A Novel Algorithm to Solve the Vehicle Routing Problem with Time Windows: Imperialist Competitive Algorithm

Geng-jia Wang, Yuan-Biao Zhang, Jia-Wei Chen

Packaging Engineering Institute, Zhuhai Campus, Jinan University, Zhuhai, 838661092@qq.com

Abstract

This paper study the vehicle routing problem with time windows based on using a novel optimization algorithm (ICA). The vehicle routing problem with time windows is a combinatorial optimization problem. It copes with route scheduling and the distribution of goods from the distribution center to geographically dispersed customers by a fleet of vehicles with constrained capacities. We first built the mathematic model for the vehicle routing problem with time. Then apply the imperialist competitive algorithm to solve the specific problem, comparing the results produced by the PSO and GA algorithms, which proved the effectiveness of the ICA.

Keywords: Vehicle routing Problem, Imperialist Competitive Algorithm, Genetic Algorithm, Particle Swarm Optimization

1. Introduction

The Vehicle Routing Problem (VRP) is one of the most studied combinatorial optimization problems and is concerned with the optimal design of routes to be used by a fleet of vehicles to serve a set of customers. Since it was first proposed by Dantzig and Ramser[1]. Hundreds of papers have been devoted to exact and approximate solution of the many variants of this problem. The VRP can be described as follows: given a set of customers C, a set of vehicles V, and a depot d, find a set of routes, starting and ending at d, such that each customer in C is visited by exactly one vehicle in V. Each customer having a specific demand, there are usually capacity constraints on the load that can be carried by a vehicle. In addition, there is maximum amount of time that can be spent on the road [2].

The vehicle routing problem with time windows (VRPTW) is a generalization of the VRR involving the added complexity of allowable delivery times, or time windows. In these problems, the service of a customer, including delivery of goods or services, can begin within the time window defined by the earliest and latest times when the customer will permit the start of service. Therefore, the times at which services begin are decision variables. Time windows arise naturally in problems with hard time windows include bank deliveries, postal deliveries, industrial refuse collection and school bus routing and scheduling [3].

Several techniques have been proposed for solving the vehicle routing problem with time window problem. Ever since then the VRPTW put forward, many methods have been developed for solving it. After retrieving a lot of literature, all the methods can be divided into two kinds. One of them is the exact solution [14]. For example, in 1987, Kolen developed the dynamic programming [4]. And in 1992, Desrochers put forward a column generation method to solve the problem [3]. In 1997, Kohl and Madsen proposed the Lagrange decomposition method and solved the classical model formulation directly [5]. Yet, exact solution methods are very time consuming, and therefore not suitable for real life applications when results must be obtained immediately.
The other type of methods is approximation methods, which involves the heuristics and met heuristics. In 1964, Wright studied the saving heuristic [6]. In 1975, Holland proposed the genetic algorithms in dealing with this problem [7]. In 1987, Solomon solved the 11route construction by using heuristics [8]. And 1991, Colorni developed the ant algorithms into the problem [9]. In 1993, Osman put forward the D search for the VRPTW [10]. In 2007, Zhong Y. proposed the simulated annealing method to solve the problem [11]. In 2010, Yang peng used a particle swarm optimization to vehicle routing problem with fuzzy demands [12]. The same year, Joseph Gallart Suarez solved the vehicle routing problem considering split delivery by two GRASP metaheuristic [13].

This paper focuses on using the imperialist competitive algorithm to solve the vehicle routing problem with time window. The imperialist competitive algorithm is a novel algorithm, which was first proposed by Atashpaz-Gargari in 2008. he used the ICA to design an optimal which not only decentralizes but also optimally controls an industrial Multi Input Output distillation column process [17]. The total vehicle travel cost is chosen as the objective function. Proposed method for the optimization has some advantages, such as simplicity, accuracy, and time saving compared with other algorithms.

2. The optimization model of vehicle scheduling

Vehicle scheduling problem can be described as following: using more than one delivery vehicle to deliver the goods to multiple customers. Each customer’s location and the demand for goods are certain. Each vehicle’s load capacity and the maximum distance for each distribution are both certain too. It is requested to arrange the vehicle delivery routes reasonably. Optimizing the objective function with satisfying the following conditions: The sum of each customer’s demand on the distribution path does not exceed the sum of the load capacity of the vehicle. The length of each distribution path does not exceed the vehicle’s maximum travel distance every customer should be served and only can be by one vehicle.

We suppose that there are $K$ delivery vehicles in the distribution center. Each vehicle’s load capacity is $Q_k$ ($k = 1, 2, ..., K$). And the maximum travel distance of each vehicle is $D_k$. There are $L$ customers to serve and the demand of each customer is $q_i$ ($i = 1, 2, ..., L$). The distance between the demand $i$ & $j$ defines as $d_{ij}$ ($i, j = 1, 2, ..., L$). And use $n_k$ to represent the sum of the customers of the $k$th vehicle serving. Use the set $R_k$ to indentify the $k$th distribution path. $r_{ki}$ is the element of $R_k$ represents that the order of customer $r_{ki}$ in the distribution $k$ is $i$ (not including the distribution center). Therefore, the distribution center can be identified by $k$. Use the total delivery length as the function objection and build the mathematical model for the scheduling problem mentioned above.

2.1. The object function for the model

In this problem, this paper aim at the minimum of the delivery cost, which means the total delivery length, should be minimized. And the sum of the distance is equal to the sum of each vehicle’s distance in each distribution [15]. There,

$$\min Z = \sum_{k=1}^{K} \left[ \sum_{i=1}^{n_k} d_{r_{ki-1}/r_{ki}} + d_{r_{iy}/r_{iy}} \cdot \text{sign}(n_k) \right]$$

(1)
Where, \( Z \) is the objection function; \( n_k \) is the sum of the customers that the \( k \)-th vehicle serves; \( r_{ki} \) is the order of customer \( r_{ki} \) in the distribution \( k \); \( d_{r_{ki}r_{kj}} \) is the distance between the customer \( r_{ki} \) and customer \( r_{kj} \); and \( \text{sign} \) is the distributed function for deciding whether the vehicle is involved in the distribution.

2.2. The limit conditions of the model

2.2.1. The vehicle capacity limit

In the real life, obviously, the load capacity of each vehicle is limited. Therefore, the distribution task cannot be finished by one vehicle. In order to ensure the sum of each customer’s demand on the distribution path does not exceed the sum of the load capacity of the vehicle. There is the condition for it.

\[
\sum_{i=1}^{n_k} q_{r_{ki}} \leq Q_k \tag{2}
\]

Where, \( q_{r_{ki}} \) is the demand of the customer \( r_{ki} \); \( Q_k \) is the load capacity of each vehicle.

2.2.2. The vehicle traveling capacity limit.

For each vehicle, it can only travel for a certain distance without refueling. Hence, in order to ensure the length of each distribution path does not exceed the vehicle’s maximum traveling distance. There is the condition for it.

\[
\sum_{i=1}^{n_k} d_{r_{ki}r_{ki}} \text{sign}(n_k) \leq D_k \tag{3}
\]

Where, \( D_k \) represents the vehicle’s maximum traveling distance; other signs are the same as above.

2.2.3. The customer serving limit

According to assumption, there is \( L \) customers to serve. So there is no way that the sum of the customers on the each delivery path does to exceed the sum of the entire customer.

\[
0 \leq n_k \leq L \tag{4}
\]

For the customers in the vehicle routing model, each of them should receive the delivery service. Meanwhile, each customer can only served by once. Hence,

\[
\sum_{k=1}^{K} n_k = L \tag{5}
\]
2.2.4. The vehicle serving times limit

Use the following set to identify the composition of each delivery path. Each delivery path corresponds to a vehicle.

\[ R_k = \{ r_{ki} \mid r_{ki} \in \{1, 2, ..., L\}, i = 1, 2, ..., n_k \} \]  

According to the hypothesis, each customer could only served by one vehicle. That is to say, there doesn’t any intersection point between two different delivery paths exist. Then, two different delivery paths can not share with the same customer.

\[ R_{k_1} \cap R_{k_2} = \emptyset, \forall k_1 \neq k_2 \]  

Where, \( R_k \) is the \( k \)th delivery path; \( r_{ki} \) represents the order of the customer \( r_{ki} \) in the \( R_k \) in the \( R_k \) is \( i \).

2.2.5. The vehicle using limit

In some allocation plan, it doesn’t need to use every vehicle, which may be only part of them. So, we create a function to differentiate these two situations. When the number of the customers that each vehicle serves is more than one, it means that the vehicle involves in the distribution. Otherwise, it is out of the distribution system.

\[ \text{sign}(n_k) = \begin{cases} 1 & n_k \geq 1 \\ 0 & \text{else} \end{cases} \]  

Where, \( n_k \) is the number of the vehicle serving; \( \text{sign}(n_k) = 1 \), it represents the vehicle involves in the distribution, \( \text{sign}(n_k) = 0 \) it represents that the vehicle doesn’t involve in the distribution.

2.2.6. The vehicle serving time window limit

In order to satisfy every customer, it should not only send the goods to the customers, but also send to them in time. The time of each vehicle arriving to the next customer on the each delivery path is the sum of the arriving time of the present customer and the waiting time spending on the present customers and the loading time and the delivering time from the present customer to the next customer.[14]

\[ S_{h(i-1)} + t_{h(i-1)} + t_{h(i-1)/i} = S_{h(i)} \quad i = 1, 2, ..., n_k \]  

In order to ensure the loading time is not earlier than the time requested by the customers’ time window. There,

\[ t_i = \max \{ a_i - s_i, 0 \} , i = 1, 2, ..., L \]
Where, $S_{ki}$, is the time that the vehicle arrives at the present customer; $t_{ki}$, is the waiting time that the vehicle spends waiting the present customer; $t_{ki, k+1}$, is the traveling time that the vehicle travels from the present customer to the next customer.. $S_{ki+1}$, is the time that the vehicle arrives at the next customer; $i_1$, is the time difference between the vehicle arriving time and the beginning time of the time window that the customer requests. $i_2$, is the beginning time of the time window requested by the customer;

This model is aimed at the obtaining the minimum of the sum of the distance the vehicle traveling. Actually, for this situation, the minimum of the cost is equal to the minimum of the distance because the only cost of this model is the gasoline consume, which is in direct proportion to the distance the vehicle traveling. Therefore, we choose the minimum of the distance of all the vehicles as our objection.

3. Imperialist Competitive Algorithm in General

Imperialist Competitive Algorithm (ICA) is a new evolutionary algorithm in the Evolutionary Computation field based on the human’s socio-political evolution. The algorithm starts with an initial random population called countries. Nevertheless, its effectiveness, limitations, and applicability in various domains are currently being extensively investigated. In Atashpaz-Gargari [17], ICA is used to design an optimal controller which not only decentralizes but also optimally controls an industrial Multi Input Multi Output distillation column process.

The following figure shows the flowchart of the ICA similar to other evolutionary algorithms, this algorithm starts with an initial population. Each individual of the population is called a country. Some of the best countries in the population selected to be the imperialists and the rest form the colonies of these imperialists. All the colonies of initial countries are divided among the mentioned imperialists based on their power. The power of each country, the counterpart of fitness value in the GA, is inversely proportional to its cost.

After forming initial empires, the colonies in each of them start moving toward their relevant imperialist country. The total power of an empire depends on both the power of the imperialist country and the power of its colonies. This fact is imperialist country plus a percentage of mean power of its colonies. The imperialists compete with each other for possessing other’s colonies to increase the total number of theirs and spreading their power or at least not decreasing so as to succeed in the competition. Otherwise, it will collapse finally. This possession picks out the powerful empires and collapses the weaker ones, which leads to the optimization solution [17].

![Figure 1. The process of the ICA](image-url)
4. Imperialist Competitive Algorithm to solve the vehicle routing problem with time window

4.1. Create initial empires

A candidate solution is represented as an array. We form an array of variable values to be optimized. In the GA terminology, this array is called “chromosome,” but in ICA the term “country” is used for this array.

In this case, country is an $S \times R$ array, which is randomly generated. The initial population is N. In the array, S represents the number of vehicles in the distribution center, and R is the distribution route for vehicle. So an integrated solution can be represented by this array. For instance, for a three-vehicle and nine-customer problem, country can be represented by a $3 \times 9$ array, which $R$ range from 0 to 9.

\[
\text{country} = \begin{bmatrix}
5 & 6 & 9 & 8 & 0 \\
2 & 3 & 4 & 1 & 7 \\
0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]  (11)

The country tells one of the solutions. It just uses two vehicles to solve the problem instead of three. The first rows is the route for vehicle one, second for vehicle two and the third for vehicle three.

4.1.1. The Cost Function

The cost of each country can be built according to the specific problem. The cost function is to evaluate implement condition of each solution. In this case, we use $C_i$ to evaluate each solution. Each county’s cost can be calculated by:

\[
C_i = Z_i + W \times M_i
\]  (12)

Where, $C_i$ is the cost of county $i$, $M$ is the number of infeasible routes, $W$ is the punish weight for each infeasible route.

4.1.2. The Remaining Steps to Solve the Problem

The main steps in the algorithm are summarized in the pseudo code shown as follows:

1) Select some random points on the function and initialize the empires
2) Move the colonies toward their relevant imperialist
3) If there is a colony in an empire which has lower cost than that of imperialist, exchange the positions of that colony and the imperialist
4) Compute the total cost of all empires (Related to the power of both imperialist and its colonies)
5) Pick the weakest colony(colonies) from the weakest empire and give it(them) to the empire that has the most like hood to possess it (Imperialistic competition)
6) Eliminate the powerless empires (If there is just one empire, stop, if not go to 2)

In the next part we apply the proposed algorithm to the vehicle routing problem with time windows.
A Novel Algorithm to Solve the Vehicle Routing Problem with Time Windows: Imperialist Competitive Algorithm
Geng-jia Wang, Yuan-Biao Zhang, Jia-Wei Chen
Advanced in Information Sciences and Service Sciences. Volume 3, Number 5, June 2011

5. Simulation Experiment and Results

<table>
<thead>
<tr>
<th>Customer number</th>
<th>x-axis /km</th>
<th>y-axis /km</th>
<th>q/t</th>
<th>Time window/h</th>
<th>Customer number</th>
<th>x-axis /km</th>
<th>y-axis /km</th>
<th>q/t</th>
<th>Time window/h</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>12.8</td>
<td>8.5</td>
<td>0.1</td>
<td>[4.7,10.5]</td>
<td>11</td>
<td>6.7</td>
<td>16.9</td>
<td>0.9</td>
<td>[4.1,10.1]</td>
</tr>
<tr>
<td>2</td>
<td>18.4</td>
<td>3.4</td>
<td>0.4</td>
<td>[1.5,6.0]</td>
<td>12</td>
<td>14.8</td>
<td>2.6</td>
<td>1.3</td>
<td>[3.4,8.1]</td>
</tr>
<tr>
<td>3</td>
<td>15.4</td>
<td>16.6</td>
<td>1.2</td>
<td>[4.7,10.2]</td>
<td>13</td>
<td>1.8</td>
<td>8.7</td>
<td>1.3</td>
<td>[0.0,6.0]</td>
</tr>
<tr>
<td>4</td>
<td>18.9</td>
<td>15.2</td>
<td>1.5</td>
<td>[5.1,9.5]</td>
<td>14</td>
<td>17.1</td>
<td>11.0</td>
<td>1.9</td>
<td>[5.3,10.3]</td>
</tr>
<tr>
<td>5</td>
<td>15.5</td>
<td>11.6</td>
<td>0.8</td>
<td>[3.7,8.9]</td>
<td>15</td>
<td>7.4</td>
<td>1.0</td>
<td>1.7</td>
<td>[2.1,6.3]</td>
</tr>
<tr>
<td>6</td>
<td>3.9</td>
<td>10.6</td>
<td>1.3</td>
<td>[6.7,12.3]</td>
<td>16</td>
<td>0.2</td>
<td>2.8</td>
<td>1.1</td>
<td>[6.8,12.0]</td>
</tr>
<tr>
<td>7</td>
<td>10.6</td>
<td>7.6</td>
<td>1.7</td>
<td>[7.9,12.9]</td>
<td>17</td>
<td>11.9</td>
<td>19.8</td>
<td>1.5</td>
<td>[7.7,13.4]</td>
</tr>
<tr>
<td>8</td>
<td>8.6</td>
<td>8.4</td>
<td>0.6</td>
<td>[0.6,5.7]</td>
<td>18</td>
<td>13.2</td>
<td>15.1</td>
<td>1.6</td>
<td>[6.0,10.4]</td>
</tr>
<tr>
<td>9</td>
<td>12.5</td>
<td>2.1</td>
<td>0.2</td>
<td>[2.6,6.8]</td>
<td>19</td>
<td>6.4</td>
<td>5.6</td>
<td>1.7</td>
<td>[5.4,9.6]</td>
</tr>
<tr>
<td>10</td>
<td>13.8</td>
<td>5.2</td>
<td>0.4</td>
<td>[2.5,8.1]</td>
<td>20</td>
<td>9.6</td>
<td>14.8</td>
<td>1.5</td>
<td>[5.8,11.7]</td>
</tr>
</tbody>
</table>

Now, we use the ICA algorithm to solve the problems. We use an example from the literature [20] to experiment. Suppose the distribution center and 20 customers distribute in a square area with a length of 20 km. Every customer’s good demand is not more than 2 tons. There are 5 distribution cars with the maximum capacity 8 tons and maximum traveling distance 50 km in the distribution center. The average velocity of car is 20 km/h. Use the same position coordinates of the customer in the literature. The coordinate of the distribution center is (14.5 km, 13.0 km). The coordinates of the customers and their service time windows are shown in the table 1.

This problem consists of customers and the total number of its all-permutation reaches to $2.433 \times 10^{18}$. Due to the limitation of the time, this problem can not be solved by exhaust algorithm. In order to verify the accuracy of the ICA result, we also apply the Particle Swarm Optimization and Genetic algorithm to solve the problem and compare the results of these three heuristic algorithms. The results are shown in the following table 2.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Deliver distance/km</th>
<th>Iterative number</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ICA</td>
<td>PSO</td>
</tr>
<tr>
<td>1</td>
<td>145.5</td>
<td>158.3</td>
</tr>
<tr>
<td>2</td>
<td>133.4</td>
<td>152.3</td>
</tr>
<tr>
<td>3</td>
<td>152.3</td>
<td>138.9</td>
</tr>
<tr>
<td>4</td>
<td>130.2</td>
<td>167.3</td>
</tr>
<tr>
<td>5</td>
<td>125.4</td>
<td>155.4</td>
</tr>
<tr>
<td>6</td>
<td>134.7</td>
<td>148.1</td>
</tr>
<tr>
<td>7</td>
<td>136.7</td>
<td>135.2</td>
</tr>
<tr>
<td>8</td>
<td>146.8</td>
<td>162.8</td>
</tr>
<tr>
<td>9</td>
<td>138.4</td>
<td>148.3</td>
</tr>
<tr>
<td>10</td>
<td>131.4</td>
<td>157.1</td>
</tr>
<tr>
<td>Avg</td>
<td>137.4</td>
<td>152.3</td>
</tr>
</tbody>
</table>

6. Conclusions

In this paper, we formulated the mathematical model for the vehicle routing problem with time windows. Then, we applied a meta-heuristic algorithm named Imperialist Competitive Algorithm (ICA)
for the first time to solve such kind of problem. Meanwhile, we also use the PSO algorithm and GA algorithm to solve the same specific problem. Compare the result produced by ICA with the result produced by PSO and GA, we find that ICA do better and less time consuming, with a more optimal result and less iterative number, which proves that ICA is an effective algorithm to solve the vehicle routing problem.

7. Acknowledgements

The authors acknowledge the financial support of this research by project supported by Research Cultivation and Innovation Foundation of Jinan University of China, as well as the Key Laboratory Foundation of Universities of Guangdong Province of China.

8. References