Improved Ant Colony Genetic Algorithm for Solving Traveling Salesman Problem

Wenming Wang*, Jiangdong Zhao, Ji Huang

Department of Experimental and Practical Education, West Anhui University, Lu’an, China

*Corresponding author e-mail: hellowwming@foxmail.com

Abstract. A new improved ant colony genetic algorithm (IACGA) is proposed to solve the problems of the basic ant colony optimization algorithm (ACO), such as the slow convergence rate and the local optimal solution. Based on the ACO model, local pheromone increment is updated differently according to the path quality constructed by ants, the current optimal path pheromone residual factor is adjusted adaptively, pheromone updating rules are improved, genetic operator is embedded and genetic algorithm (GA) is dynamically fused. It makes full use of the positive feedback mechanism of ACO and the searching ability of GA to realize the dynamic fusion of ACO and GA. The experimental results show that compared with the ACO and the GA, the improved algorithm can find the global optimal solution quickly, and the solution quality is relatively good, which improves the efficiency of the ant colony algorithm in solving the traveling salesman problem (TSP).

1. Introduction

Ant colony optimization algorithm (ACO) and genetic algorithm (GA) are widely used intelligent search algorithms in recent years. ACO[1] is a heuristic algorithm that preserves relatively good information in the optimization process through the positive feedback mechanism, so as to find the optimal path. The GA[2] draws on the evolutionary laws of natural organisms and searches for the optimal solution based on the natural laws of survival of the fittest and survival of the fittest. It uses selection, crossover, mutation and other operations to find good individuals to solve the problem. In order to overcome the deficiencies of ant colony algorithm and genetic algorithm and improve the efficiency and optimization ability of the algorithm, many researchers have carried out a lot of research on the improvement and fusion of ant colony algorithm and genetic algorithm.

In order to expand the search space of ants and diversify the potential solutions, Wei Gao[3] proposed a new ant colony algorithm that uses a pair of search ants combination strategy and introduces a threshold constant to reduce the impact of a limited number of encountering ants. Dongping Qiao et al.[4] proposed to introduce positive and negative feedback mechanism to update each path pheromone adaptively, so that the algorithm can make full use of the better path information. At the same time, pairwise exchange strategy is used to locally search the optimal path in each cycle. In terms of algorithm fusion, Lihua Tao[5] et al. proposed a new dynamic ant colony genetic algorithm, which dynamically fused the genetic algorithm and the ant colony algorithm, and improved the re-insertion operation of the traditional genetic algorithm.
2. The Principle of Solving TSP with Basic ACO

2.1. Path Construction
In the basic ACO, the ants are initially randomly distributed in each city. The next city the ant will visit will be chosen according to the rule of random proportion.

\[
P_{ik}^k(t) = \left\{ \begin{array}{ll}
\frac{\tau_{ij}(t)\eta_{ij}(t)^\beta}{\sum_{j \in allowed_k} \tau_{ij}(t)\eta_{ij}(t)^\beta}, & j \in allowed_k \\
0, & j \notin allowed_k
\end{array} \right.
\]  

(1)

In formula (1), \(\tau_{ij}(t)\) denotes the pheromone quantity on the path \((i, j)\) at time \(t\). \(\eta_{ij}(t)\) is the heuristic quantity on the path \((i, j)\), usually defined as \(\eta_{ij}(t) = 1/d_{ij}\). where \(\alpha\) and \(\beta\) are two parameters that determine the relative influence of the pheromone trail and the heuristic information. \(allowed_k\) is the feasible neighborhood of ant \(k\) when it is at city \(i\), that is, the set of cities that ant \(k\) has not visited yet.

2.2. Pheromone update
After the ant completes a round trip, the pheromones in the path are updated according to the following rules:

\[
\tau_{ij}(t + n) = (1 - \rho) \cdot \tau_{ij}(t) + \Delta\tau_{ij}
\]

(2)

\[
\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^k
\]

(3)

\[
\Delta\tau_{ij}^k = \begin{cases} 
\frac{Q}{L_k}, & \text{if ant } k \text{ travels on edge } (i, j) \\
0, & \text{otherwise}
\end{cases}
\]

(4)

Where \(\rho\) is the pheromone volatility coefficient to avoid unlimited accumulation of pheromones, it must be less than 1. \(\Delta\tau_{ij}\) represents the increment of pheromone on the path \((i, j)\) during the tour, that is, the amount of pheromone accumulated by ants passing through the path. \(\Delta\tau_{ij}^k\) represents the pheromone amount released by the ant \(k\) through the path during the tour. Where \(Q\) is the quantity of pheromone, which is a constant. \(L_k\) represents the path length of the ant \(k\) in this tour.

3. Improved Ant Colony Genetic Algorithm

3.1. Differential update of local pheromone increment
In an iteration, when the path length of ant search is less than the average path length of all ant searches in this iteration, the pheromone increment on the current path is strengthened to make the subsequent ants continue to explore better paths on the path. When the path length of the ant search is greater than the average path length of the current iteration, the pheromone increment on the current path will be weakened, so that the later ants will weaken the search on the poor path. By updating the local pheromone increment of different quality paths, the attraction of the better path to the ant is increased, and the interference of the poor path to the ant selection is reduced. The pheromone of the better path is fully utilized to prevent the algorithm from falling into the local optimum. The development ability of the algorithm is strengthened, and the better global search ability of the algorithm is maintained.

The local pheromone incremental differential update rule is calculated as follows:

\[
\Delta\tau_{ij}^* = \sum_{k=1}^{m} \Delta\tau_{ij}^k
\]

(5)

\[
\Delta\tau_{ij}^k = \begin{cases} 
\frac{L_{iave} - L_k}{L_{iave} - L_{ib}} \cdot \frac{1}{L_{ib}}, & \text{if } L_{iave} \neq L_{ib} \\
\frac{Q}{L_{ib}}, & \text{if } L_{iave} = L_{ib}
\end{cases}
\]

(6)
Where $\Delta \tau_{ij}$ represents the differential increase of local pheromone on the path $(i,j)$ after a trip. When the path length($L_k$) searched by ant $k$ is greater than the average path length($L_{ave}$), the pheromone increment left by ant $k$ on the path $(i,j)$ is negative, which weakens the pheromone on the path. When the path length of ant search is less than the average path length, the pheromone increment left on the path is positive, and the pheromone on the path is strengthened. When the average path length in this iteration is equal to the optimal path length($L_{opt}$) in this iteration, it indicates that the algorithm has converged, find the optimal path, and enhance the pheromone on the path.

3.2. Adaptive adjustment of pheromone residual factor of current optimal path

In the initial stage of the iteration, the path that the ant seeks is relatively single, the heuristic guidance of ants is stronger, after the ants pass through easily affected by the heuristic cause path pheromone amount increase, cause local optimum, so at the beginning of the iteration, adjusting pheromone residual factor, reduce the residual pheromone, increase diversity in the early iterations, improve the global search ability. With the increase of the number of iterations, the ants traverse more paths and release the amount of pheromone on the better path. At this time, the pheromone residual factor slowly increases and gradually searches for the better path. At the late stage of iteration, when the ant finds a better path, it needs to conduct continuous search on the better path. At this time, pheromone residual factor is the largest, which makes the ant conduct continuous search towards the optimal path and improves the convergence speed in the later stage.

The current optimal path pheromone residual factor $\rho$ is set and make its value obey exponential distribution with the increase of iteration times. Improvement of pheromone residual factor $\rho$:

$$\rho(\text{iter}) = 0.9 \times e^{\frac{\text{iter}-\text{iter}_{max}}{\text{iter}_{max}}}$$

(7)

Where $\text{iter}$ is the current number of iterations, $\text{iter}_{max}$ represents the total number of iterations. 0.9 is the control coefficient, and the maximum value of $\rho$ is 0.9.

3.3. Improved pheromone update rules

After the ant completes a tour, it updates the pheromone. The improved pheromone updating rule is divided into four steps: the first step is the pheromone evaporation in the path traveled by each ant, the second step is the local pheromone incremental differential update, the third step is the pheromone enhancement in the optimal path of the current iteration, and the fourth step is the adaptive pheromone enhancement in the current global optimal path. The improved pheromone updating process is as follows:

$$\tau_{ij}(t + 1) = (1 - \rho)\tau_{ij}(t) + \Delta \tau^*_ij + \Delta \tau^{ib}_{ij} + \Delta \tau^{gb}_{ij}$$

(8)

$$\Delta \tau^{ib}_{ij} = \begin{cases} \frac{Q}{L^{ib}}, & \text{if edge } (i,j) \text{ is on the path } T^{ib} \\ 0, & \text{otherwise} \end{cases}$$

(9)

$$\Delta \tau^{gb}_{ij} = \begin{cases} e \times \frac{Q}{L^{gb}}, & \text{if edge } (i,j) \text{ is on the path } T^{gb} \\ 0, & \text{otherwise} \end{cases}$$

(10)

$$\Delta \tau^{gb}_{ij} = \rho(\text{iter}) \times \tau^{gb}_{ij}$$

(11)

Where $\Delta \tau^{ib}_{ij}$ is the current iteration pheromone increment. $T^{ib}$ is the optimal path for the current iteration, $L^{ib}$ is the length of the optimal path for the current iteration. The optimal path in the current iteration may be the same length with different paths or the same path. Only different paths of the same optimal path length are preserved here in the current iteration. Where $\Delta \tau^{gb}_{ij}$ is the global pheromone increment, $T^{gb}$ is the current global optimal path, $L^{gb}$ is the length of the current global optimal path. The current global optimal path length includes the same and different global optimal paths. Where $e$ represents the number of edges $(i,j)$ in the current global optimal path.
3.4. Embedded genetic operator and dynamic fusion genetic algorithm

The genetic operator was embedded in the ant colony algorithm to optimize the results of ant colony traversal again, and the better individual was retained. The pheromone distribution in the path of the better individual was updated according to the better individual to increase the ant's exploration ability. At the same time, the positive feedback mechanism of ant colony algorithm is used to improve the initial population of genetic operator and improve the search efficiency of genetic operator. By embedding genetic operator into ant colony algorithm, the optimization performance of the algorithm is improved.

With the increase of iteration times, ant colony algorithm will continue to seek optimization in the direction of better solution set (better path), and the algorithm will gradually converge to a better solution set, so it may fall into a local optimal solution. Cumulative ant colony algorithm for finding the optimal path length of the number of iterations, when the number of iterations reaches a certain algebra, the ant colony algorithm to obtain the optimal solution set (global optimal path) as the initial population of genetic algorithm, dynamic integration of crossover and mutation genetic algorithm with heavy insert operation, continue to the optimal solution set in the current exploration on the basis of the optimal solution. The ant colony algorithm is used to explore the current optimal path, and the genetic algorithm is dynamically integrated to avoid the local optimal solution of the algorithm, which enhances the exploration ability of the algorithm.

4. Simulation Experiments

In order to verify the effectiveness of the improved algorithm, the experiment uses Matlab2016a software to complete the program development of ant system algorithm(AS[6]), ant colony system algorithm(ACS[7]), GA and IACGA. Simulation experiments and algorithm tests are carried out for typical TSP models in TSPLIB benchmark library. The simulation experiment environment is: Windows10 operating system, AMD processor, main frequency 2.0GHZ, 8G memory. Each instance is solved 20 times, and 600 iterations are carried out each time.

The parameters of the algorithm are set as follows: \( n \) represents the number of cities, \( m \) represents the number of ant and \( m = n \), \( \alpha = 1 \), \( \beta = 5 \), \( \rho = 0.1 \), \( Q = 20 \), \( \text{iter\_max} = 600 \), the selection probability is set as \( p_s = 0.9 \), the crossover probability is set as \( p_c = 0.9 \) and the mutation probability is set as \( p_m = 0.1 \).

| Instances | Algorithm | Best  | Worst | Average | PD (%) | opt   |
|-----------|-----------|-------|-------|---------|--------|-------|
| att48     | AS        | 34562 | 35187 | 35036   | 3.10   | 33522 |
|           | ACS       | 34442 | 35014 | 34791   | 2.69   |       |
|           | IACGA     | 33549 | 33930 | 33647   | 0.00   |       |
| st70      | AS        | 700   | 713   | 708     | 3.70   | 675   |
|           | ACS       | 693   | 702   | 698     | 2.67   |       |
|           | IACGA     | 732   | 770   | 751     | 8.44   |       |
| lin105    | AS        | 14952 | 15122 | 15028   | 3.98   | 14379 |
|           | ACS       | 14807 | 14975 | 14905   | 2.98   |       |
|           | IACGA     | 15857 | 24401 | 18868   | 10.28  |       |

As shown in Table 1, in solving att48, st70 and lin105 respectively the three different city model, the minimum values of the optimization results of the IACGA are all smaller than those of the AS, ACS and GA. The average of 20 trials is also the smallest among the four algorithms, and in the process of solving the smaller att48 city model, the IACGA can stably find the currently known global optimal solution. The results show that the optimization ability and stability of the IACGA are much better than that of the AS, ACS and GA. Table 2 shows the optimal path of IACGA for solving att48, st70 and lin105 city models.
Table 2 Optimal Path Obtained by Improved Ant Colony Genetic Algorithm

|     | att48 | st70 | lin105 |
|-----|-------|------|--------|
| IACGA | Shortest path length | Average path length | Worst path length |
|      |         |         |        |

Fig.1 is a graph of the change of the shortest path length, average path length and worst path length with the number of iterations using the IACGA, taking the att48 city model as an example. In the early stage of the iteration, improved pheromone update rules and embedded genetic operators are used to quickly accumulate pheromone on a better path and speed up the convergence of the algorithm. It can be seen from Fig.1 that the shortest path length, average path length, and worst path length converge quickly as the number of iterations increases, and the algorithm explores a better solution set faster. In the middle phase of iteration, the algorithm will come to a standstill. At this point, the GA is fused to further search for the optimal solution and explore the global optimal solution. The improved algorithm shows faster convergence speed and stronger global optimization ability.

Fig.2 shows the iterative convergence curve of AS, ACS and IACGA to solve the att48 city model. It can be seen from the figure, the AS converges too early and is prone to premature convergence, resulting in the occurrence of local optimal solutions. ACS can further optimize and explore better solutions compared with the AS, but it may also search for local optimal solutions. Combined with
Table 1, it can be seen that the IACGA has good effect compared with the AS, ACS and GA in terms of average value, optimal value and iterative convergence curve.

Fig. 2  Iterative convergence curves

5. Conclusion
In this paper, an improved ant colony genetic algorithm is proposed. According to the quality of the path constructed by ants, the local pheromone increments of different paths are updated differently. Meanwhile, the pheromone residual factor of the current optimal path are adaptively adjusted and improved the pheromone update rules. After all ants complete a random traversal, the genetic operator is embedded to optimize, and the pheromone is updated according to the optimization results, when the improved ant colony algorithm falls into the current optimal solution set, the dynamic fusion genetic algorithm is continuously explored to further optimize the feasible solution. The simulation results show that the improved algorithm has fast convergence speed and strong global optimization ability in solving TSP.

Acknowledgments
This work was financially supported by the West Anhui University Natural Youth Project (No. WXZR201912), the Ministry of Education Industry-University Cooperation Collaborative Education Project (No. 201902289017), the Quality Engineering Project in Anhui Province (No.2018xfsyxm041) and the Anhui Province University Outstanding Young Talent Support Program Project (No. gxyq2019072).

References
[1] M. Dorigo, T. Stutzle. Ant Colony Optimization[M]. The MIT Press, Cambridge, 2004.
[2] Yong Deng, Yang Liu and Deyun Zhou. An Improved Genetic Algorithm with Initial Population Strategy for Symmetric TSP[J]. Mathematical Problems in Engineering, 2015,2015(Pt.19):212794.1-212794.6.
[3] Wei Gao. New Ant Colony Optimization Algorithm for the Traveling Salesman Problem[J]. International Journal of Computational Intelligence Systems,2020,13(1):44-55.
[4] Dongping Qiao, Jie Pei, Hao Li, et al. Research on Improving Ant Colony Algorithm to Solve TSP Problem [J]. Machinery Design & Manufacture, 2019, (10):144-149.
[5] Lihua Tao, Zhennan Ma, et al. Dynamic Ant Colony Genetic Algorithm Based on TSP[J]. Machinery Design & Manufacture, 2019, (12):147-149, 154.
[6] M. Dorigo, V. Maniezzo and A. Colorni. Ant system: optimization by a colony of cooperating agents[J]. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 1996, 26(1): 29-41.

[7] M. Dorigo and L. M. Gambardella. Ant colony system: a cooperative learning approach to the traveling salesman problem[J]. IEEE Transactions on Evolutionary Computation, 1997, 1(1): 53-66.