Journal scientific and applied research, vol. 5, 2014
Association Scientific and Applied Research
International Journal

Original Contribution ISSN 1314-6289

RANDOMNESS TESTING OF SEQUENCES PRODUCED BY P-ARY GENERALIZED SELF-SHRINKING GENERATOR USING APPROXIMATE ENTROPY

Zhaneta Tasheva*, Antoniya Tasheva**

** FACULTY OF TECHNICAL SCIENCES, KONSTANTIN PRESLAVSKY UNIVERSITY OF SHUMEN; FACULTY OF ARTILLERY, AAD AND CIS, NATIONAL MILITARY UNIVERSITY, BULGARIA, E-MAIL: zh.tasheva@mail.bg*
*** FACULTY OF COMPUTER SYSTEMS AND CONTROL, TECHNICAL UNIVERSITY OF SOFIA, BULGARIA,*
E-MAIL: atasheva@tu-sofia.bg

ABSTRACT: *In this paper a supervene testing for sequences generated by p-ary Generalized Self-Shrinking Generator (pGSSG) is made. Thanks to the Approximate Entropy (ApEn) approach certain randomness properties are proved to be possessed by them. In order to test the applicability of the generator the pGSSG sequence minimum length in excess of which it can be considered that the sequence behaves as truly random is detected.*

KEY WORDS: *PRNG, pGSSG, Approximate Entropy, Encryption, Stream Cipher, Security.*

1. Introduction

Nowadays, Pseudo-Random Sequences (PRSs) are widely used in such applications as computer simulation and modeling, statistics, experimental design and cryptography.

A pseudorandom bit generator is a deterministic method to produce a large set of random locking bits, called pseudorandom sequence, from a small set of bits, called seed. Pseudorandom sequences are widely used in modern communication and information systems because of their characteristics such as easier generation in comparison with truly random generators which use physical sources; easy reproducibility due to the deterministic nature of the algorithm and because they determine the security of widely used in cryptography symmetric cryptosystems.

Variety algorithms have been proposed to produce pseudorandom sequences [1], [2]. [3], [11], [14], [16], [17]. The main property that must hold

in order to pseudo-random sequence is applicable in cryptography is its unpredictability. This defines the following paradox [2]. If a deterministic function is unpredictable, it is difficult to prove anything about it, including its unpredictability. Some useful principles to construct deterministic functions with pseudorandom behavior are expansiveness, nonlinearity and computational complexity.

The mostly used element in pseudorandom bit generator is Linear Feedback Shift Registers (LFSRs), because they can generate *m*-sequences. To generate nonlinear sequences researchers utilize structures based on LFSR registers, like filter generators, combinatorial generators and clock controlled generators.

Recently some clock controlled generators which use a *p*-ary PRS instead of binary PRS have been proposed [11], [15]. They generalize the work of Shrinking Generator [1]. Another similar generator that summarizes the work of Self-Shrinking Generator (SSG) [3] is a *p*-ary Generalized Self-Shrinking Generator (pGSSG) [14]. It is built from only a single *p*-ary LFSR and it is proven that it has long period, balance property [13] and good statistical characteristics. Moreover it is resistant against exhaustive search and entropy attacks [12].

In this paper, first we use Approximate Entropy (ApEn) approach to prove that sequences generated by *p*-ary Generalized Self-Shrinking Generator possess certain randomness properties. Then, we also use ApEn to detect the minimum length in pGSSG sequence excess of which can be considered that the sequence behaves as truly random.

2. Related Work

In this section we make brief review of algorithm of *p*-ary Generalized Self-Shrinking Generator and Approximate Entropy approach for randomness testing. Finally, we describe the ApEn Test.

2.1. *p*-ary Generalized Self-Shrinking Generator

The pGSSG generator is proposed in 2011 as a generalization of Meier's Self-Shrinking Generator. Its idea is to implement a simple, fast, and at the same time secure way to encrypt stream data. Main difference of the two generators consist of bringing in a generalization and use of Extended Galois Field $GF(p^n)$.

The schema of pGSSG is given in Figure 1. As seen it consists of a single LSFR register *A*, whose length will be denoted by *L*. It generates sequence $(a_i)_{i\geq 0}$ with *p*-ary digits (i.e. $(a_i)_{i\geq 0}, 0 \leq a_i \leq p-1$) and $0 \leq i \leq L-1$. The multipliers of the feedbacks are given by coefficients $q_1, q_2, ..., q_L$, $q_L \in [0, 1, ..., p-1]$ of the primitive polynomial in GF(p^L). Each element can remember one *p*-ary number. The register is initialized by *p*-ary sequence $(a_0, a_1, \ldots, a_{L-1})$.

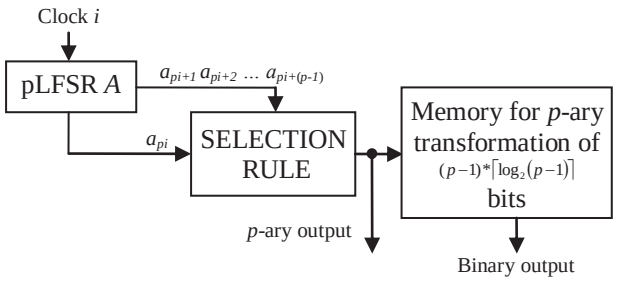

Figure 1. *p*-ary Generalized Self-Shrinking Generator

The pGSSG selects a portion of the *p*-ary output LFSR sequence by controlling the *p*-ary LFSR itself using six-step algorithm. The *p*-ary LFSR is clocked with period *T* and its output sequence is split into *p*-tuples. If the first element in the current *p*-tuple is nonzero the element with number equal to the value of that first element is output. In other case the *p*-tuple is discarded.

The *p*-ary output sequence could be transformed into a binary one using a special scheme for substitution for the zero and non-zero elements.

2.2. Approximate Entropy

First, in 1991 Steven Pincus [4] suggests approximate entropy (ApEn) as a measure of system complexity from at least 1000 given data values for both deterministic chaotic and stochastic processes. Then, in 1996 Pincus and Singer [6] propose approximate entropy as a characteristic to measure a degree of randomness of the tested sequence. Defined in such a way ApEn measures the logarithmic frequency with which blocks of length *m* that are similar remain also similar for blocks of length *m* + 1. And thus, small values of ApEn imply strong regularity and alternatively, large values of ApEn indicate irregularity (randomness) in tested sequences. After that, in 1997 Pincus and Kalman [5] propose using ApEn to quantify the extent to which given sequences differ from maximal irregularity.

In 2000, Andrew Rukhin [7] shows that ApEn and his modified version converges in distribution to χ^2-random variable in both cases when the length of blocks *m* is fixed and when *m* increases to infinity. These facts are the basis of statistical tests for randomness through approximate entropy. This idea is embedded as part of the empirical tests of NIST statistical tests for randomness has been applied to a study of the various random number generators such as Data Encryption Algorithm, Advanced Encryption Standard Finalist Candidates, Secure Hash Algorithm, Digital Signature Algorithm and many others [9], [10].

2.3. Approximate Entropy Test

Approximate Entropy Test [7], [8] checks the frequency of occurrence of all possible overlapping *m*–bit patterns in the pseudo-random generator output sequence. The purpose of the test is to compare the occurrence frequency of overlapping blocks with two consecutive lengths (m and $m + 1$) with the theoretical results for the real random sequence. Test [8] verifies the specific zero hypothesis H_0: „The sequence to be tested is random“ as it calculates the statistical *P-value*. To conduct the test the following parameters are needed: pattern length in bits – m; bit count (length) of generated pseudorandom sequence – n and the level of significance $\alpha \in [0.001, 0.01]$, which defines the acceptable level of error in the test.

To properly perform the approximate entropy test it is necessary to choose a value m that satisfies the following requirement:

$$m < \lfloor \log_2 n \rfloor - 2, \tag{1}$$

where $\lfloor x \rfloor$ is an integer greater than or equal to the real x.

Let $\varepsilon = (\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n)$ is the sequence of bits generated by the pseudorandom generator. The Approximate Entropy Test consists of the following steps:

1. An enlarged sequence ε' is formed: the sequence ε is enlarged by adding the first $m - 1$ bits of ε in the end of the n bit sequence. The result is the new sequence $\varepsilon' = (\varepsilon_1, \ldots, \varepsilon_n, \varepsilon_1, \ldots, \varepsilon_{m-1})$.

2. All the overlapping *m*–bit blocks are counted. If we denote the count of all overlapping blocks with value i and length m bits with N_i^m, therefore 2^m different values of N_i^m can be found, $i = 0, \ldots, 2^m - 1$.

3. The relative frequency of occurrence $C_i^m = \frac{N_i^m}{n}$ of all possible overlapping *m*-bit blocks is determined for $i = 0, \ldots, 2^m - 1$.

4. The Entropy of the empirical distribution is calculated:

$$\varphi^{(m)} = \sum_{i=0}^{2^m-1} C_i^m \log_2 C_i^m . \tag{2}$$

5. Steps 1 ÷ 4 are repeated for $m + 1$ instead of m.

6. The summary test statistics are calculated:

$$\chi^2 = n\left[\ln 2 - H^*(m)\right], \tag{3}$$

where

$$H^*(m) = \varphi^{(m)} - \varphi^{(m+1)} \tag{4}$$

is the Approximate Entropy of order m [6] and $H^*(0) = -\varphi^{(1)}$.

7. Calculate

$$P\text{- value} = Q(2^{m-1}, \chi^2), \tag{5}$$

where Q is the incomplete gamma function $Q(a, x)$ [8].

8. Evaluation of the results of the Approximate Entropy test: If the calculated *P-value* is less than the chosen level of significance α, than the test sequence is not random. Otherwise, the sequence generated by a pseudorandom generator can be considered as truly random.

3. The Experiments

In this section we describe some experiments carried out to test randomness of *p*-ary GSSG sequences via Approximate Entropy Test and to find the minimum length of the pGSSG sequence over which it can be considered as truly random.

3.1. Randomness Testing via Approximate Entropy

The study is conducted with Galois Field GF(257^{32}) due to the ease of byte representation and therefore the possibility of faster software implementation of the pGSSG (p = 257). Some primitive feedback polynomials used for construction of the pLFSR register (Figure 1) with prime p = 257 and length L = 32 are shown in Table 1.

As a result 300 different *P–values* have been calculated. The chosen length of the pattern is m = 10 and the level of significance is $\alpha = 0.01$. That indicates that one from 100 sequences could be rejected.

Table 1. Feedback polynomials in pGSSG.

№	Feedback Polynomial
1	$x^{32} + x + 10$
2	$x^{32} + 75\,x^2 + 174\,x + 33$
3	$x^{32} + 188\,x^2 + 200\,x + 107$

In order to determine how much empirical results coincide with the theoretical ones the distribution of the $P-values$ evenly is tested for uniformity using χ^2 criteria

$$\chi^2 = \sum_{i=1}^{10} \frac{(F_i - s/10)^2}{s/10}, \qquad (6)$$

where F_i is the count of the *P-values* in the subinterval i and s is the size of the interval. The built histograms show that all $P-values$ are distributed equally into the 10 subintervals (see Figure 2).

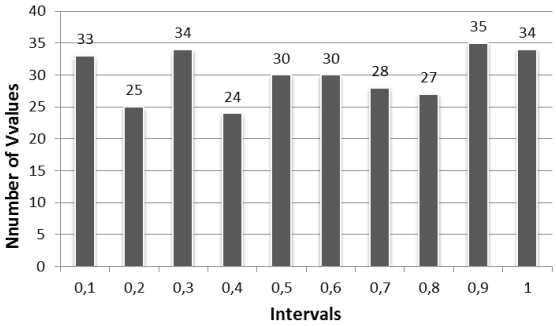

Figure 2. Histogram of *P*-values by Approximate Entropy Test

The calculated generalized *P-value* is

$$P\text{-}value = Q\left(9/2, \chi^2/2\right) =$$
$$= Q(4.5, 4.66667) = 0.5308.$$

Because the *P-value* ≥ 0.0001 [9], the sequence generated by pGSSG can be considered ***truly random with confidence level of 99%***.

3.2. Length vs. True Randomness

For finding the minimum length of the pGSSG sequence over which it can be considered as truly random the fact [5], [6], [7] that in one long random sequence with fixed length of the blocks m the Approximate Entropy $H^*(m)$ will converge to ln(2) = 0.693147 is used.

Parts of the results for 3 sequences of all 300 tests are shown in Table 2. The deviation O of the Approximate Entropy H^* of a pGSSG sequence from the truly random sequence

$$O = \ln(2) - H^*(m), \qquad (7)$$

is calculated. The results are similar for all generated sequences. Therefore the dependence of the deviation O when changing the length of the generated sequence n is shown on Figure 3. As can be seen from the figure, when the length of the sequence is $n \geq 5.10^5$ bits the deviation is almost constant and less than 1.10^{-3}. That fact shows that even shorter pGSSG sequences (less than the recommended 10^6 bits [8]) have close to the truly random Entropy characteristics and that can determine the pGSSG generator as random.

4. Conclusion and future work

The randomness of sequences produced by p-ary Generalized Self-Shrinking Generator is tested via approximate entropy. It is proven that

Table 2. Approximate Entropy $H^*(10)$ and deviation O from the truly random one.

Length n	Sequence 1		Sequence 2		Sequence 3	
	$H^*(10)$	O	$H^*(10)$	O	$H^*(10)$	O
40000	0,680055	0,013092	0,680886	0,012261	0,679772	0,013375
80000	0,687088	0,006059	0,68672	0,006427	0,686721	0,006426
120000	0,689034	0,004113	0,688915	0,004232	0,688937	0,00421
160000	0,690028	0,003119	0,689924	0,003223	0,690143	0,003004
200000	0,69058	0,002567	0,690754	0,002393	0,690607	0,00254
240000	0,690976	0,002171	0,691059	0,002088	0,691067	0,00208
280000	0,691305	0,001842	0,691439	0,001708	0,691233	0,001914
320000	0,691493	0,001654	0,691695	0,001452	0,69139	0,001757
360000	0,691715	0,001432	0,691764	0,001383	0,691611	0,001536
400000	0,69185	0,001297	0,691838	0,001309	0,691926	0,001221
440000	0,691969	0,001178	0,691941	0,001206	0,692003	0,001144
480000	0,69209	0,001057	0,692063	0,001084	0,692117	0,00103
520000	0,69217	0,000977	0,692128	0,001019	0,692172	0,000975
560000	0,692268	0,000879	0,692176	0,000971	0,692238	0,000909
600000	0,692319	0,000828	0,692282	0,000865	0,692259	0,000888
640000	0,692375	0,000772	0,692335	0,000812	0,692325	0,000822
680000	0,692411	0,000736	0,692378	0,000769	0,692358	0,000789
720000	0,692467	0,00068	0,692432	0,000715	0,692404	0,000743
760000	0,692513	0,000634	0,692467	0,00068	0,692417	0,00073
800000	0,69254	0,000607	0,692522	0,000625	0,692487	0,00066
840000	0,692567	0,00058	0,69253	0,000617	0,692527	0,00062
880000	0,692592	0,000555	0,692564	0,000583	0,692561	0,000586
920000	0,692618	0,000529	0,692582	0,000565	0,692616	0,000531
960000	0,692639	0,000508	0,692608	0,000539	0,692636	0,000511
1000000	0,692662	0,000485	0,692644	0,000503	0,692654	0,000493

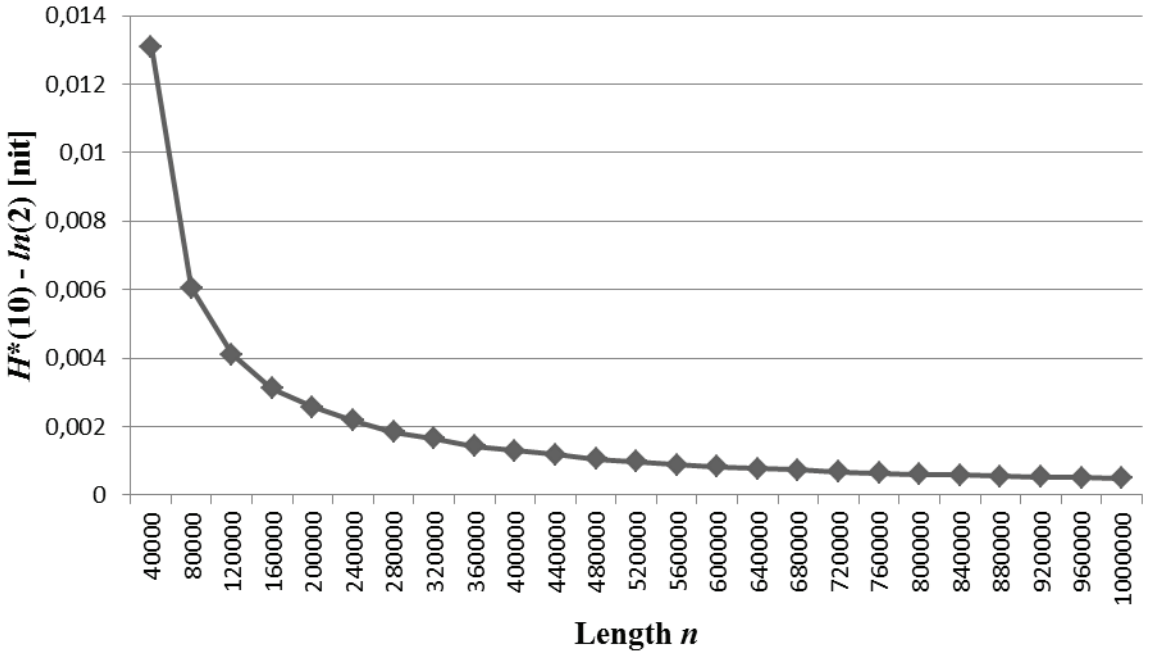

Figure 3. Approximate Entropy Deviation from the truly random one by the length of the sequence generated by pGSSG.

the pGSSG sequences can be considered as truly random with confidence level of 99%. Also, it is shown that the sequences with length more than 5.10^5 bits differ from maximal irregularity (randomness) with less than 1.10^{-3}, i.e. they have truly random properties.

However, even though this tests show satisfactory results some additional practical issues that need to be addressed. The min-entropy, which determines the probability of guessing the correct key value at first attempt, should be used in order to analyze the problem of finding the secret key (seed) of pGSSG. Furthermore, another entropy type – the guessing entropy may help us find the average number of needed guesses to determine the key, which is a task for your future work.

Acknowledgements

This paper is supported by the Project BG051PO00l-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 found of the European Union.

References:

[1] Coppersmith D., H. Krawczyk, Y. Mansour (1993). The shrinking generator, *Advances in Cryptology – EUROCRYPT'93*, vol. 773 of LNCS, Berlin, Springer-Verlag, 22-39.

[2] Lagarias, J. C. (1993). Pseudorandom numbers. *Statistical Science*, *8*(1), 31-39.

[3] Meier W., O. Staffelbach, (1995). The self-shrinking generator. In A.De Santis, editor, *Advances in Cryptology – EUROCRYPT '94*, vol.950 of LNCS, Berlin, Springer-Verlag, 205-214.

[4] Pincus, S. M. (1991). Approximate entropy as a measure of system complexity. *Proceedings of the National Academy of Sciences*, 88(6), 2297-2301.

[5] Pincus, S., and Kalman, R. E. (1997). Not all (possibly) "random" sequences are created equal. Proceedings of the National Academy of Sciences, 94(8), 3513-3518.

[6] Pincus, S., and Singer, B. H. (1996). Randomness and degrees of irregularity. *Proceedings of the National Academy of Sciences*, 93(5), 2083-2088.

[7] Rukhin, A. L. (2000). Approximate entropy for testing randomness. *Journal of Applied Probability*, 37(1), 88-100.

[8] Rukhin, A., Soto, J., Nechvatal, J., Smid, M., and Barker, E. (2001). *A statistical test suite for random and pseudorandom number generators for*

cryptographic applications. BOOZ-ALLEN AND HAMILTON INC MCLEAN VA.

[9] Soto, J. (1999, October). Statistical testing of random number generators. In *Proceedings of the 22nd National Information Systems Security Conference* (Vol. 10, No. 99, p. 12). Gaithersburg, MD: NIST.

[10] Soto, J., and Bassham, L. (2000). *Randomness testing of the advanced encryption standard finalist candidates*. BOOZ-ALLEN AND HAMILTON INC MCLEAN VA.

[11] Tashev, T., Bedzhev, B., Tasheva, Zh. (2007). The Generalized Shrinking-Multiplexing Generator, *ACM International Conference Proceeding Series 285*, Article number 48, *Proceedings of the 2007 international conference on Computer systems and technologies CompSysTech '07*.

[12] Tasheva A. (2012). Some cryptanalysis of a *p*-ary generalized self-shrinking generator. In *Proceedings of the 13th International Conference on Computer Systems and Technologies* (CompSysTech'12), Boris Rachev and Angel Smrikarov (Eds.). ACM, New York, NY, USA, 126-133.

[13] Tasheva A. T., Nakov O., Zh. A. Tasheva. (2013). About balance property of the p-ary generalized self-shrinking generator sequence. In Proceedings of the 14th International Conference on Computer Systems and Technologies (CompSysTech '13), Boris Rachev and Angel Smrikarov (Eds.). ACM, New York, NY, USA, 299-306.

[14] Tasheva A. T., Zh. N. Tasheva, A. M. Petrov (2011). Generalization of the Self-Shrinking Generator in the Galois Field GF(p^n), *Advances in Artificial Intelligence*, vol. 2011, Article ID 464971, 10 pages, 2011. doi:10.1155/2011/464971

[15] Tasheva Zh. N. (2012). Design and Analysis of 3-ary Generalized Shrinking Multiplexing Generator, *International Journal of Advance in Communication Engineering 4 (2)*, 129-140.

[16] Tsankov, T., Trifonov, T., and Staneva, L. (2013). A Survey of Phase Manipulated Signals with High Structural Complexity and Small Loses after Processing with Mismatched Filters. *Journal Scientific & Applied Research, 4*, 88-97.

[17] Tsankov, T., Trifonov, T., and Staneva, L. (2013). An algorithm for synthesis of phase manipulated signals with high structural complexity. *Journal Scientific & Applied Research, 4*, 80-87.