Hybrid Lion Swarm Optimization Algorithm for Solving Traveling Salesman Problem

Yan Yang*, Shengjian Liu and Lin Luo
Department of Games, South China Institute of Software Engineering Guangzhou University, Guangzhou, Guangdong, 510990
*e-mail: yangyan_08@yeah.net

Abstract. In view of Traveling Salesman Problem (TSP) is a NP hard problem, Lion Swarm Optimization (LSO) Algorithm is a new swarm intelligence optimization algorithm. Therefore, in this paper, it uses improved hybrid LSO algorithm to solve TSP. Because the basic LSO algorithm is mainly used to solve the continuous problem, and the TSP is a discrete combinatorial optimization problem, this paper improves the LSO algorithm, introduces the greedy strategy, and combines with the Population Competitive Search (PCS) algorithm in the later stage of the search. As a result, the convergence speed is accelerated and the search efficiency is improved. The classic TSP example is tested, and the simulation results show that the algorithm proposed in this paper has strong stability and is an effective method for solving TSP.

1. Introduction
Traveling salesman problem (TSP) has been proved to be a typical NP-hand problem which is difficult to solve. It is widely used in the fields of production line layout, inventory distribution and engineering optimization. Therefore, finding the solution of TSP has important theoretical significance and practical application value. The current methods for solving TSP are mainly divided into two categories: one is the accurate algorithm, such as dynamic programming method, branch and bound method, Djikstra algorithm, Floyd algorithm, Bellman-Ford algorithm and improved SPFA algorithm, etc. [1]. The ideas of these algorithms are easy to understand and can get the optimal solution in theory, but their computational efficiency is not satisfactory and it is difficult to solve large-scale problems. The other is heuristic algorithm, such as genetic algorithm, simulated annealing algorithm [2], particle swarm optimization (PSO) algorithm [3], discrete artificial bee algorithm [4], discrete differential evolution algorithm [5], Shuffled Frog-Leaping Algorithm [6] and ant colony optimization algorithm [7]. Although these algorithms can not guarantee that the optimal solution of the problem can be obtained in a limited time, a large number of experiments show that the error of the optimal solution obtained by these heuristic algorithms can reach a satisfactory degree. Therefore, the heuristic group optimization method has become an effective method for solving TSP.

LSO algorithm is a new swarm intelligence algorithm for solving continuous domain optimization problems [8]. It was proposed by Liu Shengjian, Zhou Yongquan and me in 2018. The algorithm has good robustness and strong global optimization ability, and the application of the algorithm has not appeared yet. In view of this, this paper improves the LSO algorithm. Combined with the PCS algorithm [9], the hybrid LSO algorithm is formed and applied to solve the TSP. Finally, in order to verify the performance of the algorithm proposed in this paper, several classical TSP problems are simulated. The experimental results show that the algorithm proposed in this paper can get the optimal
solution of the problem quickly in solving small and medium-sized TSP problems. The effectiveness of the algorithm is verified.

2. Basic lion swarm optimization (LSO) algorithm [8]

The main idea of the LSO algorithm is as follows. Firstly, the algorithm randomly initializes the positions of the lions in the search space, and the position with the best fitness value is set to the position of the lion king. Secondly, several lions are chosen as hunting lions (also called lioness) and cooperate with each other to hunt. Once the prey is better than the prey of lion king, the position of the prey will be owned by the lion king. Then, the cubs follow the lioness to learn to hunt or eat near the lion king, and will be driven out of the herd in adulthood. In order to survive, the driven lions will try to get closer to the best place in memory. Finally, according to the division of labor and cooperation, the lions keep searching repeatedly to get the optimal value of the objective function.

The lion king, lionesses and cub update their positions according to formulas (1), (2) and (3) respectively:

\[
x_{i}^{k+1} = g^{k} (1 + \gamma \cdot \| p_{i}^{k} - g^{k} \|) \\
x_{i}^{k+1} = \left( \left( p_{i}^{k} + p_{c}^{k} \right) / 2 \right) (1 + \alpha_{f} \cdot \gamma) \\
x_{i}^{k+1} = \left( \left( p_{i}^{k} + p_{c}^{k} \right) / 2 \right) (1 + \alpha_{c} \cdot \gamma) \\
\]

Where \( \alpha_{f} \) and \( \alpha_{c} \) are the disturbance factors of the moving range of lionesses and cubs, and \( \gamma \) is a random generated according to the normal distribution \( N(0,1) \), \( p_{i}^{k} \) is the historical optimal position of the lion \( i \) \((i = 1,2,\ldots,N)\) in the \( k \)-th iteration, \( g^{k} \) is the best position of the population in the \( k \)-th iteration, and \( p_{c}^{k} \) is the historical best position of a hunting partner randomly selected from the other lioness in the \( k \)-th iteration. \( g = \text{low} + \text{high} \cdot g^{k} \) and it is the position where the cub \( i \) is expelled in the hunting range. \text{low} and \text{high} are the minimum and maximum values of each dimension in the living space of lions. \( p_{m}^{k} \) is the best historical position of the lioness followed by the young in the \( k \)-th iteration, and the probability factor \( q \) is the uniform random value generated according to the uniform distribution \( U[0,1] \).

3. Hybrid LSO (HLSO) algorithm for solving TSP

The basic LSO algorithm is suitable for solving the continuous function problem, and the TSP is a discrete combinatorial optimization problem. Therefore, it is necessary to transform the model of TSP.

The general description of the TSP is as follows: a traveling salesman wants to visit \( n \) locations, visit each location only once, and finally return to the initial location to find a shortest access path [10].

Let \( S \) be a sequence encoded as city numbers, for example, the loop \( S = (0,1,2) \) is: \( c0 \rightarrow c1 \rightarrow c2 \rightarrow c0 \), the length of the loop is denoted by \( \text{len} (S) \), and the goal of TSP is to minimize \( \text{len} (S) \).

3.1. Improved LSO algorithm for solving TSP

The optimization process of LSO algorithm is to find the optimal value by updating the respective positions of lions. In the TSP problem, the optimal value found by lions is the set of all cities on the shortest path, and the updating process of lion swarm location is the process of finding the next city in the optimal path. According to the different ways of position updating of different kinds of lions, three methods of position updating are proposed in this paper.
3.1.1 Method 1 (random mutation method). Randomly choose one of the 10 cities closest to the current city as the next city. If there are no more than 10 cities around, then randomly choose one of the remaining cities as the next city, and this city is the new location of the lion.

3.1.2 Method 2 (greedy-weigh method). The six cities nearest to the current city are set as a neighborhood, and the next city must be one of the neighborhoods. Every city in the neighborhood has a weight $Q_i$ with a value of $[1/(\text{dis}_i + 10^{-6})]^{\beta}$, where $\beta = 1, 2, ..., 6$, $\text{dis}_i$ is the distance between the city $i$ and the current city, and greedy factor $\beta$ is the random value generated by the interval [0,10). Set the probability $p_i$ of roulette algorithm to $Q_i/(Q_1 + Q_2 + ... + Q_6)$, and use the roulette algorithm to get the next city which is the new position of the lion.

3.1.3 Method 3 (two-edge exchange method). One city is randomly selected from the six cities in the neighborhood, and exchanges to the latter city of the current city with two-edge exchange method [9]. Then the city sequence $S$ is re-adjusted, and a city selected randomly is taken as the next city which is the new position of lion.

When the improved LSO algorithm is used to solve TSP, the lion king updates the position according to method one, the lioness updates the position according to method two, and the position update of the cub can be divided into three situations. Generate a random value $\text{rnd}$ in interval (0,1). If $\text{rnd}$ less than 0.3, the cubs feed near lion king, and the position of lion king is taken as the current position. Then update the position of cub by method 2. If $\text{rnd}$ less than 0.6, the cubs hunt with the lioness, and the following lioness is calculated through the serial number of cub and total number of lionesses. Then the position of lioness is taken as the current position, and the new position of the cub is calculated by method 2. Otherwise, update the cub position by method 3.

When solving the TSP, the updating process of the lion position is the process of finding the next city in the optimal path. Whether the lion position is the next city or not is determined by the value $rt$ which has two values, true and false. If $fit$ is less than $p_{\text{BestFit}}$, the value of $rt$ is true, otherwise it is false. Where $fit$ is the fitness value calculated according to the lion position, and $p_{\text{BestFit}}$ is the individual historical optimal adaptation value. If $rt$ is true, the position of lion is the next city, otherwise, it is necessary to determine whether the position is the next city by a random value $r$ generated according to uniform distribution $U[0,1]$. If $r$ less than $0.1(1/\text{max}_\text{iter})(1/(1+\text{fit}-p_{\text{BestFit}}))$, the position is the next city, otherwise, need to re-search the new position, where $t$ is the current number of iterations and $\text{max}_\text{iter}$ is the maximum number of iterations.

3.2. HLSO algorithm for solving TSP

The optimal path obtained by LSO algorithm can not be regarded as the final result, and it is necessary to use PCS algorithm [9] to solve the optimal path of TSP. If the optimal path obtained by LSO algorithm is better than that calculated by PCS algorithm, the optimal value solved by LSO algorithm is the final result of the TSP, otherwise the optimal value solved by PCS algorithm is taken as the optimal value of TSP.

The steps of solving TSP with HLSO algorithm are as follows:

Step1: Initialize parameters. Population size is $N$, proportion of lioness is $\theta$, maximum number of iterations is $\text{max}_\text{iter}$, number of trials is $m$.

Step2: Give each city a number and form a sequence $S$. For example, $S = (0, 1, 2, ..., n-1)$ indicates that the loop is $c_0 \rightarrow c_1 \rightarrow c_2 \rightarrow \ldots \rightarrow c_{n-1} \rightarrow c_0$. Set the first city as the initial position of the lions.

Step3: Check each city in the city sequence $S$. For the current city $c_1$, check each city $c_2$ in its neighborhood (nearest to $c_1$, marked as $N(c_1)$), and try to replace the latter city of $c_1$ in the sequence
\( S \) with \( c_2 \). After the change, the sequence \( S' \) is obtained. If \( S' \) is better than \( S \), then replace \( S \) with \( S' \). City \( c_2 \) is searched using the LSO algorithm.

Step4: Search the next city \( c_2 \) using PCS algorithm, and try to replace the latter city of \( c_1 \) in the sequence \( S \) with \( c_2 \) to get a new sequence \( S'' \). If \( S'' \) is better than \( S \), then replace \( S \) with \( S'' \). There are two methods of replacement, two-sided interchange and single-point movement [9].

Step5: Repeat step3 and step4. If the loop \( S \) cannot be optimized after repeated \( m \) times, the search is stopped and the optimal loop is \( S' \) or \( S'' \).

Step6: Compare \( S' \) and \( S'' \), take the better loop, output the optimal solution, and end the algorithm.

4. Simulation experiment
In this paper, a hybrid lion optimization swarm (HLSO) algorithm is proposed to solve TSP, and simulation experiments are carried out with international standard TSPLIB data sets. The experimental environment of the algorithm is python3.7, NumPy, Intel Xeon E3-1230V2@3.30GHz and 16G of RAM.

4.1. Parameter setting
The algorithm parameters are set as shown in Table 1.

| Parameter definition             | parameter name | value |
|----------------------------------|----------------|-------|
| Size of population               | \( N \)        | 20    |
| Proportion of lionesses          | \( \theta \)    | 0.1   |
| Maximum number of iterations     | \( \text{max}_\text{iter} \) | 80    |
| Number of trials                 | \( m \)         | 50    |

4.2. Simulation experiment and result analysis
Table 2 shows the test results of the HLSO algorithm on 13 TSPLIB standard data sets. The experimental results are obtained by running each dataset 30 times independently. The column "Instance" represents the TSPLIB example used in the test, the column "Optimal" represents the known optimal value of the example, the column "Best" represents the optimal result obtained by running the example for 30 times independently, and the column "Worst" represents the worst result obtained by running the example for 30 times independently. Column "Avg" represents the average value of 30 independent runs of the example, column "Std" represents the standard deviation of the series composed of values running 30 times independently, column "Mean_Iter" represents the average number of iterations used for 30 times of independent operation of the example, and column "Time" represents the average time taken for 30 times of independent operation of the example.

| Instance | Optimal | Best | Worst | Avg | Std | Mean_Iter | Time  |
|----------|---------|------|-------|-----|-----|-----------|-------|
| demo6    | 56      | 56   | 56    | 56  | 0   | 1         | 0.1   |
| demo10   | 10127   | 10127| 10127 | 10127| 0   | 1         | 0.22  |
| att48    | 10628   | 10628| 10628 | 10628| 0   | 3.8       | 9.23  |
| eil51    | 426     | 426  | 427   | 426.1| 0.3 | 17.6      | 36.11 |
| krod100  | 21294   | 21294| 21294 | 21294| 0   | 22.5      | 145.61|
| eil101   | 629     | 629  | 633   | 630.5| 1.57| 45        | 254.25|
The HLSO algorithm is a combination of basic LSO algorithm and PCS algorithm. Compare the HLSO algorithm with the PCS algorithm, and the results are shown in Table 3. The experimental environment of the two algorithms is the same, and each algorithm runs 30 times independently. The data marked in bold show that the algorithm has found the optimal value. The experimental results show that HLSO algorithm is feasible and slightly better than PCS algorithm.

| Instance | Best | AVG | Mean_Iter | Best | AVG | Mean_Iter |
|----------|------|-----|-----------|------|-----|-----------|
| demo6    | 56   | 56  | 56        | 1    | 56  | 56        |
| demo10   | 10127| 10127| 10127    | 1    | 10127| 10127    |
| att48    | 10628| 10628| 10628    | 3.8  | 10628| 10628    |
| eil51    | 426  | 426 | 426.1    | 17.6 | 426  | 426.1    |
| krod100  | 21294| 21294| 21294    | 22.5 | 21294| 21294    |
| eil101   | 629  | 629 | 630.5    | 45   | 629  | 629.9    |
| pr107    | 44303| 44303| 44316.5  | 32.2 | 44303| 44315.8 |
| pr136    | 96772| 96772| 96812    | 71.9 | 96772| 96810.2 |
| pr144    | 58537| 58590| 58751    | 80   | 58537| 59070.2 |
| kroA200  | 29368| 29368| 29390.3  | 80   | 29384| 29507.7 |
| a280     | 2579 | 2579 | 2582.3   | 9.25 | 2579 | 2586.9   |
| lin318   | 42029| 42421| 42656.7  | 80   | 42304| 42664.8 |
| pr439    | 107217| 108323| 109027.1 | 80   | 108607| 109652.2 |

Six tsp examples are selected to test, and the optimization results are compared with algorithms in different literatures. The results are shown in Table 4. The data marked in bold show that the algorithm has found the optimal value or the result is better than other algorithms. The results show that HLSO algorithm can find the optimal solution of the problem.

| Instance | Optimal | Literature [11] | Literature [12] | Literature [13] | Literature [14] | HLSO |
|----------|---------|-----------------|-----------------|-----------------|-----------------|------|
| eil51    | 426     | ----            | ----            | ----            | 426             | 426  |
| eil101   | 629     | ----            | ----            | ----            | 629             | 629  |
| pr136    | 96772   | ----            | ----            | 97046           | 96772           | 96772|
| kroA200  | 29368   | 29382           | ----            | ----            | 29368           | 29368|
Figures 1 to 4 are the convergence curves of examples eil51, pr136, kroA200 and pr439, respectively. These figures show that the HLSO algorithm is feasible and the convergence speed is fast.

5. Conclusion
In this paper, a HLSO algorithm is proposed, which improves the basic LSO algorithm and combines it with PCS algorithm, and shows good performance on TSP. The experimental results show that, compared with the algorithms in literatures, the HLSO algorithm has better convergence and stability in solving the TSP.

Acknowledgments
This research was supported by the Key platform and Major Program of Guangdong Province under Grant No.2018KQNCX392 and by South China Institute of Software Engineering of Guangzhou University under Grant No. ky201823.

References
[1] Yu H T, Gao L Q, Tian W H. Discrete artificial Li F, Zhuang D. (2014) Operations Research. China Machine Press, Beijing.
[2] WU Y, BAO J J. (2006) New mixed quantum-inspired evolutionary algorithm for TSP. Computer Applications, 26(10):2433-2436.
[3] WANG K P, HUANG L, ZHOU C G, etal. (2003) Particle swarm optimization for traveling salesman problem. In: International Conference on Machine Learning and Cybernetics. IEEE. pp.1583-1585.

[4] Yu H T, Gao L Q. (2015) Tian W H. Discrete artificial bee colony algorithm for TSP. Journal of Northeastern University: Natural Science, 36(8): 1074-1079.

[5] NING G Y, CAO D Q, ZHOU Y Q. (2017) A Discrete Differential Evolution Algorithm for TSP Problem. Computer & Digital Engineering, 45(11): 2136-2142.

[6] ZHANG J M, MA L, LI Y Y. (2012) Improved shuffled frog-leaping algorithm for traveling salesman problem. Computer Engineering and Applications, 48(11): 47-50.

[7] Zhang C, Tu L, Wang J Y. (2015) Application of Self-adaptive ant colony optimization in TSP. J of Central South University: Science and Technology, 46(8): 2944-2949.

[8] LIU S J, YANG Y, ZHOU Y Q. (2018) A Swarm Intelligence Algorithm-Lion Swarm Optimization. Pattern Recognition and Artificial Intelligence, 31(5):431-441.

[9] YUAN Z. (2018) An efficient approach for small-scale TSP and its application in examination timetabling. MICROELECTRONICS & COMPUTER, 35(1):56-60.

[10] HE Q, WU Y L, XU T W. (2018) Application of improved genetic simulated annealing algorithm in TSP optimization. Control and Decision, 33(2): 219-225.

[11] Yassine Saji, Mohammed Essaid Rififi. (2016) A novel discrete bat algorithm for solving the travelling salesman problem. Neurl Comput & Applic, 27:1853-1866.

[12] Aziz Ouaarab, Belaid Ahiod, Xin-She Yang. (2014) Discrete cuckoo search algorithm for the travelling salesman problem. Neural Comput & Applic, 24:1659-1669.

[13] Pavlyukevich I. (2007) Cooling down L’evy flights. J Phys A: Math Theor, 40:12299-12313.

[14] ZHANG Z C, HAN W, MAO B. (2018) Adaptive Discrete Cuckoo Algorithm Based on Simulated Annealing for Solving TSP. ACTA ELECTRONICA SINICA, 46(8): 1849-1857.