Information & Security

Michal Turčaník & Martin Javurek
vol. 43, no. 1 (2019): 54-61
https://doi.org/10.11610/isij.4305

Published since 1998 ISSN 0861-5160 (print), ISSN 1314-2119 (online)

Research Article

Cryptographic Key Generation by Genetic Algorithms

Michal Turčaník and Martin Javurek

Armed Forces Academy, Liptovsky Mikulas, Slovakia, http://www.aos.sk

ABSTRACT:

One of the security conditions of Vernam's cipher is that the encryption key must be greater than or equal to the open text we want to encrypt. At the same time, this key must not be repeated in another encryption. Then, each change of the encryption key adds security to the encryption process. If a cipher is changed several times while encrypting a single open text, it becomes very difficult to decrypt the message. Therefore, our goal is to design a mechanism to generate an encryption key using a Tree Parity Machine and a Genetic Algorithm that will be able to create the same encryption keys on both sides that enter the encryption process. These keys should change during encryption. One of the first tasks is to create an input population for the genetic algorithm from the synchronized Tree parity machine. Therefore, this article presents one of the possible ways to create an input population without using too many synchronizing TPMs.

ARTICLE INFO:

RECEIVED: 19 JUN 2019
REVISED: 22 AUG 2019
ONLINE: 16 SEP 2019

KEYWORDS:

genetic algorithms, tree parity machine, cryptographic keys generation

Corresponding Author: michal.turcanik@aos.sk; martin.javurek@aos.sk

Introduction

Security of the internet is still more and more important along with spreading the internet connections over the world. Availability of commercial communication systems cause the potential risk of their misusing from adversary side.[9] There are several scenarios how communication can be misused. One of them is to coordinate of adversary attacks against friendly units during operations.

The motivation of the paper is to create a method to generate cryptographic keys for Tree Parity Machine. Tree Parity Machine (TPM) is a special type of multilayer forward neural network that is used in cryptography.[1] The neural network is composed of neurons and synapses. The behaviour of TPM can be changed by changing the values of synapses. For secure communication, TPM must be set on both sides of the chain the same way.[4] We do not intend to send all parameters (cryptographic keys) by insecure channel but we would like to create them by genetic algorithms independently on both sides of the communication chain. Results of the genetic algorithm generation on both sides of the chain must be identical and the process must be deterministic. The algorithm and the results of the new method of cryptographic key generation by the genetic algorithm are presented in the paper.

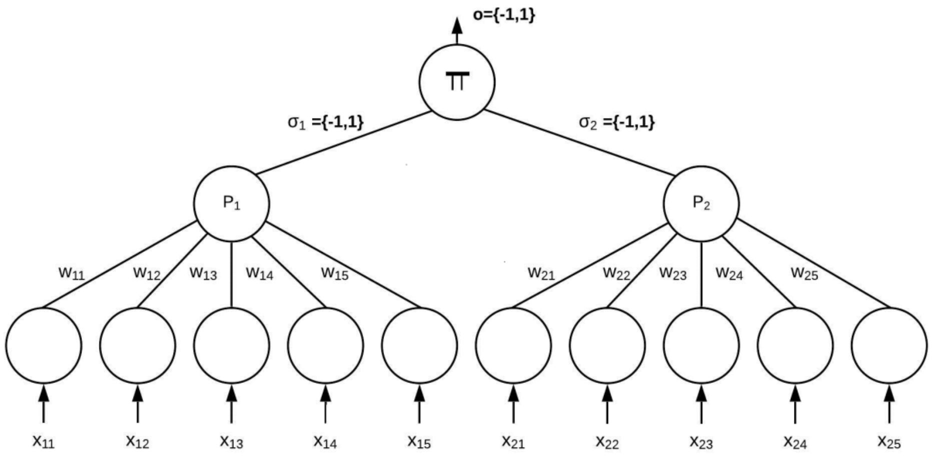

Figure 1: Algorithm of GA Key Generation.

1. Tree Parity Machine

Tree Parity Machine (TPM) is a special type of multilayer forward neural network that is used in cryptography. It is an artificial neural network topology for neural cryptography. The basis of TPM topology cryptography is the use of two identical artificial neural networks that are able to synchronize after mutual learning. TPM consists of K hidden neurons, N inputs to each hidden neuron and one output o. Each entry into a hidden neuron has a generated random weight that can take values from –L to + L. [1] These weights are changed by the learning rules so that after synchronization, the synapses values, between the respective inputs to each hidden neuron in both synchronizing TPMs, are the same. This means that the weights values are the same in both TPMs after synchronization. Furthermore, these weights may be used directly as an encryption key or may be used in any of the encryption key creation algorithms. The number of TPM weights depends on the number of hidden neurons and from inputs into each hidden neuron.

The basic assumption for using TPM in cryptography is that synchronizing TPMs are initially identical. This means that the number of hidden neurons, inputs into each hidden neuron, as well as the possible values that can gain weights are identical and are kept secret from attackers. The weight values are different due to their random generation but must be kept secret.[2]

The TPM synchronization process consists of generating random input x to be used as input to both TPMs to calculate the output in both TPMs. Only if the output value of both TPMs is the same, some of the learning rules is applied. This learning rule must be the same for both TPMs. On the basis of the learning rule, there will be updates, that is, a change in weights. The learning rule is applied until the both TPMs have the same values of weights.[3, 6, 7]

2. Genetic Algorithms

Genetic algorithms (GAs) are adaptive heuristic algorithms mainly applied in the tasks of the optimization and the searching. They are based on principles of natural selection and natural genetics. They belong to the class of evolutionary algorithms and they are using biological evolutionary tools (operator of the mutation, crossover and selection). GA's heavily rely on inheritance to find solutions for optimization problems.[8]

The main idea involved in the GAs is to replicate the randomness of nature where the population of individuals adapts to its surroundings through the natural selection process and behaviour of the natural system. This means that the survival and reproduction of a specimen are promoted by the elimination of weak features. GAs creates a population in such a way that the feature which is dominant that has higher fitness value is replicated more likewise rest of the population. Evolution makes GAs a good candidate for the process of generating keys.

The main task of GAs is to create set of keys, which will be used for securing communication, on the side of source and the destination of communication simultaneously. From set of keys the final user will have a possibility to choose the same key on both sides which will be used for setting of parameters of TPM. When transfer of data is done new key can be potentially applied for new communication.[5]

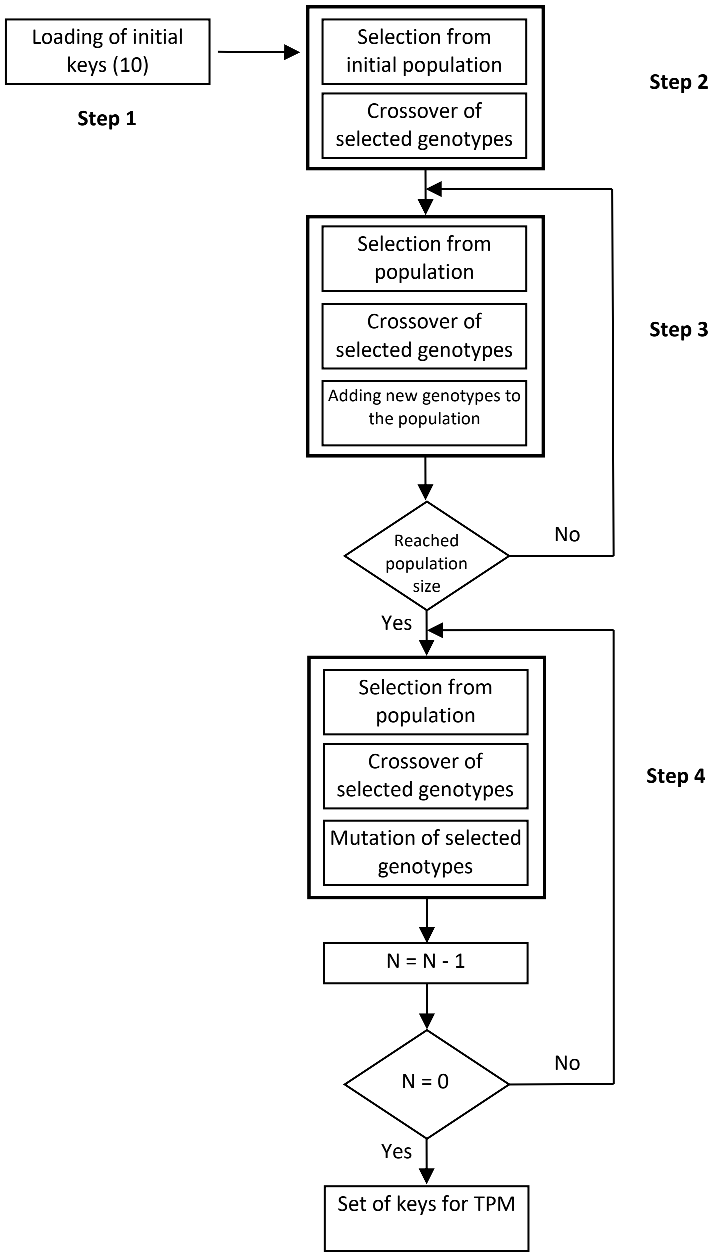
Loading of initial keys (10)
Step 1
Selection from initial population
Crossover of selected genotypes
Step 2
Selection from population
Crossover of selected genotypes
Adding new genotypes to the population
Step 3
Reached population size
No
Yes
Selection from population
Crossover of selected genotypes
Mutation of selected genotypes
Step 4
N = N - 1
N = 0
No
Yes
Set of keys for TPM

Figure 2: Algorithm of GA Key Generation.

3. Algorithm of Cryptographic Key Generation (by Genetic Algorithms)

The algorithm of key generation for TPM is divided to 4 basic steps (Figure 2). Last two steps are repeated until final condition is met. The 3rd step is repeated until the expected size of population is reached (e.g. 80 or 100 genotypes). The 4th step uses all three basic genetic operations (selection, crossover and mutation) to produce final population, which is used to define parameters of TPM. The final condition is represented by realisation of defined numbers of iterations.

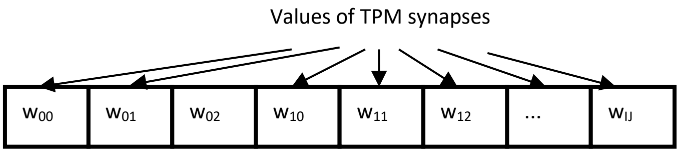

Figure 3: Genotype structure.

Step 1 Loading of initial keys set

The initial key set is input to the process of key generation. The initial key set consists of fixed number of keys (e.g. 10) and each key is composed of values of TPM synapses (Figure 3). The range for every synapse is from -7 to 7.

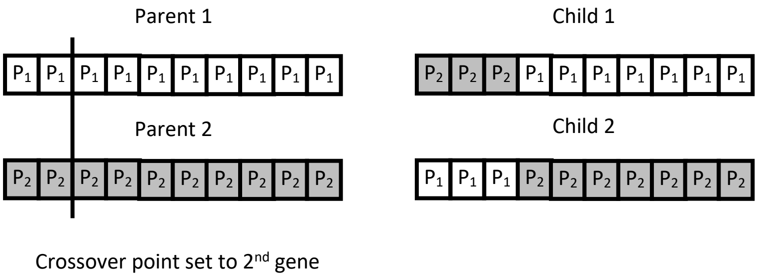

Figure 4: Single point crossover.

Step 2 Crossover of initial key set

On the base of the loaded key set the initial population of genotypes is created. Each genotype represents one key. After that selected genotypes are used to create new population with the same size by single point crossover operator, which creates two new offspring genotypes on the base two of parent´s genotype (Figure 4). The only one crossover point was set after 2nd gene of the parent´s genotypes. During the execution of the algorithm, the position of the crossover point is changed in the range between 1st and 9th genes.

Step 3 Population size increase

Input to the step 3 is population created in step 2. At first they are selected two genotypes for crossover operation. After that single point crossover operator is realized over two parent genotypes and creates two new offspring genotypes. The only one crossover point was set after randomly generated position in the parent´s genotypes. This operation is repeated for whole population. The new created genotypes are added to the actual population. The step 3 is repeated until the final size of population is reached (e.g. 100 genotypes) and it can be set by user. The optimal size of population is 100 genotypes which represents 100 possible setting for TPM.

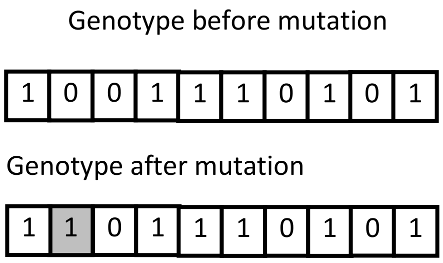

Figure 4: Mutation operation.

Step 4 Population size increase

Input to the step 3 is population created in step 2. At first, they are selected two genotypes for crossover operation. After that single point crossover operator is realized over two parent genotypes and creates two new offspring genotypes. The only one crossover point was set after randomly generated position in the parent´s genotypes. This operation is repeated for whole population. The new created genotypes are added to the actual population. The step 3 is repeated until the final size of population is reached (e.g. 100 genotypes) and it can be set by user. The optimal size of population is 100 genotypes which represents 100 possible setting for TPM.

The mutation rate is set to value 0.002. It can be used several types of mutation: flipping of bits, boundary mutation, non-uniform mutation, uniform mutation and Gaussian mutation.

The size of population after realisation of 4th step is the same (no new genotypes are added to the population).

4. Frequency Test of Generated Cryptographic Keys

Frequency test tells us whether the output of the genetic algorithms fulfils the requirements expected from the new cryptographic keys. Moreover, the quality of new cryptographic keys can be commented considering test results. Any new cryptographic key is composed of some parts of the initial population which was received during initialization of the communication chain. The major requirement is continuous uniform distribution of values of final population. Each member of the analysed population is equally probable for the continuous uniform distribution.

The result of the frequency test of the proposed method after 10 iterations of the GAs is shown in Fig. 5. In this figure ten different colours are used to depict the origin of the cryptographic key values. Every colour is equal to the one of the initial keys which was received in the beginning. Whole string of synapses (cryptographic key) is represented by the one column in Fig. 5. Every value from the final population has frequency of appearance 1%, which is appropriate result.

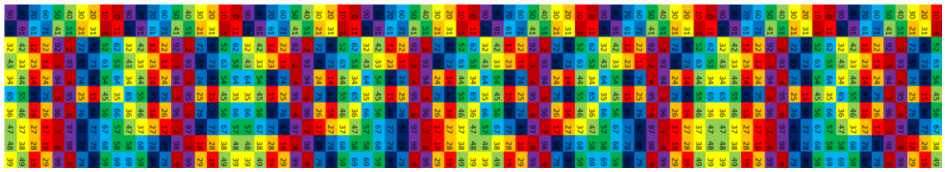

Figure 5: Population of genotypes on the base of parents from initial population.

Conclusions

In today communication technology, data encryption algorithms are used to provide secure communication between users.[4] So in this study, the possibility of using of genetic algorithms for cryptographic keys generation was analysed.

For a mechanism of generation of the cryptographic key using Tree Parity Machine and genetic algorithm that will be identical on both sides of the communication chain for the encryption algorithm, we have designed new method to create a sufficiently large population without using too many synchronizing TPMs. By doing so, we can ensure a great variety of weights that can be used to create the encryption key. Thus, the encryption key can be changed quickly while encrypting a single message, making it difficult for the attacker to decrypt the encrypted message. Our next effort will be aimed to create a model of a cryptographic system that will use the resulting population.

References

1 Ido Kanter and Wolfgang Kinzel, “The Theory of Neural Networks and Cryptography,” *Quantum Computers and Computing* 5, no. 1 (2005): 130-140.

2 Andreas Ruttor, Wolfgang Kinzel, and Ido Kanter, “Dynamics of Neural Cryptography,” *Statistical, Nonlinear, and Soft Matter Physics, Physical review. E 75, 056104* (2007).

3 Wolfgang Kinzel and Ido Kanter, “Neural Cryptography,” 9th International Conference on Neural Information Processing, Singapore, vol. 3 (2002): 1351-1354.

4 S. Santhanalakshmi, K. Sangeeta, and Gopal Krishna Patra, "Design of Stream Cipher for Text Encryption Using Soft Computing based Techniques," *IJCSNS International Journal of Computer Science and Network Security* 12, no. 12 (2012): 149-152.

5 Andreas Ruttor, Wolfgang Kinzel, Rivka Naeh, and Ido Kanter, "Genetic Attack on Neural Cryptography," *Phys. Rev. E* 73.036121 (2006).

6 Martin Javurek and Michal Turčaník, "Synchronization of Two Tree Parity Machines," 2016 New Trends in Signal Processing (NTSP), Demanovska Dolina (2016): 1-4, https://doi.org/10.1109/NTSP.2016.7747782.

7 Martin Javurek and Michal Turčaník, "Synchronization Verification Improvement of Two Tree Parity Machines Using Polynomial Function," 2018 New Trends in Signal Processing (NTSP), Demanovska Dolina (2018): 1-5, https://doi.org/10.23919/NTSP.2018.8524106.

8 Michal Turcanik, "The Optimalization of the Artificial Neural Network and Production Systems by Genetic Algorithms," MATLAB 2002: Proceedings of the Conference, Prague (2002): 562-568.

9 Michal Turcanik and Martin Javurek, "Hash Function Generation by Neural Network," 2016 New Trends in Signal Processing (NTSP), Demanovska Dolina (2016): 1-5.

About the Authors

Michal **Turčaník** is an associate professor at the Department of Informatics at the Armed Forces Academy in Liptovsky Mikulas. He has been teaching different subjects for more than 20 years. He is a Panel Member of the STO IST organization for the Slovak republic. His scientific research is focusing on reconfigurable logic, artificial intelligence and computer networks.

Martin **Javurek** is an assistant professor at the Department of Informatics at the Armed Forces Academy in Liptovsky Mikulas. In 2005, he graduated (MSc) at Armed Forces Academy in Liptovsky Mikulas as a civil student. He holds a Ph.D. degree in the field of Informatics, received in 2017 from Armed Forces Academy in Liptovsky Mikulas. His scientific research is focusing on artificial intelligence and cryptology.