82$.

Fig. 1 is a graphical representation of the chaotic map with described variables and parameters.

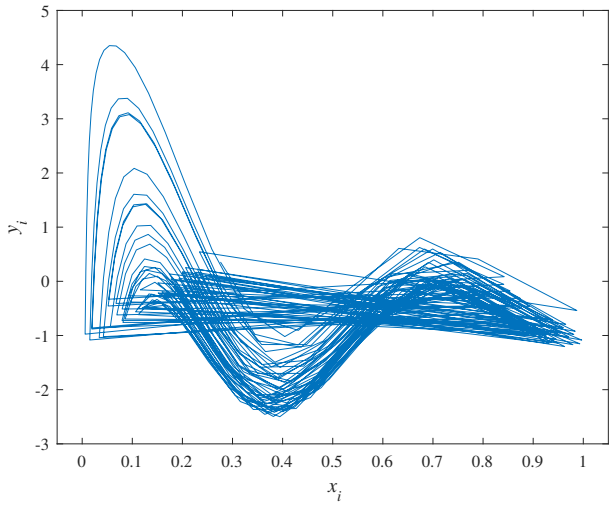


Figure 1: Kaplan-Yorke Map Plot using x_i and y_i.

2.2. Circle map

The Circle map is another chaotic map often used in cryptography [8, 10]. The equation for standard circle map iterations is:

$$\theta_{i+1} = (\theta_i + \Omega - \frac{K}{2\pi}\sin(2\pi\theta_i)) \mod 1, \tag{2}$$

where Ω and K are constant parameters with values

$$\Omega = 0.7128281828459045, \;\; K = 0.5$$

and the initial value for $\theta_0 = -0.254321234$.

The values are chosen by considering the randomness testing results in the previous research for constructing PRGs based on a circle map [8].

Fig. 2 is a graphical representation of the circle map with described variables and parameters.

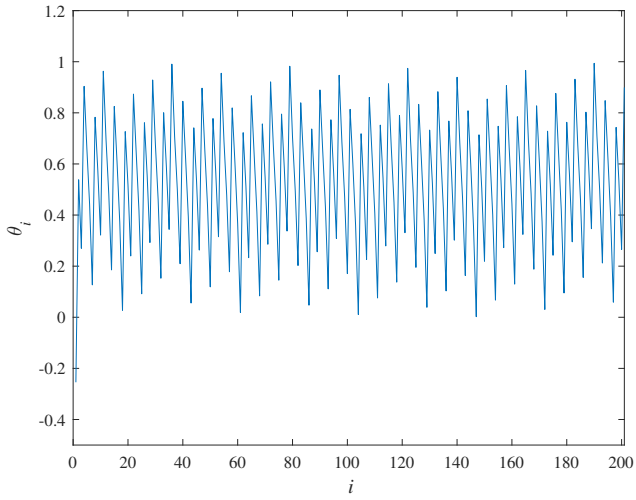


Figure 2: Circle Map Plot using θ_i and i.

2.3. Bit generation scheme

By using Eq. (1) and Eq. (2) is constructed a random bit generation scheme by performing the following steps:

- The initial values of the variables and the parameters from Eq. (1) and Eq. (2) are determined (described in the previous sections).
- The first N iterations from Eq. (1) and the first M iterations from Eq. (2) are skipped for additional security(in our case we have randomly chosen $N = M = 752$).
- In every iteration x_i and y_i from Eq. (1) are post-processed as follows:

$$\begin{aligned} p_i &= |integer(x_n \times 10^9)| mod\ 2, \\ q_i &= |integer(y_n \times 10^9)| mod\ 2 \end{aligned} \tag{3}$$

  and θ_n from Eq. (2) is post-processed as follows:

$$s_i = |integer(\theta_i \times 10^9)| mod\ 2. \tag{4}$$

- The result bit $rbit_i$ is extracted using p_i, q_i and s_i, obtained from the

previous step, by following the next rule:

$$rbit_i = \begin{cases} q_i, p_i \oplus s_i = 1, \\ !q_i, p_i \oplus s_i = 0. \end{cases} \tag{5}$$

- The previous step is repeated until the necessary random binary sequence is reached.

2.4. Statistical tests

The proposed pseudorandom generator is for randomness with the most popular statistical packages - NIST [12], DIEHARD [13] and ENT [14]. Evaluation is performed by producing 1 billion bits and the binary sequence is tested for randomness.

The first statistical test is NIST Statistical Test Suite. In Table 1 are the results of all 17 tests. The pass rate for every test is > 980 from 1000 bit subsequences (for Random-excursions and Random-excursions Variant > 613 from 627) with P-value in the range $[0, 1)$. The results indicate that all of the NIST tests are passed.

The second statistical test for randomness evaluation is DIEHARD statistical software with 19 tests. The P-value for successful passing of every test needs to be again in the range $[0, 1)$. Table 2 represents the results from all the tests and the obtained P-values which indicates that all the tests are passed.

The third statistical software ENT has another 6 different randomness tests that are presented and described in Table 3.

Statistical Test	**P-value**	**Pass rate**
Frequency (monobit)	0,355364	994/1000
Block-frequency	0,131122	988/1000
Cumulative sums (Forward)	0,962688	992/1000
Cumulative sums (Reverse)	0,896345	992/1000
Runs	0,043650	988/1000
Longest run of Ones	0,046870	994/1000
Rank	0,127393	990/1000
FFT	0,236810	983/1000
Non-overlapping templates	0,516877	990/1000
Overlapping templates	0,012829	991/1000
Universal	0,055714	990/1000
Approximate entropy	0,560545	983/1000
Random-excursions	0,409237	619/627
Random-excursions Variant	0,517565	620/627
Serial 1	0,729870	989/1000
Serial 2	0,935716	992/1000
Linear complexity	0,110734	994/1000

Table 1: NIST - statistical testing

The obtained results in Table 1, Table 2 and Table 3 prove that the proposed PRG can be used for random bit extraction for secure encryption process.

2.5. Key-space analysis

The key-space calculates all the possible secret keys that can be used in the cryptographic algorithms. The secret key defines the initial conditions of the proposed PRG. The values that define the secret key are the values of the variables from Eq. 1 and Eq. 2 - x_0, y_0 and θ_0. These three double variables define a key space about $(10^{15})^3 \approx 2^{149}$, considering IEEE floating-point standard [15]. The key space can be increased also with the $(2^{32})^2$ if the integer variables N and M are considered. In cryptographic systems, secret keys larger than 2^{100} are considered strong enough to resist the brute-force attacks.

Statistical Test	**P-value**
Birthday spacings	0,5367850
Overlapping 5-permutation	0,0510920
Binary rank (31 x 31)	0,6517030
Binary rank (32 x 32)	0,3286780
Binary rank (6 x 8)	0,7572360
Bitstream	0,5398200
OPSO	0,4168304
OQSO	0,5621179
DNA	0,4948742
Stream count-the-ones	0,5336660
Byte count-the-ones	0,5242471
Parking lot	0,5853325
Minimum distance	0,0659740
3D spheres	0,2703250
Squeeze	0,3765960
Overlapping sums	0,8715050
Runs up	0,4185100
Runs down	0,3857160
Craps	0,3678520

Table 2: DIEHARD - statistical testing

Statistical Test	**Result**
Entropy	7.999998 bits per byte
Optimum compression	OC would reduce the size of this 125000000 byte file by 0 %.
χ^2 distribution	For 125000000 samples is 260.95, and randomly would exceed this value 38.55 % of the times.
Arithmetic mean value	127.5072 (127.5 = random)
Monte Carlo π estimation	3.141576626 (error 0.00 %)
Serial correlation coefficient	0.000032 (totally uncorrelated = 0.0)

Table 3: ENT - statistical testing

3. Text Encryption/Decryption Scheme

The proposed symmetric text encryption algorithm performs the same steps for encryption and decryption process [16]. The mandatory condition is to be used

exactly the same secret key for encryption and decryption. In order to correctly process the printable and non-printable characters in digital text information, every character is being processed as binary sequence and the ASCII equivalent of the characters is used for the encryption and the decryption result. The basis of the proposed algorithm is PRG described in the previous section:

1. The text information is transformed to vector V of the binary values of the characters.
2. The proposed Pseudorandom generator is initialized with the initial conditions and N empty rotations are performed. This forms the secret key.
3. From vector V character is extracted and modified with XOR operation with 8 bits produced from PRG and the result is recorded in temporary vector T.
4. Previous step is repeated until the end of text information is reached.
5. Vector T is transformed back to text information into vector V'.
6. The result vector V' contains the encrypted/decrypted text information.

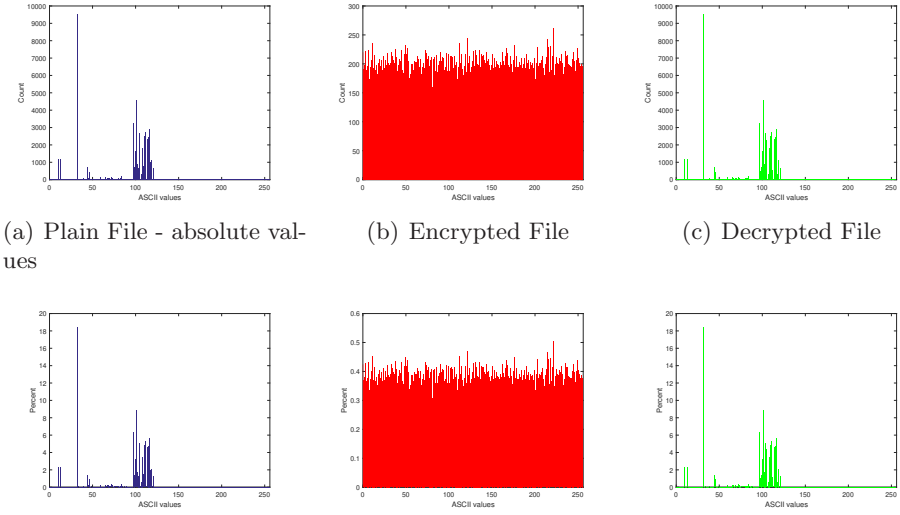
4. Cryptographic Analysis

Evaluating of the security of the proposed encryption scheme is obligated with additional empirical test to determine the level of security. The obtained result of the experiments are presented in this section.

4.1. Frequency

This test measures the characters' distribution in plain and encrypted texts. The experiment is performed by analyzing part of a sample book in text format with about 51K characters and it is compared with the corresponding encrypted and decrypted equivalents. Fig. 3 represents the characters distribution in count values and percents.

The performed analysis shows high frequency of some characters in the plain (Fig. 3(a) and Fig. 3(d)) and the decrypted file (Fig. 3(c) and Fig. 3(f)), which is normal for text information in books. The most used characters are the letters, numbers, punctuation marks and symbols "space" and "newline". However the encrypted file (Fig. 3(b) and Fig. 3(e)) has uniform distribution of all ASCII values, which is an indication of strong encryption process.



(a) Plain File - absolute values (b) Encrypted File (c) Decrypted File

(d) Plain File - percentage distribution (e) Encrypted File (f) Decrypted File

Figure 3: Character distribution analysis

4.2. Common correlation coefficient analysis

Correlation coefficient calculation is used to determine the dependency of groups of values. This test is designed to evaluate if there is a connection between the corresponding characters in plain and encrypted text messages. Values close to $|1|$ indicate strong correlation and strong dependency between the values. Correlation coefficient can be calculated as follows:

$$r_{xy} = \frac{cov(x,y)}{\sqrt{D(x)}\sqrt{D(y)}}, \tag{6}$$

where

$$D(x) = \frac{1}{N}\sum_{i=1}^{N}(x_i - \overline{x})^2,$$

$$D(y) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \overline{y})^2,$$

$$cov(x,y) = \sum_{i=1}^{N}(x_i - \overline{x})(y_i - \overline{y}),$$

x_i and y_i are the corresponding character ASCII values of the plain and the encrypted files, $\overline{x}$ and $\overline{y}$ are the mean values, N is the number of characters, and $cov(x, y)$ is covariance between both files. In Table 4 are presented the results of the common correlation coefficient analysis.

Plain File	**Encrypted File**	**Common correlation coefficient**
File1.txt	File1e.txt	-0,000259242827317
File2.txt	File2e.txt	0,009491194018969
File3.txt	File3e.txt	-0,000824196008672
File4.txt	File4e.txt	0,000362147144416
File5.txt	File5e.txt	0,001747262045057
File6.txt	File6e.txt	0,006581226654306
File7.txt	File7e.txt	0,003600204771712
File8.txt	File8e.txt	-0,000326466581749
File9.txt	File9e.txt	0,001300967561625
File10.txt	File10e.txt	-0,001016000321816

Table 4: Common correlation coefficient analysis

The results in Table 4 show that all the correlation coefficients are close to 0, indicating no dependency between the plain and encrypted text files. This means that even knowing the ASCII values of the characters from one of the files, they cannot lead to the values of the other files.

4.3. Entropy analysis

The entropy calculation is an often used cryptographic analysis. Applied to text (transformed to ASCII values) the information entropy measures the probability of a certain value' appearance in the text. The entropy is calculated as follows:

$$H(X) = -\sum_{i=0}^{N} p(x_i) log_2 p(x_i), \tag{7}$$

where for the variable X - $p(x_i)$ is function of the probability of x to have a certain value - x_i. All possible ASCII values are in the range $[0, 255]$. The best possible chaotic behavior is considered if the entropy is $H(X) = 8$. For this test 10 digital books are encrypted and the results are presented in Table 5.

The results in Table 5 indicate that in normal text there is a certain order (all entropy values are about 4.5), but in the encrypted text files we have chaotic

Plain File	**Entropy**	**Encrypted File**	**Entropy**
File1.txt	4.63372214878846	File1e.txt	7.99980650721820
File2.txt	4.50687474579827	File2e.txt	7.99659638576170
File3.txt	4.43204266208225	File3e.txt	7.99977777925915
File4.txt	4.62653721079964	File4e.txt	7.99983471077134
File5.txt	4.59139408370702	File5e.txt	7.99979622689444
File6.txt	4.47766468358831	File6e.txt	7.99792887384925
File7.txt	4.43945053407018	File7e.txt	7.99860517050420
File8.txt	4.48606832125213	File8e.txt	7.99910549127124
File9.txt	4.50698780137107	File9e.txt	7.99956017534163
File10.txt	4.55903703577167	File10e.txt	7.99981687414274

Table 5: Entropy analysis

behavior because all the entropy values are very close to 8. This indicates strong encryption process of the proposed encryption algorithm.

4.4. Key-sensitivity analysis

The secure encryption algorithms are very sensitive to changes in the used secret key. To evaluate the key sensitivity we performed an experiment by changing the initial values of the proposed PRG. In Eq. 1 and Eq. 2 the described initial values are forming Secret Key 1(K1), x_0 is slightly changed to 0.275968042 for K2 and to 0.275968044 for K3. From Eq. 2 θ_0 is changed to -0.254321233 for K4 and to -0.254321235 for K5. Test 1 is shown on Fig. 4 where the Plain message (Konstantin Preslavsky University of Shumen) is encrypted with secret keys K1-K5. Test 2 is presented in Fig. 5 where the Cipher message (encrypted with K1) is decrypted with K1-K5.

Both Test-1 and Test-2 show that the proposed algorithm is very sensitive to changes in the Secret key. Fig. 4 shows that using similar, but different keys leads to different encryption and Fig. 5 shows that an encrypted message can only be decrypted with the right key. Using very similar secret keys different from the original one, cannot restore the plain message. Both tests demonstrate high sensitivity to the Secret key.

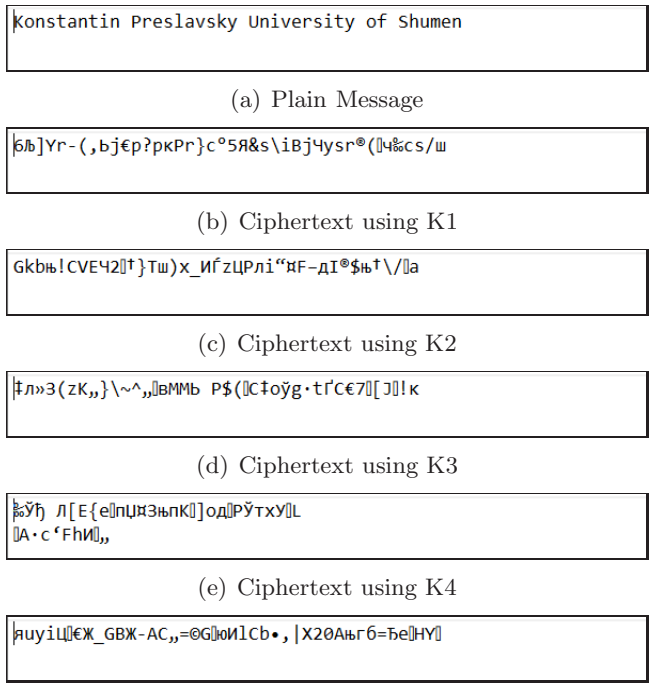


Figure 4: Key-sensitivity - Test 1

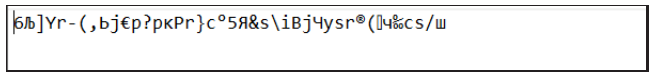
4.5. Speed analysis

Estimating the encryption/decryption time we tested 10 files (books) with a different size and number of letters with a middle-class computer system - Laptop 2.40 GHz Intel® Core™ i7-3630QM Dell Inspiron, 8 GB RAM, Windows 10. The obtained results are presented in Table 6.

The results in Table 6 demonstrate fast encryption time of the proposed algorithm (about 1,6 MB/sec.). Possible faster encryption could be achieved with parallel processing for multiple large text files [17, 18].

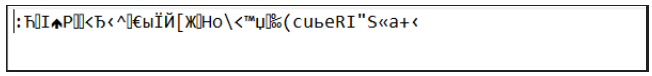
5. Conclusion

This paper presents a new secret text message encryption algorithm. The proposed algorithm is based on two chaotic maps. The performed statistical tests

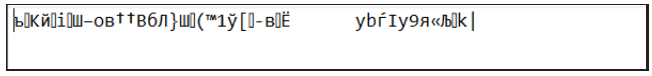


(a) Ciphertext - encrypted with K1

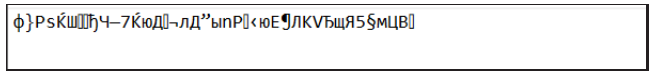
(b) Decrypted message with K1



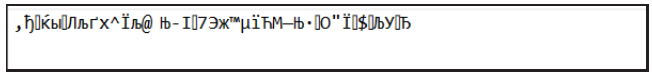
(c) Decrypted message with K2



(d) Decrypted message with K3



(e) Decrypted message with K4



(f) Decrypted message with K5

Figure 5: Key-sensitivity - Test 2

File Name	File Size	File Length (chars)	Encryption Time
File1.txt	907 KB	929 103	0h. 0m. 0.663s.
File2.txt	50,6 KB	51 833	0h. 0m. 0.038s.
File3.txt	676 KB	692 585	0h. 0m. 0.507s.
File4.txt	1,05 MB	1 104 033	0h. 0m. 0.826s.
File5.txt	914 KB	936 250	0h. 0m. 0.690s.
File6.txt	96,2 KB	98 595	0h. 0m. 0.075s.
File7.txt	108 KB	111 391	0h. 0m. 0.080s.
File8.txt	190 KB	194 858	0h. 0m. 0.141s.
File9.txt	412 KB	421 894	0h. 0m. 0.322s.
File10.txt	968 KB	991 502	0h. 0m. 0.732s.

Table 6: Speed test

and cryptographic analysis show high level of security. The results from the frequency analysis, common correlation analysis, entropy analysis demonstrate strong encryption properties. The key sensitivity analysis shows that the encryption model is highly sensitive to changes in the secret keys. The speed tests demonstrate that the encryption algorithm has fast performance even if a middle class computer system is used.

Acknowledgments

This research is supported by the European Regional Development Fund and the Operational Program "Science and Education for Smart Growth" under Contract UNITe No. BG05M2OP001-1.001-0004-C01 (2018–2023)".

References

[1] C.K. Volos, I.M. Kyprianidis, I.N. Stouboulos, Text encryption scheme realized with a chaotic pseudo-random bit generator, *Journal of Engineering Science & Technology Review*, **6**, No 4 (2013).

[2] M.A. Murillo-Escobar, F. Abundiz-Prez, C. Cruz-Hernndez, R.M. Lpez-Gutirrez, A novel symmetric text encryption algorithm based on logistic map, In: *Proc. of the International Conference on Communications, Signal Processing and Computers*, **4953** (2014).

[3] M. Mishra, V.H. Mankar, Text encryption algorithms based on pseudo random number generator, *International Journal of Computer Applications*, **111**, No 2 (2015).

[4] P.K. Shukla, A. Khare, M.A. Rizvi, S. Stalin, S. Kumar, Tapplied cryptography using chaos function for fast digital logic-based systems in ubiquitous computing, *Entropy*, **17**, No 3 (2015), 13871410.

[5] S.J. Sheela, K.V. Suresh, D. Tandur, Secured text communication using chaotic maps, In: *2017 International Conference on Algorithms, Methodology, Models and Applications in Emerging Technologies (ICAMMAET)*, **IEEE** (2017), 16.

[6] M.Z. Abdullah, Z.J. Khaleefah, Design and implement of a hybrid cryptography textual system, In: *2017 International Conference on Engineering and Technology (ICET)*, **IEEE** (2017), 16.

[7] K.M. Kordov, Modified chebyshev map based pseudo-random bit generator, *AIP Conference Proceedings*, **1629**, No 1 (2014), 432436.

[8] K. Kordov, Modified pseudo-random bit generation scheme based on two circle maps and xor function, *Applied Mathematical Sciences*, **9**, No 3 (2015), 129135.

[9] K. Kordov, Signature attractor based pseudorandom generation algorithm, *Advanced Studies in Theoretical Physics*, **9**, No 6 (2015), 287293.

[10] K. Kordov, A novel audio encryption algorithm with permutation-substitution architecture, *Electronics*, **8** No 5 (2019), # 530.

[11] J.L. Kaplan, J.A. Yorke, Chaotic behavior of multidimensional difference equations, In: *Functional Differential Equations and Approximation of Fixed Points*, Springer, Berlin-Heidelberg (1979), 204227.

[12] A. Rukhin, J. Soto, et al., A statistical test suite for random and pseudorandom number generators for cryptographic application, *NIST Special Publication 800-22*, NIST, Gaithersburg (2001).

[13] G. Marsaglia, The marsaglia random number cdrom including the diehard battery of tests of randomness, `http://www.stat.fsu.edu/pub/diehard/` (2008).

[14] J. Walker, Ent: a pseudorandom number sequence test program, `http://www.fourmilab.ch/random/` (2008).

[15] IEEE Computer Society, 754-2008 - IEEE standard for floating-point arithmetic, *Revision of ANSI/IEEE Std 754-1985*, (2008), 170.

[16] K. Kordov, S. Zhelezov, Steganography in color images with random order of pixel selection and encrypted text message embedding. *PeerJ Computer Science*, **7** (2021), # e380.

[17] B. Stoyanov, K. Kordov, Pseudorandom bit generator with parallel implementation. In: *International Conference on Large-Scale Scientific Computing*, Springer, Berlin-Heidelberg (2013), 557564.

[18] S. Zhelezov, H. Paraskevov, Possibilities for steganographic parallel processing with a cluster system,*Contemporary Engineering Sciences*, **8**, No 20 (2015), 809816.