

Improved Adaptive and Multi-group Parallel Genetic Algorithm Based on Good-point Set

Ruijiang Wang^{1,2}

1.School of Economics and Management, Beijing Jiaotong University, Beijing 100044, China

2.College of Economics and Management Hebei University of Science and Technology, Shijiazhuang, Hebei ,050018, China

E-mail:ruijiang_wang@126.com

Yihong Ru and Qi Long

School of Economics and Management, Beijing Jiaotong University, Beijing 100044, China

E-mail:yhru@center.njtu.edu.cn; longqi_1985@sina.com

Abstract: This paper puts forward an adaptive genetic algorithm to solve the multi-group homogenization in the solution space. The use of good-point set approach improves the initial population, ensuring them a uniform distribution in the solution space. In the evolution, each population implements independent genetic operations (selection, good-point set crossover, and mutation). The introduction of adaptive operator makes crossover and mutation operator self-adaptive. As the algorithm adopts a strategy of retaining the best, a space compression strategy can be designed based on information entropy theory through the information of all sub-populations in the evolution process, which ensures the algorithmic stability and fast convergence to the global optimal solution. Furthermore, in order to explore the feasibility and effectiveness of the improved multi-group parallel algorithm, optimization tests are implemented on some of the typical multi-peak functions, and the results are compared with the analytic solution and optimal solution of basic GA. The outcome suggests that the global searching ability and convergence of the improved algorithm is far better than the basic one.

Index Terms: Good-point Set; Genetic Algorithm; Adaptive Operator; Information Entropy

I. INTRODUCTION

First pioneered by John Holland in his "Adaptation in Natural and Artificial Systems" in the 1970s, Genetic Algorithm (GA) is an optimization solution based on population searching according to C.R.Darwin's biological evolution theory and G.Mendel's genetics. Its main idea is to constantly evolve a randomly generated initial population by reproduction, crossover, mutation or other genetic operators, and ultimately achieve the optimal solution^[1]. Compared with other optimization methods, GA uses only a single string to describe a problem. And no derivative function or other information is needed as a fitness function is used to optimize computing. The algorithm is particularly suitable to resolve complex and nonlinear problems which other technologies cannot or will be difficult to solve. It is a favored area of artificial intelligence besides expert system and artificial neural network, which has always been a heated topic for research. It has also been widely used in combinatorial

optimization, machine learning, adaptive control, intelligent machine system, intelligent manufacturing system, systems engineering, artificial intelligence, artificial life and other areas, becoming one of the key technologies in intelligent computing in the 21st century^[2-4].

GA is implemented as follows: ① randomly generating an initial population in the range of possible solutions; ② evaluating the fitness of each individual in the current population through certain selection methods (fitness function); ③ generating a second generation of population from those selected through genetic operators: crossover and/or mutation; ④ re-implementation of the above process until certain conditions are met. The fitness function is similar to the power of natural selection, and the genetic operators: reproduction, crossover and mutation are corresponding to the proliferation, mating and genetic mutation in nature. The selection operator chooses the fitter individuals in the father generation to ensure the optimal search direction; the crossover operator analogs genetic recombination and random exchange of information to ensure the search space; the mutation operator reflects the gene mutation to ensure the global search capability of the algorithm.

Although GA is intuitive and easy to operate, premature convergence may be brought in by some super-individuals. In the simulation of biological evolution, as these super-individuals soon domain the whole population, global optimal solution is unlikely to be found due to the lack of new genes. Another reason is that the selection and crossover operators may lead to early loss of some excellent gene fragments, thus limiting the scope of the search space, so that searches can only be made in the local area to find the local optimization instead of the global one^[5]. In particular, for the large-scale and high-precision problems, global optimizations are usually missed. Recently many improvements to GA have been put forward, mostly concentrated in initialization^[6], probabilities of selection, crossover and mutation, selection of the fitness function^[7,8], and process design of the evolution. They are all targeted, but a fundamental solution to the above deficiencies hasn't been provided.

In response to the above deficiencies and combined with the solving mechanism of GA, we propose an improved multi-group parallel and adaptive Genetic Algorithm based on good-point set (IMPGA-GPS). We use simulation technology to analyze the convergence of the algorithm, and good performance is proved under the strategy of optimal individual reservation, so that it can effectively avoid premature convergence. It is believed to be feasible for large-scale and high-precision optimization, and have broad application prospects in numerical optimization of complex systems.

II. (IMPGA-GPS) DESIGN

A. Strategy for Improvement

GA is a simulation of biological evolution, which has been widely used because of its global search capability, strong adaptability and robustness. However, there are still many problems to be improved. Recent efforts to improve the performance of GA mainly focus on the implementation strategy, encoding, and design of the selection, crossover and mutation mechanism. Implementation strategy improvements include hybrid genetic algorithm^[9-11], messy genetic algorithm^[12], chaos genetic algorithm^[13], forking genetic algorithm^[14], symbiotic genetic algorithm^[14], nonlinear genetic algorithm^[15] and parallel genetic algorithm^[17, 18]; Encoding improvements include string-coding, real-coding, structured coding and orderly coding; space searching strategies include space adaptive shrinkage^[19, 20] and space shrinkage based on statistical theory^[21].

On the basis of the above analysis, we apply the good-point set theory of number theory to uniformly design the initial population and the crossover operation, improving the overall performance of genetic algorithm. In the large-scale and high-precision optimizations, we can find the space compression factors according to the entropy strategy. Thus without the loss of optimal solutions, the scope of optimization can be narrowed, which is conducive to achieve more accurate satisfactory solution.

B. Key Point of Algorithm Designed

1) Initialization of population

The search speed of basic GA depends on crossover and mutation operators. Although it has a global search capability, it is dependent on the populations in a certain range. As a result, the initial population has a great influence on the convergence, search efficiency and stability in the algorithm^[22]. In the absence of a predictable region of the optimal solution, the initial population must stands for the individuals of the whole space, represents all individual information to the maximum, and fully represents the characteristics of the solution space, thus ensuring the algorithm a fast approach to the optimal solution^[23]. The initial population of basic GA is randomly selected (Monte Carlo method), and its coverage space is uncertain, which is prone to the phenomenon of premature convergence. In addition, maintaining the diversity of the population can improve the global convergence of the algorithm. So while ensuring a reasonable distribution of the initial individuals, a reasonable diversity should also be

considered. Accordingly, in this paper, we set up initial population in a view of the good-point set.

a) Good-point Set Theory

● Assumptions

- (1) Let G_s be an unit cube in an s -dimensional Euclidean space, then $x \in G_s$, $x = (x_1, x_2, \dots, x_s)$, where $0 \leq x_i \leq 1, i = 1, 2, \dots, s$.
- (2) Define a point set (with n points within) in the unit cube.
- (3) For any point $r = (r_1, r_2, \dots, r_s)$ in a given G_s , we can define a $N_n(r) = N_n(r_1, r_2, \dots, r_s)$ representing the number of the points satisfying the following inequality in $P_n(k)$. $0 \leq x_i^n(k) \leq r_i, i = 1, 2, \dots, s$.

Definition 1 Deviation: Define $\phi(n) = \sup_{r \in G_s} \left| \frac{N_n(r)}{n} - |r| \right|$,

where $|r| = r_1, r_2, \dots, r_s$. So we say the point set $P_n(k)$ has a deviation of $\phi(n)$. If for any n , there is $\phi(n) = O$, we say $P_n(k)$ has a coincident distribution with a deviation of $\phi(n)$

Definition 2 Good-point Set Suppose $r \in G_s$, and $P_n(k) = \{\{r_1 * k\}, \{r_2 * k\}, \dots, \{r_s * k\}\}, k = 1, 2, \dots, n\}$ has a deviation of $\phi(n)$. If $\phi(n) = C(r, \varepsilon)n^{-1+\varepsilon}$ where $C(r, \varepsilon)$ is a constant only related to r, ε (any small positive), $P_n(k)$ is called a good-point set and r a good point.

Definition 3 Good-point set: Let

$$r_k = \left\{ 2 \cos \frac{2\pi k}{p} \right\} (1 \leq k \leq s) \quad (1)$$

If p is the smallest prime satisfying $(p-s)/2 \geq s$, or $r_k = \{e^k\} (1 \leq k \leq s)$, then we say r is a good point. $\{a\}$ Denotes the decimals of a .

Theorem 1

If $P_n(k) (1 \leq k \leq n)$ has a deviation of $\phi(n)$ and $f \in B_s$ (s -dimensional limited function), then

$$\left| \int_{G_s} f(x) d_x - \frac{1}{n} \sum_{k=1}^n f(P_n(k)) \right| \leq V(f) \phi(n) \quad (2)$$

where $V(f)$ is the total variation of f .

Theorem 2

If theorem 1 is valid for any $f \in B_s, P_n(k) (1 \leq k \leq n)$ is a point set with a deviation less than $\phi(n)$. This conclusion can be seen as the converse theorem of theorem 1.

Theorem 3

Suppose $f(x)$ satisfies:

$$(1) |f| \leq L, \left| \frac{\partial f}{\partial x_i} \right| \leq L, \quad 1 \leq i \leq s;$$

$$(2) \left| \frac{\partial^2 f}{\partial x_i \partial x_j} \right| \leq L, \quad 1 \leq i \leq j \leq s;$$

$$(3) \left| \frac{\partial^2 f}{\partial x_i \partial x_s} \right| \leq L, \quad 1 \leq i \leq s,$$

Then if we use any weighed sum of any given n points' function values to approximately calculate the integrals of the function on G_s , we could never expect the error to be smaller than $O(n^{-1})$.

Note 1: This theorem is why the above set is called "Good-point Set"

Note 2: Recalling theorem1, 2 and 3, we can know that using a good-point set for approximate integrals, the order of its error is related to n only rather than the dimension s , which provides a superior algorithm for higher order approximately calculation.

Note 3: According all above, if we define n points and use the linear combination of their function values to approximate the corresponding integral, the good-point set can serve as the best solution.

Theorem 4

If $x_1, x_2 \dots, x_n$ is a uniform distribution on *i.i.d.* D_i and $P_n = (x_1, x_2 \dots, x_n)$, the probability of its deviation written as $D(n, P_n) = O(n^{-1/2} (\log \log n)^{1/2})$ is 1.

Note 4: Theorem 4 implies that for an unknown distribution, we can randomly select n points whose deviation is usually $O(n^{-1/2})$. If we use good-point set to find the n points, its deviation is usually $O(n^{-1+\epsilon})$. For instance, if $n = 10,000$, the deviation of the former is $O(10^{-2})$, while the latter is $O(10^{-4})$. Accordingly, the method of good-point set has a much smaller deviation than random selection. That's the theoretical basis we implement good-point set to improve the crossover and initialization of GA^[24,25].

b) The Establishment of Initial Population

The establishment of the initial population is essentially an optimal design of how to use limited individuals to scientifically and globally represent the characteristics of the whole solution space. The initial population can neither be randomly generated nor traverse all the conditions, particularly in the multi-variable problems. Only if we set the most representative individuals who best reflect the inherent characteristics of the solution space as the initial population can we better characterize the space^[24,26].

Uniform design is a scientific and effective method to solve this problem, which has been used to determine the

operating parameters of GA^[27]. Therefore, researchers have put forward several methods for structuring uniform design tables. Good lattice point, Latin square and expansion of orthogonal designs have been used in population initializations, which have achieved good results^[22]. But when the number of factors and levels increase, not only the experiment size increases, and the corresponding uniform design tables become very difficult to get, which bring difficulties to the uniform design methods. However, if we use a good-point set strategy whose accuracy has nothing to do with dimension to initialize the population, the mentioned disadvantages can be solved, and better diversity would be generated in the initial population.

Suppose the size of the initial population is N , and the chromosome is $A_N^i = \{a_1^i, a_2^i, \dots, a_s^i\}$. Firstly we define a good-point set containing N points in a s -dimensional space H . The method is as follows:

Establish a good-point set containing N points in a s -dimensional space H , $P_n(i) = \{\{r_1 * i\}, \{r_2 * i\}, \dots, \{r_s * i\}\}$

$$\text{With } i = 1, 2, \dots, n, r_k = \left\{ 2 \cos \frac{2\pi k}{p} \right\}, 1 \leq k \leq s$$

If p is the smallest prime satisfying $(p - s) / 2 \geq s$, or $r_k = \{e^k\}$ ($1 \leq k \leq s$), then r is a good point. $\{a\}$ denotes the decimals of a .

(1) If a_k^i ($k = 1, 2, \dots, s, i = 1, 2, \dots, N$) is a single-digit binary, define $a_k^i = [\{r_k * i\}]$ (If the decimals of $a < 0.5$, $[a] = 0$, otherwise $[a] = 1$)

(2) If a_k^i ($k = 1, 2, \dots, s, i = 1, 2, \dots, N$) is a multi-digit binary, define the range of a_k^i as $a_k \leq a_k^i \leq \beta_k$. Let $a_k^i = a_k + [\{r_k * i\}] * (a_k - \beta_k)$, and then transfer a_k^i to a binary according to the binary coding rules.

(3) If a_k^i ($k = 1, 2, \dots, s, i = 1, 2, \dots, N$) is a real code, define its range as $a_k \leq a_k^i \leq \beta_k$, and simply let $a_k^i = a_k + \{r_k * i\} * (a_k - \beta_k)$.

Other codes can be optimally structured alike.

2) *Good-point Set Crossover*

Implement the good-point set crossover of the selected individuals under an adaptive crossover probability p_c . Let $A_i = (a_1^i, a_2^i, \dots, a_L^i)$ and $A_j = (a_1^j, a_2^j, \dots, a_L^j)$ be the father-generation individuals selected for crossover, and structure A_i and A_j . Assume H is an aggregate of the

positions in which components differ between A_i and A_j , so $H = \{t \mid a_t^i \neq a_t^j, 1 \leq t \leq L\}$. If t_1, t_2, \dots, t_s denote elements of H , and $t_1 < t_2 < \dots < t_s$, then we can establish a good-point set containing n points in the s -dimensional space:

$$P_n(i) = \{\{r_1 * i\}, \{r_2 * i\}, \dots, \{r_s * i\}\}, \text{ where}$$

$$i = 1, 2, \dots, n, r_k = \{2 \cos \frac{2\pi k}{p}\}, 1 \leq k \leq s.$$
 If p is the smallest prime satisfying $(p-s)/2 \geq s$, or $r_k = \{e^k\}$ ($1 \leq k \leq s$), then r is a good point. $\{a\}$ denotes the decimals of a .

In the n offspring given birth by the crossover, the k^{th} chromosome $C_k = (c_1^k, c_2^k, \dots, c_L^k)$, where $c_m^k = \begin{cases} a_m^j, m \notin H \\ [r_{ij} \times k], m \in H \end{cases} 1 \leq k \leq n, 1 \leq m \leq L$, $1 \leq j \leq s$. And if the decimals of $a < 0.5$, $[a] = 0$, otherwise $[a] = 1$.

In this way, n offspring are generated in this "family". We take the one with the largest fitness as the crossover offspring. The above-mentioned operation is known as the good-point set crossover operation [28].

3) Adaptive Operator

The selections of crossover probability p_c and mutation probability p_m have direct impact on the convergence of the algorithm [29]. With a larger p_c , new individual will be generated faster, but the possibility of destruction of the genetic pattern will also be greater; if p_c is too small, the search process will be slow even stagnant. As for the mutation probability p_m , if it is too small, new individual structures will not come out; if it is too large, GA then becomes a pure random search algorithm. For different problems, repeated experiments are required to determine crossover probability p_c and mutation probability p_m , which is a cumbersome task. Besides, it is difficult to find a best value that applies to every problem. So in the operation of GA, making amendments on the probabilities based on real-time feedback is an effective means to improve the performance of the algorithm.

When the individual fitness tend to be the same or toward the partial optimization, we can increase p_c and p_m , and when individual fitness diverse, we can decrease p_c and p_m . Better individuals whose fitness are above the average should be given smaller p_c and p_m , protecting them to the next generation, while the others be given relatively larger p_c

and p_m . Besides, to avoid local optimal solutions at an early stage, the crossover probability and mutation probability of best-fitted individual should be increased respectively to p_{c2} and p_{m2} , correspondingly making a increase in the probabilities of the better individuals. Hence they will not stay in stagnation, but help a faster approach to the global optimization. The commonly used adaptive operators are as follows [30,31].

The crossover probability can be expressed as:

$$p_c = \begin{cases} k_1(f_{\max} - f)/(f_{\max} - f_{\text{arg}}), f \geq f_{\text{arg}} \\ k_2, f < f_{\text{arg}} \end{cases} \quad (3)$$

$$p_c = \begin{cases} k_1 \exp[-k_2(f - f_{\text{arg}})], f \geq f_{\text{arg}} \\ p_{c1}, f < f_{\text{arg}} \end{cases} \quad (4)$$

The mutation probability can be expressed as:

$$p_m = \begin{cases} k_3(f_{\max} - f)/(f_{\max} - f_{\text{arg}}), f \geq f_{\text{arg}} \\ k_4, f < f_{\text{arg}} \end{cases} \quad (5)$$

$$p_m = \begin{cases} k_3 \exp[-k_4(f - f_{\text{arg}})], f \geq f_{\text{arg}} \\ p_{m1}, f < f_{\text{arg}} \end{cases} \quad (6)$$

Where

k_1, k_2, k_3, k_4 : Integers no larger than 1.0

f_{\max} : The largest fitness value in the population

\bar{f} : The average fitness value of each generation

f : The fitness value of the mutating individuals

Adaptive approach can provide the best p_c and p_m . So the adaptive GA guarantees the convergence of the algorithm as well as a diversity of the population.

4) The Determination of Space Compression Factor

Multi-population algorithm firstly generates M initial population whose search spaces are the same and genetic operations are implemented independently on each population according to the constraints of designed variable sizes. As there is a certain randomness of genetic evolution, the evolutionary results of the M groups are varied, but they can reflect the information which will help determine the best solution. With the processing of evolution, the design will be gradually approaching the optimal solution, that is, the optimal solution will gradually clear from the initial uncertainty in the design space, and search space will also be gradually reduced and finally close to zero (or a given accuracy), making the design tend to fixed-point to achieve optimization.

Set the initial search space as $D_0[a_0, b_0]$, and the space compression factor of each population is $R_j, j = 1, 2, \dots, M$, where M is the number of population. After all M populations go through K generations of evolution respectively, the search space will change to:

$$D_j(K + 1) = R_j D_j(K) \tag{7}$$

Go on with the genetic iteration in the compressed space until convergence. To define the space compression factor, we construct the following information entropy optimization model:

$$\begin{aligned} \min &= \sum_{j=1}^M p_j F(x) \\ \min H &= -\sum_{j=1}^M p_j \ln p_j \tag{8} \\ \text{s.t.} & \sum_{j=1}^M p_j = 1, 0 \leq p_j \leq 1 \end{aligned}$$

$p_j, j = 1, 2, \dots, M$ is the probability the optimal solution falls in population j , $F(x)$ is the objective function. When the optimal solution falls in population j , $\sum_{j=1}^M p_j F'(x^*) = F'(x^*)$. The information entropy H constantly optimizes the uncertainty of the optimal solution during the evolution. At the beginning, the solution is totally unknown, so $p_j = 1/M, (j = 1, 2, \dots, M)$, the entropy get maximum value. With the processing of optimization, the uncertainty of optimal solution is reducing, and p_j and H will change accordingly. When the optimal solution is approached, the uncertainty reduces to zero, that is, $\min H = 0$.

Formula (8) is a multi-objective optimization problem. We introduce this model because the explicit solution of p_j can be easily approached in it, thus the space compression factor can be found.

$$R_j = 1 - p_j \tag{9}$$

Obviously, the best population has the largest space compression. To solve p_j , we use the weighted coefficient method of the multi-objective optimization to rewrite Formula (8) to a single-objective optimization problem.

$$\begin{aligned} \min &= -(1 - \lambda) \sum_{j=1}^M p_j F(x) - \lambda \sum_{j=1}^M p_j \ln p_j \\ \text{s.t.} & \sum_{j=1}^M p_j = 1, 0 \leq p_j \leq 1 \tag{10} \end{aligned}$$

Where λ and $1 - \lambda$ are the weighted coefficient So the Lagrangian merit function of formula 17 is

$$\begin{aligned} L(x, \lambda, \mu) &= - \\ & (1 - \lambda) \sum_{j=1}^M p_j F(x) - \lambda \sum_{j=1}^M p_j \ln p_j + \mu [\sum_{j=1}^M p_j - 1] \\ & \lambda \in (0, 1) \tag{11} \end{aligned}$$

μ is the Lagrangian multiplier, the objective value of the best individual in J populations. We can then simply get:

$$p_j = \exp[\eta F_j(x)] / \sum_{j=1}^M \exp[\eta F_j(x)] \tag{12}$$

Where $\eta = (\lambda - 1) / \lambda$.

C. The Realization of Multi-group Parallel Genetic Algorithm

1) Implementation steps

Step1. Initialization: Use the good-point set approach to generate M initial populations in initial space $D_0(0)$. The size of each population is N , and the generation counter is set $K = 0$

Step2. For each sub-population, implement independent genetic operations to retain the best, and get the best fitness function of all sub-populations and the best individual as well. Record the results of the individual and their corresponding fitness, and then get into the next step.

Step3. Calculate the probability of the optimal solution falling in each population p_j , and get the space compression factor R_j . From formula 14 we can calculate the compressed space of all populations $D_j(K + 1)$, and correspondingly amend the upper and lower limits.

$$a_{ij}(K + 1) = \max \{ [X_{ij}(K) - D_j(K + 1) / 2], a_{ij}(0) \} \tag{13}$$

$$b_{ij}(K + 1) = \min \{ [X_{ij}(K) + D_j(K + 1) / 2], b_{ij}(0) \} \tag{14}$$

where $X_{ij}(K)$ denotes variable i of the best individual in population j in generation K

Step4. Go on searching in the compressed space $D_j(K + 1)$ of all populations: calculating the fitness function, implementing the selection, good-point set crossover and mutation operations, retaining the best and recording the results of the individual and their corresponding fitness.

Step5. Repeat step 3-4 until $D_j(K) \leq \varepsilon$ is satisfied. ε is a given small decimal. Then, we can compare the best fitness values of all populations, and get the optimal value and individual.

To sum up, we use the space compression accuracy as a criterion to judge the convergence of the algorithm. When the space is compressed to the given accuracy, the result of the evolution is optimal, so that the algorithm convergence can be effectively controlled. However, it should be noted that the

optimal solution here is just the best numerical solution the algorithm can find, which may have a certain degree of error with the theoretic optimal solution due to the given restrictions on calculation accuracy.

2) *The Parallel Genetic Algorithm Design*

Firstly, search in the initial space for m times, and m initial sub-populations are generated, uniformly distributed in the solution space and the size of each is n . Then, genetic operations are implemented independently for several

generations in each sub-population in the way designed in this paper. Through information exchanges among sub-populations, the points would be concentrated on the most promising region, thus the search space of each sub-population would become smaller. As there is more than one population and the compression speed of each population differs, the optimal solution will be always included in the search space. In each sub-population we can find one or more local extreme points. Such an algorithm has high speedup ratio and efficiency. The schematic diagram of the algorithm operation is in figure 1.

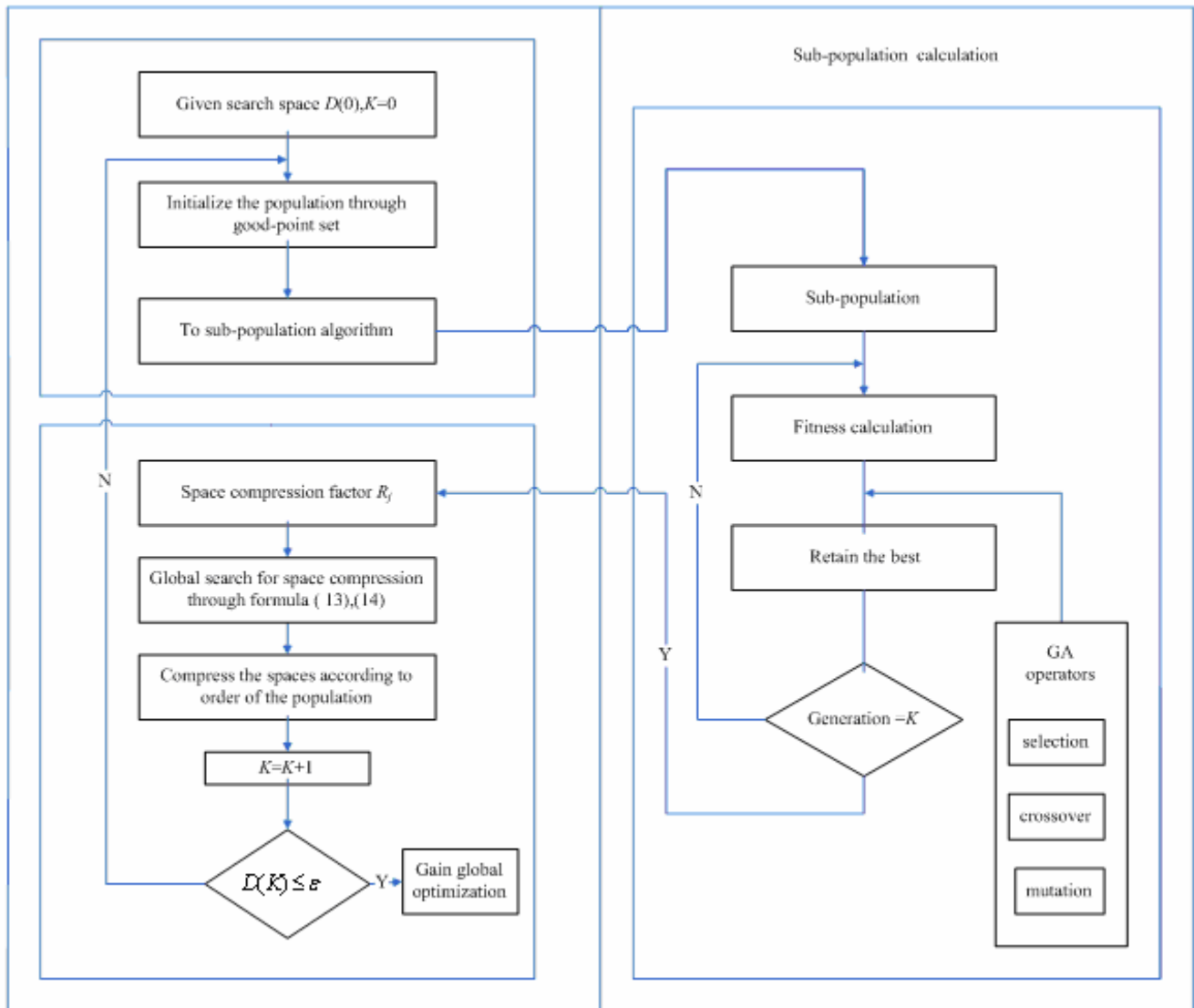


Figure1. TheThe Flow Chart of IMPGA-GPS

III. INSTANCE SIMULATION

Instance 1:

$$\text{Maximize } f(x_1, x_2) = -0.5 + \frac{\sin^2 \sqrt{x_1^2 + x_2^2} - 0.5}{[1 + 0.001(x_1^2 + x_2^2)]^2}$$

$$\text{s.t. } -100 \leq x_1, x_2 \leq 100.$$

The function is multi-peak, whose global minimum is $f_1(0,0) = -1$ at $(0,0)$. We solve the problem through the basic GA and our algorithm respectively, and set the accuracy to four decimals. Set the parameters are as follows:

Basic GA: Pop-size=80; Gen-max=100; Crossover probability $p_c = 0.6$; Mutation probability $p_m = 0.002$. And the iterative result is as Fig.2

Our algorithm: Initial sub-population=4; Pop-size=80; Gen-max=100; Crossover probability $p_c = 0.6$; Mutation

probability $p_m = 0.002$; $\alpha = 0.5$, $\varepsilon = 0.001$. And the iterative result is as Fig.3.

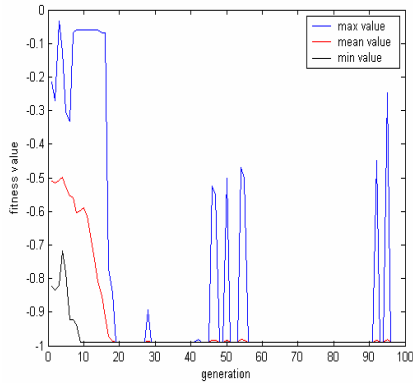


Fig.2. 100-time Iterative result of BGA

As can be seen from the figures, the 4 population only go through 5 generations before getting to the global extreme point in our algorithm.

Instance2:

$$f_1(x_1, x_2) = x_1^2 + 2x_2^2 - 0.4 \cos(3\pi x_1) - 0.6 \cos(4\pi x_2)$$

$$, \quad -100 < x_1, x_2 < 100$$

The global minimum point lies in (0,0) , and the minimum is $f_2(0,0) = -1$, around which there are numerous sub-minimum points. Due to its strong oscillation and as the minimum point is surrounded by sub-minimum points, it's

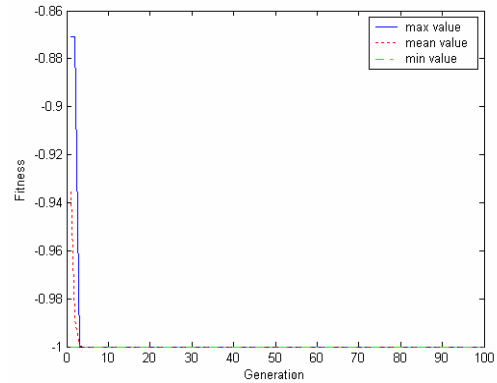


Fig.3. Iterative result of IMPGA-GPS

hard to find the global minimum point through general algorithm.

We solve the problem through the basic GA and our algorithm respectively and set the parameters are as follows:

Basic GA: Pop-size=80; Gen-max=100; Crossover probability $p_c = 0.6$; Mutation probability $p_{m_3} = 0.001$. The objective function value is -0.8235, as Fig.4 shows

Our algorithm: Pop-size=80; Gen-max=100; Crossover probability $p_c = 0.6$; Mutation probability $p_{m_3} = 0.001$; Initial sub-population=4; $\alpha = 0.3, \varepsilon = 0.001$.The objective function value is -1, as Fig.5shows.

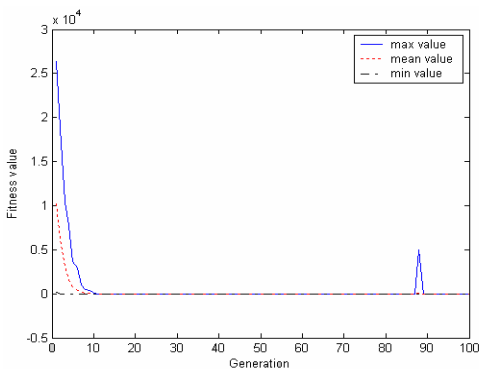


Fig.4. 100-time Iterative result of BGA

From Fig.5, we can see in our algorithm, the population is approximately convergent to the global optimal solution -1 after 5 generations. However, Fig.4 demonstrates that BGA is only convergent to -0.8235 after 100 generations, implying a bad convergence. Overall, our algorithm has greatly improved the BGA on the convergence and global optimization

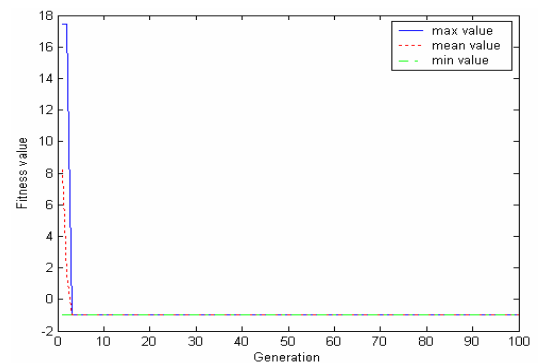


Fig.5. Iterative result of IMPGA-GPS

Moreover, for instance 2, we implement five times simulation tests on the convergence of two algorithms, and the results are as Table 1

Table 1. Five tests simulation results and comparison

	BGA (Pop-size=80, Gen-max=100, $P_{c_3} = 1$, $P_{m_3} = 0.002$)			Our Algorithm (Initial sub-population=4, Pop-size=80, $\alpha = 0.3$, $\varepsilon = 0.001$, $P_{c_4} = 0.6$, $P_{m_4} = 0.001$)		
No	Optimal Solution	Generation	Average Convergence Time (s)	Optimal Solution	Generation	Average Convergence Time (s)
1	-0.7612	17	1.3120	-0.9998	4	2.0330
2	-0.4147	25	1.3900	-1	5	2.3048
3	-0.8415	12	1.2810	-0.9999	4	2.5665
4	-0.4646	14	1.3280	-1	5	2.6440
5	-0.8415	18	1.3440	-1	4	2.1440
Average	-0.6647	17.2	1.3310	-0.9999	4.4	2.28145

From Table.1, we can find our algorithm provides a much better accuracy while a remarkably decreasing convergence generation. And its average convergence time differs little from BGA.

Still for instance 2, we use our algorithm and set the number of initial sub-populations to be 4,5,6,8 respectively, the pop-size to be 100, and $\alpha = 0.3$, $\varepsilon = 0.001$, $P_{c_4} = 0.6$, $P_{m_4} = 0.001$, then we can get results as shown in Table.2

Table 2 Iterative result of different sub-populations

number of sub-populations	Optimal Solution	Generation	Average Convergence Time (s)
4	-0.9998	4	2.0330
5	-0.9999	5	2.8370
6	-1	5	3.1440
8	-1	6	3.9650
Average	-0.99993	5	2.99475

From Table.2, we can find the accuracy of the solution improves along with the number of sub-populations, while the generations and average convergence time increase correspondingly. If the computer configuration is relatively high, the algorithm has more advantages.

IV. CONCLUSIONS

There are premature convergence, low accuracy, and other shortcomings in the practical application of Basic GA (BGA), making it difficult to find the optimal solution or satisfactory solution in the complex optimization situation with a lot of variables. However, the improved multi-group parallel and adaptive genetic algorithm we propose has a strong ability to jump out of the local extreme. The stimulation shows that this algorithm can effectively prevent the early convergence, and to a large extent improve the accuracy of solution, which is suitable for large-scale, high-precision optimization problems.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China (70671034) and the Natural Science Foundation of Hebei Province (F2006000346) and the Ph. D. Foundation of Hebei Province (B2004509, 05547004D-2) and the key project of China MOE Fund for Social Sciences (07JZD0012).

REFERENCES

- [1] J.Holland. Adaptation in natural and artificial system. University of Michigan Press, 1975.
- [2] M.Srinivas, "M.Patnaik. Genetic algorithm: A survey," .IEEE Computer, vol. 27, No.6, 1994, pp. 17-26.
- [3] D.B.Foge. "An Introduction to Simulated Evolutionary Optimization .," IEEE Trans. on SMC, vol. 24, No.1, 1999, pp. 3-14
- [4] W.Atmar. "Noteson the Simulation of Evolution ,"IEEE Trans. on SMC, vol. 24, No.1, 1994, pp. 130-147
- [5] X.M.Huang. "A Kind of Improved Hereditary Algorithm ,"Journal of Changsha University, vol. 19, No.5, 2005, pp. 1-4
- [6] Z.B.Xu, J.S.Zhang. Bionics in Computational Intelligence: Theory and Algorithm .Beijing: Science Press, 2003
- [7] D.W. Gong, X.Y. Sun, and X.J.Guo. "Novel Survival of The Fittest Genetic Algorithm ,"Control and Decision, vol. 11, No.6, 2002, pp. 908-912
- [8] W.L.Han. "The Improvement of Genetic Algorithm ,"Journal of China University of Mining & Technology, vol. 3, No.1, 2001, pp. 102-105
- [9] J.J.Deng, L.H.Xu, and Q.D.Wu. "Hybrid Genetic Algorithm for Nonlinear Function Optimization ,"Journal of Tongji University (Natural Science), vol. 29, No.11, 2001, pp. 1363-1367
- [10] J.Q.Zong. "A Kind of Mixture Adaptive Genetic Algorithm and Analysis of Property ,"Systems Engineering-Theory & Practice, vol. 21, No.4, 2001, pp. 14-18.
- [11] W.Han, Z.P.Liao. "A global optimization algorithm : genetic algorithm simplex ,"Earthquake Engineering and Engineering Vibration, vol. 21, No.2, 2001, pp. 6-12

- [12] J.R.Beveridge, K.Balasubramaniam, and D.Whitley. "Matching Horizon Features Using a Messy Genetic Algorithm," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, No.24, 2000, pp. 499-516
- [13] X.F.Yan, D.Z.Chen, and S.X.Hu. "Chaos-Genetic Algorithms for Optimizing the Operating Conditions Based on RBF-PLS Model," *Computers and Chemical Engineering*, vol.27, No.10, 2003, pp. 1393-1404.
- [14] S.Tsutsui, Y.Fujimoto, and A.Ghos. "A Forking Genetic Algorithms: Gas with Search Space Division Schemes," *Evolutionary Computation*, vol.5, No.1, 1997, pp. 61-80.
- [15] Y.K. Kim, P.Kitae, and J.Ko. "A Symbiotic Evolutionary Algorithm for the Integration of Process Planning and Job Shop Scheduling," *Computers and Operations Research*, vol.30, No.8, 2003, pp.1151-1171.
- [16] ZH.Cui, J.C.ZENG. "Studying of Nonlinear Genetic Algorithms," *Proceedings of the 4th World Congress on Intelligent Control and Automation*. Shanghai China: Press of East China University of Science and Technology, pp.812-814, 2002
- [17] M.N.A. Pereira Cl Udio, M.F.Lapacelso. "Coarse-Grained Parallel Genetic Algorithm Applied to a Nuclear Reactor Core Design Optimization Problem," *Annals of Nuclear Energy*, vol.30, No.5, 2003, pp.555-565.
- [18] A.Muhammad, A Bargiela, and G.KING. "Fuzzy and Evolutionary Modeling of Nonlinear Control Systems," *Mathematical and Computer Modeling*, vol.33, No.45, 2001, pp.533-551.
- [19] L.L.LI, S.N.Guo, S.M.Yuan, and ZH.H.Cui. "Hybrid Genetic Algorithm Based on the Strategy of Searching-Space Adaptive Shortening and Its Application," *Research and analysis*, vol.9, No.5, 2006, pp.3-7
- [20] X.Li, G.Q.Xue. "Application of the Adaptive Shrinkage Genetic Algorithm in the Feasible a Region to TEM Conductive Thin Layer Inversion.," *Applied geophysics*, vol.12, No.4, 2005, pp.204-210.
- [21] L.M.LIU, C.X.JIN, L.Y.Yang, and F.C. LI. "Structure of two-stage genetic algorithm and its performance analysis," *Journal of Hebei University of Science and Technology*, vol.28, No.1, 2007, pp.44-48.
- [22] ZH.J.Li, J.X.Cheng. "Uniform Design of Initial Population of Genetic Algorithm Based on Good Point Set," *Computer and Information Technology*, vol.15, No.4, 2007, pp.29-32.
- [23] D.K.He, F.L.Wang. "Establishment of parameters of Genetic Algorithm Based on Uniform Design," *Journal of Northeastern University (Natural Science)*, vol.24, No.5, 2003, pp.409-411.
- [24] K.T.Fang. "Uniform Design Application of Number theory in experimental design," *Acta Mathematicae Applicatae Sinica*, vol.3, No.4, 1980, pp.363
- [25] L.G.Hua, Y.WANG. *Applications of Number Theory to Approximate Analysis*. Beijing: Science Press, 1978
- [26] X.G.Chen, X.K.Li. "Application of uniform design and genetic algorithm in optimization of reversed phase chromatographic separation," *Chemometrics and Intelligent Laboratory Systems*, vol.68, No.2, 2003, pp.157-166.
- [27] K.T.Fang, H.A. Qin. "note on construction of nearly uniform Designs with large number of runs," *Statistics and Probability Letters* vol.61, No.2, 2003, pp. 215-224.
- [28] L.L.Wang, L.Zhang. "The Genetic Algorithm with Crossover Operator Employing Lattice Methods," *computer engineering & science, china, china*, vol.22, No.1, 2000, pp. 18-20.
- [29] D.Kalyanmoy, B.Hans-Georg. "Self-adaptive Genetic Algorithms with Simulated Binary Crossover," *Evolutionary Computation*, vol.9, No.2, 2001
- [30] K.Hajime. "A Comparison Study of Self-Adaption in Evolution Strategies and Real - Coded Genetic Algorithms," *Evolutionary Computation*, vol.9, No.2, 2001
- [31] P.J. Anfeline. "Adaptive and self-adaptive evolutionary computations In: Palaniswami M, Attikiouzel Y, Marks R, Fogel DB, Fukuda T, eds. *Computational Intelligence: A Dynamic systems Perspective*," IEEE Press, 1995, pp.152-163.

Rui-jiang Wang

Born on Nov 14th, 1976, in Shijiazhuang, Hebei, China, and got Master's degree in Science from Guizhou University, Guizhou Province. Major research: game theory, optimization theory and algorithms, intelligent decision-making system, etc.

He was a teacher of College of Economics and Management, Hebei University of Science and Technology from 2004 to 2007. He is now the Ph.D. of Institute of Logistics, School of Economy and Management, Beijing Jiaotong University, Beijing, China. His current research mainly focuses on logistics management and engineering, decision support systems, optimization algorithm, etc. Recently published papers: Study on reclaiming agent selection models and approaches. IEEE/SOLI 2008 and Improved genetic algorithms to fuzzy bimatrix game. ICIC 2007.

Yi-hong Ru

Born on Oct 9th, 1953 in Jiangsu, China, and got her doctor's degree in public health in the National Institute of Public Health, Japan in 1995.

She is now the Professor of Institute of Logistics, School of Economy and Management, Beijing Jiaotong University, Beijing, China, and used to be a Senior Research Scholar of the Institute of Asia-Pacific Studies, Waseda University, Japan. She is the chief editor of "Logistics Operation Management" (Beijing, China: Tsinghua University Press, 2006), "Logistics" (Beijing, China: China Railway Publishing House, 2003), etc. Her current research mainly focuses on exploring the laws of material flows in the circular economy, so that seeking out an effective management means to realize the sustainable development of human society.

Prof. Ru is the executive member of China Society of Logistics, member of the Logistics Education Steering Committee of Ministry of Education and executive member of the Logistics Branch of Chinese Mechanical Engineering Society, etc. She is also the winner of the 1st prize of Scientific Progress Award in China Federation from Logistics & Purchasing (2006) and 1st prize of Teaching Achievement Awards in Beijing Jiaotong University (2008).

Qi Long

Born on July 17th, 1985 in Sichuan, China, and is a candidate for Master in Logistics Management and Engineering, School of Economy and Management, Beijing Jiaotong University, Beijing, China