Applying Genetic Algorithm for Cargo Loading Problem

Chunyu Ren

Abstract--The present study is focused on the cargo loading problem (CLP). Genetic algorithm is used to get the optimization solution. First of all, adopt binary code so as to make the problem more succinctly. On the basis of cubage-weight balance algorithm, construct initial solution to improve the feasibility. Secondly, the study controls selection strategy through individual amount so as to guarantee group diversity, adopts 2-exchange mutation operator to strengthen the partial searching ability of chromosome. Finally, the good performance of improved algorithm can be proved by experiment calculation and concrete examples.

Index Terms--Cargo loading problem; binary code; cubage-weight balance algorithm; control selection strategy; genetic algorithm

I. INTRODUCTION

Bulk cargos have the characteristics of low-volume, multi-variety and multi-batch. Cargo distribution problem is a complex and constraint combinatorial optimization problem, which is NP-hard problem. According to the amount of cargo and vehicle, loading problem can divided into two types. One requests to utilize the minimized vehicle for limited loading cargos. Considering relatively limited loading vehicles, the other requests to fully utilize the capacity and quality of vehicle so as to load cargos as much as possible and reach to the maximum utilization of the vehicle.

The main research methods of loading problem include precise algorithm, heuristic algorithm and intelligence optimization algorithm. Scheithauer offered approximation algorithm based on dynamic programming. To about 100 cargos distribution, computation time is acceptable [1]. Francois adopt new precise algorithm to solve two-dimension loading problem [2].

Heuristic algorithm can only offer a local optimal solution of the problem, and is absent of ability of global optimization. Especially when solving the big scale problem, the efficiency is not high. Berghammer adopted linear approximation method to optimize loading problem. However, it is difficult to confirm the convergence conditions [3]. Carlo designed approximate algorithm to solve one-dimensional loading problem with fixed weight, variable quantity of cargos [4].

When solving the big scale and multi-restriction problem, intelligent algorithm is used widely. Bortfeldt solved the cargo loading problem on weak difference of single container through taboo searching algorithm of partial filling base on homogeneous block [5]. Hongmei Cao designed ant algorithm to solve cargo loading problem with multi-variety. And the result can be realized to the optimization of loading weight and capacity [6]. Lei Pu designed heuristic algorithm to solve cargo loading problem with multi-variety [7].

The traditional algorithm is easy to set the utilization rate of weight and capacity to the opposite side. Therefore, the study design genetic algorithm from overall optimization to solve cargo loading problem.

II. MODEL

\[
\text{Max } Z = \sum_{i \in N} \sum_{j \in K} \lambda g_i x_{ij} + \sum_{i \in N} \sum_{j \in K} (1 - \lambda) v_i x_{ij}
\]  

Restraint condition,

\[
\lambda G \min_j \sum_{i \in N} g_i x_{ij} \leq \eta_i G_i, \quad j \in K
\]

\[
(1 - \lambda) V \min_j \sum_{i \in N} v_i x_{ij} \leq \eta_j V_j, \quad j \in K
\]

\[
\sum_{j \in K} x_{ij} \leq 1, \quad i \in N - D_0
\]

\[
\sum_{j \in K} x_{ij} = 1, \quad i \in D_0
\]

\[
x_{ij} + \frac{1}{|D_i|} \sum_{j \in D_i} x_{ij} \leq 1, \quad i \in N, j \in K
\]

\[
x_{ij} = 0, 1 \quad i \in N, j \in K
\]

In model, all variables can be shown as followings.

\(N(i = 1, 2, \ldots, n)\) is the muster of loading cargo. \(K(j = 1, 2, \ldots, m)\) is the muster of loading tools. The maximum loading weight of j tools is \(G_j\). The maximum loading capacity is \(V_j\). The weight of \(i(i \in N)\) is \(g_i\). Volume is \(v_i\). \(\min_j\) is the minimum loading weight of j loading tools. \(\min_j\) is the minimum loading capacity of j loading tools. \(\eta_2(0 \leq \eta_2 \leq 1)\) is the elasticity coefficient of loading capacity. \(D_i(D_i \subset N)\) is the cargo mustering not mixing...
cargo. \( D_0 (D_0 \subset N) \) is the loading cargo muster of priority mandatory installation. \( x_{ij} (i \in N, j \in K) \) is the decision variable of cargo.

In the above models, the value of parameter \( \lambda \) is \( 0 \leq \lambda \leq 1 \). When only getting the maximum loading weight, \( \lambda \) is 0. When getting the maximum loading weight and capacity at the same time, \( \lambda \) is 1/2. In formula (1), target function is the maximum number of loading weight and capacity. Constraint condition (2) is the total constraint weight. Constraint condition (3) is the total constraint capacity. Constraint condition (4) is that the same cargo can only be installed in same vehicle. Constraint condition (5) is priority mandatory installation constraints. Constraint condition (6) is that cannot be equipped at the same time.

III. APPLICATION IN MCLP OF GENETIC ALGORITHM

A. Binary code

For that single vehicle loading is mixing cargos of different consignee in the same vehicle, don’t consider the vehicle selection when coding. Therefore, usually adopt binary code, namely, 0 and 1 string. Loading condition of i cargo can be thought as i chromosome gene. There are n genes in each chromosome. The concrete coding form is \( (x_1, x_2, \ldots, x_i, \ldots, x_n) \). Value of each gene is \( x_i = 1 \) or \( (i = 1, 2, \ldots, n) \). \( x_i = 1 \) can be shown that cargo i is in the loading condition. Otherwise, \( x_i = 0 \).

B. Initial solution forming

Suppose that \( N(i = 1, 2, \ldots, n) \) is the muster of loading cargo, \( K(j = 1, 2, \ldots, m) \) is the muster of loading tool, the maximum weight of j loading tool is \( G_j \), the maximum loading capacity is \( V_j \). The weight of j(i \( \in N \)) cargo is \( g_{ij} \), volume is \( v_i \). If all cargos are installed, it needs \( n_k = \max(n_g, n_v) \) vehicles, in which

\[
N_g = \left\lfloor \sum_{i=1}^{n} \frac{g_i}{\min(G_1, G_2, \ldots, G_m)} \right\rfloor
\]

and

\[
N_v = \left\lfloor \sum_{i=1}^{n} \frac{v_i}{\min(V_1, V_2, \ldots, V_m)} \right\rfloor.
\]

If \( n_g > n_v \), \( \omega_l = \frac{g_i}{v_i} (i \in N) \). Otherwise, \( \omega_l = \frac{v_i}{g_i} (i \in N) \). Taking \( n_g > n_v \), as example, the concrete steps are as follows.

1. Initial data are supposed \( n_k = 0, V_{sum} = 0, G^1 = 0, S^0 = \Phi, K = m \).

2. Calculate \( \omega_l = \frac{g_i}{v_i} (i \in N) \), sorting according to non-increasing order. Take this muster as \( p^1 \).

3. Suppose \( p = p^1 \), calculate muster p. Review it from beginning to ending. The order is \( p[1], p[N], p[2], p[N-1], \ldots \).

4. If \( G^1_{sum} + g_{p[1]} \leq G_j (j \in m) \) and \( V^1_{sum} + v_{p[1]} \leq V_j (j \in m) \), \( G^1_{sum} = G^1_{sum} + 1 \) and \( V^1_{sum} = V^1_{sum} + 1 \).

5. If \( G^1_{sum} + g_{p[N]} \leq G_j (j \in m) \) and \( V^1_{sum} + v_{p[N]} \leq V_j (j \in m) \), \( G^1_{sum} = G^1_{sum} + 1 \) and \( V^1_{sum} = V^1_{sum} + 1 \). Otherwise, turn into step 7.

6. Repeat step 4 and 5 to \( G^1_{sum} > G_j (j \in m) \) and \( V^1_{sum} > V_j (j \in m) \).

7. Record current status of \( G^1_{sum}, V^1_{sum}, S^1, n_k = n_k + 1, K = K - 1 \) and \( S^2 = S^1 \cup S^0 \).

8. Suppose \( p^1 = p - S^2 \).

9. Repeat step 3 to step 8 until to \( K = 0 \). And turn into \( K = 0 \).

10. Output the cargo muster \( \{S^1, S^2, \ldots, S^{nk}\} \), total weight of loading cargo \( G_{sum} \), total volume of loading cargo \( V_{sum} \).

C. Select Operator

The selected probability is the following formula.

\[
P_i = f_i / \sum_{i=1}^{N} f_i \quad (8)
\]

According to model theorem of genetic algorithm, sample amount of model H in t generation is the following formula.

\[
m(H, t) = m(H, 0). (1 + c)^t \quad (9)
\]

Here, \( m(H, 0) \) is the sample amount of model in initial population. When some individual number exceeds marginal value \( \varepsilon \), the individual number should be reduced so as to the individual random can make up the population scale. The description is following.

Step 1: t-1 generation group forming after t-1 genetic operation has selecting operation to create group \( p(t) \) according to proportion fitness.

Step 2: Calculate every individual number in group \( p(t) \).

Step 3: Have the following operation to group \( p(t) \) to create group \( p^1(t) \). If some individual number exceeds the marginal value \( \varepsilon \) of t individual, delete this individual so as to control individual number in the extent \( \varepsilon \) otherwise, copy all
individuals.

Step 4: If the number of group \( p_1(t) \) is less than group scale \( N \), then randomly operate \( N - p_1(t) \) new individuals. And new individual can take part in following cross and mutation operation.

D. Crossover Operator

This study adopts the partially matched crossover method. The main thought of crossover operation is that total operation process can be finished by two steps. Firstly, execute the routine crossover operation to individual coding string. Then, modify the gene value if each gene locus beyond crossover zone according to mapping relationship of each gene value in crossover zone.

E. Mutation Operator

The mutation strategy of this study is adopted 2-commutated mutation strategy, namely, randomly selecting mutated individual chromosome according to some mutation probability and two gene locations in this chromosome, exchanging gene in two places and form into new gene clusters. If it is continuously appeared with zero code in gene clusters, exchange zero code and non-zero code in random place. Execute this step much times till that new gene cluster become the legal child generation individual.

F. Step of Genetic Algorithm

Step 1: \( g_{en} = 0 \), adopt \( n \) real number to take code. Initial group is constructed based on cubage-weight balance algorithm and controlled parameters are input. Crossover operator is \( p_c \), initial mutation operator is \( p_m \), group scale is \( N \), maximum operating algebra is \( K \).

Step2: Construct initial solution

Step3: calculate fitness.

Step 4: if \( g_{en} < \max g_{en} \) and \( g_{en} = g_{en} + 1 \), enter into step four. Otherwise stop calculation and input optimization solution.

Step 5: select crossover mutation chromosome according to proportional fitness.

Step 6: calculate each individual amount in group \( p(t) \).

Step 7: if amount of some individuals exceed the threshold value \( \varepsilon \) of individual amount in \( t \) generation, delete these individuals so as to control individual amount in the extent of threshold value \( \varepsilon \). Otherwise, copy all of individuals.

Step 8: if the amount of group \( p(t) \) is less than group scale \( N \), randomly bring up \( N - p(t) \) new individuals.

Step 9: have partially matching and crossover operation.

Step 10: have 2-exchange mutation and mutation operation.

Step 11: repeat step three to step ten.

IV. EXPERIMENTAL CALCULATION AND ANALYSIS

The data of study is from reference [7], utilize TBJ10 type container to load bulk cargos of 42 freight invoices. Quality and outer diameter of each cargo can be shown in Table 1. The maximum loading weight of TBJ10 type container is \( G = 10t \).

The maximum loading capacity is \( V = 16.81m^3 \). The cargo of No.1 freight invoice must be loaded firstly. And nature and package of cargo don’t occur conflicting. Confirm the scheme of maximum loading rate.

<table>
<thead>
<tr>
<th>No.</th>
<th>( g_{i}/t )</th>
<th>( v_{i}/m^3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.221</td>
<td>1.05</td>
</tr>
<tr>
<td>2</td>
<td>1.156</td>
<td>1.98</td>
</tr>
<tr>
<td>3</td>
<td>0.7</td>
<td>2.00</td>
</tr>
<tr>
<td>4</td>
<td>1.243</td>
<td>3.14</td>
</tr>
<tr>
<td>5</td>
<td>1.600</td>
<td>2.86</td>
</tr>
<tr>
<td>6</td>
<td>1.612</td>
<td>2.17</td>
</tr>
<tr>
<td>7</td>
<td>2.300</td>
<td>4.80</td>
</tr>
<tr>
<td>8</td>
<td>1.930</td>
<td>5.20</td>
</tr>
<tr>
<td>9</td>
<td>1.850</td>
<td>2.30</td>
</tr>
<tr>
<td>10</td>
<td>1.900</td>
<td>3.80</td>
</tr>
<tr>
<td>11</td>
<td>1.120</td>
<td>2.00</td>
</tr>
<tr>
<td>12</td>
<td>1.431</td>
<td>4.02</td>
</tr>
<tr>
<td>13</td>
<td>0.600</td>
<td>2.78</td>
</tr>
<tr>
<td>14</td>
<td>0.306</td>
<td>3.22</td>
</tr>
<tr>
<td>15</td>
<td>1.040</td>
<td>2.60</td>
</tr>
<tr>
<td>16</td>
<td>0.805</td>
<td>1.23</td>
</tr>
<tr>
<td>17</td>
<td>1.220</td>
<td>0.65</td>
</tr>
<tr>
<td>18</td>
<td>1.000</td>
<td>2.40</td>
</tr>
<tr>
<td>19</td>
<td>1.782</td>
<td>0.87</td>
</tr>
<tr>
<td>20</td>
<td>1.100</td>
<td>1.54</td>
</tr>
<tr>
<td>21</td>
<td>1.030</td>
<td>5.60</td>
</tr>
</tbody>
</table>

A. Solution of new genetic algorithm

Take C program on loading problem of NGA using C language and make experiment calculation in computer of CPU1.8G and memory 1G.

After many trails, adopt following parameters, \( \lambda = 0.5 \), group scale \( NN = 50 \), maximum iterations \( t_{max} = 200 \).

Mutation operator is \( p_m = 0.01 \). Randomly get the solutions for 10 times. The results can be shown in Table 2.

<table>
<thead>
<tr>
<th>Calculation order</th>
<th>Calculation order</th>
<th>Solution</th>
<th>Solution</th>
<th>Loading</th>
<th>Capacity</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1,4,17,18,21,35,39,41</td>
<td>95.54</td>
<td>94.23</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1,3,11,15,24,28,31</td>
<td>95.61</td>
<td>94.29</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1,5,15,18,20,24,28</td>
<td>95.41</td>
<td>94.29</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1,3,5,11,21,34,39</td>
<td>94.71</td>
<td>93.46</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1,17,11,16,17,18,41</td>
<td>94.66</td>
<td>93.46</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1,2,10,11,15,16,17,18</td>
<td>94.62</td>
<td>93.46</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>1,3,11,15,24,28,31</td>
<td>95.61</td>
<td>94.29</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>1,10,18,23,32,39,41</td>
<td>94.35</td>
<td>93.46</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>1,3,5,11,21,34,39</td>
<td>94.71</td>
<td>93.46</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>1,2,8,13,16,29,37</td>
<td>94.64</td>
<td>93.46</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| Average value     | 95.004             | 93.786   |
| S.D               | 0.4693             | 0.4212   |

Table 2 can be shown that it can get comparatively higher solution during the course of 10 times solutions by genetic
The average value of loading utilization is 95.004%. The average value of capacity utilization is 93.786%. The calculation results are quite stable. The standard deviation of load utilization is only 0.4693. And the standard deviation of capacity utilization is only 0.4212.

From the view of calculation efficiency, two times can reach to optimal solution in ten times, four times can reach to best solutions. Therefore, efficiency is comparatively higher.

B. Solutions of genetic algorithm

Reference [7] is adopted genetic algorithm to get the solution. The main parameters are as followings. Group scale is $N = 80$, the maximum iterative times is $K = 300$, crossover operator is $p_c = 0.95$, mutation operator is $p_m = 0.01$, initial temperature is $T_0 = 250$, temperature coefficient $\delta = 0.89$, and randomly get the solutions for 30 times. The optimal loading weight and capacity utilization rate is 83.80% and 91.13%. The concrete loading schemes can be shown in Table 3.

<table>
<thead>
<tr>
<th>No.</th>
<th>Loading utilization /%</th>
<th>Capacity utilization /%</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, 9, 18, 23, 28, 33, 40</td>
<td>83.80</td>
<td>91.13</td>
</tr>
</tbody>
</table>

C. Analysis on two algorithms

The study adopts new genetic algorithm to get solutions. The optimal loading weight and capacity utilization rate are 100% and 100%. The concrete scheme can be shown in Table 4.

<table>
<thead>
<tr>
<th>No.</th>
<th>Loading utilization /%</th>
<th>Capacity utilization /%</th>
</tr>
</thead>
<tbody>
<tr>
<td>1, 3, 11, 15, 24, 28, 31</td>
<td>95.61</td>
<td>94.29</td>
</tr>
</tbody>
</table>

The loading weight and capacity utilization rate of designed hybrid genetic simulated annealing algorithm are all improved, and loading weight utilization rate is improved 14.09% and loading capacity utilization rate is improved 3.468%than above references.

V. CONCLUSIONS

In general, the proposed new genetic algorithm has strong searching ability, rapid convergence rate, strong ability to overcome the fall into local optimum and high solving high quality. Therefore, it is more practical significance and value so as to reduce operating cost and improve economic benefit.

REFERENCES