A Bi-level Programming for the Multi-echelon Supply Chain Distribution Network Design

Yu-Ju YANG
Senior Engineer
Material Procurement Division
AU Optronics Corporation
No.1 Xinhe Rd., Aspire Park, Lungtan
Taiwan 32543, R.O.C.
Fax: +886-3-407-8899 Ext.1150
E-mail: Corrie.Yang@auo.com

Ching-Jung TING
Associate Professor
Department of Industrial Engineering and Management
Yuan Ze University
135 Yuan Tung Road, Chung-Li
Taiwan 32003, R.O.C.
Fax: +886-3-4638907
E-mail: ietingcj@saturn.yzu.edu.tw

Abstract: This paper considers a multi-plant and multi-distribution centers (DCs) supply chain distribution network (SCDN). The decisions are to locate the open DCs, allocate customers to DCs, and DCs to plants to minimize the total logistics cost including the fixed costs and warehousing cost of DCs, and shipping cost from plants to DCs and from DCs to customers. Also, it takes workload balance requirements into consideration to avoid those open DCs being congested or idle. A bi-level programming model is proposed to deal with this problem. The upper level considers DC location problem and production-allocation problem. The lower level focuses on allocating customers to DCs such that the workload of each open DC is balanced. We propose a double threshold accepting (TA) approach to solve this problem. Four different examples are tested and the results show that our double threshold accepting algorithm can obtain a good solution efficiently.

Key Words: supply chain distribution network, bi-level programming, threshold accepting

1. INTRODUCTION

Supply chain management (SCM) may be considered as an integrated process in which a group of several organizations, such as suppliers, manufacturers, distributors and retailers, work together to acquire raw materials with a view to converting them into end products which they distribute 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 influence tactical and operational decisions. The design task involves the decision of facilities, such as manufacturing plants and distribution centers (DCs), to be opened. In addition to locating DCs and assigning customers to them, the problem also determines the allocation of plants to DCs and the commodity flows between DCs and plants. Thus, SCDN affects the cost of the distribution system, operation efficiency at DCs and the quality of the customer service level.

Sabri and Beamon (2000) proposed an integrated multi-objective supply chain (SC) model in simultaneous 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 Drexl (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, in which two kinds of variables are generally included and solved at the same time: 0-1 integer variables for locating facilities and continuous variables for determining flows of products through the network. However, the location and demand distribution decisions might be decided by two different decision makers in practice. In fact, logistics directors usually choose the nearest customers to serve and want to balance workload among operating DCs in the network. Obviously, the location problem of distribution centers can be represented as a leader-follower or Stackelberg game where the manufacturing managers are the leaders, and logistics managers are the followers who choose the DCs to serve customers. Thus, such a distribution network design problem could be formulated as a bi-level programming model.

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, which extends the traditional distribution network design problem and incorporates cross-docking in the supply chain environment. 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 modeling approach comprising two linear programming models, one for production planning and one for distribution planning. These 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.

Roghanian et al. (2007) presented a bi-level stochastic multi-objective linear programming model, one level for production planning and another for 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. Yamada et al. (2009) proposed a bi-level model for 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.

In this paper, we assume that bi-level programming problem is made up of the upper-level problem that the decision maker of production network minimizes total logistics cost, while the lower level decision maker tries to balanced workload at open 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. We offer two important contributions to the TA literature. First, we extend the breadth of applications by studying a bi-level programming problem. Second, we evaluate the computational performance under a variety of problem scenarios and TA control parameter settings.

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 SCDN design problem are presented. In section 3, the proposed double 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

In the bi-level programming problem (BLPP), each decision maker tries to optimize its own objective function(s) without considering the objective(s) of the other level, but the decision of each level affects the objective value(s) of the other level as well as the decision space. Using the common notation (Anandalingam and Friesz, 1992), the general formulation of a bi-level programming problem can be formulated as

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

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 \( x \in \mathbb{R}^n \) and the lower level (follower) variables \( y \in \mathbb{R}^m \). The functions \( F(x, y) \) and \( f(x, y) \) are the upper level and lower level objective functions, respectively, while the functions \( G(x, y) \) and \( g(x, y) \) are the upper level and lower level constraint sets, respectively. The decision sequence is as follows: the upper level minimizes the objective \( F(x, y) \) 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 \( f(x, y) \). 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 manager 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 upper level problem is also known as the leader’s problem, while the lower level problem is sometimes called a follower’s problem. The two problems are connected in a way that the leader’s problem sets parameters influencing the follower’s problem and the leader’s problem, in turn, is affected by the outcome of the follower’s problem. In the decision making process, a follower makes decisions using only its local information while a leader does using

678
the complete information including the follower’s possible reaction to the leader’s decision.

Compared with the traditional single-level programming models, the bi-level programming models have much more advantages. The main advantages are that (1) the bi-level programming can be used to analyze two different and even conflict objectives at the same time in the decision-making process; (2) the bi-level programming methods can explicitly represent the mutual action between the upper level and lower level decision makers.

Due to the location problem of logistics distribution centers involving two kinds of decision-makers that are system planners and the logistics managers who have different objective functions. Obviously, it is appropriate that the bi-level programming model is adopted to describe the location problem.

2.2 Assumptions
The assumptions for the bi-level programming model of the supply chain distribution network design problem are as follows.
1. The capacities of manufacturing plants are known in advance and they all produce the same product.
2. Customer demands are known and must be delivered by a single distribution center.
3. All the decisions for manufacturing and transportation are made within a single period.
4. The locations for all candidate distribution centers are known. These distribution centers have unlimited capacity.
5. The unit warehousing cost, and fixed operating charge at each distribution center are known.
6. The aggregate demand for each distribution center can be delivered from different plants.
7. The unit transportation costs from facilities to facilities are given.

2.3 Mathematical Model
The notation used is as follow.
\( c_{ij} \): Unit transportation cost from node \( i \) to \( j \)
\( c_j \): Unit warehousing cost at DC \( j \)
\( C \): Set of customers
\( d_k \): Demand of Customer \( k \)
\( D \): Set of candidate DC
\( f_j \): Fixed cost at DC \( j \)
\( l_i \): Capacity at plant \( i \)
\( P \): Set of plants
\( t_{jk} \): Transportation time from DC \( j \) to customer \( k \)
\( x_{ij} \): Flow from plant \( i \) to DC \( j \)
\( y_{jk} = \begin{cases} 1 & \text{if customer } k \text{ is served by DC } j \\ 0 & \text{otherwise} \end{cases} \)
\( z_j = \begin{cases} 1 & \text{if DC } j \text{ is open} \\ 0 & \text{otherwise} \end{cases} \)

The upper level model is to minimize the total supply chain cost while the lower level is to minimize the maximum workload among open DCs. The decision variables for upper level are the locations of the open DC \( z_j \) and the flow from each plant to the open DC \( x_{ij} \), while the lower level is the flow from the open DC to each customer \( y_{jk} \).
2.3.1 Upper Level Model
The upper level is to minimize the total cost for the supply chain network.

\[
\text{Min } F(X,Y,Z) = \sum_{i \in P} \sum_{j \in D} x_{ij} c_{ij} + \sum_{k \in C} d_k \sum_{j \in D} y_{jk} c_{jk} + \sum_{j \in D} \sum_{i \in P} x_{ij} + \sum_{j \in D} f_j z_j
\]  

(2)

S. T. \[ \sum_{i \in P} \sum_{j \in D} x_{ij} \geq \sum_{k \in C} d_k \]  

(3)

\[ \sum_{j \in D} x_{ij} \leq l_i \quad \forall i \in P \]  

(4)

\[ x_{ij} \leq l_i z_j \quad \forall i \in P, j \in D \]  

(5)

\[ \sum_{j \in D} z_j \geq 1 \]  

(6)

\[ z_j = \{0,1\} \quad \forall j \in D \]  

(7)

\[ x_{ij} \geq 0 \text{ and integer} \quad \forall i \in P, j \in D \]  

(8)

The objective function (2) is to minimize the total 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. \(X\) and \(Y\) are the matrices for \(x_{ij}\) and \(y_{jk}\), respectively. \(Z\) is the vector for \(z_j\). Constraint (3) ensures that the total flow transport to the DCs must greater than or 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 integral constraint for decision variables, respectively.

2.3.2 Lower Level Model
The lower level employs a mini-max operator to calculate the balance workload.

\[
\text{Min } G(Y,Z) = \text{MinMax} \left\{ \sum_{j \in D} t_{jk} y_{jk} \right\}
\]  

(9)

S. T. \[ \sum_{j \in D} y_{jk} = 1 \quad \forall k \in C \]  

(10)

\[ y_{jk} \leq z_j \quad \forall j \in D \]  

(11)

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

(12)

Objective function (9) is to minimize the maximum workload among those open 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 has been reported that even for the BLPP case where all the functions in (1) are linear, the problem is non-convex and NP-hard (Ben-Ayed et al., 1988; Bard and Moore, 1990; Shi et al., 2006). Since the upper level is a facility location related problem, while the lower level is the min-max assignment problem. Both problems are proved be to NP-hard problems by Gourdin.
et al. (2000) and Yamada et al. (1996). A double threshold accepting algorithm (DTA) is proposed to solve this BLPP problem in next section.

3. PROPOSED 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.

Assuming that \( S \) is the set of all feasible solutions of the problem, TA starts with an initial solution \( S_0 \in S \), which may be generated randomly or used a simple method. Then, the method proceeds in an iterative manner. In each iteration, the algorithm computes the difference between the new solution \( S_n \) and the current solution \( S_c \). If the difference is within the threshold, then the old one will be replaced by the new one; otherwise, another solution will be generated. If the current solution is less than the best solution so far \( S_b \), then the best solution will be replaced by \( S_c \). The initial threshold \( T_0 \) is set to be a ratio, which is 0.01 in this research, of the initial objective function value. For the same threshold value, the neighborhood move will be carried out for up to \( L \) times, then the threshold is updated by eq. (13). In our research, the threshold is reduced by a ratio \( \gamma \) \((0 < \gamma < 1)\). The stopping criterion is when the number of threshold updates reaches \( K \) times in our TA.

\[
T_{j+1} = \gamma \times T_j \quad j = 0, 1, \ldots, K-1
\] (13)

Our TA is a double TA approach (DTA), which includes the upper level TA to solve the DC location and flow decisions from plants to DCs and lower level TA to solve the order allocation to the open DC decided by the upper TA. Our TA is inspired on our previous study on the single source capacitated facility location problem (SSCFLP) (Chen and Ting, 2008). In SSCFLP, a trade-off between fixed costs of the selected hubs and transportation costs arises. In other words, as the number of selected hubs increases, transportation costs tend to decrease. Therefore, we adopt an add strategy to solve the SCND in our DTA: starting with a network consisting of one DC (i.e., \( N = 1 \)), and adding one DC at a time (i.e., \( N = N + 1 \)), we aim to find the corresponding best solution for each different value of \( N \), and stop the algorithm when the corresponding objective function value of \( N \) is larger than that for \( N - 1 \). In other words, we apply DTA iteratively in finding the best configuration corresponding for the given \( N \) DCs and the allocation of customers and add one DC at a time until the objective function value cannot be improved. The flowchart of the DTA is shown in figure 1. The overall procedures of the DTA are as follows.

**Step 1:** Set the number of open DC equal to 1 (\( N=1 \)), the upper level objective function value of no open DC is \( \infty \) \((F_0(X, Y, Z) = \infty)\).
Figure 1 The flowchart for the proposed DTA algorithm
Step 2: Generate the initial solution of open N DC (Z) for the upper level TA, S01(Z). Pass the open DC location decisions (Z) to lower level TA. Solve the lower level TA for the customer assignment (Y) for the open DCs.

Step 3: Feedback the best customer assignment solution S_{B2}(Y, Z) from the lower level TA to the upper level. Solve the resulting transportation problem in the upper level for given open DCs and customer allocation (Y, Z) by CPLEX. The flow (X) from plants to open DCs is then used to compute the upper level objective function value TC(S_{01}) for the current solution S_{01}(X, Y, Z).

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_N(X, Y, Z) for the best upper level solution S_{B1}(X, Y, Z).

Step 5: If F_N(X, Y, Z) > F_{N-1}(X, Y, Z), stop, output the best solution S_{B1}(X, Y, Z); otherwise, N = N + 1, go to step 2.

3.1 The Upper Level TA

3.1.1 The Initial Solution
We start the search with N = 1, and gradually increase the number of open DCs until the upper level objective function value cannot be improved. The sum of average unit transportation cost from all plants to the DC, the unit warehousing cost, the average unit transportation cost from the DC to all customers, and the fixed charge of establishing the DC of each DC is calculated. Sort this cost in ascending order. The initial solution for N = 1, the DC with the smallest average cost will be selected. When N ≥ 2, the initial solution includes the best locations for N - 1 and the one on top of the list which is not selected will be added to the solution. The partial solution is then transferred 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 open 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.

3.1.2 Neighborhood Search Moves
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). If the number of open DCs is one, only the second one will be used. Otherwise, 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.

3.2 The Lower Level TA
The lower level TA is used to solve the workload balance problem. The flowchart for the lower level TA is shown in figure 2.

3.2.1 The Initial Solution
Once the partial solution of upper level (locations of open DCs, Z) is transferred to the lower level as the known parameter, customers are sorted in ascending order based on the product of demand and the transportation time to the open DCs. The product represents the contribution of each customer demand to every open DC workload. The customer is then assigned to the DC which will yield the smallest workload among all open DCs. 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 open DC. The maximum cumulative workload among the open DCs is the objective function value for the lower level problem.

\begin{itemize}
\item \( S_{02}(X) \): Initial solution of LL
\item \( S_{N2}(X) \): New solution of LL
\item \( S_{C2}(X) \): Current solution of LL
\item \( S_{B2}(X) \): Optimal solution of LL
\item \( TC(\cdot) \): Objective value
\item \( T_2 \): Threshold value of LL
\item \( Y \): Allocation relationship from DC to customer
\item \( Z \): Combination of chosen DCs
\end{itemize}

3.2.2 Neighborhood Search Moves
Since the objective of the lower level is to balance the workload among open DCs, we will choose one the two moves to optimize the objective: (1) balance the workload between the maximum and minimum workload DCs by exchange two customers between DCs; (2) randomly select two customers and exchange their DC allocations. The idea for the first move is to reduce the objective function value directly, while the second move is to avoid that the first move might not be available for exchange. Similar to the neighborhood search moves in the upper level, the move selection probability is also controlled by \( H \).
4. NUMERICAL EXPERIMENTS

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. Four cases are tested in this paper and each one is run 10 times. The data for case 1 is from Melachrinoudis and Min (2000) and Huang and Liu (2004). Cases 2-4 are randomly generated based on the similar way in Altiparmak et al. (2009). The supply chain distribution network characteristics and the complexity of each case are shown in table 1.

<table>
<thead>
<tr>
<th>Case</th>
<th># of plants</th>
<th>Candidate DCs</th>
<th># of customers</th>
<th># of variables</th>
<th># of constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>7</td>
<td>21</td>
<td>168</td>
<td>46</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>15</td>
<td>50</td>
<td>915</td>
<td>227</td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>30</td>
<td>80</td>
<td>3030</td>
<td>732</td>
</tr>
<tr>
<td>4</td>
<td>30</td>
<td>50</td>
<td>100</td>
<td>6550</td>
<td>1682</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>T0</th>
<th>K</th>
<th>L</th>
<th>H</th>
</tr>
</thead>
<tbody>
<tr>
<td>ULTA</td>
<td>0.01× F(X, Y, Z)</td>
<td>30</td>
<td>20</td>
</tr>
<tr>
<td>LLTA</td>
<td>0.01× F(X, Y, Z)</td>
<td>100</td>
<td>L(N+1) = L(N)+50</td>
</tr>
</tbody>
</table>

For case 1, the data for the transportation cost between each pair of plant and DC, fixed charge and unit warehousing cost at each DC are shown in table 3, while the transportation time and cost from an open DC to the customer, and customer demand are presented in table 4. For cases 2-4, 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[100000, 300000]. The unit cost of warehousing cost for DCs is 1% of the transportation time. The unit transportation costs from the plant to the open DC are generated from U[1, 10]. The fixed cost of DCs are generated from U[10000, 50000], the fixed cost of plants are generated from U[50000, 150000]. The transportation time from an open DC to the customer is generated according to U[50, 1500]. The total capacity of plants is calculated as two times of total customer demand.

<table>
<thead>
<tr>
<th>i</th>
<th>j</th>
<th>Transportation cost ($/unit)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>1.8</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>3.5</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>3.35</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>2.84</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>3.5</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>2.6</td>
</tr>
</tbody>
</table>

For cases 2-4, 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[100000, 300000]. The unit cost of warehousing cost for DCs is 1% of the transportation time. The unit transportation costs from the plant to the open DC are generated from U[1, 10]. The fixed cost of DCs are generated from U[10000, 50000], the fixed cost of plants are generated from U[50000, 150000]. The transportation time from an open DC to the customer is generated according to U[50, 1500]. The total capacity of plants is calculated as two times of total customer demand.

<table>
<thead>
<tr>
<th>i</th>
<th>Fixed cost</th>
<th>Warehousing cost ($/unit)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1,500,000</td>
<td>4.00</td>
</tr>
<tr>
<td>2</td>
<td>3,215,569</td>
<td>2.29</td>
</tr>
<tr>
<td>3</td>
<td>3,197,605</td>
<td>2.26</td>
</tr>
<tr>
<td>4</td>
<td>3,327,844</td>
<td>2.44</td>
</tr>
<tr>
<td>5</td>
<td>3,000,000</td>
<td>2.00</td>
</tr>
<tr>
<td>6</td>
<td>3,094,311</td>
<td>2.13</td>
</tr>
<tr>
<td>7</td>
<td>3,251,497</td>
<td>2.34</td>
</tr>
</tbody>
</table>
Table 4  Transportation time, transportation cost and demand for different customers  

<table>
<thead>
<tr>
<th>k</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>demand(unit)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>389/3.1</td>
<td>560/3.31</td>
<td>444/3.3</td>
<td>595/3.8</td>
<td>524/3.28</td>
<td>372/3.2</td>
<td>1,098,953</td>
</tr>
<tr>
<td>2</td>
<td>386/3</td>
<td>0/2.5</td>
<td>232/2.83</td>
<td>76/2.9</td>
<td>263/3</td>
<td>347/3</td>
<td>78/2.7</td>
<td>1,818,654</td>
</tr>
<tr>
<td>3</td>
<td>554/3.5</td>
<td>230/2.9</td>
<td>0/2.47</td>
<td>323/3.2</td>
<td>474/3.5</td>
<td>485/3.15</td>
<td>251/3</td>
<td>1,318,788</td>
</tr>
<tr>
<td>4</td>
<td>127/2.1</td>
<td>269/2.9</td>
<td>440/3.14</td>
<td>560/3.31</td>
<td>444/3.3</td>
<td>595/3.8</td>
<td>372/3.2</td>
<td>1,098,953</td>
</tr>
<tr>
<td>5</td>
<td>1154/5.8</td>
<td>955/3.9</td>
<td>733/3.62</td>
<td>885/4</td>
<td>734/4.3</td>
<td>722/3.6</td>
<td>916/4.1</td>
<td>248,418</td>
</tr>
<tr>
<td>6</td>
<td>927/4.8</td>
<td>652/3.5</td>
<td>429/3.14</td>
<td>582/3.6</td>
<td>440/3.5</td>
<td>419/3.12</td>
<td>613/3.6</td>
<td>746,540</td>
</tr>
<tr>
<td>7</td>
<td>974/4.9</td>
<td>699/3.5</td>
<td>476/3.19</td>
<td>629/3.6</td>
<td>487/3.6</td>
<td>466/3.17</td>
<td>660/3.6</td>
<td>730,039</td>
</tr>
<tr>
<td>8</td>
<td>877/4.6</td>
<td>548/3.3</td>
<td>344/2.99</td>
<td>478/3.4</td>
<td>395/3.07</td>
<td>515/3.4</td>
<td>686,354</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>904/4.7</td>
<td>565/3.3</td>
<td>371/3.04</td>
<td>495/3.4</td>
<td>427/3.13</td>
<td>541/3.5</td>
<td>1,062,399</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>770/4.1</td>
<td>495/3.2</td>
<td>273/2.88</td>
<td>425/3.3</td>
<td>284/3</td>
<td>262/2.86</td>
<td>456/3.3</td>
<td>1,288,812</td>
</tr>
<tr>
<td>11</td>
<td>1121/6.1</td>
<td>1050/4</td>
<td>827/3.76</td>
<td>980/4.1</td>
<td>839/4.6</td>
<td>817/3.74</td>
<td>1011/4.2</td>
<td>412,680</td>
</tr>
<tr>
<td>12</td>
<td>808/4.4</td>
<td>426/3.1</td>
<td>438/3.1</td>
<td>376/3.3</td>
<td>419/3.4</td>
<td>554/3.28</td>
<td>451/3.3</td>
<td>845,273</td>
</tr>
<tr>
<td>13</td>
<td>81/2</td>
<td>434/3.1</td>
<td>606/3.38</td>
<td>489/3.4</td>
<td>640/4</td>
<td>561/3.24</td>
<td>417/3.2</td>
<td>365,089</td>
</tr>
<tr>
<td>14</td>
<td>247/2.5</td>
<td>188/2.8</td>
<td>332/2.95</td>
<td>215/3</td>
<td>368/3.04</td>
<td>144/2.8</td>
<td>2,542,647</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>172/2.3</td>
<td>322/3</td>
<td>462/3.18</td>
<td>356/3.2</td>
<td>307/3.6</td>
<td>358/3.03</td>
<td>285/3</td>
<td>1,754,643</td>
</tr>
<tr>
<td>16</td>
<td>713/4.1</td>
<td>384/3.1</td>
<td>180/2.74</td>
<td>314/3.2</td>
<td>127/2.6</td>
<td>227/2.82</td>
<td>350/3.2</td>
<td>752,373</td>
</tr>
<tr>
<td>17</td>
<td>195/4.4</td>
<td>194/3.2</td>
<td>365/2.88</td>
<td>248/3.3</td>
<td>399/2.8</td>
<td>415/2.97</td>
<td>177/3.3</td>
<td>714,202</td>
</tr>
<tr>
<td>18</td>
<td>298/2.7</td>
<td>100/2.7</td>
<td>290/2.92</td>
<td>174/3</td>
<td>392/3.05</td>
<td>104/2.8</td>
<td>1,265,406</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>1034/5.2</td>
<td>658/3.5</td>
<td>528/3.3</td>
<td>597/3.6</td>
<td>475/3.5</td>
<td>584/3.38</td>
<td>671/3.7</td>
<td>548,858</td>
</tr>
<tr>
<td>20</td>
<td>531/3.5</td>
<td>149/2.7</td>
<td>325/2.96</td>
<td>169/3</td>
<td>440/3.14</td>
<td>227/3</td>
<td>949,867</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>1024/5.1</td>
<td>749/3.6</td>
<td>529/3.28</td>
<td>679/3.7</td>
<td>537/3.8</td>
<td>516/3.27</td>
<td>710/3.7</td>
<td>552,111</td>
</tr>
</tbody>
</table>

* transportation time  
* transportation cost ($/unit)

Table 5 shows the results for the case 1 at different numbers of open distribution centers for 10 runs. The headings for the rows are number of open DCs N, the best minimum value of the upper level objective function ULSC, the average of upper level objective function value AULSC, the lower level objective function value corresponding to the best upper level value BWL, the average of the lower level objective function value in 10 runs ABWL, the minimum workload of all the open DCs corresponding the best upper level value MWL, the average workload associated with the best upper level value AWL, the balance degree associate with the best upper level value BD, and the CPU time in seconds. The balance degree is computed in eq. (14). A smaller AWL suggests that the average customer service level provided by the DCs is better. BD of 100% represents that the workload allocation has reached. We also compare our upper level objective function values with those found by GA in Huang and Liu (2004), the results show that ours are better than or equal to theirs.

Table 5 Results for different open DCs of case 1  

<table>
<thead>
<tr>
<th>N</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>1,2,3,5,7</th>
<th>1,2,3,5,6,7</th>
<th>1,2,3,4,5,6,7</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC open</td>
<td>2</td>
<td>2.5</td>
<td>2.57</td>
<td>1.2,3,5</td>
<td>1.2,3,5,7</td>
<td>1.2,3,5,6,7</td>
<td>1,2,3,4,5,6,7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ULSC (million)</td>
<td>167.38</td>
<td>165.85</td>
<td>169.32</td>
<td>172.08</td>
<td>175.37</td>
<td>179.48</td>
<td>183.80</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(0.00%)</td>
<td>(0.00%)</td>
<td>(-0.34%)</td>
<td>(-1.52%)</td>
<td>(-1.01%)</td>
<td>(-0.86%)</td>
<td>(-1.01%)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AULSC (million)</td>
<td>167.38</td>
<td>165.85</td>
<td>169.32</td>
<td>172.08</td>
<td>175.37</td>
<td>179.48</td>
<td>183.80</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BWL</td>
<td>7236.14</td>
<td>2822.32</td>
<td>1862.95</td>
<td>1201.70</td>
<td>979.63</td>
<td>798.28</td>
<td>681.60</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ABWL</td>
<td>7236.14</td>
<td>2822.32</td>
<td>1862.95</td>
<td>1232.22</td>
<td>975.54</td>
<td>801.38</td>
<td>687.98</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MWL</td>
<td>7236.14</td>
<td>2789.99</td>
<td>1851.12</td>
<td>1049.87</td>
<td>884.45</td>
<td>677.87</td>
<td>618.05</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AWL</td>
<td>7236.14</td>
<td>2806.15</td>
<td>1856.10</td>
<td>1160.48</td>
<td>953.99</td>
<td>747.08</td>
<td>657.01</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BD (%)</td>
<td>100</td>
<td>98.85</td>
<td>99.36</td>
<td>86.92</td>
<td>90.02</td>
<td>83.88</td>
<td>90.33</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPUt (sec.)</td>
<td>0.98</td>
<td>13.02</td>
<td>19.64</td>
<td>28.20</td>
<td>33.63</td>
<td>42.86</td>
<td>77.55</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* improvement over the solution in Huang and Liu (2004).
Table 6 shows the final results for all four cases. The headings for each row are the same as those in table 5 except the lower bound. As mentioned in previous section, the conventional distribution network design problem that relaxes the workload balance constraint considered in this case and only minimizes the total cost can be solved as a single objective function model as 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. The balance degree in four cases are all above 96% which means our DTA can provide good solution for the workload balance at open DCs. It is also observed that the computational times increase when the supply chain distribution network becomes larger. The gap, which is measured by the difference between the lower bound and our ULSC value, in four cases ranges from 0.65% to 5.44%. The gap increases when the network size becomes larger.

<table>
<thead>
<tr>
<th>Case</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC open</td>
<td>2, 5</td>
<td>8, 10, 14</td>
<td>5, 7, 18, 20, 27, 29</td>
<td>1, 19, 33, 35, 36, 41</td>
</tr>
<tr>
<td>ULSC (million)</td>
<td>165.85</td>
<td>635.63</td>
<td>942.18</td>
<td>870.50</td>
</tr>
<tr>
<td>(0.65%)</td>
<td>(3.79%)</td>
<td>(3.62%)</td>
<td>(5.44%)</td>
<td></td>
</tr>
<tr>
<td>Lower bound (million)</td>
<td>164.78</td>
<td>612.41</td>
<td>909.24</td>
<td>825.56</td>
</tr>
<tr>
<td>BWL</td>
<td>2822.32</td>
<td>10618.15</td>
<td>6201.68</td>
<td>5374.21</td>
</tr>
<tr>
<td>MWL</td>
<td>2789.99</td>
<td>10574.49</td>
<td>5970.79</td>
<td>5287.64</td>
</tr>
<tr>
<td>AWL</td>
<td>2806.15</td>
<td>10595.49</td>
<td>6117.90</td>
<td>5333.71</td>
</tr>
<tr>
<td>BD (%)</td>
<td>98.85</td>
<td>99.59</td>
<td>96.23</td>
<td>98.38</td>
</tr>
<tr>
<td>CPUt (sec.)</td>
<td>13.02</td>
<td>98.23</td>
<td>168.78</td>
<td>456.45</td>
</tr>
</tbody>
</table>

5. CONCLUSIONS

In this paper, a bi-level programming model is proposed to describe and solve the multi-echelon supply chain distribution network design problem, in which both the total logistics cost and logistics manager’s concern are taken into account. 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 gives a customer demand allocation by minimize the maximum workload of the open DCs. Since both upper level and lower level problems are NP-hard, a double threshold accepting algorithm is proposed to solve the problem. Finally, four numerical examples are used to illustrate the applications of the method. The results show that the proposed TA algorithm is feasible and advantageous.

The bi-level model offers a flexible framework, which enables decision makers to consider other objectives as they prefer, for example, set the sum of the mean square error of all DC utilization ratio to be minimized in the lower level while minimizing total logistics cost in the upper level. The computation time is still long to solve a larger network by our TA approach, designing other effective algorithms will be another possible extension. The distribution
centers are assumed to be uncapacitated in this paper. However, the capacity decision is also important in supply chain network design problem. We are also study the capacitated distribution center case, the results will be reported in the near future.

REFERENCES

multi-objective programming problem to supply chain planning, Applied Mathematics and Computation, Vol. 188, 786-800.


