Hybrid Heuristic Approaches For a Multi-Production Forward/Reverse Logistics System

Liu Changshi, Lai Mingyong

Economics and trade college, Hunan University, P.R.China, 410079
(Tel: 86-731-8684625; e-mail: liulongstone964@126.com)

Abstract: We consider a multi-production two-stage forward/reverse logistics system design problem where a fixed number of capacitated distribution/reclaiming centers are to be located with respect to capacitated suppliers and retail locations while minimizing the total costs, and take the random of demand/reclaiming into consideration. We also provide hybrid heuristic procedures for the solution of the problem, and develop transshipment heuristic to improve the duration of the hybrid approaches. Finally we present extensive computational results that show the high performance and effectiveness of the solution approaches.

1. INTRODUCTION

In general, a production forward/reverse logistics system design involves the determination of the best configuration, regarding location and size of the plants and distribution/reclaiming centers, their technology content, and product offerings and transportation decisions. In this paper, we consider a multi-production two-stage forward/reverse logistics system design problem where a fixed number of capacitated distribution/reclaiming centers are to be located with respect to capacitated suppliers and customers while minimizing the total costs. DCs can be seen as breakbulk and consolidation locations where the consolidated product shipments from plants are sorted and combined for shipment to retail locations to satisfy the demand at these sites.

Under these assumptions, we are interested in locating $P$ DCs/RCs so that the total fixed location costs at the second level and the transportation costs at the stages is minimized without violating the capacity restrictions at the plant and DC levels. Thus, the problem considered in this paper involves strategic DC/RC location decisions and, mostly tactical, retailer-to-DC and DC-to-plant assignment decisions.

Considerable attention has been devoted to these models for the location of plants and DCs. Several comprehensive reviews have been published in the last years (Geoffrion et al., 1995; Erenguc et al., 1997).

This paper makes several contributions to the existing literature. First, we consider a two-stage multi-production forward/reverse logistics system design problem, and handle the random of demand/reclaiming. Second, we develop customized meta-heuristic approaches with memory components. Third, the mathematical model reflects the potential cost savings that are due to economies of scale. The heuristic solution approaches can easily accommodate such changes in models and are equally applicable under these operational characteristics.

2. RELATED LITERATURE

Solution approaches related to our problem are optimization algorithms within the framework of Benders’ decomposition (Geoffrion, 1974), heuristics based on branch-and-bound (Kaufman et al., 1977; H. Ro et al., 1984), and Lagrangian relaxation (Pirkul and Jayaraman, 1996, 1998).
However, these techniques consume extensive amounts of time and effort in finding optimal solutions for realistically sized problems. On the other hand, in (Syarif, et al., 2002; Keskin, et al., 2007), the genetic algorithms are shown to be effective in solving relatively large size problems, respectively.

How ever, the common ground of above approaches is that they do not take reverse logistics into consideration, not to mention handling the random of demand/reclaiming. Our work integrates some assumptions of (Geoffrion, et al., 1974; Pirkul, et al., 1998) with (Keskin, et al., 2007). Geoffrion and Graves develop and implement a solution technique. There are several distinctions between the assumptions in (Geoffrion, et al., 1974) and our study. The first distinction is that Geoffrion and Graves include the restriction that customers cannot be served by more than one DC (single-sourcing). Second, they employ four-index variables to represent the flow of products from plants to retailers through DCs. In our study, we employ seven-index variables for both stages that are connected via flow conservation at the DCs. Furthermore, in (Geoffrion, et al., 1974), each plant has a specific capacity dedicated to each product. In our work, we assume that each plant has an overall flexible capacity that can be utilized for different products via switch-overs.

H. Pirkul and Jayaraman (1998) consider locating both capacitated plants and DCs by heuristic based on Lagrangian relaxation. However, some of their model assumptions are similar to ours. First, the capacity restrictions at the plants are not dedicated for each product. Second, they consider single-sourcing on the transportation links (plants to DCs and DCs to customers) as in our model. Third, they restrict, with an upper bound, the number of facilities and plants to be located.

B urcu B. Keskin and Halit Uster consider a two-stage production/distribution system design problem. However they only take the forward logistics into consideration.

3. THE MODEL

We consider a multi-production two-stage forward/reverse logistics system, in the first stage, the products are transported from plants to capacitated distribution centers (DCs); and, in the second stage, they are transported from DCs to retailer or customer. At the same time, some used products or returned products are transported from retailer or customer to reclaiming centers (RCs). For simplicity, the DCs are the RCs. We assume that product mixes and capacities at established plant locations as well as the demand for different products at retail locations are known, and there exists a set of DCs/RCs to be located with capacity limitations. There are multi-commodities flowing, the demand at the retailers is constant, it is specific for each retailer and each product, the demand quantity and the reclaiming quantity are independent variables.

In this setting we determine how much of each product to transport from plants to DCs and from DCs to retailers, and how much reclaiming to transport from customers to RCs in such a way that the total transportation costs are minimized. We use the following notation in a mixed integer programming formulation of our problem.

\[
\begin{align*}
I & \quad \text{set of customers, } i = 1, \ldots, m \\
J & \quad \text{set of potential DCs, } j = 1, \ldots, n \\
K & \quad \text{set of plants, } k = 1, \ldots, f \\
L & \quad \text{set of products, } l = 1, \ldots, g \\
p & \quad \text{number of DCs to be located} \\
D_x & \quad \text{demand at customer } i \in I \text{ for product } l \in L \\
D'_x & \quad \text{reclaiming at customer } i \in I \text{ for product } l \in L \\
W_j & \quad \text{capacity limit at DC } j \in J \\
W'_j & \quad \text{capacity limit at reclaiming center in DC } j \in J \\
B_k & \quad \text{capacity limit at plant } k \in K \\
c_{il} & \quad \text{unit transportation cost of product } l \in L \text{ from DC } j \in J \text{ to customer } i \in I \\
t_{kj} & \quad \text{unit transportation cost of product } l \in L \text{ from plant } k \in K \text{ to DC } j \in J \\
r_{lj} & \quad \text{unit transportation cost of reclaiming product } l \in L \text{ from customer } i \in I \text{ to DC } j \in J \\
u'_0 & \quad \text{unit disposal cost of product } l \in L \text{ in DC } j \in J \text{ while reclaiming quantity exceeds the capacity of reclaiming center} \\
u_0^0 & \quad \text{unit opportunity cost of product } l \in L \text{ in DC } j \in J \text{ while demand be satisfied partially} \\
u_0^r & \quad \text{unit storage cost of product } l \in L \text{ in DC } j \in J \text{ while distribution quantity exceeds the demand}
\end{align*}
\]

We also define three sets of decision variables:

\[
\begin{align*}
z_j & \quad 1 \text{ if DC and reclaiming center at location } j \text{ is used, } 0 \text{ o.w., } j \in J \\
x_{il} & \quad \text{amount of product } l \in L \text{ transported from DC } j \in J \text{ to customer } i \in I \\
y_{kj} & \quad \text{amount of product } l \in L \text{ transported from plant } k \in K \text{ to DC } j \in J \\
x_{ijl} & \quad \text{amount of product } l \in L \text{ transported from customer } i \in I \text{ to DC } j \in J \\
m_{ijl} & \quad \text{amount of disposal product } l \in L \text{ while reclaiming quantity exceeds the capacity of reclaiming center}
\end{align*}
\]
amount of the lack of product \( l \in L \) while demand be satisfied partially

amount of the surplus product \( l \in L \) distribution quantity exceeds the demand

Then, the problem can be formulated as follows:

\[
\begin{align*}
\sum_{i \in I} \sum_{j \in J} \sum_{l \in L} c_{ijl} \cdot x_{ijl} \cdot z_j &+ \sum_{i \in I} \sum_{j \in J} \sum_{k \in K} \sum_{l \in L} t_{ijkl} \cdot y_{ijkl} \cdot z_j \\
+ \sum_{i \in I} \sum_{j \in J} \sum_{l \in L} r_{ijl} \cdot m_{ijl}^c \cdot z_j &+ \sum_{i \in I} \sum_{j \in J} \sum_{l \in L} u_{ijl}^m \cdot m_{ijl}^s \\
+ \sum_{j \in J} \sum_{l \in L} u_{ijl}^m \cdot m_{ijl}^d &+ \sum_{j \in J} \sum_{l \in L} u_{ijl}^s \cdot m_{ijl}^s \\
\text{s.t.}
\end{align*}
\]

\[\sum_{j \in J} x_{ijl} = d_{ijl}, \forall i \in I, \forall l \in L, \quad (1)\]

\[\sum_{i \in I} \sum_{j \in J} m_{ijl} \leq d_{ijl}^l, \forall j \in J, \forall l \in L, \quad (2)\]

\[\sum_{i \in I} \sum_{j \in J} y_{ijkl} = \sum_{i \in I} \sum_{j \in J} y_{ijkl}, \forall j \in J, \forall l \in L, \quad (3)\]

\[\sum_{i \in I} y_{ijkl} \leq B_k, \forall k \in K, \quad (4)\]

\[z_j = P, \forall j \in J, \quad (5)\]

\[x_{ijl} \leq s_{ijl} \cdot z_j, \forall i \in I, \forall j \in J, \forall l \in L, \quad (6)\]

\[\sum_{i \in I} \sum_{j \in J} y_{ijkl} \cdot t_{ijkl} \leq r_{ijl} \cdot z_j, \forall j \in J, \forall k \in K, \quad (7)\]

\[x_{ijl} \geq 0, \forall i \in I, \forall j \in J, \forall l \in L, \quad (8)\]

\[z_j \in \{0,1\}, \forall j \in J. \quad (9)\]

In the objective function, the first term represents the costs from the DCs to the customers, and the second term is the costs from the plants to the DCs, the third term is the reverse costs from customers to the RCs, the fourth term represents the storage cost of superfluous product in DCs while distribution quantity exceeds the demand quantity, the fifth term represents the disposal cost of superfluous product in RCs while reclaiming quantity exceeds the capacity of reclaiming center, and the sixth term is the opportunity cost of the lack of demand in DCs.

Constraints (1) ensure that demand for each product \( l \) at each customer \( i \) is satisfied, and the reclaiming for each product \( l \) is less than the capacity limitation of RC \( j \). Constraints (2) and (4) ensure the capacity restrictions at the DCs, RCs and the plants, respectively. There is no specific capacity requirement for each product at the plants and DCs. Constraints (3) are for flow conservation at each DC. Constraint (5) specifies the number of DCs to be located. In constraint set (6), \( S_{ijl} \) is given by \( \min \{ D_{ijl}, W_{ij} \} \) for \( i \in I, j \in J \) and \( l \in L \). These constraints state that a customer can only be assigned to an open DC. In constraint set (7), \( R_{ijl} \) is given by \( \min \{ W_{ij}, B_k \} \) for \( j \in J \) and \( k \in K \). They are added to obtain a stronger formulation and are implied by constraints (2), (4) and (5). Finally, constraints (8) and (9) are non-negativity and integrality constraints.

4. HYBRID HEURISTIC SEARCH APPROACHES

In order to determine the set of open DCs/RCs, we use scatter search and tabu search meta-heuristics in our solution procedures. Scatter search has been implemented in various optimization problems (M. Laguna et al., 2003).

Our hybrid search heuristic operates on a binary vector \( z \) of size \( n \), which represents the locations of the DCs/RCs. Given the locations of the DCs/RCs, i.e., an instance of \( z \), we solve the corresponding capacitated transshipment problem to optimality. For a given solution vector \( z \), we find its objective function value \( Z \) by adding the optimal objective value of the capacitated transshipment problem. The parameters used in the scatter search are the population size \( 2^{*}h_{\text{max}} \) where \( h_{\text{max}} \) is the diversification parameter, and the number of quality and diverse solutions in the reference set, which are denoted by \( b_1 \) and \( b_2 \), respectively. The selection of the parameters has an important effect on the performance of the scatter search. We randomly select our initial seed solution, a binary vector \( z^* \), as an input to the diversification generation method where we generate a population of diverse solutions.

Step 1: Diversification. This step aims to produce trial solutions that differ from each other as significantly as possible. We summarize the ideas behind the procedure that we select in our implementation (Glover, 1997). Let \( Z^* \) be the elements of a binary solution vector \( z^* \). Given \( z^* \) as the seed solution, the generator creates \( 2^{*}h_{\text{max}} \) new solutions. According to Burcu B. Keskin and Halit Uster, the first \( h_{\text{max}} \) solutions \( Z_1, \ldots, Z_{h_{\text{max}}} \) are generated using an incremental parameter \( k \) and the following expressions:

\[z_{ijl}^{k+1} = z_{ijl}^k - z_{ijl}^{k-1}, \quad k = 0, \ldots, \left[ h / h_{\text{max}} \right].\]

The maximum value of the incremental variable \( k \) changes for each value of \( h \). The second \( h_{\text{max}} \) solutions are obtained using the relation \( 1 - Z_k \) where \( h = 1, \ldots, h_{\text{max}} \).

Step 2: Feasibility. The trial solutions are generally infeasible due to violations of constraints (2) and constraint (5). First, it is likely some of these initial solutions will not satisfy constraint (5) since the number of open facilities can be lower or higher than the prespecified \( P \)-value. The simple rule to correct this infeasibility is to open facilities randomly among the unused ones if \( P < \sum Z_{ijl} \) and, otherwise, to close facilities randomly among the open ones until the total
number of open facilities is equal to $P$. Second, we need to consider the feasibility of a trial solution in terms of the total capacity it provides. If the total capacity provided is less than the total demand, we open a facility chosen randomly among the unused ones and close a randomly chosen open facility. We swap only a pair of facilities between the open and unused facility groups until the total capacity feasibility is obtained. Therefore, constraint (5) is not violated. We continue to correct infeasibilities due to constraints (2) and (5) until all of the trial solutions in the population are feasible.

Step 3: Reference set construction. The reference set contains a set of good quality and most diverse solutions. The $b_1$ quality solutions are selected based on the objective function values, i.e., the solutions with the lower total cost values are selected as the quality solutions. In order to find $b_2$ diverse solutions, we use the distance between two solutions as a diversity measure. We first include the quality solutions in the reference set. Then, in the population, we look for a solution that is not currently in the reference set and that maximizes the minimum distance to all solutions currently in the reference set. We include that solution in the reference set as the most diverse solution. We continue in this manner until we have chosen a total of $b_2$ diverse solutions.

Step 4: Subset generator. We generate subsets of the reference set solutions that will later be combined into new solutions. We use three subset generators to obtain subsets of size 2, 3 and 4, known as Type II, Type III and Type IV, respectively. Subsets of Type II correspond to the pairs of solutions taken from the reference set. A Type III subset is formed by including the best quality solution and a pair of other solutions from the reference set. The third subset generator, Type IV, selects the two highest quality solutions from the reference set and joins them with pairs of the remaining solutions.

Step 5: Solution combination. The trial solutions in each subset that are generated in the previous steps are combined into a new solution by taking a linear combination of the solutions it includes. We use the reciprocal of the objective values of the solutions as coefficients of the trial solutions. Therefore, a solution with a lower objective value receives a higher weight. Clearly, an element of a combined solution can be a fractional value. In order to obtain a legitimate solution, we round each element of the final solution to its closest binary value.

Step 6: Reference set update. After obtaining a new solution, we determine whether this solution should be included in the reference set. First, if the objective value of the new solution is better (i.e., lower objective value) than any of the quality solutions in the reference set, we drop the worst solution among the quality solutions and add the new solution to the reference set. If this is not the case, we check the diversity of the new solution. We calculate the distance between the new solution and each solution in the reference set. If it is more diverse (the most distant) than any of the diverse solutions in the reference set, we drop the worst solution (with respect to the distance criteria) among the diverse solutions and add the new solution to the reference set. Finally, if the new solution does not satisfy these two conditions, then it is not eligible to be a member of the reference set. In which case, we move to the next subset to generate another new solution. If the reference set is updated, the algorithm goes back to Step 4 to generate new subsets. When the reference set cannot be updated further, the scatter search procedure is terminated.

We implement a tabu search heuristics to improve the result.

Step 7: Improvement with tabu search meta-heuristics. Tabu search provides an opportunity to escape local optima and explore a larger subset of the solution space (Sun, 1998; Gendreau, et al., 1994).

As before, we represent a solution to our problem with a binary vector $z = (z_1, ..., z_n)$ where $z_j = \{0,1\}$, $j \in J$. Since tabu search uses a pair-exchange neighborhood over the sets of opened and unopened DC given by $\beta$ and $\phi$, respectively. A move in this neighborhood corresponds to simultaneously changing the $z_j^+$ value for a $j \in \beta$ to zero and the $z_j^-$ value for a $j \in \phi$ to one.

Tabu move restrictions are employed to prevent cycling and revisiting previously visited solutions. We classify a solution obtained by a pair-exchange as a tabu if it corresponds to closing a DC which was opened in a recent accepted solution in the course of the procedure. The recency refers to the fact that tabu status is not permanent for an open DC. We employ a tabu tenure that is equivalent to the number of iterations an open DC remains a tabu. In this study, we choose the fixed tabu tenure approach.

The tabu search algorithm uses a tabu list $T$ where $T = (T_1, ..., T_n)$ with each $T_j$ representing the tabu tenure of DC $j$, $j \in J$. If $T_j > 0$ for some $j \in J$, then the DC $j$ is tabu. Any DC with a corresponding entry that is equal to zero in the tabu list is non-tabu. At each iteration, when a candidate solution results in opening a new DC $j$, $T_j$ is assigned the tabu tenure and all other positive entries in the tabu list are decreased by one. This is a type of recency-based, or short-term memory, since the tabu list shows how recently the solutions were visited.

An aspiration criterion is used to overrule the tabu restrictions so that we can avoid escapes from attractive unvisited solutions. The aspiration criterion states that if a solution involving a tabu move has a better objective value than the best known solution, then the tabu status is disregarded. Otherwise, if the aspiration criterion is not satisfied, of course we continue to the next iteration with the best non-tabu solution.

4.1 Solving problem with two-stage transshipment heuristic

To improve the duration of the overall heuristic, we develop a two-stage transshipment heuristic for this problem after DCs being located by above heuristics. The heuristics help to reduce the total duration of the algorithm at least by half as reported in Section 5. Let $j \in J$ be the set of open DCs so
that $|J| = P$. In this heuristic, we first solve the transportation problem between the DCs and the customers. Then, using the implied shipment requirements at the DCs as the demands for the DCs, we solve the transportation problem between the plants and the DCs. The algorithm of the two-stage transhipment heuristic is given in Display 1.

Display 1. Pseudocode of the transshipment heuristic

```
Input: $j, c, w, t, m$, $i \in I, j \in J, k \in K, l \in L$
Output: $x_{ij}, y_{ij}, m_{ij}, m_{ij}^*, m_{ij}^*$
1: Sort $c_{ij}$ in an increasing order.
2: for each $c_{ij}$ do
3: if $D_{ij} > 0$ then
4: if $W_j > D_{ij}$ then
5: $x_{ij} \leftarrow D_{ij}, W_j = W_j - D_{ij}, D_{ij} \leftarrow 0$
6: else
7: $x_{ij} \leftarrow W_j, D_{ij} = D_{ij} - W_j, W_j \leftarrow 0$
8: end if
9: end if
10: if $D_{ij} > 0$ then
11: if $W_j > D_{ij}$ then
12: $m_{ij} \leftarrow D_{ij}^*, W_j = W_j - D_{ij}^*, D_{ij}^* \leftarrow 0$
13: else
14: $m_{ij} \leftarrow W_j, m_{ij}^* = D_{ij} - W_j, D_{ij}^* \leftarrow 0$
15: end if
16: end if
17: end for
18: Calculate $F_{ij} = \sum x_{ij}, \forall j, \forall l$
19: Sort $t_{ij}$ in an increasing order.
20: for each $t_{ij}$ do
21: if $F_{ij} > 0$ then
22: if $P_l > F_{ij}$ then
23: $y_{ij} \leftarrow F_{ij}, P_i = P_i - F_{ij}$
24: else
25: $y_{ij} \leftarrow P_l, m_{ij}^* = P_l - F_{ij}, P_l \leftarrow 0$
26: end if
27: end if
28: end for
```

5. COMPUTATIONAL RESULTS

We employ C++ and traditional bit string code for solving the proposed problem. In order to test the performance and effectiveness of the heuristic approaches, we conduct a series of numerical studies as shown in Table 1 (Appendix A) on randomly generated problems ranging from small to moderately large sized ones according to schedule described above. We consider five sets of problems with 10 instances in each; thus we solve a total of 50 instances. We run the C++ algorithms on a Pentium IV 3.2 GHz machine with 1 GB memory.

We summarize the results for the hybrid heuristic approaches in Table 2 (Appendix A). i.e., the scatter search heuristic with, and without, the tabu heuristic improvement.

For small and medium problem instances, data sets 1, 2 and 5, the average gap between the hybrid heuristic approaches with and without improvement is less than 1%. For large instances, data set 3 and 4, the average gap can be as high as 4.5% and the maximum gap as high as 6.5%. Moreover, in 50 of the 90 cases, we find the optimum result.

6. CONCLUSIONS

We investigate a multi-production two-stage forward/reverse logistics system design problem. Although there have been a number of studies on our problems, there have not been sufficient studies involving solutions taking the opportunity cost, disposal cost and storage cost into consideration to handle the random of demand/reclaiming. This paper originated in an attempt to design effective and efficient hybrid heuristics approaches for the proposed problem, including scatter search and tabu search. We present scatter search and tabu search algorithms that perform very well even on large size problems. The scatter search heuristic combined with the tabu heuristic results in better optimality gaps and a high number of optimal solutions, thus, proving to be an effective method for solving the proposed problem. In our computational analysis, we observe the scatter search solutions can be improved by tabu heuristic.

We also develop a transshipment heuristics to reduce the total duration of the algorithm, and it can be easily adapted by changing the parameter $P_k$ to $P_l$ to solve the problem when the plants have product specific capacities.

REFERENCES


**Appendix A.**

<table>
<thead>
<tr>
<th>K</th>
<th>J</th>
<th>Y</th>
<th>X</th>
<th>W</th>
<th>M</th>
<th>W’</th>
<th>L</th>
<th>C</th>
<th>S</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[3,6]</td>
<td>[10,20]</td>
<td>[40,50]</td>
<td>[38,45]</td>
<td>[40,45]</td>
<td>[5,10]</td>
<td>[5,10]</td>
<td>0</td>
<td>[2,17]</td>
<td>[0,5]</td>
</tr>
<tr>
<td>2</td>
<td>[3,6]</td>
<td>[10,20]</td>
<td>[50,60]</td>
<td>[55,65]</td>
<td>[60,65]</td>
<td>[5,10]</td>
<td>[5,10]</td>
<td>[5,15]</td>
<td>0</td>
<td>[0,5]</td>
</tr>
<tr>
<td>3</td>
<td>[3,6]</td>
<td>[30,50]</td>
<td>[70,90]</td>
<td>[70,85]</td>
<td>[75,85]</td>
<td>[8,12]</td>
<td>[5,10]</td>
<td>0</td>
<td>[0,15]</td>
<td>[3,7]</td>
</tr>
<tr>
<td>4</td>
<td>[3,6]</td>
<td>[30,40]</td>
<td>[50,60]</td>
<td>[50,65]</td>
<td>[70,80]</td>
<td>[8,12]</td>
<td>[5,10]</td>
<td>[0,15]</td>
<td>0</td>
<td>[3,7]</td>
</tr>
<tr>
<td>5</td>
<td>[3,6]</td>
<td>[15,25]</td>
<td>[40,45]</td>
<td>[45,50]</td>
<td>[40,48]</td>
<td>[1,6]</td>
<td>[5,10]</td>
<td>[5,10]</td>
<td>0</td>
<td>[3,7]</td>
</tr>
</tbody>
</table>

Table1. Data sets optimum solution durations for test problems

| Data set | Without tabu improvement | With tabu improvement |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Ave.gap | Max.gap | Ave.time | Max.time | No.opt | Ave.gap | Max.gap | Ave.time | Max.time | No.opt |

Scatter search

| 1 | 0.00 | 0.00 | 305.2 | 402.8 | 10 | 0.00 | 0.00 | 401.3 | 602.5 | 10 |
| 2 | 0.26 | 0.75 | 406.9 | 996.1 | 5 | 0.00 | 0.00 | 680.6 | 1153.2 | 8 |
| 3 | 4.51 | 6.35 | 2256.4 | 5536.3 | 1 | 0.46 | 0.85 | 3482.6 | 6573.1 | 2 |
| 4 | 1.59 | 3.02 | 2157.6 | 4963.7 | 6 | 0.23 | 0.62 | 2833.4 | 5634.8 | 5 |
| 5 | 0.12 | 0.63 | 598.6 | 1025.8 | 7 | 0.00 | 0.00 | 765.2 | 1268.5 | 8 |

Two-stage transhipment heuristics

| 1 | 0.04 | 0.25 | 60.1 | 116.3 | 8 | 0.00 | 0.00 | 69.5 | 146.8 | 10 |
| 2 | 0.86 | 2.64 | 186.4 | 301.2 | 2 | 0.18 | 0.64 | 234.6 | 634.7 | 4 |
| 3 | 1.43 | 3.28 | 387.3 | 826.4 | 1 | 0.59 | 1.44 | 528.3 | 968.6 | 1 |
| 4 | 0.96 | 1.43 | 1063.2 | 2013.5 | 0 | 0.75 | 1.12 | 1684.2 | 2563.1 | 0 |
| 5 | 0.00 | 0.00 | 98.6 | 265.3 | 10 | 0.00 | 0.00 | 171.5 | 368.4 | 10 |