A Modified Ant Colony Algorithm for Traveling Salesman Problem

X. Wei, L. Han, L. Hong

Xianmin Wei*, Liqi Han, Lu Hong
School of computer engineering, Weifang University, Weifang 261061, P.R. China
weixm@wfu.edu.cn, liqi.han@wfu.edu.cn, sdkfz234@163.com
*Corresponding author: weixm@wfu.edu.cn

Abstract: Ant colony algorithms such as Ant Colony Optimization (ACO) have been effectively applied to solve the Traveling Salesman problem (TSP). However, traditional ACO algorithm has some issues such as long iterative length and prone to local convergence. To this end, we propose we embed ACO into Cultural Algorithm (CA) framework by leveraging the dual inheritance mechanism. Best solutions are evolved in both population space and belief space, and the communication between them is achieved by accept and influence operations. Besides, we employ multiple population spaces for parallel execution. Experiments show that the performance of our proposed algorithm is greatly improved.

Keywords: Traveling Salesman Problem (TSP), Ant Colony Optimization (ACO), Cultural Algorithm (CA).

1 Introduction

Traveling Salesman Problem (TSP) is a typical NP hard problem [1]. Given a number of cities and the distances between them, TSP aims to calculate the shortest tour for the salesman to visit each city exactly once and return to the starting point. Solving TSP problem can be beneficial to applications such as deployment of network devices [2], transportation [3] and traffic control [4].

Ant Colony Optimization (ACO) algorithm [5] is one possible solution for solving TSP problem. ACO simulates the foraging process of ants, where the routes with more pheromones are more likely to be selected. The process of applying ACO algorithm for TSP can be described as follows. Suppose there are ants and cities. The movement of each ant is to choose the next unvisited city to visit by certain rules, and meanwhile update the amount of pheromones on that route.

Although the advantages of ACO, such as positive feedback, distributed, parallel and self-organization, facilitate the solving of TSP problem, there still remain some issues. For example, at earlier iterations, the amount of pheromones is relative exile, so the accumulation of enough pheromones costs a long time. Besides, at later iterations, the positive feedback mechanism greatly increases the possibility of local convergence.

To deal with the above long iteration time and local convergence issues, we propose an improved algorithm in this paper. The general idea is to accelerate the evolution and revise the best solution at each iteration. To this end, we leverage the Cultural Algorithm (CA) [6] framework to embed the traditional ACO algorithm. Indeed, as indicated in [7, 8], it is feasible and efficient to adapt CA for optimize ACO. Our experiments prove that the dual inheritance mechanism of CA helps to improve the performance of ACO.

The remainder of this paper is organized as follows. Section 2 reviews related work. The proposed algorithm is discussed in Section 3. Empirical experiments of evaluating the improved algorithm are conducted in Section 4. Finally, the paper is concluded in Section 5.
2 Related work

Many efforts have been made to solve the problems of traditional ACO algorithm. For example, Dorigo et al. [9] proposed an ant system with positive feedback, distributed computation, and a constructive greedy heuristic. Bullnheimer et al. [10] proposed a new rank based version of ant system. Gutjahr et al. [11] proposed a graph based ant system. Gambardella et al. [12, 13] introduced Q-learning to ant system for solving TSP problem. Cornei et al. [14] developed a hybrid algorithm with ant colony and genetic algorithm, so that the proposed algorithm can achieve faster convergence and better search capability for global best solution. Stutzle et al. [15] designed a Max-Min Ant System (MMAS) for better exploitation ability and accelerated the accumulating the pheromone of the best solutions. Walter and Thomas et al. proved the convergence of ACO algorithm [16–18].

Indeed, ACO algorithms have been applied to many problems such as the TSP problem. Zhou et al. [19] provided the theoretical convergence analysis on the traditional ACO, and proved the feasibility of applying ACO algorithm for TSP. Zhang et al. [20] proposed an adaptive heterogeneous multiple ACO to solve TSP, which used an evolution coefficient to evaluate the solutions obtained by the ant colony and then update the phenomenon. Tuba et al. [21] modified ACO by improving the pheromone correction mechanism, where the pheromone values for highly undesirable links are significantly lowered by a posteriori heuristic. Negulescu et al. [22] presented an adapted ACO that incorporates methods and ideas from genetic algorithms (GA) by simulating artificial ants with different behaviors as synthetic genes. Besides, ACO has been employed in many applications. Gajpal et al. [23] applied ACO to the route selection of tucks, and the search all the customers during the routing path using a multi-route search principle. Hu et al. [24] employed a new method to update the phenomenon to deal with the continuous optimization problems. Ghoseiri et al. [25] developed an algorithm to solve the dual-object shortest path problem with a multi-objective ACO algorithm. Gajpal et al. [26] proposed a modified ACO method for a vehicle routing problem. Secui et al. [27] applied ACO for optimal allocation of capacitor banks in electric power distribution networks.

Cultural Algorithm (CA) was first proposed by Reynolds [6]. Then, Chung et al. [28] developed a new framework to embed the evolutionary programming into the population space to simulate the evolution process. Liu et al. [29] combined CA with particle swarm optimization algorithm and applied it to the numerical optimization problem. Ma et al. [30] applied CA to solve the TSP problem and utilized the idea from simulated annealing to ensure the accuracy and efficiency. Unlike existing work, in this paper we design a modified ACO algorithm by leveraging CA framework, that is, to employ a dual inheritance mechanism into the traditional ACO algorithm. Besides, we apply the modified algorithm on TSP problem. Note that, for the consideration of parallelism, we use multiple population spaces.

3 Proposed algorithm

The disadvantages of the traditional ACO algorithm include slow convergence and prone to stagnation. Although there are some modifications over the traditional ACO version, most of them are built upon the traditional computing structure to determine the internal status of the algorithm and estimate the search capabilities of the ant colony through qualitative analysis. For example, the improvements are made by updating pheromones, changing evaporation coefficient or the amount of released pheromones in order to avoid stagnation.

Different from existing efforts, we embed the ACO algorithm into another framework, Cultural Algorithm (CA) [6], which could essentially improve the performance. The intuitive is that both algorithms are population based and share information through interactions among the
population. We perceive that the ACO algorithm can benefit from the dual inheritance of CA. Besides, the inherent nature of parallelism of ACO indicates that parallel algorithm can improve the efficiency due to the mature of hardware and software. Therefore, in this section, we propose a parallel ACO algorithm based on CA algorithm, and then apply it for solving the traditional TSP problem.

![Figure 1: The framework of culture algorithm](image)

CA simulates the evolution process of human society, which regards culture as the information carrier. While the culture is accepted by the whole population of the society, it can be used to direct the behavior of each individual in turn. There are two evolution spaces in CA algorithm: belief space, which is composed of the experience knowledge acquired during the evolution process; and population space, which is composed of the individuals. These two spaces communicate through specific protocols. The framework of CA is illustrated as Figure 2. Generally, the lower space contributes to the upper space periodically, and the upper space continuously evolves, which in turn influences the lower space. This is called dual inheritance mechanism.

As shown in Figure 2, the major operations of CA algorithm are `accept()` and `influence()`, while other operations are performed inside belief or population space independently. Therefore, it is feasible to embed other algorithms into CA framework by adding specific logics into belief and population spaces and implementing `accept()` and `influence()` operations between them. The parallel version of CA is illustrated as Figure 3, where exist multiple population spaces. Each population space performs parallel evolution, and then produces local best individuals for next generation. Then `accept()` operation is performed by all population spaces periodically to update belief space in a synchronous way. On the other hand, belief space conducts evolution as well, and calls `influence()` operation to direct the evolution process of each population space.

### 3.1 Designing population space

Let $m$ be the number of ants, $n$ be the number of cities $C = \{C_1, C_2, \ldots, C_n\}$, $d_{ij}(i, j = 1, 2, \ldots, n, \ i \neq j)$ be the distance between any two cities, and $\tau_{ij}$ be the amount of pheromones from $i$ to $j$. At the beginning, $\tau_{ij}$ is initialized as a constant, that is, $\tau_{ij}(0) = c$, where $c$ is a constant. At $t$-th iteration, the pheromones on path $i, j$ is noted as $\tau_{ij}(t)$. According to MMAS [12] algorithm, $\tau_{ij}(t)$ is restricted between $\tau_{\text{min}}$ and $\tau_{\text{max}}$ in order to avoid premature local convergence.
\[ \tau_{\text{max}}(t) = \frac{1}{2(1 - \rho)L(t)} + \frac{\sigma}{L(t)} \] (1)

\[ \tau_{\text{min}}(t) = \frac{\tau_{\text{max}}(t)}{20} \] (2)
where $L(t)$ is the length of the best solution at iteration $t$, $\rho$ is the evaporation coefficient of pheromones, and $\sigma$ is the number of best solutions at iteration $t$.

Suppose $\eta_{ij}(t)$ denotes the heuristic information on path $(i, j)$. The general idea is that ants would select closest path with larger amount of pheromones. Therefore, the probability of ant $k$ transferring from city $i$ to $j$ is calculated as:

$$p_{ij}^k(t) = \frac{\sum_{j \in \text{allowed}_k} [\tau_{ij}(t)]^{\alpha} [\eta_{ij}(t)]^{\beta}}{\sum_{j \in \text{allowed}_k} [\tau_{ij}(t)]^{\alpha} [\eta_{ij}(t)]^{\beta}}, \quad j \in \text{allowed}_k,$$

where $\text{allowed}_k$ denotes the available nodes to choose at the next step for ant $k$, $\alpha$ is the heuristic factor, meaning the importance of path with remaining pheromones, $\beta$ is the heuristic factor for $\eta_{ij}(t)$, denoting the affect of heuristic information.

After ants iterate all the nodes, remaining pheromones are updated as follows:

$$\tau_{ij}(t + 1) = (1 - \rho)\tau_{ij}(t) + \sum_{k=1}^{m} \Delta \tau_{ij}^k(t)$$

where $1 - \rho$ is the residual coefficient of pheromones, and $\rho \in [0, 1]$. $\Delta \tau_{ij}^k(t)$ is the amount of pheromones remaining on the path at current iteration for ant $k$, which can be calculated as:

$$\Delta \tau_{ij}^k(t) = \begin{cases} \frac{Q}{L_k}, & \text{if ant } k \text{ passes } (i, j) \text{ at current iteration}, \\ 0, & \text{otherwise}, \end{cases}$$

where $Q$ is a constant, and $L_k$ is the total length of ant $k$’s tour.

### 3.2 Designing belief space

Belief space is responsible for updating knowledge, that is, to further optimize the best solutions provided by the population space, which are transferred through $\text{accept()}$ operation. For TSP problem, we employ 3-OPT algorithm to evolve belief space. Suppose there exist any three nodes $i, j, k$, and current best solution is $C = \{C_n \cdots C_iC_{i+1} \cdots C_jC_{j+1} \cdots C_kC_{k+1} \cdots C_l\}$. As shown in Figure 4, if $d(C_i, C_{i+1}) + d(C_{j+1}) > d(C_i, C_j) + d(C_{i+1}, C_{j+1})$, then reverse sort is performed on $C = \{C_{i+1}, \ldots, C_j\}$; if $d(C_j, C_{j+1}) + d(C_k, C_{k+1}) > d(C_j, C_k) + d(C_{j+1}, C_{k+1})$, then reverse sort is performed on $C = \{C_{j+1}, \ldots, C_k\}$.

### 3.3 Accept operation

We accept a fixed ratio of all population spaces as the belief space, e.g. 20%. The $\text{accept()}$ operation passes the local best solution of each population space $R^*$ and the corresponding length of route $L^*$ to the belief space.

When $\text{accept()}$ operation is performed, the relative poor individuals are replaced by the current best one of each population space. Suppose $t_{\text{current}}$ is the current iteration, $t_{\text{end}}$ is the predefined maximum iterations, and $t_{\text{accept}}$ is the iteration where $\text{accept()}$ is called, which can be calculated as:

$$t_{\text{accept}} = \text{fix}\left(C_1 + \frac{t_{\text{current}}}{t_{\text{end}}}C_2\right)$$

where $\text{fix()}$ is a rounding function, and $C_1, C_2$ are constants.
3.4 Influence operation

After further optimization via 3-OPT in belief space, influence() operation passes the shortest path and its length to each population space periodically, and updates the global pheromones. Suppose the iteration where influence() operation is performed is $t_{\text{influence}}$, which can be represented as:

$$t_{\text{influence}} = \text{fix}\left(C_1 + \frac{t_{\text{end}} - t_{\text{current}}}{t_{\text{end}}}C_2\right)$$  \hfill (7)

where fix() is a rounding function, and $C_1$, $C_2$ are constants.

The improved algorithm has one belief space and multiple population spaces. Each space evolve independently, and the communication between them is achieved by accept() and influence() operations. The process flow of belief space is shown in Figure 5. In belief space, the update is achieved by 3-OPT algorithm for computing the shortest path. When accept() is called, the solution is updated by the local best from population space. When influence() is called, the global best solution in belief space is transferred to each population space. If the termination condition is satisfied, the algorithm is completed.

As shown in Figure 6, the update of population space is achieved by ACO algorithm. When accept() is called, the local best solution from each population space is updated up to the belief space. When influence() is called, the global best solution in belief space is transferred down to each population space. If the termination condition is satisfied, the algorithm is completed.

4 Experiment

The environment of our experiments is Intel 2.3 GHz, 2GB RAM, Windows 7 operating system, and MATLAB 7.0 for programming. The dataset is achieved from TSPLIB library. The parameters settings are as follows: $\rho = 0.5$, $\alpha = 1$, $\beta = 5$, $Q = 100$, $t_{\text{end}} = 200$. We use 4 population spaces, and the maximum number of evolution of both belief space and population space is 200. We compare our algorithm with the traditional ACO. Both algorithms are ran independently for 50 times, and the average execution time is supposed to the performance.

As shown in Table 1, we evaluate three examples from TSPLIB, i.e., eil51, berlin52 and st70. We can see that our proposed algorithm outperforms the typical ACO method for solving TSP problem. Moreover, the best solution provided by our method is much better than the known best.
Figure 5: Flow chart of evolution of belief space

Table 1: Results of ACO and our algorithm with different datasets

<table>
<thead>
<tr>
<th>Example</th>
<th>Example cities</th>
<th>Example solution</th>
<th>Example ACO</th>
<th>Our algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>eil51</td>
<td>51</td>
<td>426</td>
<td>428.87</td>
<td>412.12</td>
</tr>
<tr>
<td>berlin52</td>
<td>52</td>
<td>7542</td>
<td>7542</td>
<td>7088.34</td>
</tr>
<tr>
<td>st70</td>
<td>70</td>
<td>675</td>
<td>677.11</td>
<td>652.80</td>
</tr>
</tbody>
</table>

Besides, take eil51 as an example, we compare the best tour and convergence speed between our algorithm and traditional ACO for TSP in Figure 7. The blue line denotes the performance of ACO, while the green line is for our algorithm. The result shows that our algorithm can achieve shorter tour with rapid speed of convergence.
Figure 6: Flow chart of evolution of population space

Figure 7: Evolution of best tour length of eil51 for ACO and proposed algorithm
5 Conclusions

In this paper, we propose a new ant colony algorithm and apply it to the typical traveling salesman problem. Specifically, we embed the traditional ACO algorithm into cultural algorithm framework, where the set of ants makes up the population space, and the best solutions found at specific iteration are updated up to the belief space. The evolution of population space is indeed based on ACO, and that of the belief space is based on 3-OPT algorithm for calculating the shortest path. The communication between population space and belief space is performed by accept and influence operations. Moreover, to minimum the execution cost, we design multiple population spaces for parallel execution. The experiments evaluate that our algorithm outperform the traditional ACO algorithm.

Acknowledgments

This work was partially supported by Shandong Natural Science Foundation (ZR2011FL006), Shandong Science and Technology Development Plan (2011YD01044), Shandong Spark Program (2012XH06005), and Weifang municipal Science and Technology Development Plan (201301050).

Bibliography


