A Hybrid Model for Optimal Pseudorandom Bit Sequence Generation

Chapter · June 2019

DOI: 10.1007/978-981-13-8578-0_14

Ramen Pal
Assam University
6 PUBLICATIONS **4** CITATIONS

A Hybrid Model for Optimal Pseudorandom Bit Sequence Generation

Ramen Pal[1](✉) and Somnath Mukhopadhyay[2](✉)

[1] Department of Computer Science and Engineering, University of Kalyani, Kalyani, India
ramen.pal673@gmail.com

[2] Department of Computer Science and Engineering, Assam University, Silchar, India
som.cse@live.com

Abstract. Chaotic map gained its importance in the field of cryptography, due to its properties like, randomness, unpredictability, sensitivity on initial condition, aperiodicity, which is used to generate pseudorandom bit streams. In this paper the optimal values of chaos parameters are generated through Real Coded Genetic Algorithm (RCGA), which is optimal, unpredictable, and optimally sensitive. Here, a non-deterministic RCGA based optimal pseudo-random bit sequence generator based on Chaotic maps, such as Logistic Chaotic map, Skew Tent Chaotic map, Cross Coupled Logistic Chaotic map, Cross Coupled Skew Tent Chaotic map is proposed. A real coded crossover and mutation technique is proposed for RCGA. Seed values for chaotic map have been optimized by using all sub-functions of GA (RCGA). These seed values are used to generate optimal pseudorandom bit stream of finite length. The randomness of the bit stream is tested by using *NIST statistical test suit*.

Keywords: Real coded genetic algorithm · Logistic Chaotic Map · Skewtent chaotic map · Cross Coupled Logistic Chaotic Map · Cross coupled skewtent chaotic map · NIST

1 Introduction

The tool is used to generate a random sequence is called Random Number Generator or RNG. RNG can be divided in two classes, viz, Pseudo Random Number Generator (PRNG) and Truly Random Number Generator (TRNG) [2]. PRNG algorithms are deterministic in nature [6], the input to this type of algorithm is called seed and the output of this type of algorithm is a pseudorandom bit sequence [1]. The deterministic nature of the aforementioned algorithm is useful to generate same bit sequence for a same seed multiple times.

Chaotic maps are evolution functions. In the domain of mathematics a chaotic map can be used to produce chaotic and unpredictable number sequence, for its deterministic simplicity and chaotic behavior. In last two decades it is seen that Chaotic map can be used to design deterministic random numbers generators or

PRNG [21]. In 2013, Hu et al. proposed a Pseudorandom bit sequence generator(PRBG) based on Chen chaotic system [10]. In 2015 Martinez and Canton proposed a PRBG based on K-model maps [9]. In 2015 Sadkhan and Mohammad proposed a PRBG based on Unified chaotic map [19]. In 2014 Akhshani et al. proposed a PRBG based on Quantum chaotic map, and it showed that the output chaotic sequence is non-periodic [2]. In 2014 Francois et al. proposed a Secure PRNG three-mix, from where it is seen that rather than using a single chaotic map, mixing of several chaotic maps for designing a PRNG can be done [13]. In 2016, Liu et al. proposed a PRNG based on new multi-delayed Chebyshev map [11]. From the aforementioned PRNGs, it has been observed that these PRNGs generate a random bit sequence of finite length by using chaotic map. The main inputs to these PRNGs are the optimal seed values of the chaotic map, but it does not optimized the seed values for the respective chaotic map and the optimality should be checked mathematically. This task is difficult to perform for each execution to get a random bit sequence. So, they must rely on a set of optimal seed values. A predefined optimal seed value or a set of optimal seed values will not only degrade the security, but also not generate a unique bit sequence in each and every execution of a PRNG. In this research it has been established that the seed values for a chaotic map can be optimized by using evolutionary search algorithms like GA, and this optimized seed will generate a pseudorandom, unique and optimal bit sequence.

GA is a metaheuristic search algorithm. It works on randomly taken initial population of chromosomes. Each chromosome from the population is associated with a fitness value. The fitness value is calculated by using a fitness calculation strategy. GA uses the Darwin principle of natural selection and applies genetic operations, like Elitism, Crossover, Mutation, etc. in an ordered way. It is an iterating process, i.e. aforementioned genetic operations are performed iteratively until a predefined termination condition is achieved [8,14,15,18,20]. In recent years RCGA gains significant importance over the Binary Coded Genetic Algorithms or BCGA due to its properties like, fast execution rate, efficiency and straightforwardness.

In this paper a RCGA has been proposed to find the optimal values of Chaos parameters, Sect. 2 of the paper considers to obtain optimal seed values of chaotic map using various chaotic equation like Logistic, Skew-tent and Cross-coupled version of each of them. Also these optimal system parameters have been used to generate a pseudorandom bit sequence, and NIST specified tests are performed to conform the randomness. Results are discussed in Sect. 3. Conclusion and future scope is given in Sect. 4 and references are provided in the end.

2 Proposed Method for Pseudo-random Bit-Stream Generation

The propose technique consists of a three-step process. In the first step, an initial population of size n is encoded. Here n is the number of chromosomes. In the second step, different chaotic maps are considered separately. The system parameters of one of these chaotic maps are optimized by using genetic algorithm. In the third step, these optimized system parameters are used to generate a pseudorandom bit sequence by using the respective chaotic map equation. The randomness of the generated bit sequence is tested using NIST statistical test suit for randomness.

This proposed method considers Logistic, Skew tent, Cross coupled Logistic and Cross Coupled Skew tent map equations. One of these equation can be used for the generation of pseudorandom bit sequence. A seed is combination of μ and X_0, where μ is the system or control parameter and X_0 is the initial population value.

Logistic map shows complex chaotic behavior, i.e. Population values in each time step will be different [3,12,22]. The logistic map equation is presented in Eq. 1.

$$X_{i+1} = F(X_i, \mu) = \mu X_i(1 - X_i) \tag{1}$$

Skew tent chaotic map is a one dimensional chaotic map. This map is ergodic and also shows complex chaotic behavior [11,16]. The skew tent map equation is presented in Eq. 2.

$$X_{i+1} = G(X_i, \mu) = \begin{cases} \frac{X_i}{\mu} & \text{where, } 0 \leq X_i \leq \mu \\ \frac{1-X_i}{1-\mu} & \text{where, } \mu \leq X_i \leq 1 \end{cases} \tag{2}$$

The pseudorandom bit sequence for the logistic map or the skew tent map is generated by using Eq. 3.

$$P(i) = \begin{cases} 0 & \text{if, } X_i \leq mean(X_i) \\ 1 & \text{Otherwise} \end{cases} \quad \text{where, } 0 \leq i \leq n \tag{3}$$

Two chaotic maps are called cross coupled, if they have the same number time step, i.e. same in population size, and shares the values of each other's population. This paper shows that the proposed method has successfully been able to generate an optimal pseudorandom bit sequence by using cross coupled logistic map or cross coupled skew tent map. The cross coupled logistic map equation is presented in 4–5 and the cross coupled skew tent map equation is presented in 6–7.

$$X_{i+1} = F_1(X_i, \mu_1) \tag{4}$$

$$X_{j+1} = F_2(X_j, \mu_2) \tag{5}$$

$$X_{i+1} = G_1(X_i, \mu_1) \quad \text{where, } 0 \leq i \leq n \tag{6}$$

$$X_{j+1} = G_2(X_j, \mu_2) \quad \text{where, } 0 \leq j \leq n \tag{7}$$

The pseudorandom bit sequence for the cross coupled maps are generated by using 8.

$$P(i) = \begin{cases} 0 & \text{if, } X_i < X_j \\ 1 & \text{Otherwise} \end{cases} \quad \text{where, } 0 \leq i \leq n \tag{8}$$

The length of the pseudorandom bit sequence and n are taken as a user input. A seed is constituted by using two parameters X and μ of a chaotic map. So, the length of the chromosomes is *2*. The propose method assumes the values of P_c and P_m are *0.9* and *0.1* respectively. The value of P_c and P_m can also be taken as user input, but it has to keep in mind that P_c must be a very high probabilistic value and P_m must be a very low probabilistic value. Total number of generations is *200*. The proposed algorithm is presented in Algorithm 1.

Algorithm 1. Proposed method for Pseudorandom Bit stream Generation

```
Input: Number of chromosomes $n$, Length of the pseudorandom bit sequence is $q$
Output: Optimized seed for chaos and pseudorandom bit stream
1: begin
2: Initial population encoding, where each chromosome is the seed for chaos and its
   real coded
3: Optimal seed value for chaotic map optimization by using RCGA
4: Optimal pseudorandom bit sequence generation.
5: end
```

Section 2.1 considers the initial encoding process. Section 2.2 considers the optimization of seed values. Section 2.3 considers the pseudorandom bit sequence generation process.

2.1 Initial Population Encoding

Initial mating pool or population has n number of chromosomes. Each chromosome is real coded and it represents a seed of a chaotic map. The first value and second value of a seed are X_0 and μ. The value of X_0 should be in the range *0 to 1*, and it is chosen randomly. The value of μ is different for different chaotic maps. For Eq. 1 and for Eqs. 4–5, the value of μ should be in the range *3 to 4*, and for the Eq. 2 and Eqs. 6–7, the value of μ should be in the range *0 to 1*.

2.2 Optimal Seed Value for Chaos Optimization

Optimal seed values for chaos are optimized by using a RCGA. This method is presented in Algorithm 2. Here the fitness value of each chromosome is calculated first. It generates a bit sequence of size *128* bit for each chromosome by using a chaotic map. After this the randomness of the bit sequence corresponding to each chromosome, is tested by using Poker test. The result of the poker test is a chi-square value χ^2 which is also the fitness value for a chromosome. The proposed method considers this Poker test function, i.e. Eq. 9 as the objective function to minimize the χ^2 value in each generation. In a poker test for a $k-$bit of bit-stream, a positive integer q is selected in such a way that $\frac{k}{q} \geq 5 \times 2^q$. After that $z = \frac{k}{q}$ is calculated. After that the bit sequence is divided into z non-overlapping sub blocks, each of which is q bit of length. k_i is the count of the i^{th} type of occurrences in q bit sub blocks [1,16]. The calculation of χ^2 value can be expressed mathematically in Eq. 9.

$$\chi^2 = \frac{2^q}{z}(\sum_{i=1}^{2^q} {k_i}^2) - k \tag{9}$$

Elitism is performed by comparing the fitness value of the fittest chromosome X from the previous generation with the fitness value of the weakest chromosome Y from the current generation, i.e. if $fitness(X) > fitness(Y)$, then Y is replaced by the X in the output mating pool, otherwise Y survives in that generation, the selection is done by using Binary Tournament Selection(BTS) process. BTS is easy to implement and it is also efficient. BTS can avoid the premature convergence, because it gives the chance for selection to each and every chromosome, and that can degrade the convergence [7,17]. In BTS two chromosomes are selected randomly and after comparing their fitness values, the fittest chromosome survives in the mating pool [8]. The size of the mating pool is n.

Algorithm 2. Optimal seed value for chaotic map optimization

```
Input: n, Initial population
Output: Optimized seed for chaos
 1: begin
 2: for gen= 1 to 200 do                                      ▷ Number of Generations
 3:     for each chromosome from the mating pool do
 4:         By using chromosome as a seed and the objective function, generate a pseudorandom bit sequence X.
 5:         Compute the fitness value, by applying the Poker test of randomness on X.
 6:     end for
 7:     for i= 1 to n do                                      ▷ Elitism operation is done here
 8:         Select the weakest chromosome Y from the current generation.
 9:         Select the fittest chromosome Z from the previous generation.
10:         Compute the fitness values of Y and Z. Keep the fittest chromosome in the current generation and discard the other one.
11:     end for
12:     Create a mating pool of size N by using BTS.
13:     Perform Crossover operation on the mating pool of real coded chromosomes.
14:     Perform Mutation operation on the mating pool of real coded chromosomes.
15: end for
16: end
```

The next operation is crossover. Proposed crossover technique on real coded chromosomes is presented in Algorithm 3. This method minimizes the χ^2 value of a bit sequence of size *128* bit for a seed value of a chaotic map by using the objective function 9. In this method, two chromosomes are selected from the mating pool and copied to variables randomly. Then a crossover probabilistic value $Cross_{prob}$ is also chosen within the range *0 to 1*. Then this $Cross_{prob}$ is compared with the P_c. If it is greater than the P_c, then these chromosomes are copied directly to the output mating pool, otherwise crossover operation is performed on them. In a crossover operation, first the values of X_0 and μ are subtracted and the absolute results are stored in variables $temp_1$ and $temp_2$ respectively. Then some amount of values, i.e. a and b are taken randomly from $temp_1$ and $temp_2$ within the range *0 to* $temp_1$ and *0 to* $temp_2$ respectively. Then one of the following set of operation is performed over them. The first set

Algorithm 3. Proposed crossover technique on real coded chromosomes

```
Input: Mating pool, $n$, $P_c$
Output: Mating pool contains offspring
1: begin
2: for i=1 to $n/2$ do
3:     Select two chromosomes $X_1$ and $X_2$ randomly from the mating pool.
4:     Copy $X_1$ and $X_2$ randomly in two variables $chrom_1$ and $chrom_2$.
5:     Select a crossover probability $Cross_{prob}$ within the range 0 to 1.
6:     if $Cross_{prob} \leq P_c$ then
7:         Compute the absolute difference of $\mu$'s of $chrom_1$ and $chrom_1$, i.e. $temp1 = \mid \mu_{chrom1} - \mu_{chrom2} \mid$
8:         Compute the absolute difference of $X_0$'s of $chrom_1$ and $chrom_2$, i.e. $temp2 = \mid X0_{chrom1} - X0_{chrom2} \mid$
9:         Select two random values $a$ and $b$ within the range 0 to $temp_1$ and 0 to $temp_2$ respectively.
10:        Select an integer value Choice randomly, within the range 1 to 2.
11:        if $Choice = 1$ then
12:            Add $\mu$ with $A$ and $X_0$ with $B$ of the $chrom_1$
13:            Subtract $\mu$ with $A$ and $X_0$ with $B$ of the $chrom_2$
14:        else
15:            Subtract $\mu$ with $A$ and $X_0$ with $b$ of the $chrom_1$
16:            Add $\mu$ with $A$ and $X_0$ with $B$ of the $chrom_2$
17:        end if
18:        Boundary restriction is performed here
19:    end if
20:    $chrom_1$ and $chrom_2$ are copied back to their respective positions in the resultant mating pool
21: end for
22: end
```

of operation is *Addition then Subtraction*, i.e. a and b are added with X_0 and μ of the first chromosome, and a and b are subtracted with X_0 and μ of the second chromosome. The second set of operation is *Subtraction then Addition*, i.e. a and b are subtracted with X_0 and μ of the first chromosome, and a and b are added with X_0 and μ of the second chromosome. The set of operation is also chosen randomly. The output of this procedure is the offspring, which is then copied to the mating pool.

After that mutation is executed over the mating pool. Proposed mutation technique on real coded chromosomes is presented in Algorithm 4. In this technique, one chromosome from the mating pool is selected randomly. Then a mutation probabilistic value $Mute_{prob}$ is chosen randomly within the range 0 to 1. Then $Mute_{prob}$ is compared with the P_m. If it is greater than P_m, then this selected chromosome is copied directly to the output mating pool, otherwise it gets mutated. In mutation operation the integer part of the μ is neglected here, thus the X_0 and the fractional part of μ are copied in two separate variables, $temp_1$ and $temp_2$ respectively. Like crossover, here two set of operations are also present and the set of operation is also chosen randomly. In first set of operation X is calculated, where $X = temp2 \times Mute_{prob}$. Then, this X is added with $temp_1$ and subtracted with $temp_2$. In second set of operation X is also calculated, where $X = temp1 \times Mute_{prob}$. Then, this X is subtracted with $temp_1$ and

Algorithm 4. Proposed mutation technique on real coded chromosomes

```
Input:  Mating pool, $n$, $P_m$
Output:  Mating pool contains offspring
 1: begin
 2: for i=1 to $n/2$ do
 3:     Select one chromosome chrom randomly from the mating pool
 4:     if $Mute_{prob} \leq P_m$ then
 5:         Copy the $X_0$ and the fractional part of $\mu$ of chrom in temp1 and temp2
    respectively
 6:         Select a integer value Choice randomly, within the range 1 to 2
 7:         if $Choice = 1$ then
 8:             Take $(Mute_{prob}*100)\%$ amount of value $X$ from $temp_2$. Add $X$ to $temp_1$
    and subtract $X$ from $temp_2$
 9:         else
10:             Take $(Mute_{prob}*100)\%$ amount of value $X$ from $temp_1$. Add $X$ to $temp_2$
    and subtract $X$ from $temp_1$
11:         end if
12:         Boundary restriction is performed here
13:     end if
14:     Update $\mu$ of chrom by adding the previously discarded integer part of $\mu$ with
    $temp_2$. $temp_1$ will be the updated $X_0$ of chrom
15:     chrom is copied to the resultant mating pool in the same position, where it was
    during the selection
16: end for
17: end
```

added with $temp_2$. Finally the integer part of μ, which was neglected previously, is added with current $temp_1$ to get the updated μ. The $temp_2$ is copied to X_0. The resultant seed is the offspring, which is then copied to the resultant mating pool.

2.3 Optimal Pseudorandom Bit Sequence Generation

q bit optimal pseudorandom bit sequence can be generated by using the same objective function, that was used in Sect. 2.2. The optimal seed value which is the output of the Sect. 2.2 is used here as a seed for the aforementioned objective function on chaotic map.

3 Result and Analysis

The randomness of the optimal pseudorandom bit sequence is tested by using NIST statistical test suit. This test determines the unpredictability of a Pseudo Random Number Generator (PRNG). In recent years NIST becomes the industry norm for the randomness testing and it is the most stringent test suit. It contains 15 nearly independent statistical tests, which is focused on finding all possible types of non-randomness pattern that could exist in a pseudorandom bit sequence. If any type of non-randomness pattern is found by any of this test, then the sequence is declared as non-random, otherwise it is declared as random [4]. The results of these tests are represented by *P-values*. *P-value* is the probabilistic value. If the output *P-value* of a test is greater than or equal to *0.01*, then PRNG will pass that test for randomness with a *99%* confidence. NIST specified tests are performed to conform the randomness of the result of the proposed algorithm with different chaotic map functions. The results are shown in Tables 1, 2, 3 and 4.

Table 1 represent the *P-value* and *Result* of each test, which is present in NIST test suit. A pseudorandom bit sequence of finite length is the input for each test. This bit sequence is generated by using the proposed method where Logistic map equation is taken into consideration. Obtained *P-value* for a test, if greater than or equal to *0.01*, then it can be concluded that this proposed method has passed the particular test. For the each sub test of Random Excursion and the Random Excursion Variant test, if $P-value \geq 0.01$, then the sequence will be random, otherwise it will be non-random. From this table it can be clearly observed that all the P-values are greater than or equal *0.01*. So, it can conclude that this proposed method will generate pseudorandom bit sequence by using Logistic chaotic map.

Table 1. NIST test for the proposed method with logistic chaotic map

Index	Test index	P-value	Result
1	Frequency	0.9005	Pass
2	Block frequency	0.5681	Pass
3	Runs	0.1849	Pass
4	Longest-run-of-ones	0.4746	Pass
5	Binary matrix rank	0.2701	Pass
6	DFT(Spectral)	0.8711	Pass
7	Non overlapping template matching	0.9907	Pass
8	Overlapping template matching	0.2610	Pass
9	Maurers *universal statistical*	0.9673	Pass
10	Linear complexity	0.9856	Pass
11	Serial 1	0.3644	Pass
	Serial 2	0.9231	Pass
12	Approximation entropy	0.0607	Pass
13	Cumulative sums (Cusums)	0.9555	Pass
14	Random excursion		
	−4	0.3965	Random
	−3	0.2284	Random
	−2	0.9058	Random
	−1	0.2670	Random
	1	0.9478	Random
	2	0.5497	Random
	3	0.5385	Random
	4	0.0399	Random
15	Random Excursions Variant		
	−9	0.3965	Random
	−8	0.2284	Random
	−7	0.9058	Random
	−6	0.2670	Random
	−5	0.3965	Random
	−4	0.3965	Random
	−3	0.2284	Random
	−2	0.9058	Random
	−1	0.2670	Random
	1	0.9478	Random
	2	0.5497	Random
	3	0.5385	Random
	4	0.0399	Random
	5	0.9478	Random
	6	0.5497	Random
	7	0.5385	Random
	8	0.0399	Random
	9	0.0399	Random

Table 2. NIST test for the proposed method with skew tent chaotic map

Index	Test index	P-value	Result
1	Frequency	0.1336	Pass
2	Block frequency	0.7178	Pass
3	Runs	0.5949	Pass
4	Longest-run-of-ones	0.1767	Pass
5	Binary matrix rank	0.2159	Pass
6	DFT(Spectral)	0.8711	Pass
7	Non overlapping template matching	0.2769	Pass
8	Overlapping template matching	0.9156	Pass
9	Maurers *universal statistical*	0.9272	Pass
10	Linear complexity	0.8685	Pass
11	Serial 1	0.3449	Pass
	Serial 2	0.1249	Pass
12	Approximation entropy	0.0141	Pass
13	Cumulative sums (Cusums)	0.8690	Pass
14	Random excursion		
	−4	0.0166	Random
	−3	0.7985	Random
	−2	0.8221	Random
	−1	0.0572	Random
	1	0.3503	Random
	2	0.4405	Random
	3	0.5228	Random
	4	0.4966	Random
15	Random excursions variant		
	−9	0.5211	Random
	−8	0.5259	Random
	−7	0.5294	Random
	−6	0.6485	Random
	−5	0.7055	Random
	−4	0.7210	Random
	−3	0.9326	Random
	−2	0.8273	Random
	−1	0.3447	Random
	1	0.0890	Random
	2	0.2752	Random
	3	0.3525	Random
	4	0.3531	Random
	5	0.3778	Random
	6	0.4250	Random
	7	0.4631	Random
	8	0.4945	Random
	9	0.5211	Random

Table 2 shows that all the *P-values* are greater than or equal *0.01*. So, it can be concluded that this proposed method will generate Pseudorandom bit sequence by using Skew tent chaotic map.

Table 3. NIST test for the proposed method with cross coupled logistic chaotic map

Index	Test index	P-value	Result
1	Frequency	0.8231	Pass
2	Block frequency	0.9665	Pass
3	Runs	0.0903	Pass
4	Longest-run-of-ones	0.0799	Pass
5	Binary matrix rank	0.1285	Pass
6	DFT(Spectral)	0.1443	Pass
7	Non overlapping template matching	0.6161	Pass
8	Overlapping template matching	0.0205	Pass
9	Maurers *universal statistical*	0.9616	Pass
10	Linear complexity	0.4232	Pass
11	Serial 1	0.0338	Pass
	Serial 2	0.7023	Pass
12	Approximation entropy	0.1435	Pass
13	Cumulative sums (Cusums)	0.9555	Pass
14	Random excursion		
	−4	0.1027	Random
	−3	0.1201	Random
	−2	0.3130	Random
	−1	0.7576	Random
	1	0.4629	Random
	2	0.4499	Random
	3	0.5671	Random
	4	0.5281	Random
15	Random excursions variant		
	−9	0.3960	Random
	−8	0.6056	Random
	−7	0.8352	Random
	−6	0.4739	Random
	−5	0.5320	Random
	−4	0.3447	Random
	−3	0.2404	Random
	−2	0.2199	Random
	−1	0.0801	Random
	1	0.3173	Random
	2	0.5637	Random
	3	0.4674	Random
	4	0.5083	Random
	5	0.5320	Random
	6	0.3657	Random
	7	0.2983	Random
	8	0.3017	Random
	9	0.3320	Random

Table 3 shows that all the *P-values* are greater than or equal *0.01*. So, it can be concluded that this proposed method will generate Pseudorandom bit sequence by using Cross coupled Logistic chaotic map.

Table 4. NIST test for the proposed method with cross coupled skew tent chaotic map

Index	Test index	P-value	Result
1	Frequency	0.4952	Pass
2	Block frequency	0.9918	Pass
3	Runs	0.0133	Pass
4	Longest-run-of-ones	0.1381	Pass
5	Binary matrix rank	0.6420	Pass
6	DFT(Spectral)	0.1443	Pass
7	Non overlapping template matching	0.6410	Pass
8	Overlapping template matching	0.0528	Pass
9	Maurers *universal statistical*	0.9822	Pass
10	Linear complexity	0.6767	Pass
11	Serial 1	0.0112	Pass
	Serial 2	0.8521	Pass
12	Approximation entropy	0.2402	Pass
13	Cumulative sums (Cusums)	0.9642	Pass
14	Random excursion		
	−4	0.3898	Random
	−3	0.9593	Random
	−2	0.8101	Random
	−1	0.8366	Random
	1	0.7274	Random
	2	0.8700	Random
	3	0.4736	Random
	4	0.9253	Random
15	Random excursions variant		
	−9	0.5695	Random
	−8	0.5448	Random
	−7	0.5154	Random
	−6	0.4795	Random
	−5	0.4344	Random
	−4	0.4203	Random
	−3	0.5673	Random
	−2	0.9020	Random
	−1	0.8312	Random
	1	0.6698	Random
	2	0.8055	Random
	3	0.5673	Random
	4	0.8090	Random
	5	0.6698	Random
	6	0.5629	Random
	7	0.5154	Random
	8	0.5448	Random
	9	0.5695	Random

Table 4 shows that all the *P-values* are greater than or equal *0.01*. So, it can be concluded that this proposed method will generate Pseudorandom bit sequence by using Cross coupled skew tent chaotic map.

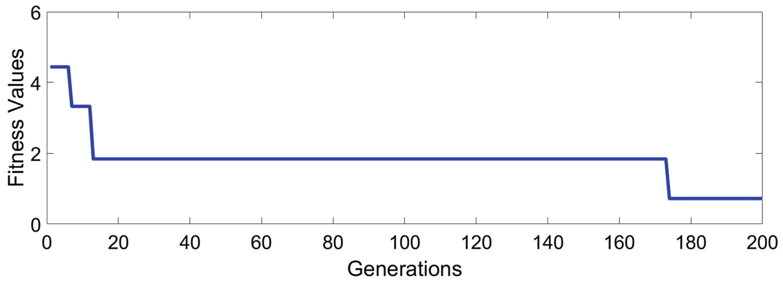

Fig. 1. Bar diagram showing minimization of fitness values in 200 iteration with logistic chaotic map

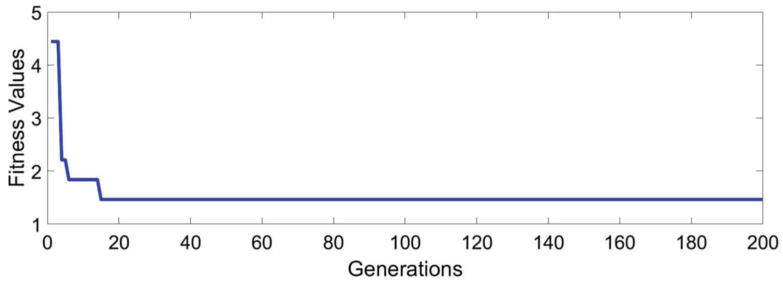

Fig. 2. Bar diagram showing minimization of fitness values in 200 iterations, with skew tent chaotic map

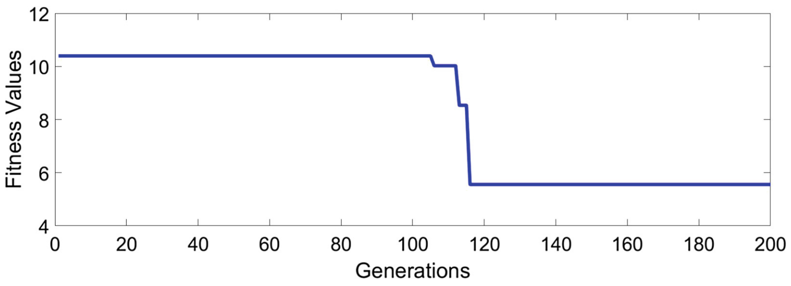

Fig. 3. Bar diagram showing minimization of fitness values in 200 iterations, with cross coupled logistic chaotic map

The generation wise growth in fitness values for *200* generations by using different type of chaotic map functions are shown as a bar diagram in Figs. 1, 2, 3 and 4. The X-axis of the graph represents the number of generations and the Y-axis represents the fitness values of the fittest chromosome in each generation.

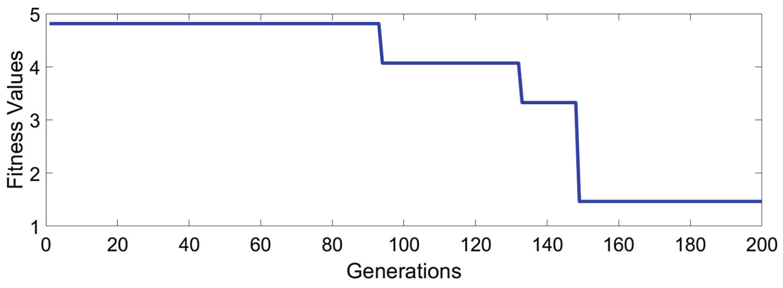

Fig. 4. Bar diagram showing minimization of fitness values in 200 iterations, with cross coupled skew tent chaotic map

From the Figs. 1, 2, 3 and 4 it can be clearly observed that the fitness value of the fittest chromosome is minimized after iterating the proposed method with Logistic or Skew Tent or Cross Coupled Logistic or Cross Coupled Skew Tent chaotic map for *200* times.

4 Conclusion and Future Scope

This research shows that the optimal seed value for a chaotic map can be optimized by using evolutionary algorithm(RCGA). This optimal seed values can be used to generate an optimal pseudorandom bit sequence. This paper proposes a RCGA enabled pseudorandom bit sequence generator, which passes all the statistical tests of NIST, by using four different types of chaotic maps. So, it can be concluded that the proposed method will generate an optimal pseudorandom bit sequence with *99%* confidence. Since the output of this generator is a pseudorandom bit sequence, i.e. it will generate same bit sequence for an optimized seed, so this bit sequence can be used as a key for Cryptographic applications.

This research is focused on finding the optimal seed values for a chaotic map for the generation of an optimal pseudorandom bit sequence. In future, finding the optimal chaotic map from a set of chaotic maps and its optimal seed value can be made. Multi-objective evolutionary algorithms will also be used for finding the optimal seed values for chaotic maps.

References

1. Menezes, A., van Oorschot, P., Vanstone, S.: Handbook of Applied Cryptography. CRC Press, Boca Raton (1996)
2. Akhshani, A., Akhavan, A., Mobaraki, A., Lim, S.C., Hassan, Z.: Pseudo random number generator based on quantum chaotic map. Commun. Nonlinear Sci. Numer. Simul. **19**(1), 101–111 (2014)
3. Bandyopadhyay, D., et al.: A novel secure image steganography method based on chaos theory in spatial domain. Int. J. Secur. Priv. Trust Manage. (IJSPTM) **3**(1), 11–22 (2014)

4. Bassham, L.E., et al.: SP 800–22 rev. 1a. A statistical test suite for random and pseudorandom number generators for cryptographic applications. National Institute of Standards & Technology, April 2010
5. Bhoskar, T., et al.: Genetic algorithm and its applications to mechanical engineering: a review. In: 4th International Conference on Materials Processing and Characterization, vol. 2, no. (4–5), pp. 2624–2630, July 2015
6. Barangi, M., Chang, J.S., Mazumder, P.: Straintronics-based true random number generator for high speed and energy-limited applications. IEEE Trans. Magn. **52**(1), 1–9 (2016)
7. Goldberg, D.E., Deb, K.: A comparative analysis of selection schemes used in genetic algorithms. Found. Genet. Algorithms **1**, 69–93 (1991)
8. Goldberg, D.E.: Genetic Algorithm in Search, Optimization and Machine Learning. Addison- Wesley, Boston (1989)
9. Garca-Martnez, M., Campos-Cantn, E.: Pseudo-random bit generator based on multi-modal maps. Nonlinear Dyn. **82**(4), 2119–2131 (2015)
10. Hu, H., Liu, L., Ding, N.: Pseudorandom sequence generator based on the chen chaotic system. Comput. Phys. Commun. **184**(3), 765–768 (2013)
11. Liu, L., Miao, S., Cheng, M., Gao, X.: A pseudorandom bit generator based on new multi-delayed Chebyshev map. Inf. Process. Lett. **116**(11), 674–681 (2016)
12. Lorenz, E.N.: The Essence of Chaos, 3rd edn. CRC Press, New York (1995)
13. François, M., Grosges, T., Barchiesi, D., Erra, R.: Pseudo-random number generator based on mixing of three chaotic maps. Commun. Nonlinear Sci. Numer. Simul. **19**(4), 887–895 (2014)
14. Mukhopadhyay, S., Mandal, J.K.: Denoising of digital images through PSO based pixel classification. Cent. Eur. J. Comput. Sci. **3**(4), 158–172 (2013)
15. Mukhopadhyay, S., Mandal, J.K.: A fuzzy switching median filter of impulses in digital imagery (FSMF). Circ. Syst. Sig. Process. **33**(7), 2193–2216 (2014). https://doi.org/10.1007/s00034-014-9739-z
16. Pareek, N.K., Patidar, V., Sud, K.K.: A random bit generator using chaotic maps. Int. J. Netw. Secur. **10**(1), 32–38 (2010)
17. Razali, N.M., Geraghty, J.: Genetic algorithm performance with different selection strategies in solving TSP. In: Proceedings of the World Congress on Engineering, IAENG II, pp. 1134–1139, July 2011
18. Sanjib Ganguly, D.S.: Distributed generation allocation on radial distribution networks under uncertainties of load and generation using genetic algorithm. IEEE Trans. Sustain. Energ. **6**(3), 688–697 (2015)
19. Sattar, B., Sadkhan, R.S.M.: Proposed random unified chaotic map as PRBG for voice encryption in wireless communication. In: International Conference on Communication, Management and Information Technology, vol. 65, no. 6, pp. 314–323, September 2015
20. Mukhopadhyay, S., Chaudhuri, T.D., Mandal, J.K.: A hybrid PSO-fuzzy based algorithm for clustering Indian stock market data. In: Mandal, J.K., Dutta, P., Mukhopadhyay, S. (eds.) CICBA 2017. CCIS, vol. 776, pp. 475–487. Springer, Singapore (2017). https://doi.org/10.1007/978-981-10-6430-2_37
21. Stoyanov, B., Kordov, K.: Novel secure pseudo-random number generation scheme based on two tinkerbell maps. Adv. Stud. Theor. Phys. **9**(9), 411–421 (2015)
22. Patidar, V., Sud, K.K., Pareek, N.K.: A pseudo random bit generator based on chaotic logistic map and it's statistical testing. Informatica **33**(4), 441–452 (2009)