Intern. J. Fuzzy Mathematical Archive
Vol. 6, No. 2, 2015,177-186
ISSN: 2320 –3242 (P), 2320 –3250 (online)
Published on 22 January 2015
www.researchmathsci.org

International Journal of
Fuzzy Mathematical
Archive

ABCRNG - Swarm Intelligence in Public key Cryptography for Random Number Generation

J.Sai Geetha[1] and D.I.George Amalarethinam[2]

[1]Department of Computer Science, Nehru Memorial College, Trichy, Tamilnadu, India
Corresponding Author e-mail: jsaigeetha99@gmail.com

[2]*D*epartment of Computer Science, Jamal Mohamed College, Trichy, Tamilnadu, India
e-mail: di_george@ymail.com,

Received 10 November 2014; accepted 3 December 2014

Abstract. Cryptography is an important tool for protecting and securing data. In public key cryptography, the key generation plays a vital role for strengthening the security. The random numbers are the seed values in key generation process in many of the public key cryptography algorithms, such as Elgamal, Rivest_Shamir_Adleman (RSA) algorithm etc. Much effort is dedicated to develop efficient Random Number Generators (RNG) with good statistical properties. In addition, many optimized functions are available for generating random numbers. Various RNG are compared against the performance of intelligent optimization algorithms. As an outcome, Intelligent optimization algorithms are very efficient to solve complex problems and also helpful for generating random numbers. The primary objective of the proposed algorithm (Artificial Bee Colony Random Number Generator - ABCRNG) is to provide quick and improved performance results having practical and feasible implementation of random number generators. To evaluate the randomness of the random numbers generated by ABCRNG, the Run test is carried out in up and down method and also in above and below mean method. Thus, pure random and non-repeating final keys are obtained by using ABCRNG which increases the key strength and security. The proposed ABCRNG can be used in many applications requiring random numbers and also in the design of secured public key cryptosystem.

Keywords: Public key cryptography, artificial bee colony (ABC), swarm intelligence, optimization, random numbers, random number generator (RNG), key generation

1. Introduction

In today's era of Information Technology, communication through Internet involves electronic transactions, mails etc. Effective communication primarily needs electronic security. The Information transmitted includes critical as well as personal data which every user desires to be safe and secure from illegal interception. Cryptography is the field of transforming messages from plaintext into unreadable format. The cipher text obtained in the process of encryption can be converted back to the original message by way of decryption by the concerned recipient. Cryptography is classified into two types: Symmetric Key Cryptography which uses single key for both encryption and decryption, and Asymmetric Key Cryptography which uses two keys, public key and private key

used for encryption and decryption processes respectively. Cryptography is essential for protecting information and also for increasing security with the advent of online transaction processing and e-commerce. Public key cryptography is one of the most important types of cryptography. Unique key has to be used in public key cryptography. Random Number Generators (RNG) are important building block for algorithms and protocols in cryptography. The need for random and pseudo random numbers arise in many cryptographic applications. For instance, common cryptosystems employ keys that must be generated in a random fashion. In addition, many cryptographic protocols require random or pseudo random inputs at various points such as, auxiliary quantities used in generating digital signatures, or for generating challenges in authentication protocols. The randomness of random number and pseudo random number generators is tested using statistical approach. The focus of this work is on those applications where randomness is required for cryptographic purposes. Random numbers are widely used in intelligent optimization algorithms such as Genetic Algorithm (GA), Ant Colony Optimization (ACO) and Particle Swarm Optimization (PSO), just to name a few. Random numbers are usually generated by deterministic algorithms called Random Number Generators (RNG) and play a key role in driving the search process. Swarm intelligence is a research branch that models the population of interacting agents or swarms that are able to be organized on their own. An ant colony, a flock of birds or an immune system is a typical example of a swarm system. Another example of swarm intelligence is that Bees' swarming around their hive. Artificial Bee Colony (ABC) Algorithm is an optimization algorithm based on the intelligent behavior of honey bee swarm. Key generation in cryptography has been dealt with many aspects but the use of ABC in the process is yet to be explored successfully. It is the most important part of encoding the data. A non repeating key guarantees better results and generates a code that is theoretically impossible to break. This work tries to explore the use of non-conventional techniques in the key generation process.

2. Literature review

Bahadori et.al. [1] described an approach for secure and fast key generation of the public key cryptographic algorithm of RSA. A new method for generating the large random prime numbers was proposed based on which a key pair can be generated in the reduced time. The key generation process is based on selecting appropriate public key from a set of pre-defined public keys and computing the private key using Euclid's extended algorithm. Mishra and Bali [2], presented an application of Genetic Algorithm in the field of cryptography. Key selection in public key cryptography is a selection process in which keys can be categorized on the basis of their fitness. Finally, it produced purely random and non-repeating final keys which increased the keys strength and security. Jhajharia et al. [3] proposed an algorithm for Public Key Cryptography (PKC) using the hybrid concept of two evolutionary algorithms, Particle Swarm Optimization (PSO) and Genetic Algorithm (GA) respectively. PSO-GA algorithms are used for generating the fittest among the fine fit keys in key domain containing best keys of highest possible strength. Mohammad et al. [4] projected the randomness of quantum key distribution system based BB84 protocol. It was investigated with suite of random number tests. Usually, the degree of randomness in QKD bits are effected by the noise and unrecognized photon polarization. Abu-Mouti and Elhawary [5] presented an overview of the literature

employing the recent Artificial Bee Colony (ABC) algorithm. The ABC algorithm is a population-based meta-heuristic optimization technique inspired by the intelligent foraging behavior of honeybee swarms. The performance characteristics and key features of ABC algorithm are also described.

Pareek et al. [6] exploited the interesting properties of chaotic systems to design a random bit generator, in which two chaotic systems are cross-coupled with each other. Statistical tests are applied to evaluate the randomness of the bit streams generated by the Cross-coupled Chaotic Tent Map Based Bit Generator (CCCBG).

3. Proposed methodology

3.1. Key generation in public key cryptography

Strong random number generation is important throughout every phase of public key cryptography. Random number generators suitable for use in cryptographic applications may need to meet stronger requirements than for use in other applications. Mandatorily, their outputs may need to be unpredictable in the absence of knowledge of the inputs.

The security of cryptographic systems depends on some secret data that is known only to the authorised persons but unknown and unpredictable to others. Some randomization is typically warranted to achieve this unpredictability.

3.1.1. RNG function

Quality in the RNG process is always required for security. Lack of quality generally provides way to attack vulnerabilities and so leads to lack of security. The RNG process is particularly attractive to attackers because it is typically a single isolated hardware or software component which can be located easily. If the intruder can substitute pseudo-random bits generated in a way by prediction, security is totally compromised, however it is generally undetectable by any upstream test of the bits. Software RNG just as with other components of a cryptosystem, a software random number generator should be designed to resist certain attacks. Typical attacks possible on a RNG includes,

i. Direct cryptanalytic attack - When an attacker obtained part of the stream of random bits and can use this to distinguish the RNG output from a truly random stream.
ii. Input-based attacks - Modify the input to the RNG to attack it, for example by "flushing" existing entropy out of the system and put it into a known state.
iii. State compromise extension attacks - When the internal secret state of the RNG is known at a particular time, this can be used predict future output or to recover previous outputs. This is possible when a generator starts up and has little or no entropy, hence an attacker may be able to obtain an initial guess at the state.

3.1.2. Hardware RNG

A number of attacks on hardware random number generators are possible which includes trying to capture radio-frequency emissions from the computer. Random numbers typically go through several layers of hardware and software before they are being put into use. A Peripheral device may be to generate Bits which are being sent over a serial cable, collected in an operating system utility and retrieved by a system call.

3.1.3. Swarm intelligence algorithm

Optimization problems arise in many application areas such as economy, engineering and management. Effective and efficient optimization algorithms are always required to tackle increasingly complex real-world optimization problems. Over the past several years, some swarm intelligence algorithms, inspired by the social behaviors of fish, birds or insects have been proposed. It is applied to solve optimization problems such as Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC) and Firefly Algorithm (FA). A recent study has shown that ABC performs significantly better or at least comparable to other swarm intelligence algorithms.

3.2. Artificial bee colony algorithm (ABCRNG)

ABC is a new swarm intelligence algorithm proposed by Karaboga in 2005[7], which is inspired by the behavior of honey bees. Since the development of ABC, it has been applied to solve different kinds of problems.

ABC algorithm consists of three kinds of bees namely employed bees, onlooker bees and scout bees. Employed bees forms half of the colony, and the rest includes onlooker bees. Employed bees are in charge of exploiting the nectar sources explored before and giving information to the waiting bees (onlooker bees) in the hive about the quality of the food source sites which they are exploiting. Onlooker bees stay in the hive and decide on a food source to exploit based on the information shared by the employed bees. Scouts search the environment randomly in order to find a new food source depending on an internal motivation or based on possible external clues.

This evolving intelligent performance in foraging bees can be represented as follows:

i. At the initial phase of the foraging process, the bees randomly initiate to explore the environment in order to find a food source.
ii. The bee becomes an employed forager after finding a food source. Then it starts to exploit the discovered source. The employed bee returns to the hive with the nectar and unloads the nectar. After unloading the nectar, she can go back to her discovered source site directly or she can share information about her source site by performing a dance on the dance area. If her source is exhausted, she becomes a scout and starts to randomly search for a new source.
iii. Onlooker bees waiting in the hive watch the dances advertising the profitable sources and choose a source site depending on the frequency of a dance proportional to the quality of the source.

In the ABC algorithm proposed by Karaboga [7], the position of a food source represents a possible solution to the optimization problem, and the nectar amount of a food source corresponds to the profitability (fitness) of the associated solution. Each food source is exploited by only one employed bee. In other words, the number of employed bees is equal to the number of food sources existing around the hive (number of solutions in the population). The employed bee whose food source has been abandoned becomes a scout.

3.2.1. Producing initial food source sites

If the search space is considered to be the environment of the hive that contains the food source sites, the algorithm starts with randomly producing food source sites that correspond to the solutions in the search space. Initial food sources are produced randomly within the range of the boundaries of the parameters.

$X_{ij}=X_j^{min}+rand(0,1)(X_j^{max}-X_j^{Min})$, where i = 1. . .SN, j = 1. . .D. (1)

SN is the number of food sources and D is the number of optimization parameters Equation (1). In addition, counters which store the numbers of trials of solutions are reset to 0 in this phase.

After initialization, the population of the food sources (solutions) is subjected to repeat cycles of the search processes of the employed bees, the onlooker bees and the scout bees. Termination criteria for the ABC algorithm might be reaching a maximum cycle number (MCN) or meeting an error tolerance.

3.2.2. Sending employed bees to the food source sites

As mentioned earlier, each employed bee is associated with only one food source site. Hence, the number of food source sites is equal to the number of employed bees. An employed bee produces a modification on the position of the food source (solution) in her memory depending on local information (visual information) and finds a neighboring food source, and then evaluates its quality. In ABC, finding a neighboring food source is defined by Eq. (2)

$$V_{ij}=X_{ij}+\phi_{ij}(X_{ij}-X_{kj}) \quad (2)$$

Within the neigbourhood of every food source site represented by x_i, a food source t_i is determined by changing one parameter of x_i. In Eq. (2), j is a random integer in the range [1, D] and kϵ{1,2, . . .SN} is a randomly chosen index that has to be different from i.ϕ_{ij} is a uniformly distributed real random number in the range [-1,1].

As can be seen from Eq. (2), as the difference between the parameters of the $x_{i,j}$ and $x_{k,j}$ decreases, the perturbation on the position $x_{i,j}$ decreases. Thus, as the search approaches to the optimal solution in the search space, the step length is adaptively reduced.

If a parameter value produced by this operation exceeds its predetermined boundaries, the parameter can be set to an acceptable value. In this work, the value of the parameter exceeding its boundary is set to its boundaries.

If $x_i > x_i^{max}$ then $x_i = x_i^{max}$;

If $x_i < x_i^{min}$ then $x_i = x_i^{min}$.

After producing v_i within the boundaries, a fitness value for a minimization problem can be assigned to the solution v_i by Eq. (3).

$$\textbf{Fitness}_i = \begin{cases} \frac{1}{1+f_i} & if\ f_i \geq 0 \\ 1+abs(f_i) & if\ f_i < 0 \end{cases} \quad (3)$$

where f_i is the cost value of the solution t_i. For maximization problems, the cost function can be directly used as a fitness function. A greedy selection is applied between x_i and t_i; then the better one is selected depending on fitness values representing the nectar amount of the food sources at x_i and t_i. If the source at t_i is superior to that of x_i in terms of profitability, the employed bee memorizes the new position and forgets the old one. Otherwise the previous position is kept in memory. If x_i cannot be improved, its counter holding the number of trials is incremented by 1, otherwise, the counter is reset to 0.

3.2.3. Calculating probability values involved in probabilistic selection

After all employed bees complete their searches, they share their information related to the nectar amounts and the positions of their sources with the onlooker bees on the dance area. This is the multiple interaction features of the artificial bees of ABC. An onlooker bee evaluates the nectar information taken from all employed bees and chooses a food

source site with a probability related to its nectar amount. This probabilistic selection depends on the fitness values of the solutions in the population. A fitness-based selection scheme might be a roulette wheel, ranking based, stochastic universal sampling, tournament selection or another selection scheme. In basic ABC, roulette wheel selection scheme in which each slice is proportional in size to the fitness value is employed Eq. (4)

$$\mathbf{P_i=fitness_i} \quad \sum_{i=1}^{SN} fitness_i \qquad (4)$$

In this probabilistic selection scheme, as the nectar amount of food sources (the fitness of solutions) increases, the number of onlookers visiting them increases, too. This is the positive feedback feature of ABC illustrated in the Fig.1.

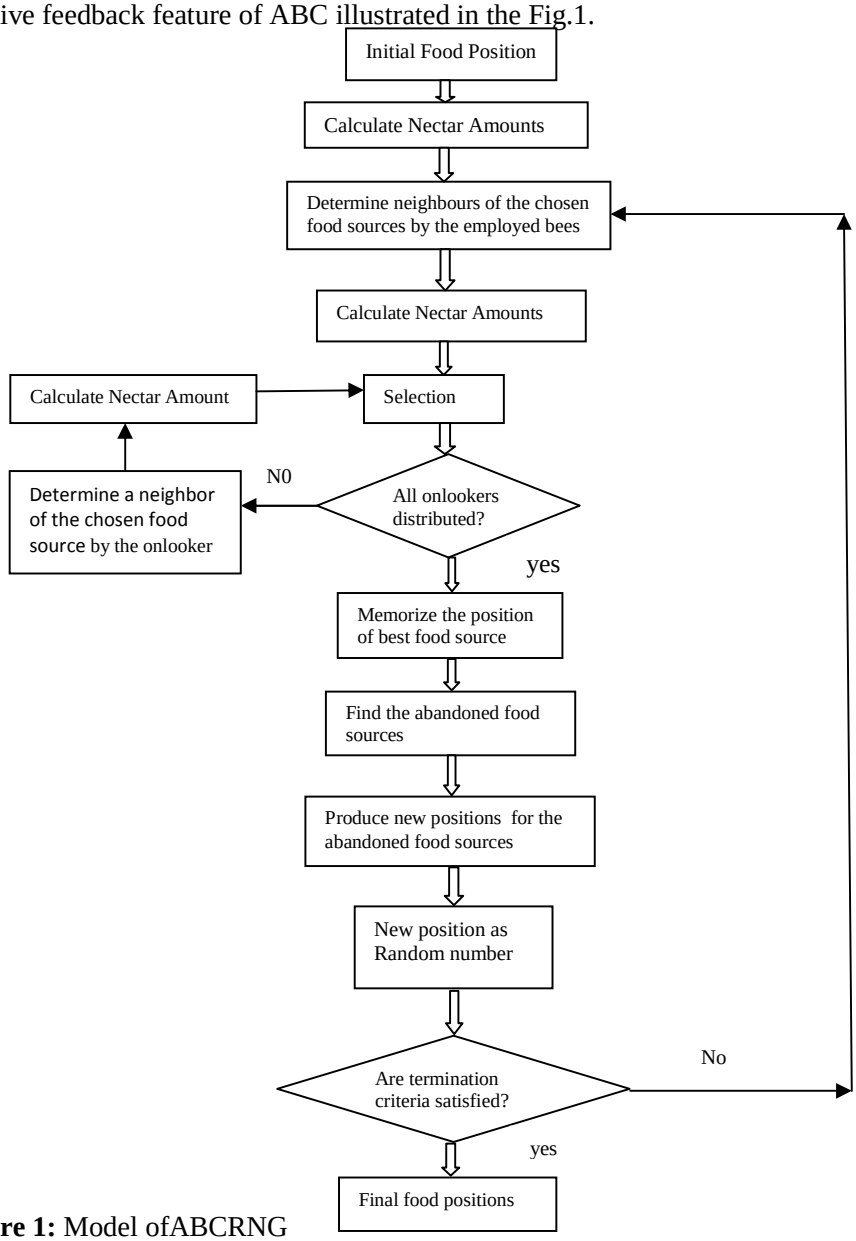

Figure 1: Model ofABCRNG

3.2.4. Food source site selection

In the ABC algorithm, a random real number within the range [0,1] is generated for each source. If the probability value (p_i in Eq. (4)) associated with that source is greater than this random number then the onlooker bee produces a modification on the position of this food source site by using Eq. (2) as in the case of the employed bee. After the source is evaluated, greedy selection is applied and the onlooker bee either memorizes the new position by forgetting the old one or keeps the old one. If solution x_i cannot be improved, its counter holding trials is incremented by 1, otherwise, the counter is reset to 0. This process is repeated until all onlookers are distributed onto food source sites.

3.3. Result and discussion

The proposed work is implemented in Java platform. It demonstrates the application of ABC algorithm in PKC. The resultant RNG is a suitable match for generation of keys of higher fitness for strengthened cryptography. The keys are found to be non-computable easily, large keys of variable size having no relation to prior history of keys generated during iterations.

There are three control parameters used in ABC

i. The number of the food sources which is equal to the number of employed or onlooker bees
ii. Colony size is equal to the population size
iii. Maximum cycle number

The experiment is repeated with different random seeds. The mean function values of the best solutions found by the algorithms for different dimensions have been recorded.

Rigorous standard testing is carried out for generation of random numbers meant for designated PKC. The statistical tests are carried out on the random numbers generated for each iteration of the algorithm. This ensures and authenticates that the generated numbers passed various statistical test to prove their randomness.

3.3.1. Run Test (Up and Down)

True Randomness of produced numbers are tested through run test with up and down method.

S.No	Sample size(n)	Calculated Value	Comparison with Tabulated value($Z_{0.025}$)
1	10	-0.28	-0.28<1.96
2	20	-0.56	-0.56<1.96
3	30	0.59	0.59<1.96
4	40	0.26	0.26<1.96
5	50	-1.36	-1.36<1.96

Table 1: Results of run test (up and down)

The calculated run test values Z with various sample sizes n is found. The value exhibits that it is less than the critical tabulated value of 1.96 as shown in Table 1. Hence result passes run test and it shows the pure random distribution of randomly generated numbers.

3.3.2. Run test (above and below mean)

The random numbers which are produced by the ABC algorithms is highly random in nature. It is evident by the run test with above and below mean method as tabulated below.

S.No	Sample size(n)	Calculated Value	Comparison with Tabulated value($Z_{0.025}$)
1	10	1.75	1.75<1.96
2	20	-0.81	-0.81<1.96
3	30	-1.12	-1.12<1.96
4	40	0.55	0.55<1.96
5	50	-0.96	-0.96<1.96

Table 2: Results of run test (above and below mean)

The calculated value Z of various sample sizes are found less than the critical value as shown in the Table 2. Hence the generated numbers are highly random.

3.3.3. Poker test

To check the anomaly of expected value and obtained value, poker test is performed. The test is conducted by three different sample sizes viz 30, 40 and 50 as tabulated in Table 3, Table 4 and Table 5 respectively.

Combination i	Observed frequency (O_i)	Expected frequency (E_i)	$(O_i-E_i)^2/E_i$
Three different digits	18	22	0.72
Three like digits	0	0	0
Exactly one pair	12	8	0.72
	30	30	1.44

Table 3: Poker test with sample size 30

Combination i	Observed frequency (O_i)	Expected frequency (E_i)	$(O_i-E_i)^2/E_i$
Three different digits	26	29	0.31
Three like digits	1	0	0
Exactly one pair	13	11	0.36
	40	40	0.67

Table 4: Poker test with sample size 40

Combination i	Observed frequency (O_i)	Expected frequency (E_i)	$(O_i-E_i)^2/E_i$
Three different digits	35	36	0.33
Three like digits	0	1	1
Exactly one pair	15	13	0.31
	50	50	1.64

Table 5: Poker test with sample size 50

The appropriate degrees of freedom are 1 less than the number of class intervals selected, where $x^2_{0.05/2} = 5.99$. All the calculated values are less than the tabulated value 5.99. Thus the independence of the random numbers is accepted on the basis of the present test.

4. Conclusion

ABC is proved to be favourable in selection of the best possible random number generator from the key domain. Main statistical tests are carried out and the results are passed proving the effectiveness and efficiency of the ABC than other methods. The final numbers generated are unique, random, non-repeating and cryptographically strong in nature with large linear complexity. Thus, numbers generated produced are also safe from related key attacks. Generated random numbers successfully passed both run test and the poker test. To ensure best results, long iterations of algorithms are implemented further increasing the complexity of keys generated. Better test results exhibit the randomness of generated numbers. It is also observed that no such plaintext can be chosen which shall reveal the key in cipher text because keys are purely random and independent of plaintext. Keys generated can also be used as input to hash function for MAC and generation of Message digest. The algorithm enjoys the add-on benefits of ABC. A number of issues related to RNG are solved in proposed algorithm. The proposed algorithm suits all kinds of public key cryptography system to strengthen the key generation process which ultimately results in secured transmission of information.

REFERENCES

1. M.Bahadori, M.R.Mali, O.Sarbishei and M.Atarodi, A novel approach for secure and fast generation of rsa public and private keys on smart card, *IEEE International Conference-NEWCAS Conference*, 2010, 265-268.
2. S.Mishra and S.Bali, Public key cryptography using genetic algorithm, *International Journal of Recent Technology and Engineering*,2(2)(2013) 150-154.
3. S.Jhajharia, S.Mishra and S.Bali, Public key cryptography using particle swarm optimization and genetic algorithm, *International Journal of Advanced Research in Computer Science and Software Engineering*, 3(6) (2013) 832-839.
4. O.K.J.Mohammad, S.Abbas, E.-S.M.E.-Horbaty and A.-B.M.Salem, A new trend of pseudo random number generation using QKD, *International Journal of Computer Applications*, 96(3) (2014). 13-17.
5. F.S.A.-Mouti, M.E.Elhawary, Overview of artificial bee colony algorithm and its applications, *IEEE International Conference–System Conference*,(2013) 1-6.
6. N.K.Pareek, V.Patidar and K.K.Sud, A random bit generator using chaotic maps, *International Journal of Network Security*, 10(1) (2010) 32-38.
7. B.Akay and D.Karaboga, *A modified Artificial Bee Colony Algorithm for Real-parameter Optimization*, Elsevier, 2010.
8. B.D.Karaboga, An artificial bee colony (abc) algorithm for numeric function optimization, *IEEE Swarm Intelligence Symposium 2006* (Indianapolis, Indiana, USA), 39(2007) 451-471.

9. G.Poornima, Naik and G.R.Naik, Asymmetric key encryption using genetic algorithm, *International Journal of Latest Trends in Engineering and Technology*, 3(3) (2014) 118– 128.
10. B.Schneier, *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, John Wiley & Sons, New York, 1996.
11. U.Maurer, A universal statistical test for random bits generators, *Journal of Cryptography*, 5 (1992) 89-106.