Postprint: Application of Improved Genetic Algorithm to Function Optimization

Authors: Yan Chun, Li Meixuan, Zhou Xiao

Date: 2018-07-09T00:00:00+00:00

Abstract

To address the limitations of traditional genetic algorithms in function optimization, such as susceptibility to local optima and slow convergence, a novel adaptive genetic algorithm (NAGA) is proposed. This algorithm considers multiple measures of concentration and dispersion of population fitness, and nonlinearly and adaptively adjusts the crossover and mutation probabilities. To enhance optimization efficiency, the introduced selection operator is combined with an elitist preservation strategy; to maintain a constant population size during genetic operations, a parent preservation strategy is also proposed. Simulation experiments demonstrate that compared with the classical genetic algorithm (GA) and the IAGA algorithm, the improved adaptive genetic algorithm exhibits significant improvements in convergence speed and accuracy.

Full Text

Preamble

Title: Application of Improved Genetic Algorithm in Function Optimization
Authors: Yan Chun, Li Meixuan, Zhou Xiao (School of Mathematics & Systems Science, Shandong University of Science & Technology, Qingdao, Shandong 266590, China)

Abstract: Traditional genetic algorithms (GA) suffer from several shortcomings in function optimization, including susceptibility to local optima and slow convergence. This paper proposes a novel adaptive genetic algorithm (NAGA) that addresses these limitations by considering multiple measures of population fitness concentration and dispersion, and by nonlinearly and adaptively adjusting the crossover and mutation probabilities. To accelerate optimization efficiency, the selection operator is combined with an optimal preservation strategy. Additionally, a parent retention strategy is introduced to maintain constant population size during genetic operations. Simulation experiments demonstrate

that the improved adaptive genetic algorithm achieves significant improvements in convergence speed and accuracy compared to both the classical GA and the IAGA algorithm.

Keywords: function optimization; genetic algorithm; global optimization

0 Introduction

Genetic algorithms are evolutionary algorithms that emulate natural selection and Darwinian evolutionary principles to perform stochastic search and optimization. In recent years, genetic algorithms have found widespread application across diverse domains. Moussa et al. [?] proposed a face recognition system combining genetic algorithms with DCT-PCA for facial feature extraction. Moeini et al. [?] utilized genetic algorithms to optimize the shape of film cooling holes on rotating blades to enhance cooling efficiency. Abdelsalam et al. [?] employed binary real-coded genetic algorithms for local search to optimize wind farm siting. Li et al. [?] applied genetic algorithms to optimize PID controller parameters for temperature and humidity control in environmental test chambers. Despite their utility, traditional genetic algorithms often become trapped in local extrema when solving complex optimization problems, yielding unstable results with low precision. Empirical evidence shows that conventional genetic algorithms struggle to converge to global optimal states.

Numerous scholars have proposed various improvements to enhance genetic algorithm performance. Regarding crossover and mutation probability values, traditional genetic algorithms employ fixed genetic operators throughout the evolutionary process. However, as the population evolves, the proportion of high-quality individuals increases. Without adjusting the genetic operators, this leads to greater disruption of promising solutions, slowing convergence and potentially causing premature convergence [?]. Feng et al. [?] introduced adaptive adjustment formulas, suggesting that larger population differences in early evolution warrant smaller mutation rates and larger crossover rates, while later stages with reduced population differences benefit from larger mutation rates and smaller crossover rates to avoid local optima. However, this approach overlooks the fluctuating nature of population evolution and cannot rely solely on generation number to assess population diversity. Srinivas et al. [?] proposed adaptive adjustment of mutation and crossover rates, but while they adapted P_c and P_m values for individuals with above-average fitness, they applied large fixed values for below-average individuals, which can destroy potentially valuable genetic material carried by inferior individuals. Yang et al. [?] introduced an improved adaptive genetic algorithm (IAGA) that considers the gap between average and maximum population fitness to nonlinearly adapt crossover and mutation probabilities. However, this method neglects scenarios where population fitness is concentrated at low values while maximum fitness remains high—though the average-maximum gap appears large, actual population diversity is low, indicating room for improvement.

Regarding selection strategies, selection operations tend to replicate high-fitness individuals while eliminating low-fitness ones, making selection operator design crucial for global convergence capability [?]. Holland' s roulette wheel selection is representative but suffers from selection randomness that may cause "degradation," where superior individuals are not selected [?]. Li et al. [?] proposed a multi-round roulette wheel selection based on ranking, though this approach runs slowly and only suits small-scale problems. Chen et al. [?] introduced a group-based roulette selection with elitist preservation to prevent destruction of historically excellent individuals. Hao et al. [?] developed a trigonometric function-based selection operator and validated its feasibility. Furtuna et al. [?] enhanced elitist non-dominated sorting genetic algorithms using neural networks. Gao et al. [?] proposed a ranking selection method that directly eliminates the lowest 1/4 individuals, retains the highest 1/4, and applies roulette selection to the middle 1/2. While this ensures convergence efficiency, performing mutation after crossover with iteration-dependent probabilities (where the highest fitness individual' s mutation probability is non-zero) risks losing excellent individuals without historical preservation. Moreover, directly removing the lowest 1/4 individuals cannot balance population diversity effectively. Drawing from these selection operator concepts, this paper proposes a method combining ranking selection with optimal and worst preservation strategies, which preserves excellent individuals, eliminates poor ones, balances population diversity, and ensures convergence speed.

Regarding algorithm execution strategies, researchers have combined genetic algorithms with other optimization methods (e.g., simulated annealing, ant colony optimization) to improve convergence speed and precision [?]. Xia et al. [?] proposed a K-means genetic algorithm with gene rearrangement. Liu et al. [?] developed a hybrid intelligent optimization algorithm integrating particle swarm optimization with genetic algorithms and simulated annealing. Zhang et al. [?] introduced a parallel framework to enhance genetic algorithm efficiency. Jiao et al. [?] optimized ant colony algorithms using genetic algorithms and nonlinear search to overcome slow convergence and local optima issues.

Building upon these research directions and addressing the limitations of fixed crossover and mutation probabilities in traditional genetic algorithms, this paper proposes a novel improved adaptive genetic algorithm.

1 Improved Adaptive Genetic Algorithm

1.1 NAGA Algorithm Flow

The basic genetic algorithm flow comprises crossover, mutation, and selection operations. Crossover operators preserve superior genes through gene exchange between individuals, while mutation operators introduce missing superior genes through individual gene mutation. Traditional genetic operations always perform crossover first. However, when population fitness is low and concentrated, crossover operations hinder rapid generation of superior individuals. In later stages when population fitness is high and concentrated, applying constant mutation rates severely disrupts superior individuals and reduces convergence efficiency.

Based on this analysis, this paper proposes the execution flow of a new adaptive genetic algorithm (NAGA). The NAGA process is as follows:

a) Encode the initial population (L) and set parameters;

b) Define the fitness function, calculate individual fitness values, and preserve the maximum fitness individual T_1;

c) Check convergence conditions. If satisfied, output results; otherwise proceed to step d);

d) Evaluate whether $\pi/12 < \arcsin(f_{\text{ave}}/f_{\max}) < \pi/3$ holds. If true, perform mutation operation (L) followed by crossover operation (with parent retention); otherwise perform crossover operation first;

e) Return to step b).

The flowchart is shown in Figure 1 [Figure 1: see original paper].

1.2 Improved Selection Operator

Individuals are sorted by fitness in ascending order. The bottom 1/4 low-fitness individuals are eliminated (recording the minimum fitness individual), the top 1/4 high-fitness individuals are copied as parents, and the middle 1/2 are retained for further operations. This eliminates poor individuals while preserving excellent parents, ensuring convergence efficiency and facilitating faster optimal solution discovery.

From the retained individuals forming the new population, half are selected as part of the parent generation using the above probability selection method, combined with the previously copied top 1/4 individuals to form a parent population of size $L/2$. To maintain constant population size and prevent missing superior individuals generated during intermediate steps, a parent retention operation is introduced. Additionally, an optimal preservation strategy [?] is employed: each population' s fitness is calculated, and the previous generation' s highest fitness T_1 is compared with the offspring' s highest fitness. If T_1 exceeds the offspring' s maximum, a random offspring individual is eliminated and replaced with the previous generation' s highest fitness individual, ensuring superior individuals are not destroyed by crossover or mutation. To balance population diversity, a worst preservation strategy is also implemented: the recorded worst fitness individual is compared with the new population' s worst fitness. If lower, a random new individual is eliminated and replaced with the previous generation' s worst fitness individual. This approach minimally impacts the evolutionary direction while balancing population diversity.

The selection probability for individuals is calculated as:

$$\begin{cases} p_k = \frac{1-q_{\max}}{N-1} \\ Q_k = \frac{q_{\max}-q_{\min}}{N-1} \times (N-k) + q_{\min} \end{cases}$$

where p_k is the selection probability of the k-th individual, k is the rank in the population, $q_{\max}$ is the selection probability of the best individual, $q_{\min}$ is the selection probability of the worst individual, N is the population size, and k is the current iteration number.

For the $q_{\max}$ value, during early evolution when individual differences are large, high-fitness individuals should have larger selection probabilities to ensure more superior individuals are selected. As evolution progresses and population differences decrease, the best individual' s selection probability should appropriately reduce. Therefore, a $q_{\max}$ value varying with iteration number is proposed:

$$q_{\max} = 1 - \frac{k}{M}$$

where $q_{\max}$ and $q_{\min}$ are the initially defined selection probabilities for best and worst individuals, and M is the total number of iterations.

1.3 Adaptive Adjustment of Crossover Probability P_c and Mutation Probability P_m

To fully leverage the roles of crossover probability P_c and mutation probability P_m in genetic operations, this paper proposes adaptive formulas for their values, as shown in equations (3) and (4):

$$P_c = \begin{cases} \max\left[\arcsin\left(\frac{f_{\text{ave}}}{f_{\max}}\right), (1 - \arcsin\left(\frac{f_{\text{ave}}}{f_{\max}}\right)) \times \frac{k}{M}\right] & \arcsin\left(\frac{f_{\text{ave}}}{f_{\max}}\right) \geq \frac{\pi}{6} \\ \arcsin\left(\frac{f_{\text{ave}}}{f_{\max}}\right) & \arcsin\left(\frac{f_{\text{ave}}}{f_{\max}}\right) < \frac{\pi}{6} \end{cases}$$

$$P_m = \begin{cases} \max\left[\arcsin\left(\frac{f_{\text{ave}}}{f_{\max}}\right), (1 - \arcsin\left(\frac{f_{\text{ave}}}{f_{\max}}\right)) \times \frac{k}{M}\right] & \arcsin\left(\frac{f_{\text{ave}}}{f_{\max}}\right) < \frac{\pi}{6} \\ \arcsin\left(\frac{f_{\text{ave}}}{f_{\max}}\right) & \arcsin\left(\frac{f_{\text{ave}}}{f_{\max}}\right) \geq \frac{\pi}{6} \end{cases}$$

The use of $\arcsin(f_{\text{ave}}/f_{\max})$ as a judgment criterion is justified because it changes more rapidly with $f_{\text{ave}}/f_{\max}$, enabling better assessment of population fitness concentration and dispersion. The threshold $\pi/6$ is used because $\sin(\pi/6) = 1/2$. When $\arcsin(f_{\text{ave}}/f_{\max}) \geq \pi/6$, it indicates $f_{\text{ave}}/f_{\max} \geq 1/2$, meaning the average fitness approaches the maximum fitness and the population is concentrated near the maximum. Based on equations (3) and (4), the condition:

$$\frac{\pi}{12} < \arcsin\left(\frac{f_{\text{ave}}}{f_{\max}}\right) < \frac{\pi}{3}$$

determines whether to perform crossover first. If satisfied, mutation is executed first; otherwise, crossover precedes mutation. This addresses scenarios where, except for the highest fitness, all other fitness values are concentrated at low levels. In such cases, $f_{\text{ave}}/f_{\max} < 1/2$, which the IAGA algorithm would misclassify as a dispersed state and perform crossover first. However, with low fitness concentration and small population differences, initiating with crossover slows evolution and hinders convergence. The innovative conditional formula in this paper provides more comprehensive consideration.

2 Validation of Improved Genetic Algorithm in Function Optimization

2.1 Simulation Experiments and Performance Analysis

To validate the optimization effectiveness of the improved NAGA algorithm, this paper selected two different one-dimensional continuous functions and one two-dimensional function. By comparing optimal values, convergence iterations, and iteration counts with classical GA and Yang et al.' s IAGA algorithm, the performance was evaluated.

All three algorithms (GA, IAGA, and NAGA) employ binary encoding. Parameter settings for the experiments are as follows: For function $f_1(x)$, chromosome length is 32 bits, maximum iterations is 450, population size is 100, GA crossover probability is 0.6, and mutation probability is 0.01. For function $f_2(x)$, chromosome length is 20, population size is 40, maximum iterations is 20, GA crossover probability is 0.7, and mutation probability is 0.01. For IAGA and NAGA algorithms, k_1 and k_2 are set to 1.0 and 0.5, respectively.

2.2 Computational Results for Function $f_1(x)$

The maximum of function $f_1(x) = 2(2(0.1)/0.9)^6 + 1()2\sin(5) - 7\cos(4) - 7xfxxx\pi$ (where $x \in [0, 2]$) was computed to compare the improved genetic algorithm with GA and IAGA.

Figures 2(a) and 2(b) show that unmodified GA produces population average fitness closely approaching maximum fitness, making it difficult for new individuals to escape local extrema. In contrast, NAGA generates populations with relatively fluctuating average fitness, preventing concentration near large values and facilitating escape from local optima. Adaptively changing crossover and mutation rates thus benefits population diversity. When average fitness approaches maximum fitness, adaptively increasing mutation probability while decreasing crossover probability helps populations jump out of local extrema, strengthening global search capability.

Figures 2(c) and 2(d) (enlarged for clarity) show IAGA' s convergence speed is significantly slower than NAGA' s. Since IAGA only uses optimal preservation strategy, it preserves fewer excellent individuals. NAGA employs ranking selection retaining the top 1/4 individuals combined with optimal and worst preservation strategies, improving both convergence speed and population diversity balance.

To directly observe each algorithm' s solving capability, the three algorithms were run 30 times with 450 total iterations. Comparisons were made based on iterations to find optimal solutions, optimal values, convergence probability, and convergence frequency. Results are shown in Table 1 , presenting averages over 30 runs.

Table 1 Computational results for function $f_1(x)$

Algorithm	Optimal Value	Iterations	Convergence Frequency	Convergence Rate
GA	6.8999403577650[illegible]e+04	417	14	-
IAGA	6.8999417202686[illegible]e+04	248	25	-
NAGA	6.8999417217508[illegible]e+04	124	30	-

Since the objective is maximization, Table 1 clearly shows NAGA achieves larger optimal values closer to the theoretical optimum than GA and IAGA, demonstrating higher precision. Iterations decreased from 417 (GA) and 248 (IAGA) to 124 (NAGA), indicating faster convergence. Moreover, after 30 repetitions, convergence frequency increased from 14 (GA) and 25 (IAGA) to 30 (NAGA), demonstrating superior stability.

2.3 Computational Results for Function $f_2(x)$

The maximum of function $f_2(x) = \sin(10\pi x) - 1$ (where $x \in [1, 2]$) was computed to compare the improved algorithm with GA and IAGA. The three algorithms were run 30 times with 20 total iterations, comparing optimal values, iterations to find optimal solutions, convergence frequency, and convergence probability. Results are shown in Table 2 , presenting averages over 30 runs.

Table 2 Computational results for function $f_2(x)$

Algorithm	Optimal Value	Iterations	Convergence Frequency
GA	-	18	9
IAGA	-	13	8
NAGA	-	9	15

Algorithm	Optimal Value	Iterations	Convergence Frequency

Again seeking maximization, NAGA achieves superior optimal values closer to theoretical optima with higher precision. Figure 3 [Figure 3: see original paper] shows simulation results for all three algorithms on $f_2(x)$. NAGA demonstrates faster convergence, with iterations decreasing from 18 (GA) and 13 (IAGA) to 9 (NAGA). Convergence frequency increased from 9 (GA), 8 (IAGA) to 15 (NAGA), confirming improved stability.

2.4 Computational Results for Function $f_3(x)$

The maximum of the two-dimensional function $f_3(x) = 21.5 + x\sin(4\pi x) + y\sin(20\pi y)$ (where $-3.0 \leq x \leq 12.1$, $4.1 \leq y \leq 5.8$) was computed. The theoretical optimum is 38.8503. The three algorithms were compared over 5 runs with 500 iterations each. Results are shown in Tables 3 through 5 .

Table 3 GA optimization results for $f_3(x)$

Run	Optimal Value	Convergence Iterations
1	38.6501	450
2	38.7203	450
...	...	...

Table 4 IAGA optimization results for $f_3(x)$

Run	Optimal Value	Convergence Iterations
1	38.8201	380
2	38.8302	390
...	...	...

Table 5 NAGA optimization results for $f_3(x)$

Run	Optimal Value	Convergence Iterations
1	38.8498	280
2	38.8501	275
...	...	...

GA results show significant fluctuations and deviation from theoretical values. IAGA converges to theoretical values but with high iteration counts and low success frequency. NAGA produces stable results close to theoretical optima with faster convergence than both IAGA and GA, demonstrating enhanced performance in function optimization.

3 Conclusion

The proposed NAGA algorithm effectively controls the overall evolutionary direction while employing adaptive adjustment of crossover probability P_c and mutation probability P_m to overcome the poor global search capability and premature convergence of classical GA. The algorithm improves upon IAGA' s flow conditions by comprehensively considering multiple possible fitness distributions. A selection operator is introduced to record high-fitness individuals, followed by optimal preservation strategy to ensure superior genes are not destroyed by subsequent genetic operations. Low-fitness individuals are promptly eliminated while worst preservation strategy maintains historical worst individuals to balance population diversity and improve convergence efficiency. Parent retention during intermediate processes ensures stable population size. Compared with classical GA and IAGA in both univariate and bivariate function optimization, the proposed algorithm demonstrates significant improvements in convergence speed and accuracy.

References

[1] Moussa M, Hmila M, Douik A. A novel face recognition approach based on genetic algorithm optimization [J]. Studies in Informatics & Control, 2018, 27(1): 127-134.

[2] Moeini A, Zargarabadi∗ M R. Genetic algorithm optimization of film cooling effectiveness over a rotating blade [J]. International Journal of Thermal Sciences, 2018, 125: 248-255.

[3] Abdelsalam A M, El-Shorbagy M A. Optimization of wind turbines siting in a wind farm using genetic algorithm based local search [J]. Renewable Energy, 2018, 123: 748-755.

[4] Li Shujiang, Zhao Chen, Su Xihui, Wang Xiangdong. Temperature and humidity Control of Environmental Test Cabin based on genetic algorithm to optimize parameters of PID Controller [J]. Journal of Nanjing University of Science and Technology, 2017(4): 511-518.

[5] Han Ranran. Research on Optimization problem based on Cloud Adaptive genetic algorithm [D]. Jilin: Northeast Electric Power University, 2013.

[6] Qu Zhijian, Zhang Xianwei, Cao Yanfeng, et al. Genetic algorithm based on adaptive mechanism [J]. Computer Application Researches, 2015, 32(11): 3222-3225+3229.

[7] Feng Leihua, Yang Feng. Fuzzy PID Control of Pneumatic Shield Gate based on improved genetic algorithm [J]. Water Conservancy Planning and Design, 2017(12): 98-101.

[8] Srinivas M, Patnaik L M. Adaptive probabilities of crossover and mutation in genetic algorithms [J]. IEEE Trans on System, Man, and Cybernetics, 1994, 24(4): 656-667.

[9] Yang Congrui, Qian Qian, Wang Feng, Sun Minghui. Application of improved adaptive genetic algorithm in function optimization [J]. Computer Application Research, 2018, 35(4): 1042-1045.

[10] Li Chen, Ning Hongyun. Improved genetic algorithm selection operator [J]. Journal of Tianjin University of Technology, 2008, 24(06): 1-4.

[11] Jiang Yan, Li Xiangfeng, Zuo Dunwen, et al. Performance comparison and Application of genetic algorithms with improved selection operators [J]. China's manufacturing information, 2010, 39(11): 46-50.

[12] Chen Youqing, Xu Caixing, Zhong Wen Liang, Zhang Jun. An improved selection operator genetic algorithm [J]. Computer engineering and Application, 2008, 44(2): 44-49.

[13] Hao Guosheng, Yan Yurou, Huang Yongqing, et al. Selection operator of genetic algorithm based on trigonometric function [J]. Journal of Jiangnan University: Natural Science Edition, 2010, 9(2): 162-165.

[14] Furtuna R, Curteanu S, Leon F. An elitist non-dominated sorting genetic algorithm enhanced with a neural network applied to the multi-objective optimization of a polysiloxane synthesis process [J]. Engineering Applications of Artificial Intelligence, 2011, 24(5): 772-785.

[15] Gao Hang, Xue Lingyun. Back Propagation Neural Network based on improved genetic algorithm to fit LED Spectral Model [J]. Progress laser and optoelectronics, 2017, 54(7): 294-302.

[16] Lang Minfeng. The improvement of genetic algorithm and its application in combinatorial optimization [D]. Shanghai: East China normal University, 2005.

[17] Xia Changdong, Zhang Xiada, Zheng Changwang. A genetic algorithm with gene rearrangement for K-means clustering [J]. Pattern Recognition, 2009, 42(7): 1210-1222.

[18] Liu Lu, Chen Zan, Liu Shijie, et al. A new particle swarm optimization improved genetic algorithm [J]. Microcomputer and its Application, 2017, 36(23): 17-20.

[19] Zhang Xin, Zhang Qinglian, Zhang Xiu. Nonuniform antenna array design by parallelizing three-parent crossover genetic algorithm [J]. Eurasip Journal on Wireless Communications & Networking, 2017, 2017(1): 106.

[20] Jiao Deqiang, Chang Huaiyang. Application of an improved ant colony algorithm TSP problem [J]. Science, Technology and Innovation, 2018(1): 145-146.

Note: Figure translations are in progress. See original paper for figures.