Journal of Computers Vol. 30 No. 5, 2019, pp. 111-127
doi:10.3966/199115992019103005009

Multi-offspring Genetic Algorithm with Two-point Crossover and the Relationship between Number of Offsprings and Computational Speed

Jiquan Wang[1*], Zhiwen Cheng[1], Okan K. Ersoy[2], Panli Zhang[1], Weiting Dai[1]

[1] College of Engineering, Northeast Agricultural University, No. 600 ChangJiang Road, Harbin, Heilongjiang, China
wangjiquan@neau.edu.cn, {chang1993621, PPenelope, dv15663580893}@163.com

[2] Purdue University, School of Electrical and Computer Engineering West Lafayette, Indiana 47907-1285
ersoy@purdue.edu

Received 23 November 2017; Revised 31 March 2018; Accepted 2 June 2018

Abstract. This paper presents a multi-offspring genetic algorithm (MGA) with two-point crossover in accordance with biology and mathematical ecological theory. For the MGA, the main existing problems are generation methods of multi-offsprings with different crossover methods, the best number of offsprings and the influence of the number of offsprings on the speed of computation. To solve these problems, the paper first studies the relationship between the number of offsprings and the computational speed of the MGA with two-point crossover. Furthermore, the relationship between the generation method of multi-offsprings, the number of offsprings and the computational speed is analyzed. The results with ten test functions show that when the number of offsprings generated by the MGA based on two-point crossover equals 6, the MGA with two-point crossover has significantly improved the computational speed and reduced the number of iterations as compared to the basic genetic algorithm (BGA) and the MGA of single-point crossover.

Keywords: computing speed, multi-offspring genetic algorithm, mutation, offspring individual quantity, two-point crossover

1 Introduction

Genetic algorithm (GA) is a random global search optimization technology based on Darwin's natural evolution theory and Mendel's genetics and mutation theory [1]. GA was proposed by Professor John H. Holland and his student at Michigan University in the late 1960s and early 1970s [2-6]. De Jong proposed an elitist reserved evolutionary strategy in his doctoral thesis in 1975, and later proposed a variety of evolutionary strategies of elitist retention and selection instead of copying [7-10]; currently, GA usually utilizes this evolutionary strategy.

In recent years, GA has attracted more attention because of its unique and superior performance. Many scholars have conducted in-depth studies on GA and proposed improved algorithms, such as hierarchical GA, CHC algorithm, messy GA, self-adaptive GA, GA based on niche technology, hybrid GA, and parallel GA [11-17]. In these studies, common approaches are two parent individuals generating two offsprings, multiple parents generating two offsprings [18-21], and one parent generating one offspring [22]. Thus, the number of offsprings is less than or equal to the number of parents. When the crossover probability equals 1, the number of offsprings equals that of parents; when the crossover probability is less than 1, the number of offsprings is less than that of parents [23-25]. This is unlike the situation in the biosphere for animals and plants for survival in nature. In recent years, some scholars have proposed the concept of multi-offspring genetic algorithm [26-27]. Reference [26] showed the advantages of MGA by

* Corresponding Author

applying the schema theorem, given the generation method of multi-offsprings based on single-point crossover. Reference [27] discusses the generation method of multi-offsprings based on single point crossover. In these references, there is neither a discussion of the relationship between the number of offsprings and the computational speed nor the estimation of the optimal number of offsprings.

To solve these problems, this paper proposes MGA with two-point crossover motivated by biology and mathematical ecology theory. A generation method of multi-offsprings based on MGA with two-point crossover, the relationship between the number of offsprings and computational speed, and the estimation of the optimal number of offsprings are discussed. Results with ten test functions show that MGA with two-point crossover has significantly reduced the average number of iterations and the average computational time as compared to BGA and MGA based on single-point crossover.

2 Definition and Its Analysis of MGA

MGA and BGA are defined as follows:

Definition 1. If the number of offsprings generated by GA iteration is more than the number of parents, then the GA is a MGA.

Definition 2. If the number of offsprings generated by GA iteration is less than or equal to the number of parents, then the GA is a BGA.

In MGA, the number of offsprings is an integral multiple of the number of parents in general. If the number of parents is *n,* and the number of offsprings is n_1, the relationship between the number of offsprings and the number of parents can be expressed as

$$n_1 = \beta n \quad \beta \in \{2,3,4,\cdots\} . \tag{1}$$

Since the number of offsprings of MGA is more than that of BGA, survival pressure of MGA is larger than that of BGA within populations. Based on the principle of survival of the fittest, winning individuals survive and failed individuals are eliminated to maintain constant population size. More intense competition with MGA results in faster computational speed as compared to BGA.

3 Theoretical Foundation of MGA

3.1 Biological Theory Foundation

Life on earth began as a result of a lengthy evolutionary process (e.g., simple to complex and disordered to ordered). Many explanations for biological evolution have been proposed, and Darwin's theory of evolution is widely accepted [28]. Natural selection theory of biological evolution includes the following factors: high fertility rate, struggle for existence, indeterminate mutation, and survival of the fittest. The number of born individuals is remarkably more than the number of surviving individuals. High fertility rate is a common phenomenon in the biosphere. For example, a plant may bear one thousand seeds each year, however, the number of fruit seeds is less than one thousand. Thus, two parents often produce more than two offsprings in the biosphere. Such species in the evolutionary process can both inherit parent characteristics (i.e., more chances for survival in a complex, ever-changing natural environment), and drive the continuous evolution of the biological species, thus producing more excellent individuals. By contrast, a species in which two parents produce less than or equal to two offsprings does not exist; even if it does exist, such species may eventually become extinct because of infertility, low fertility rate, diseases, food, water, intraspecific and interspecific competition, and many other factors.

Darwin's theory of evolution shows that a variety of biological organisms on earth universally have a strong ability to reproduce and a tendency to increase in accordance with the geometric ratio, that is, excessive reproduction is the basis of biological evolution. However, biological survival is limited by food and space. A species must fight for survival. In the process of biological evolution in which the struggle for existence is widespread, individuals with lower survival ability are eliminated, and excellent individuals with higher survival ability survive. Thus, the number of surviving offsprings is less than that of offsprings cross-generated by two parents. MGA proposed is based on such biological principles.

3.2 Mathematical Ecological Theory Foundation

To illustrate the probability of species extinction, we suppose that one species has only one individual at first. Then, at a certain time *t*, the probability of population size equaling 0 is given by

$$p_0(t \mid i=1) = \frac{\mu e^{(\lambda-\mu)t} - \mu}{\lambda e^{(\lambda-\mu)t} - \mu}. \tag{2}$$

where *i* is the size of initial population, μ is the mortality rate, and λ is the reproduction rate.

The probability for the population whose initial size equals *i* to be extinct with time is given by

$$p_0(t) = [p_0(t \mid i=1)]^i = \left(\frac{\mu e^{(\lambda-\mu)t} - \mu}{\lambda e^{(\lambda-\mu)t} - \mu}\right)^i. \tag{3}$$

As t tends to infinity, there are three situations as follows:

(1) When the reproduction rate is less than the mortality rate, i.e., $\lambda < \mu$, the exponential term in Eq. (2) would tend to 0 as $t \to \infty$, resulting in

$$\lim_{t\to\infty} p_0(t) = 1. \tag{4}$$

The extinction probability of a population is equal to 1. Hence, the species must become extinct eventually.

(2) When the reproduction rate is greater than the mortality rate, i.e., $\lambda > \mu$, Eq. (2) as t tends to ∞ can be represented as follows:

$$p_0(t) \to \left(\frac{\mu e^{(\lambda-\mu)t}}{\lambda e^{(\lambda-\mu)t}}\right)^i = \left(\frac{\mu}{\lambda}\right)^i. \tag{5}$$

According to Eq. (4), such population cannot guarantee continual existence because there is still a finite probability of extinction. However, if the reproduction rate is much greater than the mortality rate, and if the initial population size is much larger, then the probability of biological extinction will be smaller.

(3) When the reproduction rate is equal to the mortality rate, i.e., $\lambda = \mu$, Eq. (2) can be expanded in a series of exponential terms. Letting $\lambda - \mu = r$, $p_0(t)$ as t tends to ∞ can be written as

$$p_0(t) = \left[\frac{\mu(rt + r^2t^2/2! + \cdots)}{\lambda(1 + rt + r^2t^2/2! + \cdots) - \mu}\right]^i. \tag{6}$$

When $r \to 0$, ignoring r^2, and due to $\lambda - \mu = r$, we get

$$p_0(t) \to \left[\frac{\mu rt}{(\lambda-\mu) + \lambda rt}\right]^i \to \left(\frac{\lambda t}{1+\lambda t}\right)^i. \tag{7}$$

Hence,

$$\lim_{t\to\infty}\left(\frac{\lambda t}{1+\lambda t}\right)^i = 1\cdot \tag{8}$$

When the reproduction rate is equal to the mortality rate, Eq. (7) proves that the species must become extinct eventually. Although the expectation size of population is constant, the species will be extinct after a sufficiently long time while randomly fluctuating around the population expected size. Only when $\lambda > \mu$ (i.e., when population has a positive increase in rate) will the population may survive forever (not necessarily). Moreover, mathematical ecology indicates that the probability distribution of biological population size depends on the product of reproduction rate and time when the biological initial population size is known. Thus, a high reproductive rate for a shorter time and a low reproductive rate for a long time, as long as each individual increase rate equals the product of time, would provide the same results. Therefore, to get more excellent individuals in the shortest possible time, the reproduction rate of species must be improved.

In conclusion, MGA has not only biological foundation, but also mathematical ecological foundation. Therefore, MGA is feasible in theory.

4 MGA of Two-point Crossover

The basic operations of MGA and BGA include selection, crossover, and mutation. The main differences between MGA of two-point crossover and BGA are due to the differences in crossover and mutation operations. For BGA, two parent individuals generate two offspring individuals by crossover; however, for MGA, two parent individuals generate two or more offspring individuals by crossover. Since the number of cross-generated offsprings in MGA is higher, the number of mutation operations is also higher. In MGA, to keep the population size constant, worse individuals are eliminated, and n excellent individuals are preserved in accordance with the principle of survival of the fittest.

4.1 Selection

Let the population size be n, and the individuals in the population be expressed as $X(t)=(X_1(t), X_2(t), \ldots, X_i(t), \ldots, X_n(t))$, wherein $X_i(t)=(x_{i1}(\mathrm{t}), x_{i2}(t), \ldots, x_{id}(t))$, t is the iteration number. The individuals are sorted in descending order according to objective function values, yielding $\bar{X}(t)=(\bar{X}_1(t), \bar{X}_2(t), \cdots, \bar{X}_i(t), \cdots, \bar{X}_n(t))$. Suppose $\beta\in(0, 1)$, and the fitness of $\bar{X}_i(t)$ is $eval(\bar{X}_i(t))$. The fitness value of $\bar{X}_i(t)$ is computed as follows [14]:

$$eval(\bar{X}_i(t)) = \beta(1-\beta)^{i-1} \quad i = 1, 2, \cdots, n. \tag{9}$$

where $\beta \in (0,1)$ is a constant, usually chosen between 0.01 and 0.3.

Then, the roulette wheel method is used to pair members [29]. The roulette wheel method is as follows:

The selection probability of the i^{th} member in the population is given by

$$P_i = \frac{eval(\bar{X}_i(t))}{\sum_{i=1}^{n} eval(\bar{X}_i(t))}. \tag{10}$$

Letting

$$PP_0 = 0. \tag{11}$$

$$PP_i = \sum_{j=1}^{i} P_i, \quad i = 1, 2, \cdots, n. \tag{12}$$

The roulette wheel is rotated up to n times, and a random number $\eta_k\in(0, 1)$ is generated at each rotation. When this random number satisfies $PP_{i-1} \le \eta_k \le PP_i$, the i^{th} member is selected to take part in crossover.

4.2 Crossover Operation

To study the relationship between the number of offsprings and the computational speed of MGA, a method for two parents to generate a number of offsprings by crossover is given below.

Suppose there are two parents P_1 and P_2 selected to take part in crossover. Two crossover points are randomly generated, and P_1 and P_2 are divided into three sections. That is, $P_1 = D_1E_1F_1$, $P_2=D_2E_2F_2$, also D_1 and D_2, E_1 and E_2, as well as F_1 and F_2 contain the same number of binary digits. The method for two parents to generate a number of offsprings by crossover is as follows:

Two parents generating two offsprings by crossover. E_1 and E_2 exchange location to generate two offspring individuals C_1 and C_2 ($C_1=D_1E_2F_1$, $C_2=D_2E_1F_2$); the specific method is shown in Table 1.

Table 1. Two parent generate two offsprings by crossover

Parents			Offsprings		
	CP1	CP2		CP1	CP2
P_1=D_1	E_1	F_1	C_1=D_1	E_2	F_1
P_2=D_2	E_2	F_2	C_2=D_2	E_1	F_2

Two parents generating four offsprings by crossover. The generation method of C_1 and C_2 is the same as that in (1). D_1 and D_2 exchange location to generate two offspring individuals C_3 and C_4 (C_3=$D_2E_1F_1$, C_4=$D_1E_2F_2$). The specific method is shown in Table 2.

Table 2. Two parents generate four offsprings by crossover

Parents			Offsprings					
	CP1	CP2		CP1	CP2		CP1	CP2
P_1=D_1	E_1	F_1	C_1=D_1	E_2	F_1	C_3=D_2	E_1	F_1
P_2=D_2	E_2	F_2	C_2=D_2	E_1	F_2	C_4=D_1	E_2	F_2

Two parents generating six offsprings by crossover. The generation method of C_1, C_2, C_3, and C_4 is that same as that in (2). F_1 is placed between D_2 and E_2, F_2 is placed between D_1 and E_1, to obtain C_5 and C_6 (C_5=$D_1F_2E_1$, C_6=$D_2F_1E_2$). The specific method is shown in Table 3.

Table 3. Two parents generate six offsprings by crossover

Parent individuals			Offspring individuals					
	CP1	CP2		CP1	CP2		CP1	CP2
P_1=D_1	E_1	F_1	C_1=D_1	E_2	F_1	C_4=D_1	E_1	F_2
P_2=D_2	E_2	F_2	C_2=D_2	E_1	F_2	C_5=D_1	F_2	E_1
			C_3=D_2	E_1	F_1	C_6=D_2	F_1	E_2

Two parents generating eight offsprings by crossover. The generation method of C_1, C_2, C_3, C_4, C_5, and C_6 is the same as that in (3). E_2 is placed behind D_1 and F_1, E_1 is placed behind D_2 and F_2, to obtain C_7 and C_8 (C_7=$D_1F_1E_2$, C_8= $D_2F_2E_1$); the specific method is shown in Table 4.

Table 4. Two parents generate eight offsprings by crossover

Parents			Offsprings					
	CP1	CP2		CP1	CP2		CP1	CP2
			C_1=D_1	E_2	F_1	C_5=D_1	F_2	E_1
P_1=D_1	E_1	F_1	C_2=D_2	E_1	F_2	C_6=D_2	F_1	E_2
P_2=D_2	E_2	F_2	C_3=D_2	E_1	F_1	C_7=D_1	F_1	E_2
			C_4=D_1	E_2	F_2	C_8=D_2	F_2	E_1

Two parents generating 10 offspring individuals by crossover. The generation method of C_1, C_2, C_3, C_4, C_5, C_6, C_7, and C_8 is the same as that in (4). F_2 is placed behind D_1 and E_1, F_1 is placed behind D_2 and E_2, to obtain C_9 and C_{10} (C_9= $D_1E_1F_2$, C_{10}= $D_2E_2F_1$); the specific method is shown in Table 5.

Table 5. Two parents generate 10 offsprings by crossover

Parents			Offsprings					
	CP1	CP2		CP1	CP2		CP1	CP2
			C_1=D_1	E_2	F_1	C_6=D_2	F_1	E_2
			C_2=D_2	E_1	F_2	C_7=D_1	F_1	E_2
P_1=D_1	E_1	F_1	C_3=D_2	E_1	F_1	C_8=D_2	F_2	E_1
P_2=D_2	E_2	F_2	C_4=D_1	E_2	F_2	C_9=D_1	E_1	F_2
			C_5=D_1	F_2	E_1	C_{10}=D_2	E_2	F_1

Two parents generating 12 offsprings by crossover. The generation method of C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8, C_9, and C_{10} is the same as that in (5). F_2 is placed before D_1 and E_1, F_1 is placed before D_2 and E_2, to obtain C_{11} and C_{12} (C_{11}= $F_2D_1E_1$, C_{12}= $F_1D_2E_2$). The specific method is shown in Table 6.

Table 6. Two parents generate 12 offsprings by crossover

Parents			Offsprings					
CP1	CP2		CP1	CP2		CP1	CP2	
			$C_1=D_1$	E_2	F_1	$C_7=D_1$	F_1	E_2
			$C_2=D_2$	E_1	F_2	$C_8=D_2$	F_2	E_1
$P_1=D_1$	E_1	F_1	$C_3=D_2$	E_1	F_1	$C_9=D_1$	E_1	F_2
$P_2=D_2$	E_2	F_2	$C_4=D_1$	E_2	F_2	$C_{10}=D_2$	E_1	F_1
			$C_5=D_1$	F_2	E_1	$C_{11}=F_2$	D_1	E_1
			$C_6=D_2$	F_1	E_2	$C_{12}=F_1$	D_2	E_2

Two parents generating fourteen offspring individuals by crossover. The generation method of C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8, C_9, C_{10}, C_{11}, and C_{12} is the same as that in (6). E_2 is placed before D_1 and F_1, E_1 is placed before D_2 and F_2, to obtain C_{13} and C_{14} ($C_{13}= E_2D_1F_1$, $C_{14}= E_1D_2F_2$). The specific method is shown in Table 7.

Table 7. Two parents generate 14 offsprings by crossover

Parents			Offsprings					
CP1	CP2		CP1	CP2		CP1	CP2	
			$C_1=D_1$	E_2	F_1	$C_8=D_2$	F_2	E_1
			$C_2=D_2$	E_1	F_2	$C_9=D_1$	E_1	F_2
			$C_3=D_2$	E_1	F_1	$C_{10}=D_2$	E_2	F_1
$P_1=D_1$	E_1	F_1	$C_4=D_1$	E_2	F_2	$C_{11}=F_2$	E_1	E_1
$P_2=D_2$	E_2	F_2	$C_5=D_1$	F_2	E_1	$C_{12}=F_1$	D_2	E_2
			$C_6=D_2$	F_1	E_2	$C_{13}=E_2$	D_1	F_1
			$C_7=D_1$	F_1	E_2	$C_{14}=E_1$	D_2	F_2

In Table 1 to Table 7, C_i (i=1, 2, …, 14) is cross-generated with the i^{th} offspring individual.

A specific example is given below to illustrate the multi-offspring generation method. Suppose there are two parents P_1 and P_2 selected to take part in crossover. Their binary number representations are P_1 = 10101011101110 and P_2 = 01010010100011. Two different crossover points are randomly generated as 5 and 11.

	Crossover point 1	Crossover point 2
P_1=10101	011101	110
P_2=01010	010100	011

Letting D_1=10101, E_1=011101, F_1=110, D_2=01010, E_2=010100, and F_2=011, crossover operations are performed with P_1 and P_2 according to the method in Table 1. The resulting offsprings are shown in Table 8.

Table 8. Two parents generate two offsprings by crossover

Parents			Offsprings		
CP1	CP2		CP1	CP2	
$P_1=D_1$	E_1	F_1	C_1=10101	010100	110
$P_2=D_2$	E_2	F_2	C_2=01010	011101	011

P_1 and P_2 perform crossover operation according to the method of Table 2. The resulting offsprings individuals are shown in Table 9.

Table 9. Two parents generate four offsprings by crossover

Parents			Offsprings					
	CP1	CP2		CP1	CP2		CP1	CP2
P_1=D_1	E_1	F_1	C_1=10101	010100	110	C_3=01010	011101	110
P_2=D_2	E_2	F_2	C_2=01010	011101	011	C_4=10101	010100	011

P_1 and P_2 perform crossover operation according to the method of Table 3; the resulting offsprings individuals are shown in Table 10.

Table 10. Two parents generate six offsprings by crossover

Parents			Offsprings					
	CP1	CP2		CP1	CP2		CP1	CP2
P_1=D_1	E_1	F_1	C_1=10101	010100	110	C_4=10101	010100	011
P_2=D_2	E_2	F_2	C_2=01010	011101	011	C_5=10101	011	011101
			C_3=01010	011101	110	C_6=01010	110	010100

P_1 and P_2 perform crossover operation according to the method of Table 4. The resulting offsprings individuals are shown in Table 11.

Table 11. Two parents generate eight offsprings by crossover

Parents			Offsprings					
	CP1	CP2		CP1	CP2		CP1	CP2
P_1=D_1	E_1	F_1	C_1=10101	010100	110	C_5=10101	011	011101
P_2=D_2	E_2	F_2	C_2=01010	011101	011	C_6=01010	110	010100
			C_3=01010	011101	110	C_7=10101	110	010100
			C_4=10101	010100	011	C_8=01010	011	011101

P_1 and P_2 perform crossover operation according to the method of Table 5. The resulting offsprings are shown in Table 12.

Table 12. Two parents generate 10 offsprings by crossover

Parents			Offsprings					
	CP1	CP2		CP1	CP2		CP1	CP2
P_1=D_1	E_1	F_1	C_1=10101	010100	110	C_6=01010	110	010100
P_2=D_2	E_2	F_2	C_2=01010	011101	011	C_7=10101	110	010100
			C_3=01010	011101	110	C_8=01010	011	011101
			C_4=10101	010100	011	C_9=10101	011101	011
			C_5=10101	011	011101	C_{10}=01010	010100	110

P_1 and P_2 perform crossover operation according to the method of Table 6. The resulting offsprings are shown in Table 13.

Table 13. Two parents generate 12 offsprings by crossover

Parents			Offsprings					
	CP1	CP2		CP1	CP2		CP1	CP2
P_1=D_1	E_1	F_1	C_1=10101	010100	110	C_7=10101	110	010100
P_2=D_2	E_2	F_2	C_2=01010	011101	011	C_8=01010	011	011101
			C_3=01010	011101	110	C_9=10101	011101	011
			C_4=10101	010100	011	C_{10}=01010	010100	110
			C_5=10101	011	011101	C_{11}=011	10101	011101
			C_6=01010	110	010100	C_{12}=110	01010	010100

P_1 and P_2 perform crossover operation according to the method of Table 7. The resulting offspring individuals are shown in Table 14.

Table 14. Two parents generate 14 offsprings by crossover

Parents			Offsprings					
	CP1	CP2		CP1	CP2		CP1	CP2
			C_1=10101	010100	110	C_8=01010	011	011101
			C_2=01010	011101	011	C_9=10101	011101	011
			C_3=01010	011101	110	C_{10}=01010	010100	110
P_1=D_1	E_1	F_1	C_4=10101	010100	011	C_{11}=011	10101	011101
P_2=D_2	E_2	F_2	C_5=10101	011	011101	C_{12}=110	01010	010100
			C_6=01010	110	010100	C_{13}=010100	10101	110
			C_7=10101	110	010100	C_{14}=011101	01010	011

4.3 Mutation Operation

All cross-generated offsprings undergo mutation in MGA and BGA. Suppose the mutation probability is P_m, $X_i = (x_{i1}, x_{i2}, \cdots x_{ij}, \cdots, x_{id})$ is the i^{th} individual to be mutated, and x_{ij} is the j^{th} component of the i^{th} individual. Such a component is mutated as follows: A 0-1 uniformly distributed random number is generated. If the random number is less than or equal to the mutation probability P_m, this bit is mutated, otherwise, it is not be mutated. Thus, if a certain bit is 0, then the bit is 1 becomes 1 if mutated and vice versa.

4.4 Evolutionary Strategy

4.4.1 Evolutionary Strategy of MGA

The evolutionary strategy of MGA is as follows. First, the initial population is generated to become the first parent generation. All the members of the population are sorted in descending order according to their calculated objective function values (if objective function solves maximum). Then, selection and crossover operations are performed. Thirdly, retaining s elite individuals and n excellent individuals from n parent individuals and cross-generated βn offspring individuals, n excellent individuals are mutated. Finally, n excellent individuals are preserved from nP_m mutated, unmutated $n(1\text{-}P_m)$, and s elite individuals. In this way, the new population is generated. If computing requirements are met, then computing is stopped; if the computing requirements are not met, process is repeated with the new population. The above steps are repeated until the computing requirements are met. The evolutionary strategy of MGA is shown in Fig. 1.

4.4.2 Evolutionary Strategy of BGA

The evolutionary strategy of BGA is as follows. First, the initial population is generated. All the individuals in the population are sorted in descending order according to their calculated objective function values (if objective function solves maximum). Then, s elite individuals are preserved from n parents. Next, selection, crossover and mutation operations are performed. Finally, s low ranked individuals are replaced by s elite individuals. In this way, the new population is generated. If the computing requirements are met, the process is stopped; else the process is repeated until the computing requirements are met. The evolutionary strategy of BGA is shown in Fig. 2.

4.4.3 Comparative Analysis of Evolutionary Strategy

In the evolutionary strategy of BGA, the parents and cross-generated βn offsprings are likely to be mutated. Hence, excellent cross-generated individuals may be destroyed during the mutation operation, leading to lack of cross-generated excellent individuals. In the evolutionary strategy of MGA, s elite individuals are preserved from parents and cross-generated offsprings. Hence, even though some elite parents are destroyed during the crossover operation, the preserved s elite individuals within the new offspring population are no worse than the elite individuals within the parent population. Based on such analysis, in the evolutionary strategy of MGA, the crossover probability equals 1, the number of cross-generated offsprings is βn, the number of cross-generated offsprings is far more than that of BGA, leading to increase of the probabilty of generating excellent individuals and improved performance.

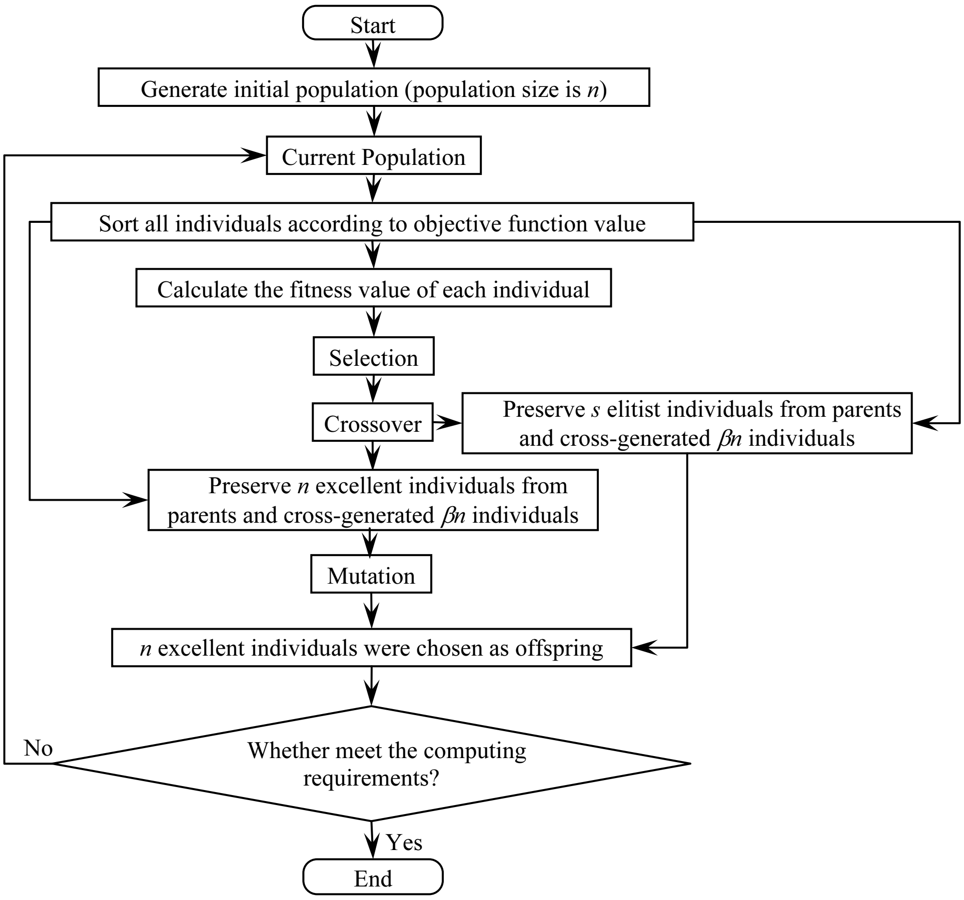

Fig. 1. The evolutionary strategy block diagram of MGA

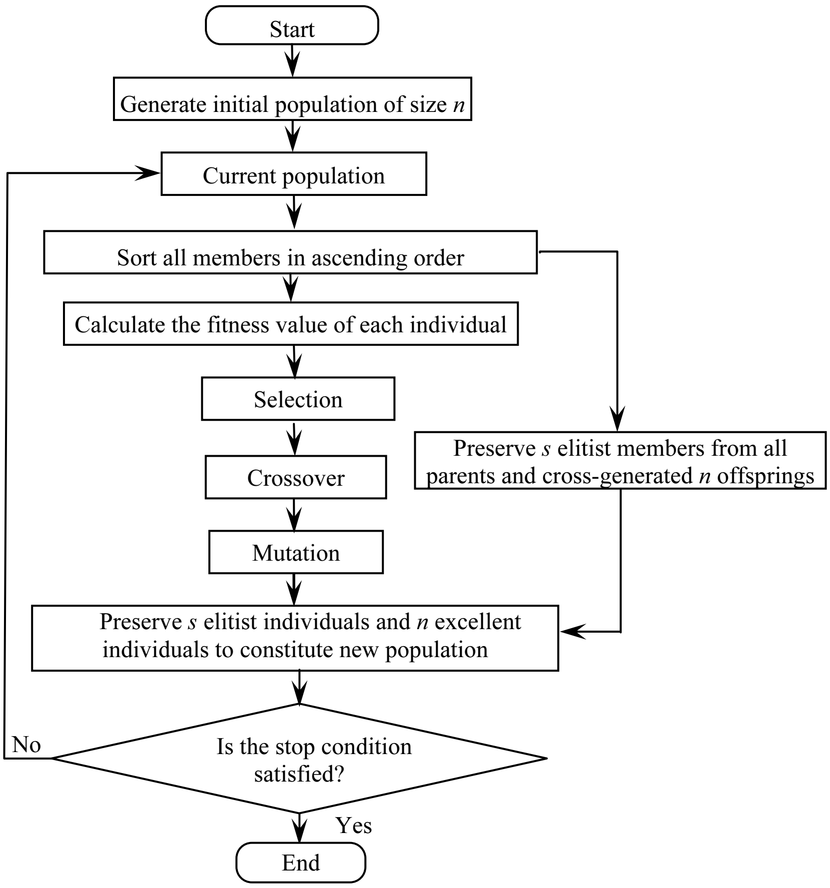

Fig. 2. The evolution strategy block diagram of BGA

5 Iteration Terminal Condition

Both with MGA and BGA, the chosen iteration termination condition is chosen as follows:

$$\left|f_i - f_i^*\right| \le \varepsilon_i, \quad i = 1, 2, \cdots, p \tag{13}$$

where f_i^* is the theoretical global maximum or minimum of the i^{th} test function, f_i is the i^{th} test function's optimal value found by MGA or BGA, and ε_i is the given precision requirement of the i^{th} test function.

6 Algorithm testing and analysis

6.1 Selection of Test Functions

To compare the performance of MGA in comparison to BGA, as well as to investigate the relationship between number of offsprings and computational speed of MGA, ten common test functions with considerable complexity were adopted as described below [30-31].

Test function 1

$$\max f_1(x, y) = \left[\frac{3}{0.05 + (x^2 + y^2)}\right]^2 + (x^2 + y^2)^2 \qquad -5.12 \le x, y \le 5.12 \cdot \tag{15}$$

The global optimal solution of function f_1 is surrounded by the worst solution, and function f_1 has four local minima; determining the global optimal solution is like searching for a needle in a haystack. Therefore, the function is also known as needle in a haystack problem. The global optimal value of f_1 is 3600, and the optimal solution is (x, y)=(0, 0).

Test function 2

$$\min f_2(x_1, x_2) = -\left|\sin(x_1)\cos(x_2)\exp\left(\left|1 - \frac{\sqrt{x_1^2 + x_2^2}}{\pi}\right|\right)\right| \qquad -10 \le x_1, x_2 \le 10 \cdot \tag{16}$$

Function f_2 is called Holder Table function. It has many local minima and four global minima. Determining the local optima is relatively easy, whereas the global optimal solutions are difficult to find. The four global optimal solutions of function are (x_1, x_2) = (8.05502, 9.66459), (x_1, x_2) = (8.05502,−9.66459), (x_1, x_2) = (−8.05502, 9.66459), (x_1, x_2) = (−8.05502,−9.66459), and the global minimum value is −19.2085.

Test function 3

$$\min f_3(x) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + \left(-4 + 4x_2^2\right)x_2^2 \qquad -3 \le x_1 \le 3, -2 \le x_2 \le 2 \cdot \tag{17}$$

Function f_3 is a six-hump camel back function, which has six local minima, two of which are global. The global minimum value of function f_3 is −1.0316, and the global optimal solutions are (x_1, x_2) = (−0.0898, 0.7126) and (x_1, x_2) = (0.0898,−0.7126).

Test function 4

$$\min f_4(x) = \sum_{i=1}^{3}(x_i^2 - 10\cos(2\pi x_i) + 10) \ \text{-}5.12 \le x_1, x_2, x_3 \le 5.12 . \tag{18}$$

Function f_4 is called generalized Rastrigin function. It has many local minima. The function is a typical complex multimodal function with a large number of local optimal points. Determining the local optima is relatively easy, whereas finding the global optimal solution is difficult. The global optimal solution of function f_4 is (x_1, x_2)=(0, 0), and the global minimum value is 0.

Test function 5

$$\min f_5(X)=\frac{1+\cos\left(12\sqrt{x_1^2+x_2^2}\right)}{0.5(x_1^2+x_2^2)+2} \qquad -5.12\le x_1,x_2\le 5.12\,. \tag{19}$$

Function f_5 is called Drop-Wave function. It is multimodal and highly complex, with many local minima, and a single global optimal solution which is (x_1, x_2)=(0, 0), with the global minimum value equal to -1.

Test function 6

$$\min f_6(x_1,x_2)=100*(x_2-x_1^2)^2+(x_1^2-1)^2 \; -10\le x_1,x_2\le 10\,. \tag{20}$$

Function f_6 is the Rosenbrock function, also referred to as the Valley or Banana function. It is a popular test problem for gradient-based optimization algorithms. The function is unimodal, and the global minimum lies in a narrow, parabolic valley. Even though this valley is easy to find, convergence to the minimum is difficult. The global optimal solution of function f_6 is (x_1, x_2)=(1, 1), and the global minimum value is 0.

Test function 7

$$\min f_7(x_1,x_2)=(x_1+2x_2-7)^2+(2x_1+x_2-5)^2 \; -10\le x_1,x_2\le 10\,. \tag{21}$$

Function f_7 is the booth function. Its global optimal solution is at (x_1, x_2)=(1, 3), and its global minimum value is 0.

Test function 8

$$\min f_8(X)=-\cos(x_1)\cos(x_2)\exp\left(-(x_1-\pi)^2-(x_2-\pi)^2\right) -100\le x_1,x_2\le 100\,. \tag{22}$$

Function f_8 is the Easom function, and has several local minima. It is unimodal, and the global minimum has a small area relative to the search space. Its global optimal solution is at f_8 is (x_1, x_2)=(π, π), and the global minimum value is -1.

Test function 9

$$\min f_9(X)=0.5+\frac{\left(\sin\sqrt{x_1^2+x_2^2}\right)^{-2}-0.5}{\left(1+0.001(x_1^2+x_2^2)\right)^2} \; -100\le x_1,x_2\le 100\,. \tag{23}$$

Function f_9 is the Schaffer function, and has thousands of local minima. It is a multimodal function with only one global minimum. Its global optimal solution is at (x_1, x_2) = (0, 0), and the global minimum value is 0.

Test function 10

$$\min f_{10}(X)=\left(\sum_{i=1}^{5} i\cos((i+1)x_1+i)\right)\left(\sum_{i=1}^{5} i\cos(i+1)x_2+i)\right) -10\le x_1,x_2\le 10\,. \tag{24}$$

Function f_{10} is Shubert function, has 760 local minima and 18 global minima. Its global optimal solution is at $(x_1, x2)$=(-1.42513,-0.80032), and the global minimum value is -186.7309.

6.2 The Relationship between the Number of Offsprings and the Computational Speed

0-1 encoding and two-point crossover are adopted with MGA, and BGA; the initial population is randomly generated; the population size is n=100; the precision of encoding is c=10; the mutation probability is P_m=0.1; the number of elite individuals is s=10; computing precisions of the ten test functions are, respectively, $\varepsilon_i=10^{-4}$(i=1, 2, …, 10), and β is equal to 0.15 in Eq. (9).

The range of variables in the test functions are given in Eqs. (15) to (24). Each method was run 500 times with each test function on the same computer. The average running times and average numbers of iterations are shown in Table 15.

Table 15. Test results

Test functions	Methods	Average running time	Average number of iterations
f_1	BGA	0.3223	19.3133
	MGA of 4 offsprings	0.3017	15.1900
	MGA of 6 offsprings	0.2824	12.7367
	MGA of 8 offsprings	0.3662	11.5900
	MGA of 10 offsprings	0.4164	11.2340
	MGA of 12 offsprings	0.4455	10.6420
	MGA of 14 offsprings	0.5029	10.4620
f_2	BGA	0.5741	38.1740
	MGA of 4 offsprings	0.5318	23.8460
	MGA of 6 offsprings	0.1778	6.6460
	MGA of 8 offsprings	0.2172	5.8620
	MGA of 10 offsprings	0.2362	5.3420
	MGA of 12 offsprings	0.2649	5.1900
	MGA of 14 offsprings	0.2890	5.0760
f_3	BGA	0.1972	12.3533
	MGA of 4 offsprings	0.1327	5.9000
	MGA of 6 offsprings	0.1118	4.0133
	MGA of 8 offsprings	0.1215	3.5467
	MGA of 10 offsprings	0.1300	3.4680
	MGA of 12 offsprings	0.1455	3.3980
	MGA of 14 offsprings	0.1663	3.3460
f_4	BGA	0.5861	29.2000
	MGA of 4 offsprings	0.5621	20.1633
	MGA of 6 offsprings	0.5416	15.0233
	MGA of 8 offsprings	0.6045	13.7133
	MGA of 10 offsprings	0.6633	12.8240
	MGA of 12 offsprings	0.7763	12.9580
	MGA of 14 offsprings	0.8918	12.2120
f_5	BGA	0.6912	117.0000
	MGA of 4 offsprings	0.6698	96.6030
	MGA of 6 offsprings	0.2482	30.7060
	MGA of 8 offsprings	0.2596	21.4740
	MGA of 10 offsprings	0.2640	19.1200
	MGA of 12 offsprings	0.2703	17.4620
	MGA of 14 offsprings	0.2908	16.4790
f_6	BGA	6.5208	1207.9400
	MGA of 4 offsprings	4.4131	581.7100
	MGA of 6 offsprings	2.9366	310.8800
	MGA of 8 offsprings	3.0437	240.0100
	MGA of 10 offsprings	3.2007	225.5100
	MGA of 12 offsprings	3.5715	210.2500
	MGA of 14 offsprings	3.7472	203.1900
f_7	BGA	0.8546	52.3920
	MGA of 4 offsprings	0.6722	44.4530
	MGA of 6 offsprings	0.4456	19.5760
	MGA of 8 offsprings	0.4836	14.9220
	MGA of 10 offsprings	0.5096	14.6850
	MGA of 12 offsprings	0.5524	14.2900
	MGA of 14 offsprings	0.6336	14.0760
f_8	BGA	1.4451	272.7260
	MGA of 4 offsprings	1.2930	209.0300
	MGA of 6 offsprings	1.2159	178.9560
	MGA of 8 offsprings	1.2647	131.4720
	MGA of 10 offsprings	1.3665	124.7500
	MGA of 12 offsprings	1.6981	113.4340
	MGA of 14 offsprings	1.7279	103.926

Table 15. Test results (continue)

Test functions	Methods	Average running time	Average number of iterations
f_9	BGA	0.8213	142.4320
	MGA of 4 offsprings	0.7915	105.6830
	MGA of 6 offsprings	0.2291	23.0140
	MGA of 8 offsprings	0.2447	19.6690
	MGA of 10 offsprings	0.2731	18.1400
	MGA of 12 offsprings	0.2860	17.2120
	MGA of 14 offsprings	0.3309	16.9250
f_{10}	BGA	0.6224	50.4980
	MGA of 4 offsprings	0.2782	38.0200
	MGA of 6 offsprings	0.1600	17.9570
	MGA of 8 offsprings	0.1820	15.9470
	MGA of 10 offsprings	0.1909	15.1260
	MGA of 12 offsprings	0.2040	14.1280
	MGA of 14 offsprings	0.2670	13.9170

Corresponding to each test function, the average number of iterations and average running times as a function of the number of offsprings are shown in Fig. 2 and Fig. 3.

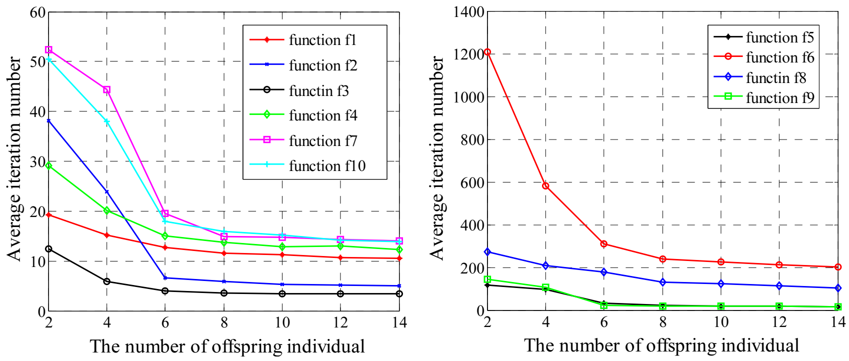

Fig. 2. Average iteration number as a function of number of offsprings

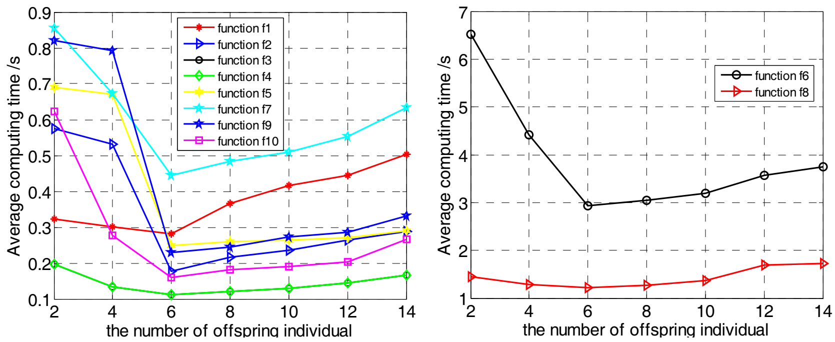

Fig. 3. Average computational time as a function of number of offsprings

The number of offsprings generated by crossover with two parents is significantly increased in MGA as compared with BGA. The increase of number of offsprings improves the probability of generating excellent individuals, and improves the convergence speed of MGA, and saves search time. On the other hand, the calculation time of the crossover operation increases due to generation of more offspring. Thus, a certain relationship exists between the number of cross-generated offspring and the computational speed of MGA.

Table 15, Fig. 2, and Fig. 3 show that the average number of iterations decreases gradually, and the average number of iterations of MGA is significantly less than the average number of iterations of BGA.

The average running time first decreases gradually with the increase in the number of cross-generated offsprings. When the number of cross-generated offsprings exceeds a threshold value, the average running time gradually increases with the number of cross-generated offsprings. The test results show that, for all functions, when the number of cross-generated offsprings exceeds 6, the average running time of MGA gradually increases with the number of cross-generated offsprings; when the number of cross-generated offsprings is less than or equal to 6, the average running time of MGA gradually decreases with the increase of the number of cross-generated offsprings, and the average running time of MGA is significantly less than the average running time of BGA. For functions f_2, f_3, f_5, f_6, f_7, f_9 and f_{10}, when the number of cross-generated offsprings exceeds 6, the average running time of MGA is less than the average running time of BGA. For function f_1, f_4 and f_8, when the number of cross-generated offspring individual exceeds some threshold value, the average running time of MGA is greater than the average running time of BGA.

In conclusion, when the number of cross-generated offsprings by two parents equals to 6, the average number of iterations and average running time of MGA are significantly less than those of BGA.

6.3 Comparison and Analysis of Algorithm Performance

In order to verify the validity and feasibility of MGA, it was compared with references [20, 26-27]. The selection, crossover, mutation, evolution strategy and parameter setting of MGA are as described above, and the number of offsprings of MGA is chosen equal to 6. In addition, the selection, crossover, mutation, evolutionary strategy, and parameter setting were chosen the same as those in [20, 26-27]. The iterative termination conditions for all algorithms are given in Section 5. The average running time and average number of iterations are shown in Table 16.

Table 16. The computational results with all the algorithms

Test function		f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}
Reference [20]	Average computational time	0.3263	0.5583	0.1806	0.5780	0.6870	6.2270	0.7362	1.4062	0.8062	0.6046
	Average number of iterations	16.3890	33.6240	11.5342	28.7250	116.2680	1183.6820	50.2680	268.5732	140.2680	49.6752
Reference [26]	Average computational time	0.3027	0.5283	0.1407	0.5716	0.6432	4.3869	0.6682	1.2832	0.8023	0.2760
	Average number of iterations	16.1460	24.7580	6.2640	20.6840	95.7820	579.6320	43.6582	206.2040	106.3210	37.9020
Reference [27]	Average computational time	0.3125	0.5267	0.1377	0.5672	0.6742	4.4632	0.6702	1.2906	0.7982	0.2774
	Average number of iterations	16.1720	24.6640	6.0820	20.1730	96.5320	582.1020	44.3230	208.6750	106.3620	38.1620
MGA	Average computational time	0.2824	0.1778	0.1118	0.5416	0.2482	2.9366	0.4456	1.2159	0.2291	0.1600
	Average number of iterations	12.7367	6.6460	4.0133	15.0233	30.7060	310.8800	19.5760	178.9560	23.0140	17.9570

From Table 16, it can be seen that the average running time and the average number of iterations of the MGA are significantly better than those of the references [20, 26-27].

7 Conclusions, Significance and Limitations

7.1 Conclusions

The conclusions can be itemized as follows:

(1) A MGA method is proposed in accordance with biological and mathematical ecological theories; the feasibility of MGA is discussed in theory.

(2) A method to generate multi-offsprings by two-point crossover is given.

(3) In MGA, when the number of cross-generated offsprings by two parents is less than or equal to some threshold value, the average running time of MGA is less than those of BGA and references [20, 26-27].

(4) The paper gives the change rule of the average number of iterations and average running time of MGA with the increase of number of cross-generated offspring individual.

(5) Test results with ten test functions show that MGA is faster than BGA and the methods used in references [20, 26-27]. When the number of cross-generated offsprings by two parents equals 6, the running speed of MGA is the fastest. In addition, the average number of iterations of MGA significantly decreases with the increase in the number of cross-generated offsprings.

The conclusions given above are valid for MGA based on the two-point crossover. There are many issues worth further studying, such as offspring generation method in multi-offspring real-coded genetic algorithm, the corresponding best number of offsprings and the relationship between the number of offsprings and the computational time.

7.2 Significance, Limitation and Expectation

The significance of this paper lies in: (1) Gives the theoretical basis of biological and mathematical ecology of MGA; (2) Gives the offspring generation method of MGA based on two-point crossover; (3) Gives the relationship between the offspring number and running speed and the optimal number of offspring individuals of MGA based on two-point crossover; (4) Lays the foundation for subsequent research on MGA.

The limitations of this paper lies in: the conclusions given in this paper are valid only for MGA based on the two-point crossover.

MO-GA has many issues worth further studying, for example, offspring generation method of multi-offspring real-coded genetic algorithm, the best number of offspring individuals and the influence of the number of offspring individuals on the running speed, and so on.

Acknowledgments

The authors thank the editors and anonymous referees who commented on this paper. This work was supported in part by project of the social science foundation of Heilongjiang Province (Grant No. 16JYB06), and in part by Liaoning Provincial Department of Education Project (Grant No. JQW201714407). The authors thank the anonymous reviewers for their valuable and constructive comments that greatly helped improve the quality and completeness of this paper.

References

[1] Z.M. Liu, J.L. Zhou, Q. Ao, The analysis on running mechanism of crossover in genetic algorithms, Journal of Sichuan University (Natural Science Edition) 39(5)(2002) 857-860.

[2] M.Q. Li, The Basic Theory and Application of Genetic Algorithm, Science Press, Beijing, 2002.

[3] Y.L. Zhu, Research on optimization problems based on genetic algorithm, [dissertation] Xian: University of Electronic and Technology of China, 2010.

[4] J.H. Holland, Adaptation in Natural and Artificial Systems: an Introductory Analysis With Application to Biology, Control and Artificial Intelligence, University of Michigan Press, Ann Arbor, 1975.

[5] D.L. Liu, S.W. Xu, Inward-outward crossover based genetic algorithm for constrained optimization problem, System Engineering-Theory & Practice 32(1)(2012) 189-195.

[6] T. Dede, S. Bekiroglu, Y. Ayvaz, Weight minimization of trusses with genetic algorithm, Applied Soft Computing 11(2)(2011) 2565-2575.

[7] K.A. De Jong, An analysis of the behavior of a class of genetic adaptive systems, [dissertation] Holland: University of Michigan, 1975.

[8] Q. Liu, X.Y. Wang, Q.M. Fu, Double elite coevolutionary genetic algorithm, Journal of Software 23(4)(2012) 765-775.

[9] S.Y. Yu, S.Q, Kuang, Convergence and convergence rate analysis of elitist genetic algorithm based on martingale approach, Control Theory & Applications 27(7)(2010) 843-848.

[10] R.B. Qi, F. Qian, W.L. Du, Multiobjective genetic algorithm based on elitist selection and individual migration, Control and Decision 22(2)(2007)164-168.

[11] X.P. Wang, L.M. Cao, Genetic Algorithm-Theory Application and Software Implementation, Xi'an Jiaotong University Press, Xian, 2005.

[12] M. Li, Q. Wang, J.Y. Dai, PID parameter adjustment based on niche hybrid genetic algorithm, Computer Simulation 26(4)(2009) 233-236.

[13] G. Garai, B.B. Chaudhuri, A distributed hierarchical genetic algorithm for efficient optimization and pattern matching, Pattern Recognition 40(2007) 212-228.

[14] F.L. Wang, J.Q. Wang, C.Y. Wu, The improved research on actual number genetic algorithm, Journal of Biomathematics 21(1)(2006) 153-158.

[15] L. Wang, H. Wu, F. Tang, Hybrid quantum genetic algorithm and performance analysis, Control and Decision 20(2)(2005) 156-158.

[16] C.Y. Liu, An improved adaptive genetic algorithm for the multi-depot vehicle routing problem with time window, Journal of Networks 8(5)(2013) 1035-1042.

[17] T.M. Cheng, R.Z. Yan, Integrating messy genetic algorithms and simulation to optimize resource utilization, Computer-Aided Civil and Infrastructure Engineering 24(6)(2009) 401-415.

[18] P. Li, J.Y. Wu, J.H. Zheng, Theoretical analysis and application research on multi-parent genetic algorithm, Computer Engineering and Design 27(4)(2006)581-583.

[19] I. Saracoglu, S. Topaloglu, T. Keskinturk, A genetic algorithm approach for multi-product multi-period continuous review inventory models, Expert Systems with Applications 41(18)(2014) 8189-8202.

[20] F.H. Tseng, X. Wang, L.D. Chou, H.-C. Chao, V.C.M. Leung, Dynamic resource prediction and allocation for cloud data center using the multiobjective genetic algorithm, IEEE Systems Journal PP(99)(2017) 1-12.

[21] M.Z. Ali, N.H. Awad, P.N. Suganthan, A.M. Shatnawi, R.G. Reynolds, An improved class of real-coded genetic algorithms for numerical optimization, Neurocomputing 275(2018)155-166.

[22] M.J. Li, Partheno genetic algorithm theory and its application, [dissertation] Changsha: Hunan University, 2002.

[23] J. Wu, H. Wang, Partheno genetic algorithm for the foundersequence reconstruction problem, Journal of Computers (Finland) 8(11)(2013) 2934-2941.

[24] K. Katayama, H. Narihisa, On fundamental design of parthenogenetic algorithm for the binary quadratic programming problem, in: Proc. the 2001 Congress on Evolutionary Computation, 2001.

[25] S.C. Tsai, S.Y. Fu, Genetic-algorithm-based simulation optimization considering a single stochastic constraint, European Journal of Operational Research 236(1)(2014)113-125.

[26] F.L. Wang, X.M. Fu, H.X. Zhu, The multi-child genetic algorithm, International Journal of Innovative Computing Information & Control 11(5)(2015) 1811-1821.

[27] L. Ma, Q.T. Li, M.Q. Ma, et al., Optimization and application of single-point crossover and multi-offspring genetic algorithm, International Journal of Hybrid Information Technology 9(1)(2016) 1-8.

[28] C. Darwin, On the Origin of Species by Means of Natural Selection, John Murry, London, 1859.

[29] D.W. Wang, J.W. Wang, H.F. Wang, Intelligent Optimization Method, Higher Education Press, Beijing, 2007.

[30] L. Ma, G. Zhu, A.B. Ning, Ant Colony Optimization Algorithm, Science Press, Beijing, 2008.

[31] Z. Ji, H.L. Liao, Q.H. Wu, Particle Swarm Optimization and Application, Science Press, Beijing, 2009.