Research Journal of Applied Sciences, Engineering and Technology 4(21): 4458-4462, 2012
ISSN: 2040-7467

Submitted: May 01, 2012 Accepted: June 15, 2012 Published: November 01, 2012

The Evolutionary Generation of Orthomor Phisms in the Finite Field F_2^8

Haiqing Han and Xiaofang Xu
School of Mathematics and Physics, Hubei Polytechnic University, Huangshi 435003, Hubei, China

Abstract: The orthomorphisms on the finite field GF (2^8) play a important roles in the information security area, This study has obtained the orthomorphism on GF(2^8) by genetic algorithms and particle swarm to search, after the orthomorphisms are transformed into the orthomorphic polynomials, it can help us to understand the degree distribution of the polynomials.

Keywords: The finite field, the genetic algorithm, the orthomorphism, the particle swarm algorithm

INTRODUCTION

The information security has become more and more important because of Network openness and anarchy. Cryptology is one of the crucial technologies of information security. Lots of practice show that cryptology is the most basic and familiar techniques in information security. Cryptography requires people to design a safe and efficient cryptosystem. With the development of communications and computer, the design of ciphers has changed from simple handwork into computer-aided. Especially, the most ciphers are based on mathematical hard problems today, which is not conducive to the new ciphers have been designed because calculation and workload are very great by mental and manual design. Many studies indicate that the main index of a cipher against attack can be found out (Zhang *et al*., 2003; Han and Zhang, 2010). We can disassemble the cryptosystem into components, well components of cryptology can build up a cryptosystem by a certain structure, which improved the efficiency of cipher design.

Along With the progress of science and technology, people put new desires and expectations for computer science, including high-speed computing and intelligent computing. People inspired by "imitate" and "bionic" have found some good inspiration algorithms including genetic algorithms, particle swarm optimization algorithms (Mendes and Kennedy, 2004), ant colony algorithms (Ma *et al*., 2008) and so on. These algorithms can help us optimize the search by targeted manner. We can use evolutionary algorithms to help us design the cryptology parts with well character, so assembled into a secure cryptosystem. This powerful computing resources can be used today to services for the design of the cryptosystem and improve design efficiency.

Genetic algorithms, particle swarm and ant colony algorithm can be used to evolution generation and directed search according to the probability. This study has mainly searched the nonlinear orthomorphisms on the finite field F_2^8 using the evolutionary algorithms. In fact, the number of the orthomorphisms on the finite field is "combinatorial explosion" With the cardinality increasing. If you generally use exhaustive search to obtain orthomorphisms, it is not only very slow, but also difficult to find the orthomorphisms with the high nonlinearity and the small uniformity. Therefore, this study attempts to utilize the evolutionary algorithm to generate the nonlinear orthomorphisms on F_2^8, and the nonlinear orthomorphisms on the finite field F_2^8 can be directly viewed as S-boxes, the process has carried out automated design to the components of cryptosystem.

PRELIMIARIES

We let $F_2 = \{0, 1\}$ be a binary finite field, GF (2^n) is the n-degree extension field of F_2, it also can be seen as the n dimensions linear space on F_2. Sometimes GF (2^n) is also denoted by F_2^n. Let S be a bijection on F_2^n, then S is called a permutation on F_2^n.

Definition 1: Let S be a permutation on F_2^n, and I is the identity permutation on F_2^n (I (x) = x, x in F_2^n). If S+I is still the permutation on F_2^n, S is called the othomorphism on F_2^n.

Definition 1 shows that a permutation is being othomorphism if the sum between it and the identity permutation is still the permutation. When $n \geq 2$, the othomorphism on F_2^n must exist (Hall and Paige, 1957). When S is an othomorphism, h is the any linear permutation, the composite $h^{-1}Sh$ is still being an othomorphism. Further research is the following nature of othomorphisms (Lv *et al*., 2008).

Corresponding Author: Haiqing Han, School of Mathematics and Physics, Hubei Polytechnic University, Huangshi 435003, Hubei, China

Let S be a mapping on GF (2^n) → GF (2^m), that is $S:(x_1,x_2,\cdots,x_n)\mapsto(y_1,y_2,\cdots,y_m)$. For each $y_i(1\le i\le m)$ is relevant to $X=(x_1,x_2,\cdots,x_n)$, namely $y_i=s_i(X)=s_i(x_1,x_2,\cdots,x_n)$, so the mapping S is expressed as a vector function: $S=(s_1,s_2,\cdots,s_m)$, that is S(X) = (s_1 (X), s_2 (X), …, s_m (X)), where each $y_i=s_i(x_1,x_2,\cdots,x_n)$ can be viewed as a Boolean function. The mapping $S=(s_1,s_2,\cdots,s_m)$ is said the multi-output Boolean function, called Boolean functions for short. Each S-box with n bits input and m bits output in the cryptosystem can be viewed as a multi-output Boolean function. Any permutation on GF (2^n) can be seen as multi-output Boolean functions with n bits input and output.

Definition 2: If the multi-output Boolean function $S(X)=(s_1(X),s_2(X),\cdots,s_n(X))$ is a permutation on GF (2^n), each of the function $s_i(X)=s_i(x_1,x_2,\cdots,x_n)$, $(1\le i\le n)$ is Boolean function, $S(X)$ is called the Boolean permutation.

The Boolean permutation $S(X)$ is the othomorphism function S(X)+I(X) = (s_1 (X) + x_1, s_2 (X) + x_2 ,…, s_n (X) + x_n) is a Boolean permutation, if any linearly combination (the combination coefficients not all are 0) of the component function in $S(X)$ and $S(X)+I(X)$ is a balanced Boolean function (Wen *et al*., 2000). Yuan and Zhang (2007) pointed out that the othomorphism can also be represented into the permutation polynomial and there is one to one correspondence between the orthomorphic permutation polynomials and orthomorphism. Feng and Liu (1996, 1998) and Dai and Solonmen (1999) given the several generation algorithms of orthomorphisms on GF (2^n) as follows:

- Product orthomorphisms by recurrence relations
- Product orthomorphisms using the method piece orthomorphisms on F_2^k and F_2^l together orthomorphisms to get orthomorphisms on F_2^{k+l}; 3

The new Boolean permutation is obtained by directly inserting the arbitrary transformation into the orthomorphic Boolean permutation. This study has generated orthomorphisms by evolutionary algorithms.

EVOLUTIONARY COMPUTATION

This section mainly has designed the generation algorithm to product orthomorphisms on F_2^8 which possess high nonlinearity and small differential uniformity by evolutionary computation.

The genetic generation algorithm of the nonlinear orthomorphisms on F_2^8: We obtain nonlinear orthomorphisms possessed high nonlinearity and small differential uniformity through the orthomorphism on F_2^8 is regarded as the evolutionary object. Based on genetic algorithm to carry on the evolution design, the steps of algorithm are as follows.

Coding strategy: A chromosome is represented an orthomorphism of on F_2^8 and the orthomorphism is expressed into the Boolean permutation, namely $F(x_1,x_2,\cdots,x_8)=(f_1,f_2,\cdots,f_8)$ is a chromosome, each Boolean function component represents a gene (Sometimes the truth table is also used to represent the chromosome).

The initial population: First search all orthomorphisms on the finite fields F_2^2, F_2^3, F_2^4, respectively. All orthomorphisms are represented into Boolean permutations, obtained orthomorphisms on F_2^8 by way of the low-order orthomorphisms assembling into higher-order orthomorphisms. The splicing method is as follows:

"2+3+3": denote that the orthomorphisms on F_2^8 split joint by a orthomorphic Boolean permutation on F_2^2 and two orthomorphic Boolean permutations on F_2^3; "4+4": denote that the orthomorphisms on F_2^8 split joint by two orthomorphic Boolean permutations on F_2^4; "2+2+4": denote that the orthomorphisms on F_2^8 split joint by one orthomorphic Boolean permutation on F_2^4 and two orthomorphic Boolean permutations on F_2^2.

Of course, there are other methods to split joint and we have randomly chosen method to split joint such as "4+4". Determine the size of population is 16. The experiment used "4+4" combination to generate because of this initial population has been selective and easy to be initialized.

The fitness function: We consider the linear and differential attacks for the easy design to experiment. The smaller the differential uniformity is the stronger the ability to fight differential attack and the larger the non-linearity is the stronger against linear attack, so the fitness function is defined as:

$$f(x)=32N_F-15\delta_F+1/g$$

where, N_F indicated the nonlinearity of the Boolean function F, δ_F is the differential uniformity of the

Boolean function, g is the generation number of the evolution algorithm. ($1/g$ makes the offspring inferior to the parents when the individual has the same the differential uniformity and the nonlinearity).

The evolution: The evolution strategies is the form of $\mu+\lambda$, that is each generation must wash out λ chromosomes by the evaluation of the fitness function after produce λ new individuals joined to the population. The evolution operator is as follows:

- **The crossover operator:** Choose two arbitrary chromosomes $F=(F_1,F_2)$ and $G=(G_1,G_2)$, exchange of gene fragments and get $F=(G_1,F_2)$, $G=(F_1,G_2)$. The probability of the hybrid to two chromosomes is determined by the roulette.
- **The mutation operator:**
  - Arbitrary choose a chromosome $F=(F_1,F_2)$, exchange two component functions before and after on F_2^4, then the permutation after the exchange should be regarded as orthomorphisms on F_2^8.
  - Arbitrary choose a transformation on F_2^4 (not must be the permutation) as a gene chip and insert it into the chromosome to obtain $F=(F_1+H,F_2)$ or $F=(F_1,F_2+H)$, the mutation probability is determined by the roulette.
- **The selection operator:** In each generation of the evolution, the chromosomes with the dominant gene (viz. the fitness value is bigger) must be retained into the next generation involving in the evolution. We have some orthomorphisms on F_2^8 by several times Evolutionary computation. Furthermore, we have calculated the index of the nonlinearity and differential uniformity are same optimal individuals. Select the permutation that the comprehensive indexes are optimal to design the S-box.

The experiment shows that the nonlinearity and the differential uniformity of the orthomorphisms on F_2^8 by Evolutionary computation spliced joint two orthomorphisms on F_2^4 cannot be superior to the S-boxes in AES. If you want the nonlinear orthomorphism with better cryptologic character should consider improving the initial population and evolutionary strategies.

The particle swarm optimization algorithm of the nonlinear orthomorphisms on F_2^8: This section mainly has designed the generation algorithm by the particle swarm algorithm. A orthomorphism on F_2^8 represents a particle and the location of a particle, the coding strategy is same as the evolutionary computation. The specific steps are as follows:

- **Coding strategy:** A particle represents orthomorphism on F_2^8 and the orthomorphisms is expressed into Boolean permutations. Namely $F(x_1,x_2,\cdots,x_8)=(f_1,f_2,\cdots,f_8)$ means a particle, which each component Boolean function represents a coordinate position. Sometimes, the particles are expressed into a form of the truth table. The representation of particles is interchangeable to obtain the nonlinear orthomorphism with good cryptographic properties by the exercise.
- **The initial particles:** Choose arbitrary 32 orthomorphisms on F_2^4, which assemble into 16 orthomorphisms on F_2^8, the size of the initial particle swarm is 16.
- **The fitness function:** The Fitness function is as follows:

$$f(x)=32N_F-15\delta_F+1/g$$

where, N_F is the nonlinearity and δ_F is the differential uniformity of the Boolean function F, g is the generation number of the particle swarm algorithm. The fitness function can also be adjusted appropriately according to the results of the particle swarm algorithm.

The method of particles motion: First calculate the individual fitness value of the initial particle swarm, let particles with the largest fitness value do not move, other particles are subject to movement. The motion states ensure the size of the particle swarm is same, calculate the fitness value of the particle after a sport. The particles locate on the overall best state and personal best (fitness achieves to the maximum value) position in the movement not to move, but the particles located on the other position should move by the some probability which is the certain value in sometimes.

The retention of the particles' motion state: Each particle's fitness value is compared to the past after the movement, if their fitness value is increasing then the movement position was retained; if the fitness value is not greater than the past then it participate in the new movement depending on the certain probability. The advantage of this reservation is not wash out of the inferior particles in each generation.

Table 1: The parameter by the evolutionary generation

The genetic algorithm		The particle swarm algorithm	
Population size	16	Particle size	16
The crossover probability	0.03	Movement probability	0.017
The mutation probability	0.08		
Generation number	2000	Movement frequency	2000
Differential uniformity	12	Differential uniformity	10
Nonlinearity	96	Nonlinearity	96
Degree of permutation polynomials	252	Degree of permutation polynomials	251

- **The movement pattern of particles:** Particle motion by two ways:
  - Particle's self-mutate, the mutation is same as the evolution (exchange position and insert the transformation)
  - Particles have occurred by similar transformation, which generates a random 8 square invertible matrix and finds $A^{-1}FA$.
- **The probability of particle motion:** The other particles should move in addition to the individual particles whose fitness value is the best in the overall or personal. For carrying on the easy programming, the movement probability is setup at 0.017.

In common with the genetic algorithms, we have some orthomorphisms on F_2^8 based on particle swarm optimization to experiment over and over. Furthermore, we have calculated the index of the nonlinearity and differential uniformity; select the permutation that the comprehensive indexes are optimal to design the S-box.

The cryptographic properties of orthomorphisms produced by particle swarm algorithm is not greatly improved in comparison with the genetic computation, it is to consider other ways to get the initial particle swarm or in combination with other intelligent algorithms to improve the search capabilities of PSO.

THE ANALYSIS OF EXPERIMENTAL RESULTS

The orthomorphism generated the genetic algorithms experiment is as follows: 58, 56, 49, 141, 174, 48, 71, 90, 21, 173, 236, 203, 170, 89, 169, 175, 74, 93, 65, 214, 25, 138, 215, 51, 21, 22, 32, 207, 147, 218, 180, 97, 119, 59, 104, 75, 149, 27, 107, 139, 144, 179, 109, 101, 181, 117, 106, 158, 78, 130, 136, 229, 126, 193, 216, 255, 140, 132, 247, 54, 198, 168, 128, 123, 116, 110, 200, 84, 61, 38, 224, 182, 77, 242, 45, 190, 19, 32, 192, 201, 7, 127, 88, 0, 86, 150, 100, 87, 188, 114, 64, 199, 83, 129, 204, 223, 191, 133, 137, 161, 164, 50, 98, 55, 172, 52, 230, 24, 222, 9, 253, 57, 63, 43, 70, 184, 44, 82, 217, 35, 220, 68, 103, 213, 211, 91, 142, 69, 145, 79, 81, 99, 245, 254, 183, 112, 29, 20, 160, 237, 243, 30, 240, 162, 66, 5, 178, 225, 108, 111, 16, 231, 53, 228, 118, 148, 3, 47, 113, 233, 28, 163, 120, 26, 13, 221, 37, 131, 124, 42, 115, 244, 125, 146, 185, 152, 196, 197, 121, 17, 171, 153, 105, 60, 135, 186, 194, 177, 67, 156, 210, 31, 11, 34, 40, 33, 36, 46, 248, 85, 80, 23, 205, 219, 122, 134, 41, 239, 73, 208, 250, 94, 167, 12, 187, 2, 92, 10, 251, 76, 157, 15, 176, 206, 1, 252, 95, 241, 246, 96, 209, 166, 165, 18, 143, 22, 195, 227, 235, 249, 159, 226, 8, 202, 4, 238, 234, 39, 72, 6, 154, 62, 155, 14, 151, 189, 102.

The above data manifested that $\sigma(0) = 58$, $\sigma(1) = 56, \cdots, \sigma(255) = 102$. The above data represented the elements in F_2^8, as 58 = (00111010), $\cdots$, 102 = (01100110).

The orthomorphism to express by the data above has the parameters as the Table 1. One of the orthomorphisms by PSO experiment is as follows: 138, 146, 148, 113, 203, 224, 209, 111, 61, 127, 86, 163, 141, 13, 117, 123, 252, 97, 244, 15, 247, 250, 195, 46, 242, 45, 182, 60, 235, 255, 135, 254, 172, 70, 11, 74, 231, 110, 34, 164, 216, 44, 173, 94, 155, 124, 142, 139, 73, 246, 22, 43, 119, 176, 128, 183, 2, 237, 32, 154, 3, 159, 204, 150, 220, 238, 171, 143, 55, 52, 219, 38, 210, 5, 221, 80, 211, 223, 213, 222, 196, 169, 212, 104, 101, 71, 72, 122, 77, 79, 186, 75, 65, 68, 243, 78, 102, 156, 95, 81, 87, 59, 69, 188, 131, 7, 205, 152, 90, 181, 240, 214, 29, 57, 54, 58, 215, 42, 56, 49, 50, 251, 6, 208, 160, 105, 64, 67, 236, 130, 230, 37, 180, 8, 62, 225, 121, 17, 1, 228, 187, 239, 229, 217, 84, 25, 35, 202, 199, 192, 88, 39, 82, 126, 63, 151, 193, 207, 129, 191, 136, 96, 12, 248, 218, 83, 158, 134, 233, 116, 30, 165, 106, 115, 4, 137, 253, 185, 112, 184, 206, 144, 125, 100, 76, 93, 189, 14, 179, 194, 149, 190, 201, 9, 167, 10, 198, 0, 28, 147, 197, 177, 157, 140, 48, 226, 23, 227, 108, 178, 92, 145, 103, 168, 107, 249, 98, 153, 85, 27, 99, 170, 51, 245, 91, 31, 232, 26, 132, 16, 241, 162, 18, 166, 200, 89, 19, 120, 21, 133, 24, 161, 118, 33, 36, 175, 174, 40, 53, 234, 20, 114, 66, 109, 47, 41.

The above data indicated that $\sigma(0) = 138$, $\sigma(1) = 146, \cdots, \sigma(255) = 41$. The orthomorphism to express by the data above has the parameters as the Table 1 too.

Table 2: The algebraic degree distribution of the orthomorphic polynomials on F_2^8

Inexistence degree	2, 3, 5, 15, 17, 51, 85, 127, 254, 255	Nonlinear
Existence degree	252, 250, 249, 248, 224, 208	Nonlinear
	1, 4, 32, 64, 128	Linear
Other degree		Unknown

The orthomorphic polynomials of the algebraic degree 252 have emerged most frequently in the results by the experiment in 3.1 and 3.2, we have guessed that the orthomorphic polynomials of the degree 252 is likely to the most greatest amount by the maximum likelihood estimation. We have already received the orthomorphic polynomials with the algebraic degree 208, 224, 248, 249, 250 in the experiments, also found the linear orthomorphic polynomials with the degree 32, 64, 128. So the degree distribution of the orthomorphic polynomials on F_2^8 is as shown in the Table 2.

CONCLUSION

This study has mainly studied the evolution generation algorithm to generate the orthomorphism on F_2^8. Our experiments have obtained the orthomorphism to design S-boxes that the cryptographic properties is slightly worse in comparison with AES, which shows the evolutionary computation and initial population structure need to make a progress. On the other hands, we have got hold of the orthomorphic polynomials with different algebraic degree, what have built up the good foundation for in-depth study.

We consider that the differential characteristic with high probability based on the ant colony algorithm to search them. The feasibility test should be a further research. The experiment is waiting for search the differential characteristics by the ant colony algorithm.

ACKNOWLEDGMENT

This study was supported in part by Hubei Polytechnic University (11YJZ10R) under Grant Nos. 801-8852.

REFERENCES

Dai, Z. and W.G. Solonmen, 1999. Generating all linear orthemorphisms without repetition. Discret. Math., 5: 47-55.

Feng, D. and Z. Liu, 1996. On the construction of the orthomorphic permutation. Secure Commun., 2: 61-64(Ch).

Feng, D. and Z. Liu, 1998. An Iterated Method of Constructing Orthomorphic Permutation. DCS Center P.O. Box 3908, Beijing, 100039, 2: 53-54(Ch).

Hall, M. and L.J. Paige, 1957. Complete mappings of finite groups. Pac. J. Math., 5: 541-549.

Han, H. and H. Zhang, 2010. The generation algorithm of a sort of p-permutations. Wuhan U. J. Nature Sci., 15: 237-241(Ch).

Lv, S. and X. Fan etc., 2008. Complete Mapping and Application in Cryptography [M]. China University of Technology, Hefei, (Ch).

Ma, L., G. Zhu and A. Ning, 2008. Ant Colony Optimization [M]. Science Press, Beijing.

Mendes, R. and J. Kennedy, 2004. The full informed particle swarm: Simpler, maybe better. IEEE T. Evol. Comput., 8: 204-210.

Wen, Q., X. Niu and Y. Yang, 2000. The Boolean Function in Modern Cryptogranphy [M]. Science Press, Beijing, (Ch).

Yuan, Y. and H. Zhang, 2007. A Note on orthomorphic permutation polynomial. J. Wuhan U. Nat. Sci. Ed., 53(1): 33-36(Ch).

Zhang, H.G., X.T. Feng, Z.P. Qin and Y.Z. Zhen, 2003. Research on evolutionary cryptosystems and evolutionary DES. Chinese J. Comput., 26: 1678-1684(Ch).