Research Article

Yan Taishan*, Chen Xin, and Yan Baoshuang

An adaptive genetic algorithm with double populations for solving traveling salesman problems

https://doi.org/10.1515/jisys-2025-0040
received March 01, 2025; accepted May 23, 2025

Abstract: Traveling salesman problem (TSP) is a typical combinatorial optimization problem which is regarded as an NP (nondeterministic polynomial)-hard problem. When the TSP in a large scale are solved by traditional optimization methods, the computation is large, the iteration time is long, and the result may not be optimal. The genetic algorithm is a highly favored optimization algorithm, but due to the lack of diversity in population and genetic individuals, it also has the same limitations when solving large-scale TSPs. In order to improve the solving speed and solution accuracy of TSPs, an adaptive genetic algorithm with double populations (DPAGA) is proposed. A dominant population and an auxiliary population are introduced in the algorithm. The algorithm distinguishes genetic individuals into male and female individuals, and only heterosexual individuals can perform crossover operations. The dominant population and auxiliary population execute different adaptive crossover and mutation strategies during the evolution process. The DPAGA is used to solve TSPs; the experimental results prove that it is completely feasible, and its convergence speed and solution quality have been improved. When solving the TSPs with scales of 20, 30 and 50 cities, the optimal path lengths of DPAGA were reduced by the maximum of 0.2491, 9.1071 and 15.7086, respectively, compared with other optimization algorithms.

Keywords: traveling salesman problem, adaptive genetic algorithm, dominant population, auxiliary population

1 Introduction

Traveling salesman problem (TSP) is a nondeterministic polynomial (NP)-hard problem in path planning and combinatorial optimization fields [1–7]. The general definition for TSP is as follows: there are n cities connected to each other, and the distance between cities is known. A travel salesman starts from the starting city and passes through the remaining cities in sequence. He must pass through each city once, and only once, and finally return to the starting city. It is required to develop an optimal route to minimize the total distance he passes through. Its mathematical description can be expressed as follows:

$$\min Z = \sum_{i=1}^{n}\sum_{j=1}^{n} d_{i,j}x_{i,j}, \tag{1}$$

*** Corresponding author: Yan Taishan,** School of Energy and Electrical Engineering, Hunan Institute of Science and Technology, Yueyang, 414000, China, e-mail: 11991437@hnist.edu.cn
Chen Xin: School of Energy and Electrical Engineering, Hunan Institute of Science and Technology, Yueyang, 414000, China, e-mail: 14205401029@hnist.edu.cn
Yan Baoshuang: School of Energy and Electrical Engineering, Hunan Institute of Science and Technology, Yueyang, 414000, China, e-mail: 12006042@hnist.edu.cn

$$s.t.\ \sum_{j=1}^{n} x_{i,j} = 1, (i = 1,\ 2,\dots,\ n), \tag{2}$$

$$\sum_{i=1}^{n} x_{i,j} = 1,\ \ (j = 1,\ 2,\dots,\ n), \tag{3}$$

$$\textstyle\sum_{i,j\in S} x_{i,j} \le R - 1, 2 \le |R| \le n - 2, R \subset \{1,\ 2,\dots,\ n\}, \tag{4}$$

$$x_{i,j} \in [0,\ 1], (i, j = 1,\ 2,\ \dots, n). \tag{5}$$

Here, formula (1) is the objective function. $d_{i,j}$ represents the distance between cities i and j. Formulas (2), (3), and (4) indicate that a travel salesman passes through each city once and only once. Formula (5) is the decision variable value, 1 represents a city that has already been passed through, and 0 represents a city that has not been passed through.

Algorithms for solving TSPs usually include the following: (1) traditional classic algorithms such as the exhaustive method [8], branch and bound method [9], dynamic programming method [10], etc. (2) Popular intelligent algorithms such as the genetic algorithm [5,11–14], ant colony optimization (ACO) algorithm [4,15,16], particle swarm optimization algorithm [17], heuristic search algorithm [18], simulated annealing (SA) algorithm [19,20], etc. The time complexity of traditional classical algorithms is exponential; their efficiency is low for large-scale TSPs and cannot guarantee obtaining the optimal solution. Although the popular intelligent algorithms may not necessarily obtain the optimal solution to the problem, they have the advantages of faster speed and better solution results in solving large-scale TSPs. Among them, genetic algorithms are highly favored for their strong robustness and global search ability. However, traditional genetic algorithms work in a population with fixed crossover and mutation rates. For a given optimization object, finding the appropriate crossover and mutation rates requires a lot of time, which affects the efficiency of the algorithm. In this case, the algorithm is also difficult to meet the requirements of population diversity.

In order to improve the performance of genetic algorithm, many scholars have proposed various adaptive genetic algorithms. Defersha and Rooyani [21] proposed an efficient two-stage genetic algorithm to solve a flexible job-shop scheduling problem. Wang [22] proposed an adaptive genetic algorithm for real-time path optimization of mobile robots. Pan et al. [23] proposed an adaptive genetic algorithm based on chaos "micro mutation" by adding chaos disturbance to the selection operator of genetic algorithm and adaptively adjusting the crossover operator and mutation operator. Chen et al. [24] proposed a self-tuning and adaptive genetic algorithm by dynamically adjusting the mutation rate, population size, and convergence threshold. Wang et al. [25] proposed an improved adaptive genetic algorithm by designing the adaptive crossover and mutation operator. Zhang et al. [26] proposed a nonlinear adaptive genetic algorithm, which was successfully applied to parameter identification of the Bouc-Wen model. Li et al. [27] proposed an improved adaptive genetic algorithm based on the sigmoid function's adaptive crossover and mutation operators and successfully applied it to optimize water distribution analysis in canal systems. Qin et al. [28] proposed an improved adaptive genetic algorithm by optimizing the crossover and mutation rules as well as the objective function and successfully applied it to the location selection of logistics distribution centers. Most of the above algorithms are studied from the perspective of adaptive parameter settings. The usual approach is to adjust the genetic parameters, such as the crossover rate Pc and mutation rate Pm, dynamically according to the actual situation of the population during the evolution process, which improves the global convergence rate and optimization efficiency to varying degrees. However, these algorithms still have certain limitations, mainly reflected in two aspects: first, the singularity of the evolutionary population, and second, the singularity of genetic individual sex. So, it is fundamentally impossible to ensure that the algorithm extracts local optima effectively.

For the purpose of improving the ability of genetic algorithms to jump out of the local optima, a new algorithm with population diversity and genetic individual diversity is needed. Therefore, an adaptive genetic algorithm with double populations (DPAGA) is proposed in this article. The basic ideas, implementation methods, and workflow of the algorithm are explained. This algorithm is used to solve TSPs, aiming to improve the efficiency of solving TSPs and the quality of the optimal solutions.

2 DPAGA

2.1 Basic idea of the algorithm

In order to overcome the problem that traditional genetic algorithms with single evolutionary population cannot guarantee the escape from local optima, we introduce a double population mechanism of "dominant population and auxiliary population" in genetic algorithms. Different strategies are used to achieve the evolution of genetic individuals in the dominant and auxiliary populations. For the purpose of reducing the blindness of selection operations, we introduce the characteristic of sexual reproduction in the biological world, dividing genetic individuals into male and female sex types and allowing opposite sex individuals to perform crossover operations. A DPAGA is established. The basic idea of this algorithm is as follows: the individuals selected by the selection operation are used as the new evolutionary population (dominant population) and the individuals not selected by the selection operation formed another population (auxiliary population). The dominant population performs adaptive high probability crossover and low probability mutation operations during the evolution process, while the auxiliary population performs adaptive low probability crossover and high probability mutation operations during the evolution process. With the help of auxiliary population, the algorithm can effectively jump out of local optima when premature convergence occurs.

2.2 Encoding and initial population generation

For a TSP involving *n* cities, a specific number is assigned to each city. So, the solution space of TSP is formed by a lot of complete permutations consisting of *n* city numbers. For example, nine cities are numbered 1, 2, 3, 4, 5, 6, 7, 8, and 9 in sequence. Thus, $7 \rightarrow 5 \rightarrow 9 \rightarrow 8 \rightarrow 4 \rightarrow 3 \rightarrow 2 \rightarrow 6 \rightarrow 1 \rightarrow 7$ represents a touring path (chromosome), indicating that merchants depart from city 7 and pass through cities 5, 9, 8, 4, 3, 2, 6, and 1 in order before returning to city 7. The sex encoding of genetic individuals adopts 0/1 encoding, where 1 represents male individuals and 0 represents female individuals. The genetic individual coding structure is shown in Figure 1.

Sex coding	Chromosome coding

Figure 1: Genetic individual coding structure. Source: Created by the authors.

The initial population is usually generated randomly. First, the size of the initial population is set according to the requirements of the problem, then a certain number of individuals are generated randomly, and the better individuals from these individuals are selected to form the initial population.

2.3 Fitness function

In genetic algorithms, the individual fitness function is usually obtained by converting the objective function according to certain rules. The goal of the TSP is to minimize the total path length, so the fitness function can be gained by taking the reciprocal, that is:

$$F(x) = \frac{1}{D(x)}, \tag{6}$$

$$D(x) = \sum_{i=1}^{n-1} d_{i,i+1} + d_{n,1}, \tag{7}$$

where $D(x)$ represents the total distance generated by travel salesman when passing through all cities without repetition and returning to the starting city. $d_{i,i+1}$ represents the distance from city i to adjacent city $i + 1$ in the path experienced by traveling salesman.

2.4 Genetic operation

2.4.1 Selection operation

In DPAGA, a two-stage competitive ranking selection method is used to select excellent genetic individuals as candidates for crossover operation. In order to ensure effective crossover among individuals of the opposite sex, male and female individuals should be kept equal in number and sorted in the order of superiority or inferiority among the selected individuals. In this way, the genetic diversity during the evolutionary process can be maintained better, the excellent genes and patterns are not destroyed, and the evolutionary process will develop towards the direction of the global optimal solution.

2.4.2 Crossover operation

In DPAGA, the two individuals performing crossover operations are two opposite sex individuals paired in the order of superiority. The crossover method is a single point adaptive crossover, and the dominant population crossover rate P_{dc} and auxiliary population crossover P_{ac} are adaptively adjusted according to the following methods:

$$P_{\mathrm{dc}} = \begin{cases} P_{\mathrm{dc1}} - (P_{\mathrm{dc1}} - P_{\mathrm{dc2}})\dfrac{f' - f_{\mathrm{davg}}}{f_{\mathrm{dmax}} - f_{\mathrm{davg}}}, & f' \geq f_{\mathrm{davg}} \\ P_{\mathrm{dc1}}, & f' < f_{\mathrm{davg}} \end{cases}, \tag{8}$$

$$P_{\mathrm{ac}} = \begin{cases} P_{\mathrm{ac1}} - (P_{\mathrm{ac1}} - P_{\mathrm{ac2}})\dfrac{f' - f_{\mathrm{aavg}}}{f_{\mathrm{amax}} - f_{\mathrm{aavg}}}, & f' \geq f_{\mathrm{aavg}} \\ P_{\mathrm{ac1}}, & f' < f_{\mathrm{aavg}} \end{cases} \tag{9}$$

2.4.3 Mutation operation

The mutation operation of DPAGA is adaptive mutation, and its dominant population mutation rate P_{dm} and auxiliary population mutation rate P_{am} are adaptively adjusted in the following ways:

$$P_{\mathrm{dm}} = \begin{cases} P_{\mathrm{dm1}} - (P_{\mathrm{dm1}} - P_{\mathrm{dm2}})\dfrac{f - f_{\mathrm{davg}}}{f_{\mathrm{dmax}} - f_{\mathrm{davg}}}, & f \geq f_{\mathrm{davg}} \\ P_{\mathrm{dm1}}, & f < f_{\mathrm{davg}} \end{cases} \tag{10}$$

$$P_{\mathrm{am}} = \begin{cases} P_{\mathrm{am1}} - (P_{\mathrm{am1}} - P_{\mathrm{am2}})\dfrac{f - f_{\mathrm{aavg}}}{f_{\mathrm{amax}} - f_{\mathrm{aavg}}}, & f \geq f_{\mathrm{aavg}} \\ P_{\mathrm{am1}}, & f < f_{\mathrm{aavg}} \end{cases} \tag{11}$$

In formulas (8)–(11), f_{dmax} and f_{amax} denote the maximal fitness of the dominant population and auxiliary population, respectively; f_{davg} and f_{aavg} denote the average fitness of the dominant population and auxiliary population, respectively; f' denotes the higher fitness of the two crossing individuals; f denotes the fitness of the mutating individuals; P_{dc1} and P_{dc2} denote the upper limit and the lower limit of the crossover probability of the dominant population, respectively; P_{ac1} and P_{ac2} denote the upper limit and the lower limit of the crossover probability of the auxiliary population, respectively; P_{dm1} and P_{dm2} denote the upper limit and the lower limit of the mutation probability of the dominant population, respectively; P_{am1} and P_{am2} denote the upper limit and the lower limit of the mutation probability of the auxiliary population, respectively.

2.5 Workflow of DPAGA

Figure 2 illustrates the workflow of DPAGA.

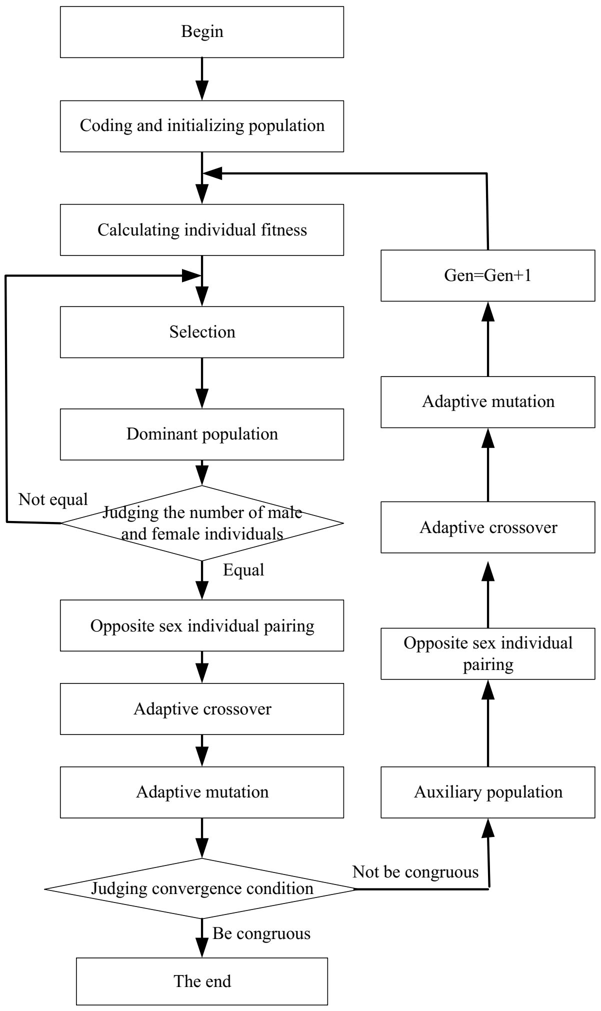
Begin
Coding and initializing population
Calculating individual fitness
Selection
Dominant population
Not equal
Judging the number of male and female individuals
Equal
Opposite sex individual pairing
Adaptive crossover
Adaptive mutation
Judging convergence condition
Be congruous
The end
Not be congruous
Auxiliary population
Opposite sex individual pairing
Adaptive crossover
Adaptive mutation
Gen=Gen+1

Figure 2: Workflow of DPAGA. Source: Created by the authors.

Table 1: Parameters of DPAGA

TSP	*T*	*M*	Parameter settings
Example 1	200	80	$P_{dc1} = 0.9$, $P_{dc2} = 0.6$, $P_{ac1} = 0.1$, $P_{ac2} = 0.001$, $P_{dm1} = 0.1$, $P_{dm2} = 0.001$, $P_{am1} = 0.5$, and $P_{am2} = 0.1$
Example 2	300	100	$P_{dc1} = 0.9$, $P_{dc2} = 0.6$, $P_{ac1} = 0.1$, $P_{ac2} = 0.001$, $P_{dm1} = 0.1$, $P_{dm2} = 0.001$, $P_{am1} = 0.5$, and $P_{am2} = 0.1$
Example 3	500	120	$P_{dc1} = 0.9$, $P_{dc2} = 0.6$, $P_{ac1} = 0.1$, $P_{ac2} = 0.001$, $P_{dm1} = 0.1$, $P_{dm2} = 0.001$, $P_{am1} = 0.5$, and $P_{am2} = 0.1$

Table 2: Solution results of DPAGA

TSP	Shortest path	Shortest path length	Solving time
Example 1	14 → 11 → 4 → 8 → 10 → 15 → 19 → 7 → 18 → 16 → 5 → 13 → 20 → 6 → 17 → 9 → 2 → 12 → 14 3 → 1 → 14	24.5222	0.097
Example 2	21 → 13 → 18 → 26 → 11 → 12 → 7 → 4 → 1 → 9 → 6 → 5 → 16 → 17 → 27 → 3 → 14 → 2 → 30 → 29 → 23 → 28 → 15 → 25 → 24 → 22 → 19 → 20 → 8 → 10 → 21	415.7623	0.1920
Example 3	5 → 8 → 6 → 7 → 9 → 10 → 11 → 12 → 13 → 22 → 17 → 18 → 19 → 20 → 21 → 42 → 43 → 44 → 45 → 46 → 32 → 33 → 34 → 35 → 36 → 23 → 3 → 25 → 26 → 41 → 15 → 16 → 1 → 2 → 14 → 28 → 27 → 5 37 → 38 → 39 → 47 → 30 → 29 → 31 → 40 → 48 → 49 → 24 → 50 → 4 → 5	453.9863	0.859

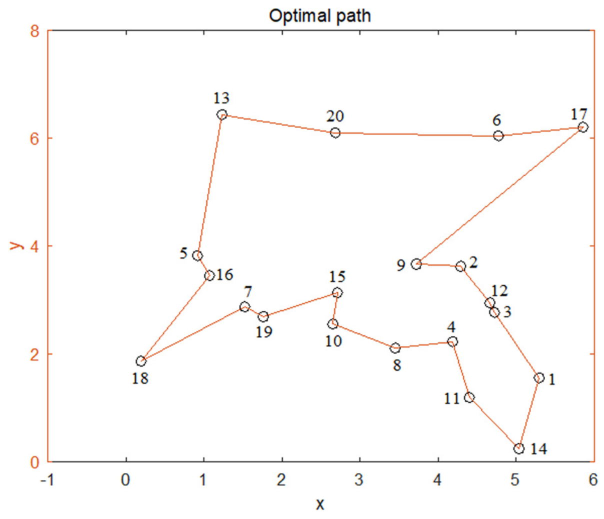

Figure 3: Optimal solution road of Example 1 solved by DPAGA. Source: Created by the authors.

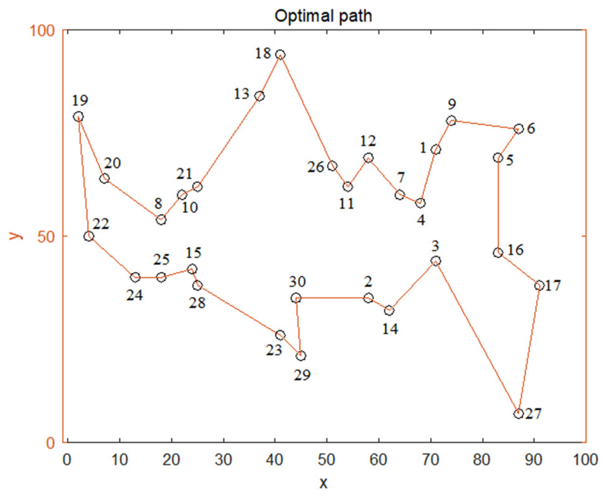

Figure 4: Optimal solution road of Example 2 solved by DPAGA. Source: Created by the authors.

3 Solving TSPs with DPAGA

3.1 TSP examples

The classic TSP examples in Guo et al., You & Lei, Duan, Tao et al., Pei et al., and Pan & Wu [2,6,7,12–14] are used in this article. Assuming that the number of cities is n, the TSP examples are described as follows:

Example 1: n = 20, city20 = {{5.294,1.558}, {4.286,3.622}, {4.719,2.774}, {4.185,2.230}, {0.915,3.821}, {4.771,6.041}, {1.524,2.871}, {3.447,2.111}, {3.718,3.665}, {2.649,2.556}, {4.399,1.194}, {4.660,2.949}, {1.232,6.440}, {5.036,0.244}, {2.710,3.140}, {1.072,3.454}, {5.855,6.203}, {0.194,1.862}, {1.762,2.693}, {2.682,6.097}};

Example 2: n = 30, city30 = {{71,71}, {58,35}, {71,44}, {68,58}, {83,69}, {87,76}, {64,60}, {18,54}, {74,78}, {22,60}, {54,62}, {58,69}, {37,84}, {62,32}, {24,42}, {83,46}, {91,38}, {41,94}, {2,79}, {7,64}, {25,62}, {4,50}, {41,26}, {13,40}, {18,40}, {51,67}, {87,7}, {25,38}, 45,21}, {44,35}};

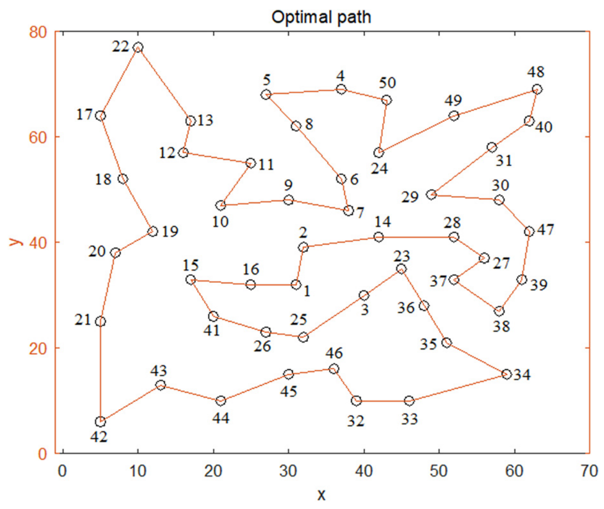

Figure 5: Optimal solution road of Example 3 solved by DPAGA. Source: Created by the authors.

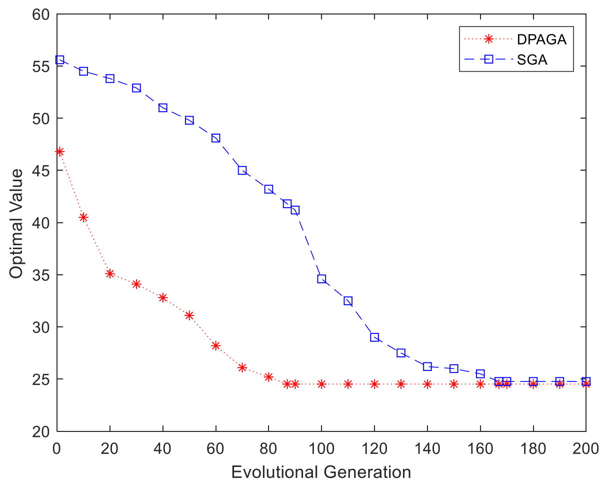

Figure 6: Evolution process of DPAGA and SGA when solving Example 1. Source: Created by the authors.

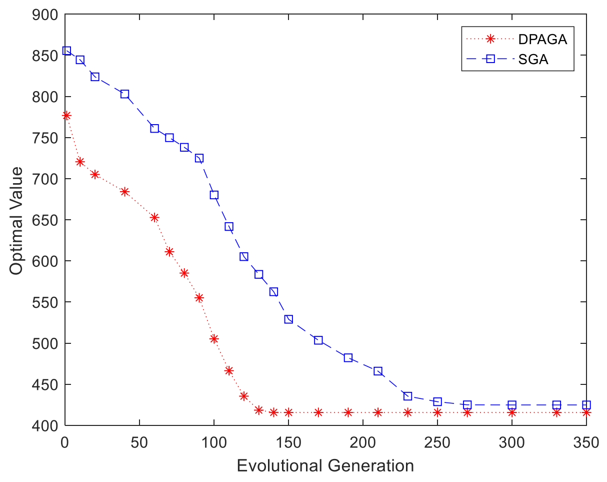

Figure 7: Evolution process of DPAGA and SGA when solving Example 2. Source: Created by the authors.

Example 3: n = 50, city50 = {{31,32}, {32,39}, {40,30}, {37,69}, {27,68}, {37,52}, {38,46}, {31,62}, {30,48}, {21,47}, {25,55}, {16,57}, {17,63}, {42,41}, {17,33}, {25,32}, {5,64}, {8,52}, {12,42}, {7,38}, {5,25}, {10,77}, {45,35}, {42,57}, {32,22}, {27,23}, {56,37}, {52,41}, {49,49}, {58,48}, {57,58}, {39,10}, {46,10}, {59,15}, {51,21}, {48,28}, {52,33}, {58,27}, {61,33}, {62,63}, {20,26}, {5,6}, {13,13}, {21,10}, {30,15}, {36,16}, {62,42}, {63,69}, {52,64}, {43,67}}.

3.2 Parameter settings of DPAGA

DPAGA is used to solve TSP examples; its parameters are shown in Table 1.

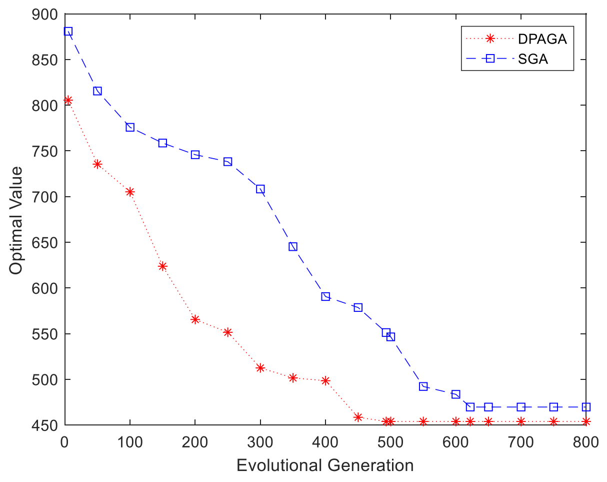

Figure 8: Evolution process of DPAGA and SGA when solving Example 3. Source: Created by the authors.

Table 3: Comparison of solution results

Problem	Algorithm	Shortest path length
Example 1	DPAGA	24.5222
	DACGA	24.5263
	ACO	24.7359
	SGA	24.7713
Example 2	DPAGA	415.7621
	ACO	417.1128
	SA	423.9493
	SGA	424.8692
Example 3	DPAGA	453.9863
	ACO	457.0813
	SA	461.4514
	SGA	469.6949

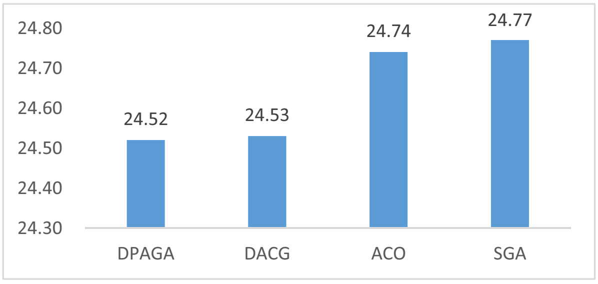

Figure 9: Optimal path length comparison of Example 1. Source: Created by the authors.

3.3 Solution results and analysis

The solution results of DPAGA for the three examples are shown in Table 2, and the optimal paths are shown in Figures 3–5. The evolutionary process of solving the three examples with DPAGA and the simple genetic algorithm (SGA) is shown in Figures 6–8.

The comparison of solution results between DPAGA and the existing basic ACO [7,12], dynamic ant colony genetic algorithm (DACGA) [12], SA algorithm [7], and SGA is shown in Table 3 and Figures 9–11.

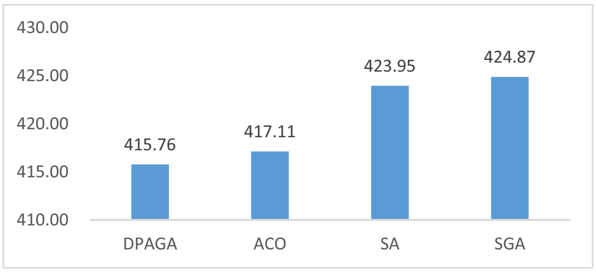

Figure 10: Optimal path length comparison of Example 2. Source: Created by the authors.

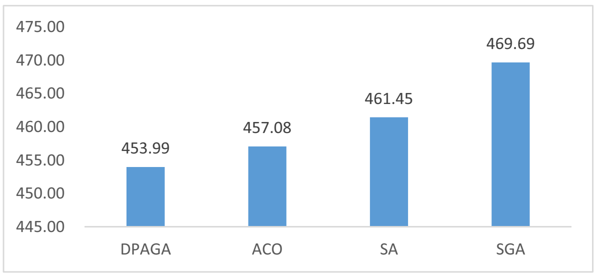

Figure 11: Optimal path length comparison of Example 3. Source: Created by the authors.

The above results indicate that DPAGA can quickly find the optimal solution for the shortest path in TSP. When solving the TSPs with scales of 20, 30, and 50 cities, the optimal path lengths of DPAGA were reduced by a maximum of 0.2491, 9.1071, and 15.7086, respectively, compared with other optimization algorithms. So, the solution results of DPAGA are better than that of existing algorithms.

4 Conclusions

A DPAGA was proposed in this article. This algorithm distinguishes genetic individuals into male and female individuals, and only individuals of the opposite sex can perform crossover operations. The algorithm introduces a dominant population and an auxiliary population, which execute different adaptive crossover and mutation strategies during the evolution process. In this way, the singularity of the population and genetic individuals in genetic algorithms is overcome, and the performance of the algorithm is improved. The DPAGA was applied to solve TSP instances; the experimental results show that the TSP solving method based on DPAGA is not only completely feasible but also has better convergence speed and solution quality than other optimization algorithms. Due to no additional information exchange between the dominant population and auxiliary population, the time complexity of DPAGA does not increase compared with the basic adaptive genetic algorithm, and it does not affect the cost-effectiveness of the algorithm. However, there are still some shortcomings in this study, and the selection of application scope is not enough. In the future, more applications can be expanded for experimentation and make the algorithm more comprehensive.

Funding information: We would like to thank the Science Research Project Foundation of Hunan Province Education Department under Grant No. 22C0357.

Author contributions: Y. T. S. was responsible for manuscript writing, research framework, and algorithm design. C. X. was responsible for algorithm realization, experimental analysis, and language proofreading. Y. B. S. was responsible for language proofreading and data analysis.

Conflict of interest: The authors declare that there is no conflict of interest regarding the publication of this article.

Data availability statement: The data used to support the findings of this study are available from the corresponding author upon request.

References

[1] Wu CY, Fu XS, Pei JK, Dong ZG. A novel sparrow search algorithm for the traveling salesman problem. IEEE Access. 2021;9:153456–71. doi: 10.1109/ACCESS.2021.3128433.
[2] Guo ZH, Jin L, Zheng CY. Study on improved neural network to solve TSP. Comput Simul. 2014;31(4):355–8. doi: 10.3969/j.issn.1006-9348.2014.04.081.
[3] Deb S, Fong S, Tian Z, Mohammed S, Fiaidhi J. Finding approximate solutions of NP-hard optimization and TSP problems using elephant search algorithm. J Super-comput. 2016;72(1):1–33. doi: 10.1007/s11227-016-1739-2.
[4] Ma XS, Gong S, Zhu J, Tang H. Ant colony algorithm of partially optimal programming based on dynamic convex hull guidance for solving TSP. J Commun. 2018;39(10):59–71. doi: 0.11959/j.issn.1000−436x.2018218.
[5] Wang GH. Research analysis of small-scale tsp based on genetic algorithm. Logist Eng Manag. 2022;44(3):111–4. doi: 10.3969/j.issn.1674-4993.2022.03.031.
[6] You ML, Lei XJ. Improved bacterial foraging algorithm for solving TSP. J Guangxi Univ (Nat Sci Ed). 2013;38(6):1436–43. doi: 10.3969/j.issn.1001-7445.2013.06.027.
[7] Duan SS. Analysis of some intelligent algorithms for solving TSP with different scales. Technol Perspect. 2020;21:21–2. doi: 10.19694/j.cnki.issn2095-2457.2020.21.009.
[8] Xu L. Searching the approximate optimal solution of traveling salesman problem by optimized exhaustive method. Microcomput Appl. 1998;17(10):20+30. doi: CNKI:SUN:WXJY.0.1998-10-006.
[9] Chen T, Zhang S. Branch and bound method for solving practical TSP problems. Comput Eng Des. 2009;30(10):2431–4. doi: CNKI:SUN:SJSJ.0.2009-10-026.
[10] Dan K, Duan LZ. A dynamic programming algorithm based on optimal insertion subset for solving traveling salesman problems. Comput Appl Software. 2022;39(12):261–5. doi: 10.3969/j.issn.1000-386x.2022.12.039.
[11] Liu HL, Lei B, Wang WY. An improved fusion genetic grey wolf optimization algorithm for solving Tsp. Comput Simul. 2023;40(9):333–8. doi: 10.3969/j.issn.1006-9348.2023.09.064.
[12] Tao LH, Ma ZN, Shi PT, Wang RF. Dynamic ant colony genetic algorithm based on tsp dynamic ant colony genetic algorithm based on TSP. Mach Des Manuf. 2019;12:147–9. doi: 10.19356/j.cnki.1001-3997.2019.12.037.
[13] Pei JM, Zhou B, Li L. TSP algorithm based on genetic algorithm for solving the shortest journey in 20 cities. Comput Knowl Technol. 2019;15(16):194–5. doi: 10.14004/j.cnki.ckt.2019.2135.
[14] Pan YM, Wu LS. An improved genetic algorithm for solving TSP. J Ningbo Inst Educ. 2010;12(2):74–6. doi: 10.3969/j.issn.1009-2560.2010.02.022.
[15] Ebadinezhad S. DEACO: Adopting dynamic evaporation strategy to enhance ACO algorithm for the traveling salesman problem. Eng Appl Artif Intell. 2020;92:103649. doi: 10.1016/j.engappai.2020.103649.
[16] Li X, Tong BL, Fang TT. TSP solved by the optimal worst ant colony algorithm based on simulated annealing. J Shanxi Norm Univ. 2023;37(2):22–7. doi: 10.16207/j.cnki.1009-4490.2023.02.015.
[17] Cheng BY, Lu HY, Huang Y. Adaptive excellence coefficient particle swarm optimization algorithm for solving TSP. Comput Appl. 2017;37(3):750–4. doi: 10.11772/j.issn.1001-9081.2017.03.750.

[18] Ran LL, Li L, Zheng XD. Solving TSP problem based on improved taboo search algorithm. J Shenyang Univ Aeronaut Astronaut. 2023;40(4):80–7. doi: 10.3969/j.issn.2095-1248.2023.04.01.

[19] Chen KS, Xian SD, Guo P. Adaptive warming simulated annealing algorithm for solving traveling salesman problems. Control Theory Appl. 2021;38(2):245–54. doi: 10.7641/CTA.2020.00090.

[20] Ezugwu ES, Adewumi AO, Marc EF. Simulated annealing based symbiotic organisms search optimization algorithm for traveling salesman problem. Expert Syst Appl. 2017;77:189–210. doi: 10.1016/j.eswa.2017.01.053.

[21] Defersha FM, Rooyani D. An efficient two-stage genetic algorithm for a flexible job-shop scheduling problem with sequence dependent attached/detached setup, machine release date and lag-time. Comput Ind Eng. 2020;9:1–19. doi: 10.1016/j.cie.2020.106605.

[22] Wang MC. Real time path optimization of mobile robots based on improved genetic algorithm. J Syst Control Eng. 2020;235(1):1–10. doi: 10.1177/0959651820952207.

[23] Pan W, Ding LC, Huang F, Sun Y. Adaptive genetic algorithm based on chaos “micro variation”. Control Decis. 2021;36(8):2042–8. doi: 10.13195/j.kzyjc.2021.0319.

[24] Chen QR, Li YL, Xu KQ, Liu YL, Wang SQ. WKNN feature selection method based on self-tuning adaptive genetic algorithm. Comput Eng Appl. 2021;57(20):164–71. doi: 10.3778/j.issn.1002-8331.2012-0021.

[25] Wang H, Zhao XJ, Yuan XJ. Robot path planning based on improved adaptive genetic algorithm. Electron Opt Control. 2022;29(5):72–6. doi: 10.3969/j.issn.1671-637X.2022.05.014.

[26] Zhang HM, Hu F, Duan YF. A nonlinear adaptive genetic algorithm for parameter identification of the BOUC-WEN model and its experimental verification. Eng Mech. 2022;39(6):191–201. doi: 10.6052/j.issn.1000-4750.2021.03.0237.

[27] Li JY, Wang ZH, Zhang JH, Zhang ZJ, Liu NN, Li M. Optimization analysis of canal system water distribution based on improved adaptive genetic algorithm. J Irrig Drain Mach Eng. 2024;42(11):1150–6. doi: 10.3969/j.issn.1674-8530.23.0114.

[28] Qin FF, Zhang JR, Zhang T, Luo SW. Location selection of logistics distribution center based on adaptive genetic algorithm. Logist Eng Manag. 2023;45(8):1–6. doi: 10.3969/j.issn.1674-4993.2023.08.001.