Journal of Advanced Engineering Vol. 7, No. 2, pp. 51-60 / April 2012

高價低量商品倉儲選擇及轉運調度之搜尋演算法
Search Algorithms in the Selection of Warehouses and Transshipment Arrangements for High-end Low-volume Products

洪一峯* 陳威志 陳建良
Yi-Feng Hung*, Wei-Chih Chen, James C. Chen

摘要

在當前競爭激烈及低利潤之市場條件之下，連鎖零售產業所面臨的挑戰是如何管理庫存以降低成本並改善客戶服務，以爭取更大的市場占有率。在本研究提出了一個新的轉運模式，從現有的零售點中選擇部分地點作為高價低量產品之倉庫，並負責這些產品之存貨管理及配送作業，藉由風險分擔的效果，最小化系統整體成本。然而，產品儲存點的選擇必須與庫存管理及轉運安排一併考量以達成系統整體之最佳化。本研究所提出之轉運模式可用一個非線性整數規劃問題表示，並發展了以模擬退火法及塔布搜尋法為基礎的二個區域搜尋法來求解非線性整數規劃問題，以解決傳統數學方法求解時間過長與無法求解大規模問題之困難。研究結果顯示，在小規模問題上模擬退火法與塔布搜尋法均可快速的求得最佳解。本研究將測試問題擴大至 200 個零售點，以測試大規模問題之下二個方法之求解效能，實驗結果顯示在大規模問題上，塔布搜尋法之求解時間及求解品質表現優於模擬退火。
關鍵詞：供應鏈管理，風險共擔，選址-庫存模型，模擬退火法，塔布搜尋法

Abstract

In this highly competitive and low-profit marketing environment, the challenge that chain-store resale industry face is how to implement inventory management techniques so that cost down is possible and quality of customer service can be improved simultaneously to broaden market share. In this study, a transshipment approach is proposed by selecting several retail stores not only served as a warehouse for high-end, low-volume products but also are responsible for inventory and delivering to share and lower their risk to minimize overall system costs. However, locations of warehouse to products must be considered along with inventory management and transshipment arrangements to achieve their maximum performance for the whole system. To solve this problem, a transshipment model, proposed in this research, is formulated by an integer nonlinear programming methodology. And two efficient local search algorithms based on Simulated Annealing and Tabu search are developed on integer nonlinear programming, instead of using traditional mathematical approaches, having difficulties in solving these problems with time-consuming drawback and restrictions on large scale. Study shows that both algorithms are efficient, obtaining optimal solutions on small-sized problems. In this research, computational results with up to 200 retail stores to evaluate the performance with these two algorithms in large scale are also presented. And based on the acquisition of data, Tabu Search outperforms Simulated Annealing either in time-consuming or calculating quality.
Keywords: supply chain management, risking-pooling, location-inventory model, simulation annealing, tabu search

國立清華大學工業工程與工程管理學系
*Corresponding author. E-mail: yifeng@ie.nthu.edu.tw
Department of Industrial Engineering and Engineering Management, National Tsing Hua University, Hsinchu, Taiwan, R.O.C.
Manuscript received 27 September 2011; revised 6 December 2011; accepted 22 December 2011

I. INTRODUCTION

Under fierce competition in today's global markets with shorter product life cycles, uncertainties and varying customer demands have forced business enterprises to focus on supply chains. Supply chain management aims to efficiently integrate suppliers, manufacturers, warehouses, and retail stores to minimize system-wide costs while satisfying service level requirements [1]. The main focus of reducing costs includes those of transportation, inventory, and manufacturing. In the 3C retail business, the emerging challenge is to find ways, by which to manage inventories to reduce costs and improve customer service under intense competition and lower profit margin. Keeping high stock levels for different product varieties is necessary to meet satisfactory customer service levels. However, this normally causes high inventory costs, especially for expensive, low demand 3C products. On the other hand, low inventory in a retail store leads to a low service level for usually impatient customers. Moreover, 3C products normally have very short product life cycles. New models may force price reduction for the stock of old models, causing a huge loss; or at times, making the old products unsalable. Two methods are commonly used to solve this problem. One is the pull-based approach, which places an order when a demand occurs. The other method establishes a centralized warehouse. The pull-based approach effectively reduces the inventory cost, but prolonged lead time may not be acceptable to customers, especially for 3C products. Having a warehouse requires huge investments on facility and stock inventory. Moreover, a centralized warehouse may be too far away from the retail store requesting a product, and thus, may also lead to prolonged lead time. Today's advanced and rapid transportation systems make transshipment a good alternative arrangement to improve the efficiency of supply chain operations.

The present study proposes a new business model for managing the inventory of high-end, low-volume products by applying transshipment on supply chain design for 3C chain stores. A store serves as a warehouse responsible for storing and supplying products via transshipment to another retail store that requires these. When demand in a neighboring retail store occurs, the warehouse-like retail store immediately transships the product to the ordering retail store. This means that among all the retail stores, a few are chosen to become storage points for a particular product. In addition, the storage point configuration of a certain product can be different from that of another. By integrating the selection of storage points, stock levels for each storage point, and transshipment arrangements, the same advantages of risk pooling effect can be achieved without using a traditional centralized warehouse. Furthermore, inventory management in each retail store still basically follows the pull-based approach, placing orders when demand occurs and keeps zero stock on most high-end, low-volume products. Also, more customer demands can be satisfied with a shorter lead time. Such a transshipment business model can help a 3C retail chain store achieve its goals of reducing costs and improving customer service.

In such a business model, the key problem is selecting the warehouse storage points and the transshipment arrangement among the current retail stores. This process involves the identification of retail stores that can keep the product stocks and those that can be responsible for delivery; these have to be simultaneously determined to minimize system-wide costs. Traditionally, supply chain optimization models are classified into strategic-level and tactical-level supply chain operations. The strategic-level supply chain design involves decisions on the number, location, and capacity of facilities and warehouses. In comparison, tactical-level supply chain operation involves production planning and inventory management. These two models are usually solved separately, because solving them together is a highly complex process. Daskin *et al*. [2] and Shen *et al*. [3] first developed location models with risk pooling to combine the strategic and tactical decisions into a single model. Considering the risk-pooling effect [4] and customer service levels, decisions on the warehouse number and locations, distribution, and associated inventory control at each warehouse must all be simultaneously determined to achieve cost reduction. Issues on integrated location-inventory models have gained increased attention in supply chain management literature in recent years [5-13].

This study formulates a mathematical program for the proposed transshipment model, and implements procedures based on search algorithms. The mathematical program uses the following assumptions. First, a set of retailer stores exists and only one product is considered in a problem. The well-known inventory approach called the (r, Q) policy is used to manage each warehouse-like retail store and fulfill a specified service level. Similar to previous studies on integrated inventory-location model, the considered problem can be formulated as an integer nonlinear programming (INLP) model and is regarded as an NP-hard problem [14]. Previous techniques for solving INLP [15-16] such as branch-and-bound method, out-approximation algorithm and Lagrangian relaxation approach, are non-intuitive and complicated procedures. Special structures are needed to decompose the original problem or convex assumption to obtain a tight bound function. These methods also have difficulties solving large-scale problems within a reasonable time. The present study applies search method solutions for the proposed INLP problem; these methods are simple, intuitive, and efficient. Search algorithms, such as simulation annealing (SA) proposed by Kirkpatrick *et al*. [17] and tabu search (TS) proposed by Glover [18], have successfully solved practical optimization problems in a wide range of domains. Using these, qualified solutions can be obtained within a relatively short time. Although optimality is not always guaranteed, the SA and TS are still widely used for solving large-scale optimization problems, because of their

rapid production of good solutions and the advantage of escaping from local optimal solutions. The present study implements efficient approaches based on these two search algorithms to solve the proposed problem. Computational experiments using randomly generated test problems verify the performance of both methods. For small-sized problems, the solutions from SA and TS procedures are compared with those of the optimal solution obtained by an enumeration approach. For large-sized problems, the approaches are evaluated by the solution objectives and computation times.

The manuscript is outlined as follows. Section 2 briefly reviews joint location-inventory models and research related to the techniques for solving the model. Section 3 describes the considered problem and presents the mathematical formulation. Section 4 discusses two heuristic approaches based on SA and TS. Section 5 describes the proposed algorithm configuration and shows the computational results. Finally, Section 6 presents conclusions and future research perspectives.

II. LITERATURE REVIEW

Choosing the optimal warehouse locations and determining inventory policy are the major issues in traditional supply management and are commonly treated separately. Literature on these issues is extensive. Drezner and Hamacher [19] provided excellent reviews in the theory and application of location analysis, while Zipkin [20] provided an excellent summary of the theory for inventory management.

In recent years, integrated models combining inventory management and distribution center locations have been proposed to achieve risk-pooling benefits and inventory cost reductions. Daskin *et al*. [2] and Shen *et al*. [3] considered a location-inventory model, in which they examined a distribution center location problem with continuous (r, Q) inventory policy in each distribution center to capture economics of scale and risk-pooling effects. The model explicitly considers expected inventory costs using an EOQ-based approximation when deciding on distribution center locations; thus, the objective function is nonlinear with two square-root terms. For simplicity, they assumed that demands have the same mean-variance ratio, such that the two square root terms can be combined into one. The problem can then be solved using the Lagrangian relaxation technique and column generation algorithm. Computational results show that if location and inventory decisions are separately optimized, the number of facilities opened can be more than that of the optimal solution. Considering stochastic parameters described by discrete scenarios [5-6] or relaxing and including several assumptions [7-10], a few researchers have proposed inventory-location models by including operation routing cost [11]. These studies focus on constructing the supply chain network usually formulated as a nonlinear discrete optimization problem. Arora and Huang [15] and Grossmann [16] reviewed the techniques for solving such problems. Well-known methods, such as the branch and bound and Lagrangian relaxation technique, have difficulty solving large-scale problems.

Numerous researchers have proposed heuristic algorithms, such as SA and TS, to solve large-scale problems in supply chain network design [21-24]. However, the objective functions (including location, transportation, and inventory) in these cases are all linear; in addition, economies of scale and the risk-pooling effect are not considered. Only a few studies have focused on developing efficient heuristic solutions for location-inventory models with large-scale problems. You and Grossmann [12] proposed several efficient heuristics for the generalized problem presented by Shen *et al*. [3], where the mean-to-variance ratio of each demand is not identical. They reformulated Shen's problem as a mixed-INLP problem, and used a convex relaxation model to generate an initial solution. One of their algorithms is based on the Lagrangian relaxation and decomposition algorithm, the aim of which is to obtain global or near-global optimal solutions. Solutions can be typically obtained within 1.2% of the global optimum, requiring much less computational effort. Park *et al*. [13] considered a three-level supply chain network design model with risk-pooling and lead times that are dependent on the distribution-center-to-supplier route. In the current paper, a two-phase heuristic solution algorithm is developed based on the Lagrangian relaxation approach and TS. The proposed approach is both effective and efficient even for large-size problems.

III. PROBLEM FORMULATION

1. Problem description

We consider the distribution system design problem with a set of I retail stores, with each store having an uncertain demand for a considered product. The retail store locations, distances between stores, transportation cost per unit product, ordering cost, and order lead time are the known parameters for a given problem. Each retailer location is also a candidate warehouse site for the considered product. A fixed setup cost is added when the retailer site serves as a warehouse. A warehouse requires safety stock to fulfill a specific service level. The daily demand of a retailer site is a random variable of normal distribution, with a known mean and variance, and is independent from that of another retail store. A warehouse can serve the demands of more than one retailer, while a retailer can only be assigned to a single warehouse responsible for satisfying its demand. Storage capacity is assumed to be infinite. The warehouse demand is the sum of the demands of all retailers assigned to that warehouse. We assume that the warehouse-like retail store orders its inventory from suppliers using an EOQ model to approximate the (r, Q) policy. Fig. 1 illustrates an example of the considered problem.

The problem is summarized in this section. Considering a single product, a set of known retail stores

and independent uncertain demand of each retail store, some current retail stores are selected as warehouses. Each retail store is assigned to a warehouse for supplying the product, with the objective of minimizing the total location costs, shipment, and inventory, while ensuring a specified service level.

2. Model formulation

2-1. Notation

The notations below are used for the mathematical description of the proposed problem.

Sets

I: set of indices retail stores; $|I|$ represents the number of retail stores

Indices

i: index for retail stores, for each $i \in I$

j: index for the warehouse-like retail store, for each $j \in I$

Parameters

f_j: annual fixed cost of retail store j to become a warehouse, for each $j \in I$

d_{ij}: unit transportation cost from warehouse store j to retail store i, for each $i \in I$, for each $j \in I$

p_j: unit transportation cost from a supplier to warehouse store j, for each $j \in I$

a_j: fixed cost of placing an order to warehouse store j, for each $j \in I$

h_j: inventory holding cost per unit per year at warehouse store j, for $j \in I$

μ_i: mean annul demand at retail store i, for each $i \in I$

σ_i^2: variance of daily demand at retail store i, for $i \in I$

L_j: order lead time for warehouse store j, for $j \in I$

Decision variables

$u_j \in \{0,1\}$ $u_j = 1$, if retail store j is selected as a warehouse, and 0 otherwise for $j \in I$

$x_{ij} \in \{0,1\}$ $x_{ij} = 1$, if retail store i is served by a warehouse store j, and 0 otherwise for $i \in I$ and $j \in I$

$D_j = \sum_{i \in I} \mu_i x_{ij}$ mean of annul demand at warehouse store j, for each $j \in I$

$\Gamma_j = \sum_{i \in I} \sigma_i^2 x_{ij}$ variance of daily demand at warehouse store j, for each $j \in I$

The demand at warehouse store j is normally distributed with mean and variance, because the demands at each retailer store are independent and normally distributed.

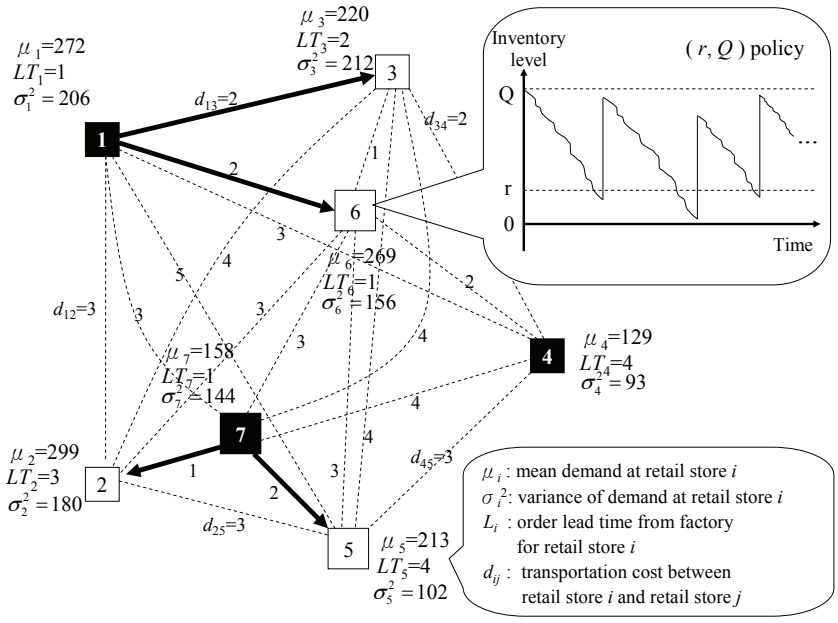

Fig. 1 Illustration of the proposed model with 7 retail stores

2-2. Formulation

The optimal ordering quantity and the reorder level at each warehouse store must first be determined. The warehouse stores are assumed to follow the (*r*, *Q*) policy, which is endogenously accounted for using the EOQ-approximation procedure [25] consisting of the EOQ cost plus a safety stock cost. The optimal ordering quantity at each warehouse store j is approximately derived by:

$$Q_j^* = \sqrt{2a_j D_j / h_j} = \sqrt{2a_j \sum_{i \in I} \mu_i x_{ij} \Big/ h_j} \quad (1)$$

The safety stock level SS_j at warehouse store *j* can be derived by:

$$SS_j = z_\alpha \sqrt{L_j \Gamma_j} = z_\alpha \sqrt{L_j \sum_{i \in I} \sigma_i^2 x_{ij}} \quad (2)$$

in which z_α is a safety factor calculated to ensure that a stock-out occurs during the lead time with a probability of α or less. The "service level" is defined by 1-α, and z_α is satisfied when $P(z \le z_\alpha) = \alpha$. The optimal inventory cost function at the warehouse store *j* is nonlinear, and can be derived using the following:

$$\begin{aligned} C_j^{Inv} &= \sqrt{2a_j h_j D_j} + h_j (z_\alpha \sqrt{L_j \Gamma_j}) \\ &= \sqrt{2a_j h_j \sum_{i \in I} \mu_i x_{ij}} + z_\alpha h_j \sqrt{L_j \sum_{i \in I} \sigma_i^2 x_{ij}} \end{aligned} \quad (3)$$

Moreover, the transportation cost function at warehouse store *j* can be derived using:

$$C_j^{Transit} = \sum_{i \in I} d_{ij} \mu_i x_{ij} + p_j \sum_{i \in I} \mu_i x_{ij} \quad (4)$$

The fixed cost for selecting retail stores as warehouses is given by:

$$C_j^{Fixed} = f_j u_j \quad (5)$$

From Equations (3), (4), and (5), the overall cost function of the problem is given by:

$$\begin{aligned} TC(u, x) = \sum_{j \in I} (& \sqrt{2a_j h_j \sum_{i \in I} \mu_i x_{ij}} + z_a h_j \sqrt{L_j \sum_{i \in I} \sigma_i^2 x_{ij}} \\ & + \sum_{i \in I} d_{ij} \mu_i x_{ij} + p_j \sum_{i \in I} \mu_i x_{ij} + f_j u_j) \end{aligned} \quad (6)$$

The INLP formulation is presented below.

Minimize

$$\begin{aligned} TC(u, x) = \sum_{j \in I} (& \sqrt{2a_j h_j \sum_{i \in I} \mu_i x_{ij}} + z_a h_j \sqrt{L_j \sum_{i \in I} \sigma_i^2 x_{ij}} \\ & + \sum_{i \in I} d_{ij} \mu_i x_{ij} + p_j \sum_{i \in I} \mu_i x_{ij} + f_j u_j) \end{aligned}$$

subject to the following:

$$x_{jj} \ge u_j \qquad \text{for each } j \in I \quad (7)$$

$$x_{ij} - u_j \le 0 \qquad \text{for each } i, j \in I \quad (8)$$

$$\sum_{j \in I} x_{ij} = 1 \qquad \text{for each } i \in I \quad (9)$$

$$u_j \in \{0,1\} \qquad \text{for each } j \in I \quad \text{and} \quad (10)$$

$$x_{ij} \in \{0,1\} \qquad \text{for each } i, j \in I \quad (11)$$

Equation (7) shows that the demand of a retail store selected as a warehouse should be served by itself. The equation $x_{jj} \ge u_j$ implies that x_{jj} = 1 when u_j = 1 for each warehouse store *j*. Equation (8) states that if a retail store is served by warehouse store j, there must be a warehouse at location j. Equation (9) states that each retail store i should be served by one and only one warehouse store j. The mathematical formulation is similar to the joint inventory-location model of Shen *et al*. [3], which focuses on the supply chain design of local blood bank, and that of You and Grossmann [12], which focuses on the supply chain of liquid oxygen. However, the proposed model in the present study differs from those of previous studies (i.e., warehouse locations). In comparison, our proposed model focuses on selecting the product storage points from current retail stores, and which warehouse store is responsible for shipping the product to a retail store. Thus, Equation (7) ensures that the demand of a warehouse store is served by itself. Moreover, the product storage point configuration can be different from that of another. Each retail store may also be responsible for storage and delivering a number of products in the transshipment approach. Furthermore, the cost function in our problem differs from that of previous studies.

The problem can be reorganized as an uncapacitated facility location (UFL) problem, which is a well-known NP-hard problem [13], when the inventory holding cost is set as $h_j = 0$. Thus, solving for large instances is very difficult due to the large number of binary variables and the nonconvex nature of the nonlinear objective function. In this work, we propose heuristic approaches based on SA and TS for efficient solutions, instead of using traditional mathematical approach. The following section presents the procedures in detail.

IV. SEARCH ALGORITHMS

Equations (6)-(11) have an integer nonlinear problem, in which decision variables are binary variables and the objective function includes nonlinear terms. Search procedures based on SA and TS are applied to efficiently obtain solutions. This section explains the basic concept and the steps involved in each of these search procedures.

1. Simulated annealing procedure

The SA concept was first proposed by Metropolis *et al*. [26] as the simulation of the evolution of a solid from a high to low energy ground state. About three decades later, Kirparick *et al*. [17] applied that criterion to introduce a combinatorial optimization algorithm called SA. The SA concept is briefly outlined here. Given the objective function value of the current solution *E*, a perturbation mechanism is then applied to generate the candidate solution S_c by a small displacement of randomly selected neighborhood solutions from the current solution *S*. The objective function value of this candidate solution is E'.

If the difference between the two corresponding objective function values, $\Delta E(=E'-E)$, is less than or equal to zero, the process is continued with this new solution. If ΔE is greater than zero, the generated solution is accepted with a probability of $P=e^{-\Delta E/T}$. The parameter T represents the current temperature, which controls the annealing process and the acceptance probability. A random number $r\in U[0,1]$ is generated and the new solution S_c is accepted if $P(=e^{-\Delta E/T})>r$. When the search CPU time within a temperature reaches a pre-specified time, a cooling process is deployed to decrease the temperature. The search continues until the stopping condition is satisfied. Prior to applying the SA process to the proposed problem, we define the following components of the SA method below.

1-1. Initial solution

According to Equations (10) and (11), the decision variables are $x_{ij}\in\{0,1\}$ for each $i,j\in I$, and $u_j\in\{0,1\}$ for every $j\in I$. In addition, the Equation (7) implies that $x_{jj}=1$ when $u_j=1$ for each warehouse store j. Thus, the number of decision variables is n^2, and the solution of the decision variables problem can be presented as a $n\times n$ matrix. Fig. 2 presents a feasible solution.

We used a greedy method to generate the initial solution, which is constructed following the steps described here. First, a set of current retail stores are randomly selected to become warehouses. A randomly generated binary value (0 or 1) is assigned to the warehouse location decision variables x_{ij} for $i=j$. Then, each remaining retail store is assigned to its nearest warehouse store. In this way, the value of demand assignment decision variables x_{ij} for $i\neq j$ becomes consistent with Equations (8) and (9).

1-2. Neighborhood solutions

We use a random neighborhood generation method in SA. A neighborhood solution is generated by changing the serving warehouse of a randomly selected retail store. Depending on whether or not the selected retail store is a warehouse, there are two possible cases. If the selected retail store is not a warehouse, its serving warehouse is randomly changed to another warehouse or the retail store itself becomes a warehouse. If the selected retail store is a warehouse and is served by itself, its warehouse role will be removed and all of retail stores served by it will served by their nearest warehouses. The focus of the neighborhood generation method is to maintain feasibility.

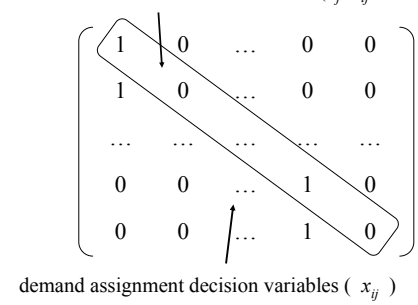

Fig. 2 Presentation of a feasible solution

1-3. Cooling schedule

The cooling function $T_k=T_{k-1}/(1+\beta\cdot T_{k-1})$ is used to compute the value T_k of temperature k from T_{k-1} of temperature $k-1$ based on Lundy and Mess [27]. If the initial temperature T_0, the final temperature T_f, and the number of cooling down n are given, then the value of β is equal to $(T_0-T_f)/(n\cdot T_0\cdot T_f)$. Furthermore, the relation between P (accepting probability) and T (temperature) is defined by $P=e^{-\Delta E/T}$. Temperatures T_0 and T_f can be computed by the relation $T=(-\Delta E)/(\ln P)$, when P_0 and P_f are given, respectively.

The SA algorithm is outlined in Algorithm 1. A summary of notations used in this algorithm is outlined below.

Variables

k: index of temperatures in SA process, assumes the temperature cools down n times, $k=0,1,\cdots,n$

S^*: the best solution found thus far

S_1: an initial solution

S: a current solution

S_c: a candidate solution selected from neighborhood of S

Parameters

P_0: probability of accepting a worse solution of first temperature

P_f: probability of accepting a worse solution of last temperature

T_0: initial temperature of SA process

T_t: trial temperature of SA process

T_f: last temperature of SA process

T_k: current temperature of SA process

t_s: computing time for temperature

β: cooling parameter

t: time

n: number of cooling down

Function

$G(S)$: the objective function of solution S

Termination condition

$STOP$: the process termination condition, which is a fixed execution time $T_{\max}$

Algorithm 1. The SA procedure

Step 1: Use a trial temperature T_t and the given P_0, P_f to compute the initial temperature T_0, and the last temperature T_f. Then, calculate the cooling parameter β by $\frac{T_0 - T_f}{n \cdot T_0 \cdot T_f}$.
Set $k = 0$, $T_k = T_0$, and set the stopping condition. Generate an initial solution S_1, and set $S = S_1$, $S^* = S_1$.

Step 2:

Step 2.1: Randomly select a candidate S_c from neighborhood of S.
Calculate $\Delta E = G(S_c) - G(S)$.
If $\Delta E \leqq 0$, then $S = S_c$ and go to step 2.2; otherwise, go to step 2.3.

Step 2.2: If $G(S_c) \leq G(S^*)$, then set $S^* = S_c$, and go to step 3.

Step 2.3: Randomly generate $X \sim U(0,1)$.
If $X \leq e^{\frac{-\Delta E}{T_k}}$, then $S = S_c$.

Step 3: Obtain CPU time t.

Step 4:

Step 4.1: If *STOP* is met, then go to step 5.

Step 4.2: If $t > t_s$, then go to step 4.3; otherwise go to step 2.

Step 4.3: Set $k = k + 1$, and compute $T_k = \frac{T_{k-1}}{1 + \beta \cdot T_{k-1}}$, then go to step 2.

Step 5: Output the best solution found and its objective value.

2. Tabu search

TS was first proposed by Glover [18], and was primarily applied to solve combinatorial optimization problems. A TS process starts from an initial solution S_1. The neighborhoods of current solution S are generated through a particular class of transformation. Short-term memory (a tabu list), is used to record the recent search trajectory and avoid cycling. Then, a best candidate solution S_c in this neighborhood and not in the tabu list is selected as the new current solution, even if it is not an improvement. This procedure is repeated until the termination condition is met. The following components of the TS algorithm applied to the proposed problem are outlined below.

2-1. Initial solution

The method of generating initial solution for TS is the same as that introduced in SA.

2-2. Neighborhood solutions

The method of generating neighborhood solutions for TS is the same as that for SA. The differences between the two methods lie in the neighborhood generation method used and the number of generated solution in each iteration. Moreover, SA randomly generates just one neighborhood solution and evaluates the solution in each iteration; whereas, TS produces and evaluates multiple neighborhood solutions in an iteration, and selects the one with best objective value as the next solution.

2-3. Short-term memory

We use a short-term memory (called tabu list) in the TS to record the recent search trajectory. Determining the number of entries in tabu list depends on the application, which can affect TS efficiency. Thus, an aspiration rule is used, which allows a tabu move prohibited by the tabu list, if the move generates a solution better than the best one found thus far.

The TS algorithm is outlined in Algorithm 2. Parameter definitions are the same as those of SA. Additional notations used in this algorithm are also defined below.

Sets

tabu_list: a short-memory of TS. The *tabu list* records the recent search trajectory of TS

Algorithm 2. The TS procedure

Step 1: Generate an initial solution S_1, and set $S = S_1$, $S^* = S_1$.

Step 2: Select S_c from neighborhood of S.
If the move $S \rightarrow S_c$ is prohibited by *tabu_list* and not allowed by aspiration criterion, then go to step 3; otherwise Set $S = S_c$, and update *tabu_list* status.
If $G(S_c) \leq G(S^*)$, then set $S^* = S_c$.

Step 3: If *STOP* is met then go to step 4; otherwise go to step 2.

Step 4: Output best solution found and the objective value.

V. COMPUTATIONAL EXPERIMENT AND RESULTS

Using computational experiments, we examine the effectiveness of SA and TS. Several sets of test problems with various sizes are randomly generated. The two heuristic methods are then coded using Visual C++ programming language and are tested on an Intel Core 2 i5-2500K 3.3 GHz CPU personal computer with 4GB RAM.

1. Experimental settings

1-1. Data parameter configuration for experimental cases

Focusing on the effectiveness of the algorithms as a function of the number of retail stores $|I|$, which is referred to as "problem size," several model parameters are kept common to the different problems sizes. The detailed common settings for each test problem are presented here. For all j, we set the annual inventory holding cost $h_j = 0.5$, safety factor $z_\alpha = 1.645$ at the 95% service level, unit transportation cost from the supplier to the warehouse $p_j = 1.5$, fixed cost of placing an order

a_j = 60, and the order lead time L_j = 5 days. The annual fixed cost f_j is generated from the uniform distribution $U[500,750]$. For a retail store i, the annul mean demand μ_i is generated from a uniform distribution $U[100,500]$. The daily variance σ_i^2 is generated from uniform distribution $U[1,50]$. The transportation cost d_{ij} between i and j is generated from uniform distribution $U[1,10]$.

The SA and TS algorithms are tested by solving 10 random problems with different sizes. The test problems are divided into two categories: small and large size problems. Problem sizes of 5-11 are classified as small. The execution times $T_{\max}$ of SA and TS for the seven problem sizes are 5, 6, 7, 8, 9, 10, and 30 seconds, respectively. The solutions for the small size problems obtained using the search algorithms are then compared with the optimal solution obtained by enumeration. A total of 7 different problem sizes at 50, 75, 100, 125, 150, 175, and 200, respectively, are then classified as large problems. For each large size, 10 random problems are tested to evaluate the performance of the proposed heuristic solutions. The execution times $T_{\max}$ of SA and TS for the seven different problem sizes are 300, 450, 600, 900, 1500, 2400, and 3600 seconds, respectively.

1-2. Configurations of algorithm parameters

(1) Simulated annealing

According to the cooling function discussed in Subsection IV-1, the initial temperature T_0, the last temperature T_f, and the SA number of temperatures n must first be determined to calculate the cooling parameter β. Temperatures T_0 and T_f are computed using the relation $T = \frac{-\Delta E}{\ln P}$ when P_0 and P_f are given, respectively. In the present study, we set P_0 = 0.999 and P_f = 0.001 to obtain a high T_0 and a very low T_f respectively, in order to achieve diversification in the early stage and intensification in the final stage of SA computation. Each problem size is tested in advance to collect the average difference of the objective values between a current solution and a new neighborhood solution as well as to estimate the value of ΔE. The number of cooling down n is set at 99. Executing SA search within 10,000 iterations under trial temperature for 10 randomly generated test problems from all problem sizes, the SA parameter configurations are set as T_0 = 1000000 and T_f = 150.

(2) Tabu search

In the present study, the tabu list size is set at 7, as suggested by Glover and Laguna [28]. We use the attribute (i, j) to record a tabu move, which is an entry in the tabu list. The first in first out is also used to regulate the tabu list.

2. Performance evaluation

The average performance of various problem instances is obtained by transforming the objective value and execution time into a unified frame of reference among different problem scales and instances. We then transform the original objective value z to a normalized objective value z/z_{init}, which is the initial solution generated by a greedy method. The SA and TS procedures start searching from the same initial solution. For execution time normalization, we let t^* represent the time when the best solution among the two methods is found. The normalized time index v is equal to $\log_{10}(10000\times t/t^*)$. We use these normalized measures to observe the performance of the two heuristic search algorithms. Change in current best solutions within the whole computational period is used to observe the behavior of the two approaches.

3. Computational results

For small size problems, the SA and TS solutions are compared to the optimal solution obtained by enumeration. Table 1 shows the average solution qualities and average CPU times for the three methods.

Both SA and TS procedures can obtain an optimal solution for most small size problems in less than one second. However, the average CPU times for obtaining optimal solutions in problem sizes 10 and 11 are lengthy at 224.9885 and 6761.8636 seconds, respectively. Larger problem sizes need longer computational time to obtain optimal solutions. By enumeration, the process of obtaining optimal solutions may go beyond a reasonable amount of time when the problem size is greater than 11. Thus, the SA and TS are reasonably reliable and efficient.

For large size problems, optimal solutions cannot be obtained within a reasonable computation time. The performances of the proposed heuristic algorithms are compared with using their objective values. Table 2 presents the average solution qualities of SA and TS for large size problems. As can be seen, TS performs better than SA in terms of objective value in all large-sized problems.

Fig. 3 shows the overall average progress curves of the two approaches. TS performs better than SA throughout the whole computation period; it also had steep improvements in the early stage of computation, but slight improvements after the initial stage. Inheriting the intensification property, TS can rapidly produce good solutions in the early computation stage. However, lacking an efficient diversification mechanism to explore more attractive areas causes TS to have only slight improvements in the final computation stage. In contrast, SA provides slight improvements in the early computation stage, and has steep improvements after the initial stage. The adaptive cooling function and fitting parameter configurations possibly make SA achieve diversification and intensification in the early and final computation stages, respectively.

Table 1 Average results of three methods for small-sized problems

# of retail stores	Enumerative method		SA			TS		
	optimal value	execution time (sec)	objective value (z)	# of optimal found	best solution obtained (sec)	objective value (z)	# of optimal found	best solution obtained (sec)
5	5392.039948	0.0045	5392.039948	10	0	5392.039948	10	0
6	6346.378258	0.0045	6346.378258	10	0	6346.378258	10	0
7	7620.363309	0.0215	7620.363309	10	0.0076	7620.363309	10	0
8	8106.297434	0.372	8106.29743	10	0.0123	8106.29743	10	0.0015
9	9420.641989	8.4216	9420.641989	10	0.0779	9420.641989	10	0.003
10	10461.41954	224.9885	10461.41954	10	0.1481	10461.41954	10	0.0031
11	10885.12735	6761.8636	10885.12735	10	0.345	10885.12735	10	0.0169

Table 2 Average results of the proposed methods for large-sized problems

number of retail stores	T_{max} (sec)	SA		TS	
		Objective value (z)	normalized objective value (z/z_{init})	objective value (z)	normalized objective value (z/z_{init})
50	300	44969.50339	0.8625058	**44786.42945**	**0.858958**
75	450	64668.5157	0.8610277	**64168.88259**	**0.8542862**
100	600	85186.56447	0.84086	**84970.90371**	**0.83869**
125	900	107455.1055	0.846939	**106895.2587**	**0.84261**
150	1500	126245.8762	0.8393492	**125117.7988**	**0.8317965**
175	2400	145858.477	0.8306153	**144137.0141**	**0.8208035**
200	3600	167889.6375	0.8336464	**166157.5951**	**0.8249967**

VI. CONCLUSIONS

Under the intense competition in retail business, especially 3C retailers, maintaining adequate stocks on a variety of products in each retail point is necessary to fulfill a certain level of customer service demand. However, this may require high inventory costs, especially for high-end, low-volume products. Establishing a centralized inventory system can help achieve lower costs and higher customer service level; however, this causes a long lead time that is usually unacceptable for customers of 3C products. As a solution, this study proposes a transshipment approach by selecting several retail stores to serve as a warehouse for high-end low-volume products and take advantage of the risk pooling effect. The warehouse store is responsible for storing and supplying the high-end low-volume product via transshipment. This is the key to reduce costs and improve customer service in our model, which can help in the effective management and operation of 3C retailer chain stores.

An INLP model is also established in this study to minimize the overall system-wide costs. Two straightforward and efficient approaches based on SA and TS are implemented to solve the proposed formulation. The performances of the two algorithms are verified by solving 10 random test problems classified into small and large size problems. Both SA and TS algorithms can obtain an optimal solution in less than one second for most small-sized problems. For large-sized problems, the proposed approaches are used for up to 200 retail stores, after which the overall average progress curves are compared. Based on the results, the TS performs better than SA throughout the whole computation period.

In an actual 3C retail environment, the inventory storage space may be limited. Thus, considering capacity constraints and multiple products in a single model provides a possible extension for the current research. In addition, obtaining a good initial solution and efficient neighborhood structures are also attractive issues for future works.

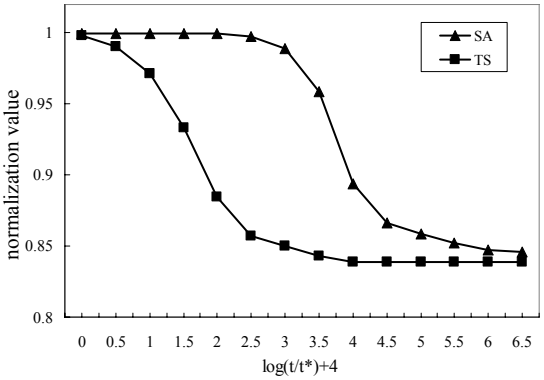

Fig. 3 Average progress of the SA and TS algorithm

REFERENCE

[1] D. Simchi-Levi, P. Kamiinsky, and E. Simchi-Levi, *Designing & managing the supply chain: concepts, strategies, and case study*, Boston: MaGraw-Hill/Irwin, 2006.

[2] M. S. Daskin, C. R. Coullard, and Z. J. M. Shen, "An inventory-location model: formulation, solution algorithm and computational results," *Ann. Oper. Res*., vol. 110, pp. 83-106, 2002.

[3] Z. J. M. Shen, C. R. Coulard, and M. S. Daskin, "A joint location-inventory model," *Transport. Sci*., vol. 37, no. 1, pp. 40-55, 2003.

[4] G. Eppen, "Effect of centralization on expected costs in multi-location newsboy problem," *Manage. Sci.*, vol. 25, no. 5, pp. 498-501, 1979.

[5] L. V. Snyder, M. S. Daskin, and C. P. Teo, "The stochastic location model with risk pooling," *Eur. J. Oper. Res.*, vol. 179. pp. 1221-1238, 2007.

[6] J. Shu and J. Sun, "Designing the distribution network for an integrated supply chain," *J. Ind. Manage. Optim.*, vol. 2, no. 3, pp. 339-349, 2006.

[7] J. Shu, C. P. Teo, and Z. J. M. Shen, "Stochastic transportation-inventory network design problem," *Oper. Res.*, vol. 53, no. 1, pp. 48-60, 2005.

[8] W. Huang, H. E. Romeijm, and J. Geunes, "The continuous-time single-sourcing problem with capacity expansion opportunities," *Nav. Res. Log.*, vol. 52, no. 3, pp. 193-211, 2005.

[9] P. A. Miranda and R. A. Garrido, "Incorporating inventory control decisions into a strategic distribution network design model with stochastic demand," *Transport. Res. E-Log.*, vol. 40, no. 3, pp. 183-207, 2004.

[10] L. Ozsen, C. R. Coullard, and M. S. Daskin, "Capacitated warehouse location model with risk pooling," *Nav. Res. Log.*, vol. 55, no. 4, pp. 295-312, 2008.

[11] Z. J. M. Shen and L. Qi, "Incorporating inventory and routing costs in strategic location models," *Eur. J. Oper. Res.*, vol. 179, pp. 372-389, 2006.

[12] F. You and I. E. Grossmann, "Mixed-integer nonlinear programming models and algorithms for large-scale supply chain design with stochastic inventory management," *Ind. and Eng. Chem. Res.*, vol. 47, no. 20, pp. 7802-7817, 2008.

[13] S. Park, T. E. Lee, and C. S. Sun, "A three-level supply chain network design model with risk pooling and lead times," *Transport. Res. E-Log.*, vol. 46, pp. 563-581, 2010.

[14] P. B. Mirchandani and R. L. Francis, *Discrete location theory*, New York: Wiley, 1990.

[15] J. S. Arora and M. W. Huang, "Methods for optimization of nonlinear problem with discrete variables: a review," *Struct. Optimization*, vol. 8, pp. 69-85, 1994.

[16] E. Grossmann, "Review of nonlinear mixed-integer and disjunctive programming techniques," *Optim. Eng.*, vol. 3, pp. 227-252, 2002.

[17] S. Kirparick, Jr. C. D. Gelett, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671-680, 1983.

[18] F. Glover, "Heuristics for integer programming using surrogate constraints," *Decision Sci.*, vol. 8, pp. 156-166, 1977.

[19] Z. Drezner and H. W. Hamacher, *Facility location: Applications and theory*, New York: Springer, 2002.

[20] P. H. Zipkin, *Foundations of inventory management*, Boston: McGraw-Hill, 2000.

[21] S. S. Syam, "A model and methodologies for the location problem with logistical components," *Comput. Oper. Res.*, vol. 29, pp. 1173-1193, 2002.

[22] V. Jayaraman and A. Ross, "A simulated annealing methodology to distribution network design and management," *Eur. J. Oper. Res.*, vol. 144, pp. 629-645, 2003.

[23] S. C. Liu and C. C. Lin, "A heuristic method for the combined location routing and inventory problem," *Int. J. of Adv. Manuf. Tech.*, vol. 26, pp. 273-381, 2005.

[24] M. Sun, "Solving the uncapacitated facility location problem using tabu search," *Comput. Oper. Res.*, vol. 33, pp. 2563-2589, 2006.

[25] S. Axäter, "Using the deterministic EOQ formula in stochastic inventory control," *Manage. Sci.*, vol. 42, no. 6, pp. 830-834, 1996.

[26] N. Metropolis, A. W. Rosenbluth, and A. H. Teller, "Equation of state calculations by first computing machines," *J. Chem. Phys.*, vol. 21, pp. 1087-1092, 1953.

[27] M. Lundy and A. Mess, "Convergence of an annealing algorithm," *Math. Program.*, vol. 34, pp. 111-124, 1986.

[28] F. Glover and M. Laguna, *Tabu search*, Norwell: Kluwer Academic Publishers, 1997.