ISPO: A New Way to Solve Traveling Salesman Problem

Xiaohua Wang¹², Aiqin Mu², Shisong Zhu¹²

¹College of Science, China University of Mining & Technology, XuZhou, China
²Foundation Departments, Xuzhou Air Force Academy, XuZhou, China

Email: plaxz@126.com

Received September 24, 2012; revised January 15, 2013; accepted January 23, 2013

ABSTRACT

This paper first introduces the concepts of mobile operators and mobile sequence, with which it redefines the rate of particle swarm optimization algorithm and the formula of position updating. Combining this discrete PSO algorithm with neighbors, the paper puts forward Hybrid Particle Swarm Optimization Algorithm, whose effectiveness is verified at the end of this paper.

Keywords: Mobile Operators; Particle Swarm Optimization; Traveling Salesman Problem

1. Introduction

The Particle Swarm Optimization algorithm [1] presented by psychologists Kennedy and Dr. Eberhart in 1995 is a new intelligent optimization algorithm which imitated the behaviors of birds. Compared with others, the PSO has shown its advantages. The PSO is intuitive and easy to realize and has high efficiency in complement. Nowadays, the PSO is widely used in many fields, such as functions optimization, training of neutral network, fuzzy system control and so on.

Traveling salesman problem (TSP) is a typical combinatorial optimization problem, which may be described as follows. Given n cities and the distances between every two cities, which is the shortest path for travelling all the cities only once?

At first, PSO algorithm as an efficient method is mainly used to solve some continuous optimization problems. However, with the development of PSO algorithm, some experts and scholars began to think how to solve the discrete optimization problems, such as, Kennedy [2] proposed a discrete PSO algorithm used to solve binary problems in the year of 1997 and he proposed a dynamic probabilistic PSO in 2005 [3]. In addition, some scholars presented the discrete PSO algorithm for solving TSP problem. Clerc [4] re-explained the velocity and position equation through replacement sequence, and Huang [5] proposed swapping sequence. Others combined crossover and mutation of genetic algorithm with PSO algorithm [6], Zeng and Cui presented a new unified model of PSO [7] which can be used in solving combinatorial optimization problem. This paper proposed mobile operators and mobile sequence for solving TSP problem. The arrangement of this paper is as follows. In Section 2, mobile operators and mobile sequence were defined. In Section 3, the velocity and position equation of PSO were re-explained and PSO algorithm was improved base on simulated annealing algorithm. In Section 4, TSP examples were used to evaluate the improved PSO algorithm, and the conclusions are given in Section 5.

2. Mobile Operators and Mobile Sequence

Definition 1. Assuming the solution space of TSP with n nodes be \( s = (n_1, n_2, \cdots, n_n) \), \( n_i \in E \), and the mobile operator \( sv(i,k) \) means the node \( n_i \) of \( s \) moving \( k \) steps \( (k \in N) \). If \( k \) is greater than zero, node \( n_i \) moves backward, on the contrary, it moves forward, and it keeps unmovable if \( k \) equals to zero. The formula \( s' = s + (i,k) \) is a new solution after processing by the mobile operator \( sv(i,k) \). And the symbol plus is given a new meaning. Because the result of solving TSP is an h-cycle, and \( (n_1, n_2, \cdots, n_n) \) and \( (n_2, n_3, \cdots, n_n, n_1) \) are the same solution actually.

If \( k \) is greater than zero, node \( n_i \) moves \( k \) steps anti-clockwise, otherwise it moves \( |k| \) steps clockwise.

Example 1. Assuming there are five nodes in TSP, and \( s \) equals to \((1, 2, 3, 4, 5)\), the mobile operator is \( sv(1,2) \), and \( s' \) can be presented as follows:

\[
\begin{align*}
  s' &= s + sv(1,2) = (1,2,3,4,5) + (1,2) = (2,3,1,4,5)
\end{align*}
\]

If the mobile operator is \( sv(3,3) \), thus
s' = s + sv(3,3) = (1,2,3,4,5)+(3,3) = (1,3,2,4,5)

Definition 2. Mobile sequence is the ordered sequence of one or more mobile operators, and it can be expressed as follows:

\[ v = (sv_1, sv_2, \cdots, sv_i) \]

where \( sv_1, sv_2, \cdots, sv_i \) are mobile operators, and the order of each operators has the rigorous significance.

A mobile sequence is running on a TSP’s solution which means all operators of the mobile sequence are running on the solution one by one.

Definition 3. Various mobile sequences running on one solution could bring same solution, all mobile sequences set with the same running result is called mobile sequences’ equivalent set.

Definition 4. More than one mobile sequence can be combined into a new sequence, \( \oplus \) is defined as the combining operator of two mobile sequences.

Definition 5. Assuming a mobile sequence \( v \), which is made up of \( n \) mobile operators, equals to \( (sv_1, sv_2, \cdots, sv_n) \), the length of this mobile sequence \( \|v\| \) is defined as \( n \), i.e. the number of mobile operators. If \( \|v\| \) is one, \( v \) is called unit mobile sequence.

Definition 6. Let \( p \) be a real coefficient on the interval \((0,1)\) and \( v \) be a mobile sequence and it equals to \( (sv_1, sv_2, \cdots, sv_n) \), \( p \odot v \) is defined by

\[ p \odot v = \begin{cases} v & \text{rand} \leq p \\ sv(1,0) & \text{rand} > p \end{cases} \]

where \( \text{rand} \) is a random number between 0 and 1. It means, \( p \odot v \) equals to \( v \) on probability \( p \).

3. IPSO—Improved PSO Algorithm

Now, we can rewrite the iterative formula of PSO algorithm as follows:

\[ v_{i(k+1)} = w_i \odot v_{i(k)} \oplus r_1 \odot (p_{gk} - x_{ik}) \oplus r_2 \odot (p_{gk} - x_{ik}) \]  \hspace{1cm} (1)

\[ x_{i(k+1)} = x_{ik} + v_{i(k+1)} \] \hspace{1cm} (2)

where

\( v_{i(k+1)} \) := velocity of particle \( i \) at time step \( k+1 \),
\( x_{i(k+1)} \) := position of particle \( i \) at time step \( k+1 \),
\( p_{gk} \) := best previous position of particle \( i \), found so far, at time step \( k \),
\( p_{gik} \) := best neighbor’s previous best position, at time step \( k \),
\( w, r_1, r_2 \) := social/cognitive confidence coefficients, and
\( r_1, r_2 \) are random numbers between 0 and 1.

As the PSO algorithm is easily trapped into local optimum in dealing with continuous problems, this paper applies the hybrid algorithm based on PSO to solve the TSP to avoid the same trouble. In each iteration, the PSO algorithm is running firstly, then produces \( n \) new positions in \( p_{gik} \)’s neighborhoods by using the simulated annealing (SA) algorithm, compares the fitness of \( n \) positions’ and \( p_{gik} \)’s, chooses the best position as \( p_{gik} \) and assigns the new position of \( p_{gik} \) to a particle as the first particle which is randomly selected. If the same solution is got \( m \) times of iterations, the velocity of all particles must be re-initialized.

The specific process of the improved algorithm is shown as follows:

1. Initialize the position and velocity of particle swarm, define velocity of each particle as the mobile sequence, and calculate \( p_{gk}, p_{gik} \);
2. Update particles’ velocity and position according to Formulas (1) and (2) shown before;
3. Generate \( l \) new locations randomly in the neighborhoods of \( p_{gik} \), compare the fitness of the new locations with \( p_{gik} \). Renew \( p_{gik} \) and assign it to the first particle;
4. Determine whether the algorithm has found the same optimal value after \( m \) times’ iteration, if the answer is yes, re-initialize all particle’s velocity.
5. Determine whether the stop condition is met. If it’s met, end the loop, otherwise, go to Step 2.

4. Experiments

The weight of inertia of PSO is linear, which can be expressed as follows:

\[ w = w_{\text{max}} - \frac{w_{\text{max}} - w_{\text{min}}}{\text{iter}_{\text{max}}} \times k \]

where

\( w_{\text{max}} \) := Initial weight
\( w_{\text{min}} \) := final weight
\( \text{iter}_{\text{max}} \) := the maximum number of iterations
\( k \) := the current number of iterations

TSP with 14 nodes \([4,8]\) is used to examine the effectiveness of the improved algorithm. Parameters setting: \( n \)-the number of particles is 40, the maximum times of iteration is 1000. The length of all particles’ velocity is initialized to 1 randomly, and the value of \( m \) is 10 as well.
as \( l \). Weight initialization: \( w_{\text{max}} = 0.9 \), \( w_{\text{min}} = 0.4 \). All feasible solution sequence begins from the first node. We executed the algorithm randomly for 30 times, the results of the experiments are analyzed in the Table 1.

The maximum search space of the improved PSO algorithm is 50,000. The search space of [8] is 200000 and is 4 times of ours, which shows that our approach is more effective. And in all the 30 runs our approach has consistently found the optimal solution, which presents our algorithm convergence rate is 100 percent.

Swapping sequence \( so(i, j) \) is proposed in references 4, which means exchanging \( n_i \) with \( n_j \) in the solution sequence \( s = (n_1, n_2, \cdots, n_N) \). Now, compare swapping sequence \( so \) with mobile sequence \( sv \). If \( A \) is \((1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7)\), and \( B \) is \((1 \ 3 \ 4 \ 5 \ 6 \ 2 \ 7)\), then \( B + sv(2, 4) \) equals to \( A \),

\[
B + (so(2, 6), so(3, 6), so(4, 6), so(5, 6))
\]
equals to \( A \). Normally, unit mobile sequence \( sv(i, k) \) is equivalent to swapping sequence

\[
( so(i, i + k), so(i + 1, i + k), \cdots, so(i + k - 1, i + k) )
\]
of length \( k \) when \( k \) is greater than zero and \( i + k \) is not greater than \( N \). On the contrary, if \( i + k \) is greater than \( N \), it can be converted into a swapping sequence \( sv(i, k) \) of length \( i + k - (N - 1) \). Furthermore, unit swapping sequence \( so(i, j) \) (if \( j \) is greater than \( i \)) can be converted into a mobile sequence

\[
(sv(i, j - i), sv(j - 1, i - j + 1))
\]
of length two, if \( j \) equals to \( i + 1 \), it can be converted into the unit mobile sequence \( sv(i, j - i) \). All of these imply that all problems the swapping sequence solved can be solved by mobile sequence, too. According to the length of exchange, mobile sequence has the advantage on swapping sequence. Moreover, if we take the definition of swapping sequence into consideration, it has no identical operator, which means there is no unit swapping sequence \( so \) can support the equation: \( A + so = A \).

However, the mobile sequence can overcome the shortcoming of swapping sequence with the equation \( A + sv(1, 0) = A \) is satisfied in all solution sequence. Therefore mobile sequence is more effective than swapping sequence.

For further verification of performance of improved algorithm, we apply the improved algorithm to solving Bays 29 [9] and Oliver 30 [10]. The data come from TSPLIB, the weight decreases from 0.9 to 0.4 during the iteration and the maximum times of iteration is 1500, and other parameters are as same as TSP with 14 nodes. All experiments independently run 30 times and the results are shown in Table 2 and the best solutions of the two plans are shown in Figures 1 and 2.

From Table 2, it is obvious that the improved algorithm gets a high convergence rate either in Bays 29 or in

<table>
<thead>
<tr>
<th>Problem</th>
<th>Bays 29</th>
<th>Oliver 30</th>
</tr>
</thead>
<tbody>
<tr>
<td>The swarm size</td>
<td>40</td>
<td>40</td>
</tr>
<tr>
<td>Iteration step</td>
<td>1500</td>
<td>1500</td>
</tr>
<tr>
<td>Opt</td>
<td>9074.1</td>
<td>423.9045</td>
</tr>
<tr>
<td>Best result</td>
<td>9074.1</td>
<td>423.9045</td>
</tr>
<tr>
<td>Average value</td>
<td>9123.316</td>
<td>429.7901</td>
</tr>
<tr>
<td>Search space</td>
<td>1500 × 40 + 1500 × 10 = 75,000</td>
<td>75,000</td>
</tr>
<tr>
<td>Solution space</td>
<td>28/2</td>
<td>29/2</td>
</tr>
<tr>
<td>Search space/solution space</td>
<td>1.22996e−25</td>
<td>4.24124e−27</td>
</tr>
<tr>
<td>Convergence rate</td>
<td>17/30</td>
<td>19/30</td>
</tr>
</tbody>
</table>

![Figure 1. The best solution of experiment BAYS 29 with our algorithm.](image-url)
Oliver 30, though the search space versus solution space is small (1.22996e−25, 4.24124e−27). It demonstrates that the improved algorithm has the advantages of small searching area and high convergence rate in solving the small TSP problems, and it is efficient.

For comparison, three other artificial algorithms including basic SA, basic GA and Basic Ant Colony Algorithm are used to solve the same TSP problem. All four algorithms are run 20 times for Oliver30, the results are shown in Table 3.

Form Table 3, it is clear that our algorithm is better than the other algorithms. For Oliver 30 problem, our improved PSO not only can achieve the best fitness value 423.7405 but also the average fitness value is better than the other algorithm’s. This means our improved PSO is a better and more effective means to solve TSP problem.

5. Conclusions

This paper has improved the PSO algorithm by introducing the concepts of mobile operator and mobile sequence. The improved algorithm can be applied to solve discrete problems. Through experiments in this paper, we can draw conclusions:

1) Using PSO algorithm with mobile operators and mobile sequence is an effective new way to solve TSP.
2) The improved algorithm has the advantages in small searching area and high convergence rate in solving the small TSP problems.

REFERENCES


Table 3. Results for Oliver30 problem.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Average Value</th>
<th>Best Value</th>
<th>Worst Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic SA</td>
<td>437.6632</td>
<td>424.9918</td>
<td>480.1452</td>
</tr>
<tr>
<td>Basic GA</td>
<td>482.4671</td>
<td>468.1531</td>
<td>504.5256</td>
</tr>
<tr>
<td>Basic ACA</td>
<td>447.3256</td>
<td>440.8645</td>
<td>502.3694</td>
</tr>
<tr>
<td>MPSO</td>
<td>429.7301</td>
<td>423.9045</td>
<td>449.4346</td>
</tr>
</tbody>
</table>

Figure 2. The best solution of experiment Oliver 30 with our algorithm.