Solving the Bi-level Programming for Supply Chain Distribution Network with Threshold Accepting

Ching-Jung Ting\(^1\) and Yu-Ju Yang\(^2\)
\(^1\)Department of Industrial Engineering and Management, Yuan Ze University, Chung-Li 32003, Taiwan
\(^2\)AU Optronics Corporation, Lantian, Taiwan
jetingcj@saturn.yzu.edu.tw

Abstract
This paper investigates a multiple echelon supply chain network where the number and location of distribution centers has to be determined to be the intermediate points between manufacturers and customers. We apply the bi-level programming to this supply chain network design problem. The upper level considers DC location problem and production-allocation problem, while the lower level focuses equality of capacity utilization ratios for all opened DCs. Since the bi-level programming problem is NP-hard, we develop a threshold accepting heuristic to solve the proposed model. Few different examples are tested and the results show that our threshold accepting algorithm can obtain a good solution efficiently.

Keywords: Bi-level programming, Supply chain network, Threshold accepting.

1. Introduction

In today’s globalization environment, individual firms no longer compete as independently, but rater as integral part of supply chains. Supply chain management (SCM) could be considered as an integrated process in which a group of several firms, such as suppliers, manufacturers, distributors and retailers, work together to acquire raw materials with a view to converting them into commodities that they deliver to retailers (Beamon, 1998). Supply chain distribution network (SCDN) design is to provide an optimal platform for efficient and effective SCM. Distribution network design problems involve strategic decisions which will affect both tactical and operational decisions. The design task involves the decision of locating facilities, such as manufacturing plants and distribution centers (DCs), and the allocation of plants to DCs and the commodity flows between DCs and plants. This integrated decision problem has been recognized as an important problem in SCM (Vidal and Goetschalckx, 1997; Ercenguc et al., 1999).

Sabri and Beamon (2000) proposed an integrated multi-objective supply chain (SC) model in strategic and operational supply chain planning. The constraints on the strategic objective are determined subjectively, and thus it is not clear how the variation bounds on them are determined. Jayaraman and Pirkul (2001) put forward an integrated model for supply chain design and planning by means of mixed integer linear programming. Zhou et al. (2002) investigated balanced allocation of customers to multiple distribution centers with a genetic algorithm. Klose and Drezel (2005) reviewed some of the contributions to the current state of facility location models for distribution system.

Most of the formulations in supply chain network design problems focus on a mixed integer programming model with single or multiple objective functions, which assume a central firm makes the decision on the integrated process or multiple firms act collaboratively to achieve common goals. However, the location and distribution decisions might be decided by two different decision makers in practice. In fact, logistics directors usually choose the nearest DC to serve customers and also must balance capacity utilization among opened DCs in the network. Obviously, the production-distribution problem can be represented as a bi-level programming problem where the manufacturing managers are the leaders, and logistics managers are the followers who choose the DCs to serve customers.

In the literature, few supply chain network design problem is formulated as a bi-level programming problem. Taniguchi et al. (1999) developed a bi-level model to determine the optimal size and location of public logistics terminals. The upper level minimizes the total cost. The lower level determined the user equilibrium assignments of vehicles on the road network for given public logistics terminal locations. Jayaraman and Ross (2003) provided a two-stage model to represent a multi-product, four-echelon distribution network. Chan and Chung (2004) proposed a multi-criterion genetic algorithm to solve a two-tier supply chain, including the demand and supply layers. Their objective was to balance the production loads and minimize the production and transportation cost of the supply chain. Ryu et al. (2004) proposed a bi-level linear programming modeling approach for the production-distribution planning problem. The models subsequently considered demand uncertainty, resources and capacities when they are reformulated by multi-parametric linear programming. Huang and Liu (2004) developed a bi-level programming model to optimize the distribution network. The upper level determines the distribution center locations, and lower level tries to balance workload for DCs. Cao and Chen (2006) proposed a bi-level model with mixed integer and continuous variables to determine the opening and closing plants in a decentralized environment. They transformed the bi-level problem into an equivalent single level model by using Karush-Krantz-Tucker condition of the lower level problem.

Roghanian et al. (2007) presented a bi-level stochastic multi-objective linear programming model for the production-distribution planning. They transformed the bi-level stochastic model into an equivalent deterministic model with multi-objective nonlinear programming to which fuzzy techniques are applied to solve it. Chen et al. (2007) proposed a model that considers substitutions of materials, components, and products when designing a production-distribution network for a decentralized supply chain. Their problem is formulated as a mixed-integer, bi-level programming problem, Sun et al. (2008) presented a bi-level programming model to seek the optimal location for logistics distribution centers by considering benefits of customers and logistics planning departments. The upper-level determines the optimal locations by minimizing the planners’ cost, while the lower level gives an equilibrium demand distribution by minimizing the customers’ cost.

Marinakis and Marinaki (2008) formulated a location routing problem as a bi-level problem. The upper level is for the strategic facility location problem while the lower level corresponds to the routing problem. A genetic algorithm combined with an expanding neighborhood search method is proposed to solve the problem. Yamada et al. (2009) proposed a bi-level model for
solving the bi-level programming for supply chain distribution network with threshold accepting.

strategic transport planning in freight terminal development and interregional freight transport network design. The upper-level problem determines the best combination of actions, whilst a multimodal multiclass user traffic assignment technique is incorporated within the lower-level problem such that the freight-related benefit-cost ratio is maximized. Three types of genetic-base local search heuristic are proposed to solve the lower level problem. Calvete et al. (2011) developed a bi-level model for production-distribution planning problem in which the upper level is to solve the distribution problem and the lower level deals with the production problem. They used an ant colony optimization algorithm to solve the upper level distribution problem, and then the production problem is solved in the lower level.

In this paper, we assume that bi-level programming problem is made up of the upper-level problem that the decision maker of the manufacturing company minimizes total logistics cost by outsourcing to a distribution company, while the lower level decision maker tries to balanced capacity utilization at opened distribution centers. Although threshold accepting studies abound, the area of distribution network design in a supply chain environment has not been addressed in the literature. It is from this point that the current study embarks. The contribution of this paper is to present a bi-level programming model for the hierarchical supply chain network design problem and to propose an approach for solving it.

The remainder of this paper is organized as follows. Section 2 introduces the bi-level programming basics and both upper and lower level models of the SC&DN design problem are presented. In section 3, the proposed threshold accepting algorithm is presented. Numerical examples are given to illustrate the applications of the model and its algorithm in section 4. Finally, the concluding remarks are given in section 5.

2. Bi-level Programming Model

2.1 General Bi-level Programming Problems

The bi-level programming problem describes a hierarchical system, which is composed of two levels of decision-makers, where the upper-level problem is the leader and the lower-level problem is the follower. The upper-level decision affects the lower-level decision and vice versa. Using the common notation (Anandalingam and Friesz, 1992), the general formulation of a bi-level programming problem can be formulated as:

\[
\begin{align*}
\text{Min} & \quad G(x, y) \\
\text{S.T.} & \quad F(x, y) \leq 0 \\
\text{Min} & \quad g(x, y) \\
\text{S.T.} & \quad f(x, y) \leq 0
\end{align*}
\]

where \( x \in \mathbb{R}^n \) and \( y \in \mathbb{R}^m \). The variables of problem (1) are divided into two classes, the upper level (leader) variables \( x \) and the lower level (follower) variables \( y \). The functions \( G(x, y) \) and \( g(x, y) \) are the upper level and lower level objective functions, respectively, while the functions \( F(x, y) \) and \( f(x, y) \) are the upper level and lower level constraint sets, respectively. The decision sequence is as follows: the upper level minimizes the objective function \( G(x, y) \) by finding an optimal solution of \( x \) for the feasible set \( X \). Given the optimal solution of \( x \), the lower level will optimally solve the objective function \( g(x, y) \). In the decision making process, the upper level decision maker uses the complete information including the lower level possible reaction to the upper level decision, while the lower level one only uses the local information to make decisions. This may be similar to the complicated supply chain distribution network design situations which may be difficult to model using other modeling methodologies. Upper level constraints involve variables from both levels and play a specific role. Indeed, they must be enforced indirectly, as they do not bind the lower-level decision-maker.

The location problem of logistics distribution centers involving two kinds of decision-makers that are manufacturing company and the distribution company who have different objectives. Obviously, it is appropriate that the bi-level programming model can be adopted to describe the location problem.

2.2 Assumptions and Notations

A third party distribution company has to serve the manufacturing company for a set of customers with known demands. The company decides which DCs are opened and which customers are assigned to each selected DC. The product is supplied by the manufacturing company that owns several plants. The manufacturing company allocates the production to its plants based on the aggregated demand at each opened DC and aims to minimize its total logistics cost involved to serve all customers. These costs includes the transportation cost from the plants to DCs, the warehousing cost and fixed cost at each opened DC, and the transportation cost from the DC to customers. The model assumptions are as follows:

1. There is only one product. The production costs at different plants are same. Thus, we do not take into account the production cost here.
2. The capacities of manufacturing plants are known in advance and they all produce the same product.
3. Customer demands are known and must be delivered by a single distribution center.
4. All the decisions for manufacturing and transportation are made within a single period.
5. The locations for all candidate distribution centers and their capacities are known. These distribution centers can be served by multiple plants.
6. The unit warehousing cost and fixed operating charge at each distribution center are known.
7. The unit transportation costs from facilities to facilities are given.

The notation used is as follow.

- \( c_{ij} \) : Unit transportation cost from node \( i \) to \( j \)
- \( c_j \) : Unit warehousing cost at \( j \)
- \( C \) : Set of customers
- \( d_k \) : Demand of Customer \( k \)
- \( D \) : Set of candidate DC
- \( f_j \) : Fixed cost at DC \( j \)
- \( L_i \) : Production capacity at plant \( i \)
- \( O_D \) : Set of opened DC
- \( P \) : Set of plants
- \( t_{ik} \) : Transportation time from DC \( j \) to customer \( k \)
- \( w_j \) : Capacity of DC \( j \)
- \( x_{ij} \) : Flow from plant \( i \) to DC \( j \)
- \( y_{jk} \) : \( = 1 \) if customer \( k \) is served by DC \( j \); \( = 0 \) otherwise
- \( z_i \) : \( = 1 \) if DC \( j \) is opened; \( = 0 \) otherwise

2.3 Mathematical Model

The upper level model is as follow.\n
\[
\begin{align*}
\text{Min} & \quad \sum_{n \in P} \sum_{i \in S} x_{i}c_{ij} + \sum_{k \in D} d_{k} \sum_{j \in D} c_{jk} y_{jk} + \sum_{j \in D} c_{j} \sum_{i \in S} x_{ij} + \sum_{j \in D} f_{j} z_{j} \\
\text{S.T.} & \quad \sum_{i \in S} x_{i} = \sum_{k \in D} d_{k} \quad \forall j \in D \\
& \quad \sum_{i \in S} x_{i} \leq L_{i} \quad \forall i \in P
\end{align*}
\]
\[
x_i \leq L \cdot z_j \quad \forall i \in P, j \in D \quad (5)
\]
\[
\sum_{j \in D} z_j \geq 1 \quad (6)
\]
\[
z_j = \{0, 1\} \quad \forall j \in D \quad (7)
\]
\[
\sum_{x_j} \geq 0 \text{ and integer} \quad \forall i \in P, j \in D \quad (8)
\]

The objective function (2) is to minimize the total logistics cost which includes the transportation cost between plants and DCs, the transportation cost between DCs and customers, the warehousing cost, and the fixed cost of open DCs. Constraint (3) ensures that the total flow transport to the DCs must equal to total demand. Constraint (4) states that the total flow transported from plant \(i\) cannot exceed its capacity. Constraint (5) guarantees that plant \(i\) can transport to DC \(j\) only if DC \(j\) is opened. Constraint (6) ensures that at least one distribution center is opened. Constraints (7) and (8) are the binary integral and integer constraint for decision variables, respectively.

The lower level model is as follow.

\[
\text{Min} \left\{ \sqrt{\sum_{j \in D} \left[ \sum_{k \in C} \left( \frac{d_{jk} y_{jk}}{w_j} \right)^2 - \sum_{j \in D} \frac{d_{jk} y_{jk}}{w_j} \right]^{2/3}} \right\} \quad (9)
\]

\[
\sum_{j \in D} y_{jk} = 1 \quad \forall k \in C \quad (10)
\]

\[
\sum_{j \in D} d_{jk} y_{jk} \leq w_j z_j \quad \forall j \in D \quad (11)
\]

\[
y_{jk} = \{0, 1\} \quad \forall j \in D, \forall k \in C \quad (12)
\]

Objective function (9) is to minimize the mean square error of utilization ratio on all opened DCs. Constraint (10) ensures that a customer must be served by one and only one DC, which indicates the demand cannot be split. Constraint (11) states that customers can be served by DC \(j\) only if the DC is open. Constraint (12) is the binary integrality constraint.

It is obviously that the proposed mixed integer bi-level problem is difficult to solve since the upper level is a capacitated facility location problem, not to mention the non-linear objective function for the lower level problem. It is shown that such a bi-level problem is difficult to solve (Bard and Moore, 1992; Shi et al., 2006). Thus, metaheuristic approaches have been applied for solving bi-level programming problems, such as genetic algorithms (Calvete et al., 2008), simulated annealing (Sahin and Ciric, 1998), tabu search (Wen and Yang, 1990), and ant colony optimization (Calvete et al., 2011). Dempe (2002) provided a recent survey that covers applications and major theoretical developments.

### 3. The Threshold Accepting Algorithm

The threshold accepting (TA) was first introduced by Dueck and Scheuer (1990) as a deterministic version of the classical simulated annealing (SA) algorithm. TA adopts a simpler acceptance criterion for new solutions and does not require the generation of random numbers and exponential functions. This implies that better solutions are always accepted while worse solutions are also accepted if their objective value is within a certain threshold from the current objective value. The key components of TA are the function that determines the lowering of the threshold during the course of the procedure, stopping criteria as well as the methods used to create initial and neighboring solutions. In general, the threshold is gradually decreased to zero as the heuristic proceeds. The main advantages of TA are its conceptual simplicity and its excellent performance on different combinatorial optimization problems.

The upper level objective function depends on both upper and lower level decision variables. In order to solve the problem, the procedure we propose constructs feasible solutions of the bi-level problem. For this purpose, the procedure uses the TA to find a feasible solution of the upper level capacitated facility location problem (CFLP). At the same time, the resolution of the lower level optimization problem is the embedded using the information about the DC needs given by the CFLP solution. The partial upper level problem has the characteristics of a transportation problem once the lower level allocation problem is solved. Hence, it can be efficiently solved to optimality by CPLEX. Since we are solving a bi-level problem, the overall evaluation takes into account the upper level objective function, which involves both upper and lower level variables.

Our TA is inspired by our previous study on the single source capacitated facility location problem (SSCFLP) (Chen and Ting, 2008), in which a trade-off between fixed costs of the selected hubs and transportation costs arises. Since the upper level problem is a facility location problem, we adopt an add strategy to solve the SCDN in our TA: starting with a network consisting of minimum required number of DCs based on the DCs’ capacities and total customer demand, and adding one DC at a time, we aim to find the corresponding best solutions of upper and lower problems for each different value of DCs, and stop the algorithm when the corresponding objective function values cannot be improved. The overall procedures of the DTA are as follows.

1. **Step 1:** Set the number of open DCs equal to the minimum required number of DCs based on the total demand \((N = \min)\). Set \(F_S(X, Y, Z) = \infty\).
2. **Step 2:** Generate the initial solution of opening \(N\) DCs \((Z)\) for the upper level TA, \(S_0(Z)\). Pass the opened DC location decisions \((Z)\) to lower level TA. Solve the lower level TA for the customer assignment \((Y)\) for the opened DCs to minimize the lower level objective function value.
3. **Step 3:** Feedback the best customer allocation solution \(S_{c1}(X, Y, Z)\) from the lower level TA to the upper level. Solve the resulting transportation problem in the upper level for given opened DCs and customer allocation \((Y, Z)\) by CPLEX. The flow \((X)\) from plants to opened DCs is then used to compute the upper level objective function value \(TC(S_{c1})\) for the current solution \(S_{c1}(X, Y, Z)\).
4. **Step 4:** Repeat steps 2 and 3 until the number of iterations for given \(N\) open DCs is reached. Compute the best objective function \(F_S(X, Y, Z)\) for the best upper level solution \(S_{c1}(X, Y, Z)\).
5. **Step 5:** If \(F_S(X, Y, Z) > F_S(X, Y, Z)\), stop, output the best solution \(S_{c1}(X, Y, Z)\); otherwise, \(N = N + 1\), go to step 2.

### 3.1 The Upper Level TA (ULTA)

Since the distribution center has capacity limitation, we can first sort the DC in descending order based on its capacity. The minimum number of required DCs \((\min)\) can be obtained. We then start the search with \(N = \min\), and gradually increase the number of open DCs until the upper level objective function value cannot be improved. For each DC, we calculated the average cost of each DC which is the sum of average unit transportation cost from all plants to the DC and from the DC to all customers, the unit warehousing cost, and the fixed charge of establishing the DC of each DC is calculated. Sort this cost in ascending order. The initial
solution for \( N = \text{min} \), the DC with the smallest average cost will be selected. The initial solution when adding one more DC. The initial solution includes the best locations for \( N - 1 \) and the one on top of the list which is not selected. This solution is then passed to the lower level, and the lower level TA is implemented to solve the lower level problem. The solution of the lower level problem is the customer allocation to the opened DCs, which should be returned to the upper level. After that, the left part of the upper level problem is simplified into a transportation problem, which can be solved by CPLEX easily.

The neighborhood search for the location decisions in the upper TA uses one of the following two ways based on the selection probability \( H \): (1) change the location of the previous \((N-1)\) open DCs; (2) change the newest selected DC to another one. For example, if \( H = 0.5 \), half of the moves will use (1) and the other half will use (2). Both moves will be chosen based on the probability. The idea is to switch between the locations for the selected DC or the new selected DC to avoid being trapped in the local optima. Two approaches are used to set the probability \( H \): fixed and adaptive. The first approach is to set a fixed probability, for example \( H = 0.5 \), while the latter one is to set the initial and final probability and the probability will be changed gradually based on the number of iterations has been implemented.

### 3.2 The Lower Level TA (LLTA)

Once the locations of opened DCs \((Z)\) is determined, they are transferred to the lower level as the known parameter, customers are sorted in ascending order based on the demand. Start from the top of the list, the customer is then assigned to the DC randomly but must meet the DC capacity. Once a customer is assigned to the DC, it will be removed from the list. Repeat this step until all the customers are assigned to a single opened DC. The objective function of the lower level is then computed.

Since the objective of the lower level is to minimize the mean square error of utilization ratio on all opened DCs, we will two DCs with the maximum and minimum utilization ratios, respectively. One customer is randomly selected from both DCs and exchange two customers’ allocation. The idea is to reduce the objective function value directly.

### 4. Numerical Examples

This section gives numerical results on the performance of DTA. The algorithm is coded in Visual Studio 2005 C++ programming language and run on an Intel Core 2 Duo 2.66 GHz CPU with 2G RAM PC in Windows XP operating system. Since there are not related works on the same problem as we discussed here, no data for comparison exist. Hence, we present the results of tests conducted on an instance derived from Altiparmak et al. (2009) for multiple objective supply chain network design problem. Two cases are tested in this paper and each one is run 10 times.

We generate two cases based on Altiparmak et al. (2009). The supply chain distribution network characteristics and the complexity of each case are shown in table 1. The customers, candidate DCs and plants are randomly generated from a uniform distribution over a square side of 100. Euclidean distances are used as transportation costs between nodes on each stage of network. All the other parameters are generated from uniform distributions with corresponding ranges. The demand of each customer is generated according to \( U[50000, 150000] \). The supply chain distribution network characteristics and the complexity of each case are shown in table 1.

![Table 1: Network characteristics and complexity of tested cases](image)

| Case | \(|P|\) | \(|D|\) | \(|C|\) | # of variables | # of constraints |
|------|--------|--------|--------|----------------|-----------------|
| 1    | 5      | 7      | 20     | 182            | 75              |
| 2    | 20     | 30     | 80     | 3030           | 761             |

Based on the preliminary runs, the parameter setting for the upper and lower level TA is shown in table 2.

![Table 2: Control parameters for the upper and lower level TA](image)

<table>
<thead>
<tr>
<th>(T_0)</th>
<th>(K)</th>
<th>(L)</th>
<th>(H)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ULTA</td>
<td>0.01 (\times f(X, Y, Z))</td>
<td>30</td>
<td>20</td>
</tr>
<tr>
<td>LLTA</td>
<td>0.01 (\times f(X, Y, Z))</td>
<td>300</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 3 shows the results for the two tested cases for 10 runs. The headings for the rows are number of opened DCs N, the best minimum value of the upper level objective function ULSC, the lower bound of upper level objective function value UB, the maximum utilization ratio among opened DCs Max UT, the minimum utilization ratio among opened DCs Min UT, the average utilization ratio Avg UT, the gap between average utilization ratios and the overall utilization ratio between total demand and overall capacity gap, the balance degree associate with the lower level utilization ratios BD, and the CPU time in seconds. The overall utilization ratio is computed as ratio between total demand and available capacity from opened DCs in eq. (13). The gap is then calculated as in eq. (14). The balance degree is defined as the utilization ratio difference between the maximum and minimum values and computed in eq. (15). BD of 100% represents that the each opened DC has the same utilization ratio. It is observed that the balance degree of both cases is reaching over 99.9%.

![Table 3: Results for the tested cases](image)

<table>
<thead>
<tr>
<th>Case</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC opened</td>
<td>2, 6, 7</td>
<td>5, 7, 8, 14, 18, 19, 20, 27, 29</td>
</tr>
<tr>
<td>ULSC (million)</td>
<td>403.34</td>
<td>1563.45</td>
</tr>
<tr>
<td>Max UT</td>
<td>84.29</td>
<td>98.08</td>
</tr>
<tr>
<td>Min UT</td>
<td>84.28</td>
<td>97.98</td>
</tr>
<tr>
<td>Avg UT</td>
<td>84.28</td>
<td>98.04</td>
</tr>
<tr>
<td>gap (%)</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>BD (%)</td>
<td>99.99</td>
<td>99.90</td>
</tr>
<tr>
<td>CPU1 (sec.)</td>
<td>93.55</td>
<td>616.86</td>
</tr>
</tbody>
</table>

\[
AVG^* = \frac{\sum_{i=1}^{N} d_i}{\sum_{j=1}^{M} w_j}
\]  
\[
gap = \frac{\text{Avg UT} - \text{AVG}^*}{\text{AVG}^*} \times 100\%
\]  
\[
BD = \left[ \frac{\text{Max UT} - \text{Min UT}}{\text{Avg UT}} \right] \times 100\%
\]
The conventional distribution network design problem that relaxes the utilization ratio balance constraint considered in this paper and only minimizes the total logistics cost can be solved as a single objective function model and is the lower bound of our bi-level model. We solve the model with CPLEX with the upper level objective function by combining constraints from both level problems as the lower bound. Table 4 presents the results by CPLEX for the upper level objective function as the single objective. We can find that the total logistics costs for both cases are lower that those in Table 3. However, balance degree for such results is very bad. To wit, the distribution company may have very congested DC and idle DC in their network. In terms of resource allocation for the company, he has to find other customers to fulfill the waste capacity or relocate the resources accordingly.

Table 4: CPLEX Results for the tested cases

<table>
<thead>
<tr>
<th>Case</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC opened</td>
<td>3, 6, 7</td>
<td>5, 7, 16, 18, 19, 20, 27, 28, 29, 30</td>
</tr>
<tr>
<td>SC (million)</td>
<td>346.05</td>
<td>953.88</td>
</tr>
<tr>
<td>Max UT</td>
<td>98.71</td>
<td>99.38</td>
</tr>
<tr>
<td>Min UT</td>
<td>27.84</td>
<td>57.80</td>
</tr>
<tr>
<td>Avg UT</td>
<td>73.20</td>
<td>90.52</td>
</tr>
<tr>
<td>BD (%)</td>
<td>3.19</td>
<td>54.07</td>
</tr>
</tbody>
</table>

5. Conclusion

In this paper, we propose a bi-level programming model for the supply chain distribution network design problem, where two decision makers are involved. The manufacturing company outsources the distribution activity to a third party distribution company. The upper-level model is to determine the optimal locations of open distribution center and flows from plants to selected DCs by minimize the total logistics cost, while the lower level determines the customer allocation to the opened DCs to minimize the mean square error of utilization ratio on all opened DCs. Since both upper level and lower level problems are NP-hard, a threshold accepting algorithm is proposed to solve the problem. The effectiveness of the TA is tested with two cases. The results are compared with those obtained by CPLEX with single objective function. Experimental study shows that the TA found much better utilization ratio for opened DCs than that of CPLEX, though the total logistics cost is larger. These results show that the proposed TA algorithm is feasible and advantageous.

References

Solving the Bi-level Programming for Supply Chain Distribution Network with Threshold Accepting


