Multi-stage Logistics Inventory for Automobile Manufacturing by Random Key-Based GA

Hisaki Inoue[1(✉)], Jung Bok Jo[2], and Mitsuo Gen[3]

[1] Miyazaki Sangyo Keiei University, Miyazaki 880-0931, Japan
inoue-lab@sankei.info
[2] Division of Computer Engineering, Dongseo University, Busan, Korea
[3] Fuzzy Logic Systems Institute, Tokyo University of Science, Tokyo, Japan

Abstract. When evaluating a logistics system, automobile companies commonly search for the minimum transportation cost, which is significantly influenced by inventory problems. These inventory problems are extensive and varied. In many actual logistics systems, there are the three-stage network models take inventory values into consideration. Safety inventories are kept in distribution centers (DCs). In this study, we adapted a model to set-up a number of plants and DCs. We then performed numerical experiments by using demand data that we created on the basis of data disclosed by an automobile company. In this study, we propose a random key-based genetic algorithm (rk-GA) with the distributed environment scheme, we compared it with random key and spanning tree-based GAs, and report the advantages of the proposed method, random key-based genetic algorithm with distributed environment scheme (des-rkGA).

Keywords: Automobile manufacturing · Multi-stage logistics · Inventory control · Random key-based genetic algorithm

1 Introduction

Many difficult inventory problems involve production control and asset management, but those that involve logistics systems are important and have garnered strong interest in recent years. Particular focus has been paid to inventory problems. Doboshas presented many papers on inventory problems. In 2001, he presented the problem of reverse logistics, adjusting the relationship of holding, production, and disposal costs [4]. In 2005, he investigated production inventory adjustment [5]. In 2007, he presented a paper on the total production cost of two companies for adjustment [6]. Minner et al. [18] have also presented papers on inventory problems. In 2003, they evaluated reverse logistics, where the inventory has several supply methods. In 2005, they presented a paper on the problem of adjusting shipping, replenishment, and lost sales opportunities for two inventories [19]. In 2008, Thangam et al. [28] presented a paper on how to determine

J. Xu et al. (eds.), *Proceedings of the Eleventh International Conference on Management Science and Engineering Management*, Lecture Notes on Multidisciplinary Industrial Engineering, DOI 10.1007/978-3-319-59280-0_63

replenishment for Poisson demand. In 2003, Mahadevan et al. [16] treated a facility as an inventory problem where the returned products are remanufactured. In 2004, Miranda et al. [20] analyzed the inventory decision problem using the Lagrangian relaxation method and subgradient methods; their ordering point method was based on the economic order quantity. In 2009, Rieksts et al. [24] analyzed the inventory problem with ordering intervals using power- of- two policies.

There are many other inventory problems, including the reduction of the total safety inventory quantity, or on-hand inventory [8], and the calculation of the inventory value at each step, or echelon inventory [12].

In this study, we propose a new model where inventories are managed at distribution centers (DCs), taking actual conditions into account. Holding costs only occur in DCs, but additional inventory costs are incurred for the product value because a product near dealers has more value. For inventory costs other than the holding cost, we can get an interest charge if the product is exchanged with cash. Another factor is the lost product value due to age depreciation. We calculated the annual supply and demand value, and created demand data that were based on the Poisson demand of time-series fluctuations. Logistics models such as these are known as the NP-hard problem [3]. Soft computing methods such as simulated annealing, neural networks, and genetic algorithms (GA) are well-suited to solve this problem [1,17,26,29].

In this study, we adopted random key-based genetic algorithm (rk-GA) with distributed environment scheme (des-rkGA) as the proposed method, which is an improved version of rk-GA; we compared the proposed method with rk-GA and spanning tree-based GA (st-GA) to confirm its suitability [9].

We propose a model that addresses many of the different inventory problems studied earlier; we demonstrate des-rkGA algorithm to solve the multi-logistics inventory problem, and present the computational results with the effectiveness by the proposed algorithm.

2 Inventory Problem in Logistics System

In this section, we detail the contents for the inventory problem treated in this study. Inventory costs result from holding costs and other factors. For the model used in this study, the product is held in a DC for general managing. All products have needed costs as their product value, and the production control cost is needed to calculate the safety inventory in a DC. The safety inventory quantity is decided by the service level but is affected by the production control and holding costs. The safety inventory cost is based on the service level and has a trade-off relation with lost costs based on lost sales opportunities.

Figure 1 shows many different kinds of inventories; they are difficult to categorize clearly, but it is important for inventory management to categorize and manage them. The inventory cost without the holding cost includes all inventory, such as the pipeline, production process, lot size and DC inventories. The inventory costs determine the product value, inventory value, inventory time, and inventory holding ratio.

Therefore, the inventory cost without holding costs and the product value, inventory value, inventory time, and inventory holding ratio have a trade-off relation. In this section, we describe the inventory holding ratio, safety inventory, production adjustment, pipeline inventory, and inventory on a production process, lot size inventory, and DC inventory as well as a summary of the inventory problems evaluated in this study. Although there are many kinds of inventories, we adopted the idea of a value chain for the inventories; we used the pipeline, lot size, production process, and DC inventories.

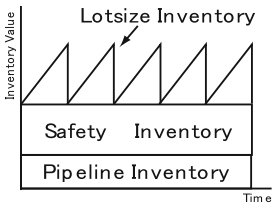

Fig. 1. Inventory category

The general outline of the logistics model used for the inventory in this study is shown in Fig. 2.

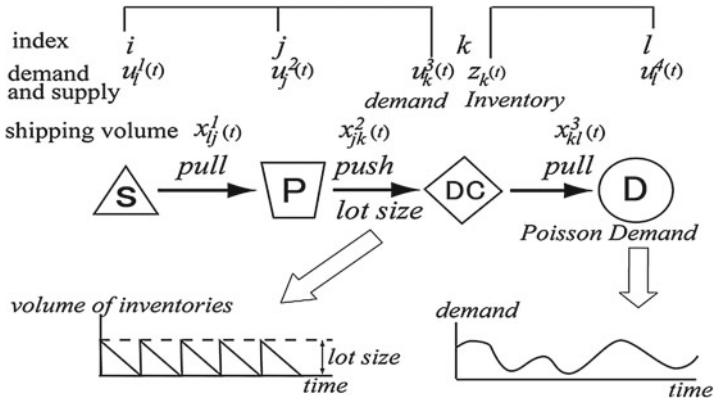

Fig. 2. General outline of logistics model with inventory

Below, we define the indices, constants, parameters, decision variables, and units used in this paper. Units of measurement are in square brackets.

Indices:
$i = 1, 2, \cdots, I$: index of suppliers;
$j = 1, 2, \cdots, J$: index of plants;
$k = 1, 2, \cdots, K$: index of DCs;
$l = 1, 2, \cdots, L$: index of customers;
$t = 1, 2, \cdots, T$: index of cycles in the logistics system. T_{cycle} intervals are described later.

Parameters:
A : number of unit parts to constitute;
c_{ij}^1 : shipping cost of unit parts or material from supplier i to plant j [yen];
c_{jk}^2 : shipping cost of unit production from plant j to DC k [yen];
c_{kl}^3 : shipping cost of unit production from DC k to customer l [yen];
g_{ij}^1 : shipping time of unit parts or material from supplier i to plant j [h];
g_{jk}^2 : shipping time of unit production from plant j to DC k [h];
g_{kl}^3 : shipping time of unit production from DC k to customer l [h];
T_{plant} : producing time of plants [h];
α : safety inventory coefficient;
σ : standard deviation of demand;
T_{cycle} : cycle time in the logistics system. It shows how much time it takes for a load to be moved once [h];
h : inventory holding ratio (2.1 value chain.);
H : holding cost of unit production in a DC [yen];
e_k : inventory cost or DC k [yen];
M_{cost} : material cost [yen];
r_j^1 : fixed cost for operating plant j [yen];
r_k^2 : fixed cost for operating DC k [yen];
U_i^1 : upper limit of supply for parts and materials in supplier i;
U_j^2 : upper limit of supply for production in plant j;
U_j^3 : upper limit of supply for production in DC k;
p_j^{const} : unit cost production at steady state [yen];
p_j^{exceed} : unit cost production at excess state, [yen];
p_j^{shortage} : unit cost production at shortage state [yen];
I_{total} : total inventory cost without holding cost;
N_{delay} : number of delays for plant production;
W_1 : number of plants that can be operated;
W_2 : number of DCs that can be operated;
R_{average} : production quantity for one period of cycle time as calculated from the annual average production quantity;
S_i^1 : supplier production ratio; ratio of supplier i to gross supplier product capability;
S_j^2 : plant production ratio; ratio of plant j to gross plant product capability;
z_k^{level} : base inventory level in DC k.

Decision Variables:
$u_i^1(t)$: supply amount of parts and materials in supplier i at cycle t;
$u_j^2(t)$: supply amont of production in plant j at cycle t;
$u_k^3(t)$: supply amont of production in DC k at cycle t;
$u_l^4(t)$: demand amount of production in customer l;
$z_k(t)$: inventory volume for DC k at cycle t [yen];
$S_k^3(t)$: DC demand ratio; demand ratio of DC k to the gross DC demand quantity;
$R_j^2(t)$: shipping quantity in plant j;
$R_k^3(t)$: receive cargo quantity in DC k;
$B_l(t)$: back order quantity in customer l;
$Dl(t)$: order quantity at this period in customer l;
$D_l^{DC}(t)$: request quantity to DC at this period in customer l;
$\Delta z_k(t)$: difference of inventory quantity and base inventory level;
$\Delta z_{\max}(t)$: maximum of difference of inventory quantity and base inventory level;
$z_k^{\text{req}}(t)$: request quantity in DC k;
p_{val}^1 : product value when delivery is completed to plants [yen];
p_{val}^2 : product value when delivery is completed to DCs [yen];
p_{val}^3 : product value when delivery is completed to customers [yen];
U_j^{exceed} : threshold for determination when exceeding production;
U_j^{shortage} : threshold for determination when reducing production;

Decision Variables:
$p_j(uj^2)$: producing cost of unit production in plant j, [yen]
$x_{ij}^1(t)$: amount supplied of unit parts or material from supplier i to plant j at cycle t;
$x_{jk}^2(t)$: amount supplied of production from plant j to DC k at cycle t;
$x_{kl}^3(t)$: amount supplied of production from DC k to customer l at cycle t;
$p_j^1(t)$: operating flag for plant j at cycle t ($= 1$ when plant j is used, $= 0$ otherwise);
$p_k^2(t)$: operating flag for DC k at cycle t ($= 1$ when DC k is used, $= 0$ otherwise).

2.1 Value Chain

When considering the product value and inventory holding ratio in addition to the holding cost, the inventory cost becomes high near the customer (dealer), as shown in Fig. 3 [23]. The inventory holding ratio is determined by the constant number by interest rate when the product is cashed, the decrease in product value, etc. In this study, we treated the inventory holding ratio as uniform because we were dealing with engineered products such as automobiles that do not experience degradation. We adopted the value chain concept for all products.

The value chain can be defined as:

I_{cost} : Inventory Cost;
P_{value} : Product Value;
I_{volume} : Inventory Amount.

$$I_{\text{cost}} = h \times P_{\text{value}} \times I_{\text{volume}} \tag{1}$$

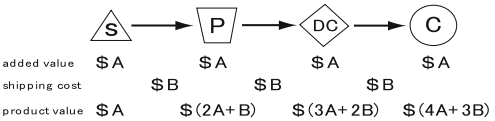

Fig. 3. Value chain

2.2 Safety Inventory

The safety inventory is the inventory required to prevent stock out. The safety inventory and service level have a mutual trade-off relation. The formula for the safety inventory is shown below.

$$I_{\text{safety}} = \alpha\sigma\sqrt{T_{\text{cycle}} + \max(g_{jk})} \quad \forall j, k, \tag{2}$$

where I_{safety} is safety inventory.

2.3 Production Adjustment

Order fluctuations mean that factories have to adjust production. Extra charges include extra pay, etc. if production is over the steady state, and extra fixed costs for the equipments if production is under steady state. The formula for production adjustment is shown below.

$$\begin{aligned} P_j(u_j^2(\mathrm{t})) = & P_{\text{const}} + \max\{P_{\text{exceed}}(u_j^2(\mathrm{t}) - U_j^{\text{exceed}}), 0\} \\ & + \max\{P_{\text{shortage}}(U_j^{\text{shortage}} - u_j^2(\mathrm{t})), 0\}. \end{aligned} \tag{3}$$

2.4 Pipeline Inventory

The pipeline inventory denotes the inventory during shipping. There is a trade-off between the pipeline inventory and shipping cost. In this study, we constructed a model based on the idea of an inventory holding ratio. We treated each product during shipping between a supplier and plant and between a DC and customer

as pipeline inventory. The formula of the pipeline inventory cost I_{pileline} is shown below.

$$I_{\text{pipeline}} = h\left\{\sum_{t=1}^{T}\left(p^1_{val}\sum_{i=1}^{I}\sum_{j=1}^{J}g^1_{ij}x^1_{ij}(t) + p^2_{val}\sum_{j=1}^{J}\sum_{k=1}^{K}g^2_{jk}x^2_{jk}(t) + p^3_{val}\sum_{k=1}^{K}\sum_{l=1}^{L}g^3_{kl}x^3_{kl}(t)\right)\right\}. \tag{4}$$

The cost is approximated as follows because we presumed that the product has the same value when received by any factory, DC, or dealer. The value of completed product is approximated by material and average shipping costs.

$$p^1_{\text{val}} = M_{\text{cost}} + \frac{\sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{j=1}^{J}c^1_{ij}x^1_{ij}(t)}{\sum_{t=1}^{T}\sum_{i=1}^{I}\sum_{j=1}^{J}x^1_{ij}(t)}. \tag{5}$$

The product value when shipped to DCs is approximated by its value when shipped to plants and the average shipping cost.

$$p^2_{\text{val}} = p^1_{\text{val}} + \frac{\sum_{t=1}^{T}\sum_{j=1}^{J}\sum_{k=1}^{K}c^2_{jk}x^2_{jk}(t)}{\sum_{t=1}^{T}\sum_{j=1}^{J}\sum_{k=1}^{K}x^2_{jk}(t)}. \tag{6}$$

The product value when shipped to dealers is approximated by its value when shipped to DCs and average shipping cost.

$$p^3_{val} = p^2_{val} + \frac{\sum_{t=1}^{T}\sum_{k=1}^{K}\sum_{l=1}^{L}c^3_{kl}x^3_{kl}(t)}{\sum_{t=1}^{T}\sum_{k=1}^{K}\sum_{l=1}^{L}x^3_{kl}(t)}. \tag{7}$$

2.5 Inventory of a Production Process

For the inventory of a production process, we constructed a model based on the idea of an inventory holding ratio. We treated an unfinished product as inventory on a production process.

The formula is shown below. The product value on a production process changes with the progress conditions of the manufacturing processes. However, in this study, the value was approximated as the material cost and half of the production cost in the plant.

$$I_{\text{production}} = hT_{\text{plant}}\sum_{j=1}^{J}u^2_j(t)(p^1_{val} + P_j(u^2_j(t))/2). \tag{8}$$

2.6 Lot-Size Inventory

The lot size inventory is the inventory of the completed products during shipping. We treated completed products as lot size inventory. All products are shipped from plants to DCs in the same interval. The lot size inventory cost was calculated as half the product of the shipping values, production values in plants, and inventory holding ratio, as shown in Fig. 4; we used this formula because all products were shipped from a plant to a DC in the same time intervals. The formula is shown below.

$$I_{\text{lotsize}} = h \sum_{t=1}^{T} \sum_{j=1}^{J} \sum_{k=1}^{K} \frac{g_{jk}^2 x_{jk}^2(t)(p_{val}^1 + P_j(u_j^2(t)))}{2}. \tag{9}$$

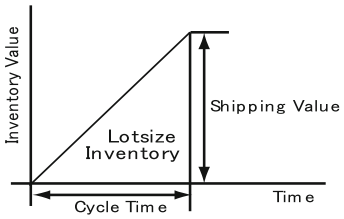

Fig. 4. Lot size inventory value

2.7 DC Inventory

We treated products in a DC as DC inventory. In this study, the DC inventory plays the central role of the inventory of the logistics system; we need extra holding costs for each car in DCs. The DC inventory cost is calculated as the product of the DC inventory amount, cycle time, and production value when shipped to DCs multiplied by the inventory holding ratio. The formula is shown below.

$$I_{DC} = hT_{\text{cycle}} p_{\text{val}}^2 \sum_{t=1}^{T} \sum_{k=1}^{K} z_k(t) + HT_{\text{cycle}} \sum_{t=1}^{T} \sum_{k=1}^{K} z_k(t). \tag{10}$$

2.8 Total Inventory Cost Without Holding Cost

As mentioned above, the formula of the total inventory cost without the holding cost used in this study consists of the pipeline, production process, lot size, and DC inventory costs; it is shown as follows.

$$I_{\text{total}=} I_{\text{pipeline}} + I_{\text{production}} + I_{\text{lotsize}} + I_{DC} \tag{11}$$

$$\begin{aligned} I_{\text{total}} = h\Bigg\{ & \sum_{t=1}^{T} (p^1_{\text{val}} \sum_{i=1}^{I} \sum_{j=1}^{J} g^1_{ij} x^1_{ij}(t) + p^2_{\text{val}} \sum_{j=1}^{J} \sum_{k=1}^{K} g^2_{jk} x^2_{jk}(t) \\ & + p^3_{val} \sum_{k=1}^{K} \sum_{l=1}^{L} g^3_{kl} x^3_{kl}(t)) + T_{\text{plant}} \sum_{j=1}^{J} u^2_j(t)(p^1_{\text{val}} + P_j(u^2_j(t))/2) \\ & + \sum_{t=1}^{T} \sum_{j=1}^{J} \sum_{k=1}^{K} \frac{g^2_{jk} x^2_{jk}(t)(p^1_{val} + P_j(u^2_j(t)))}{2} + T_{\text{cycle}} p^2_{\text{val}} \sum_{t=1}^{T} \sum_{k=1}^{K} z_k(t) \Bigg\} \\ & + HT_{\text{cycle}} \sum_{t=1}^{T} \sum_{k=1}^{K} z_k(t). \end{aligned} \tag{12}$$

2.9 Demand Data

In this study, we constructed the logistics system in terms of inventories with Poisson demand [25]. To confirm the availability, we created demand data from an automobile company's public data. Many products flow in the supply chain network (SCN) by dealer's demand, and these demands are approximated with Poisson demand time fluctuations. We created new demand data using Poisson randomization. The Poisson equation is shown below.

$$p(k) = \frac{e^{-\lambda}\lambda^k}{k!}, \tag{13}$$

here, $p(k)$ is the demand, λ is the average order values, and k is the number of order times. Figure 5 shows the outline of the Poisson randomization. In this study, we created daily demand data each experiment by applying a Poisson random number to the annual demand.

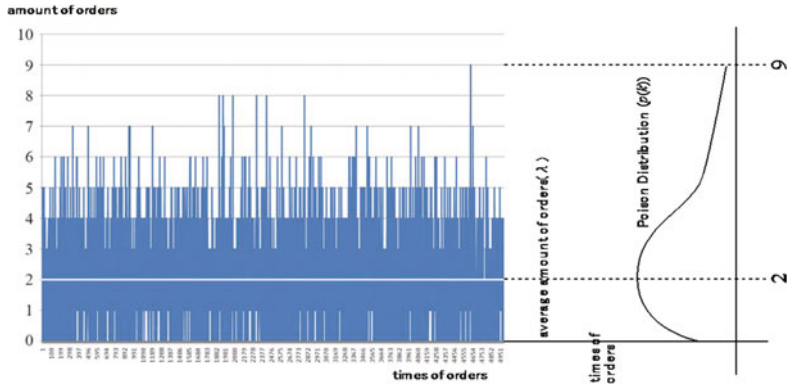

Fig. 5. Relationship between amount and times of orders

3 Mathematical Model of Multi-stage Logistics System with Inventory

3.1 Assumptions

In this study, we constructed the logistics model with the following assumptions.

A1. The transit times are known between suppliers and plants, plants and DCs, and DCs and customers.
A2. The shipping costs are known between suppliers and plants, plants and DCs, and DCs and customers.
A3. The supplies are delivered without delay to a factory in population to the planned production.
A4. In this model, suppliers provide multipurpose parts for effective optimization. The parts are examined and classified for easy assembly work. This process is carried out assuming that A package parts are used in the assembly of one car. The conditions for assembling one car require the package parts of A units. Other parts are not targeted in this model because the supply route is decided from the first time the supplier side is entrusted with delivery, few of the detailed parts have management value in the logistics system, etc.
A5. The products made at a plant are shipped to a DC by lot size.
A6. The DCs have space to accept products from plants.
A7. We consider only the inventory costs in DCs.
A8. The customer addresses are known.
A9. The existence of inventory is known in advance through an inventory check; orders for inventory that is out of stock are treated as reservations.
A10. A product has the same value when received by any factory, DC, or dealer.
A11. All products are delivered within the limits time of a T_{cycle}.

3.2 Multi-stage Logistics System

Generally, when a logistics system is seen from the functional constitutive property, it is modeled by a three-stage production network and distribution system, which is called a supply chain network. The first stage is the supplier phase, which involves parts and a supplier. The second stage is the plant phase, which consists of a production plant or outsourcing. The third stage is the DC phase and consists of a distribution center or storehouse. A sample of an actual automobile company's logistics system is shown in Fig. 6. In many cases, the actual logistics system is comprised of three stages. A three stage logistics system model is shown in Fig. 7. The logistics model used in this study had 14 suppliers, 2 factories, 4 DC, and 22 customers. The mathematical model used in this study is shown below.

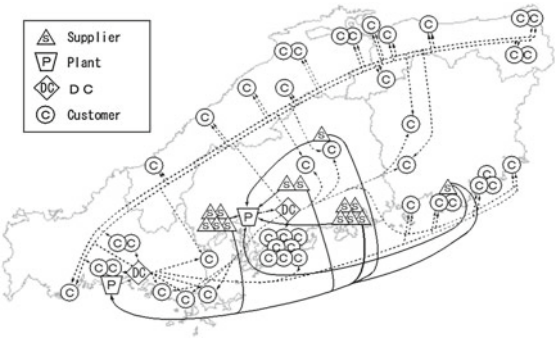

Fig. 6. Sample of actual logistics system

$$
\begin{aligned}
\min Z_1 = \sum_{t=1}^{T} \Bigg\{ & A \sum_{i=1}^{I} \sum_{j=1}^{J} c_{ij}^1 x_{ij}^1(t) + \sum_{j=1}^{J} \sum_{k=1}^{K} c_{jk}^2 x_{jk}^2(t) \\
& + \sum_{k=1}^{K} \sum_{l=1}^{L} c_{kl}^3 x_{kl}^3(t) \Bigg\} + h \Bigg\{ \sum_{t=1}^{T} (p_{\mathrm{val}}^1 \sum_{i=1}^{I} \sum_{j=1}^{J} g_{ij}^1 x_{ij}^1(t) \\
& + p_{\mathrm{val}}^2 \sum_{j=1}^{J} \sum_{k=1}^{K} g_{jk}^2 x_{jk}^2(t) + p_{\mathrm{val}}^3 \sum_{k=1}^{K} \sum_{l=1}^{L} g_{kl}^3 x_{kl}^3(t)) \\
& + T_{\mathrm{plant}} \sum_{j=1}^{J} u_j^2 (p_{\mathrm{val}}^1 + P_j(u_j^2(t))/2) \\
& + \sum_{t=1}^{T} \sum_{j=1}^{J} \sum_{k=1}^{K} \frac{g_{jk}^2 x_{jk}^2(t)(p_{\mathrm{val}}^1 + P_j(u_j^2(t)))}{2} \\
& + T_{\mathrm{cycle}} p_{\mathrm{val}}^2 \sum_{t=1}^{T} \sum_{k=1}^{K} z_k(t) \Bigg\} + H T_{\mathrm{cycle}} \sum_{t=1}^{T} \sum_{k=1}^{K} z_k(t) \\
& + \sum_{t=1}^{T} \sum_{j=1}^{J} u_j^2(t) P(u_j^2(t)) + \sum_{j=1}^{J} r_j^1 p_j^1(t) + \sum_{k=1}^{K} r_k^2 p_k^2(t)
\end{aligned} \tag{14}
$$

$$
\text{s. t.} \quad \sum_{j=1}^{J} x_{ij}^1(t) \le u_i^1(t), \quad \forall i, t \tag{15}
$$

$$
\sum_{i=1}^{I} x_{ij}^1(t) \le u_j^2(t) p_j^1(t), \quad \forall j, t \tag{16}
$$

$$\sum_{j=1}^{J} x_{jk}^2(t) \le u_k^3(t) p_k^2(t), \quad \forall k, t \tag{17}$$

$$\sum_{k=1}^{K} x_{kl}^3(t) \ge u_l^4(t), \quad \forall l, t \tag{18}$$

$$\sum_{j=1}^{J} x_{ij}^1(t) = \sum_{j=1}^{J} x_{jk}^2(t), \quad \forall j, t \tag{19}$$

$$\sum_{k=1}^{K} x_{jk}^2(t) = z_k(t-1) - z_k(t) + \sum_{k=1}^{K} x_{kl}^3(t), \quad \forall j, t \tag{20}$$

$$\sum_{j=1}^{J} p_j^1 \le W_1 \tag{21}$$

$$\sum_{k=1}^{K} p_k^2 \le W_2 \tag{22}$$

$$x_{ij}^1(t), x_{jk}^2(t), x_{kl}^3(t), z_k(t) \ge 0, \quad \forall i, j, k, l, t \tag{23}$$

$$p_j^1(t), p_k^2(t) = \{0, 1\}, \forall j, k, t \tag{24}$$

$$\begin{aligned} P_j(u_j^2) = P_j^{\text{cont}} &+ \max(P_j^{\text{exceed}}(u_j^2(t) - U_j^{\text{exceed}}), 0) \\ &+ \max(P_j^{\text{shortage}}(U_j^{\text{shortage}} - u_j^2(t)), 0) \end{aligned} \tag{25}$$

$$p_{\text{val}}^1 = M_{\text{cost}} + \frac{\sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{j=1}^{J} c_{ij}^1 x_{ij}^1(t)}{\sum_{t=1}^{T} \sum_{i=1}^{I} \sum_{j=1}^{J} x_{ij}^1(t)} \tag{26}$$

$$p_{\text{val}}^2 = p_{\text{val}}^1 + \frac{\sum_{t=1}^{T} \sum_{j=1}^{J} \sum_{k=1}^{K} c_{jk}^2 x_{jk}^2(t)}{\sum_{t=1}^{T} \sum_{j=1}^{J} \sum_{k=1}^{K} x_{jk}^2(t)} \tag{27}$$

$$p_{\text{val}}^3 = p_{\text{val}}^2 + \frac{\sum_{t=1}^{T} \sum_{k=1}^{K} \sum_{l=1}^{L} c_{kl}^3 x_{kl}^3(t)}{\sum_{t=1}^{T} \sum_{k=1}^{K} \sum_{l=1}^{L} x_{kl}^3(t)}. \tag{28}$$

4 Advanced Genetic Algorithm

4.1 Random-Key Based Genetic Algorithm

Gen and Lin [7] surveyed genetic algorithms in Wiley Encyclopedia of Computer Science and Engineering and recently many researchers applied GA to various areas in logistics systems. Inoue and Gen [10] reported multistage logistics system with inventory considering demand by hybrid GA, Neungnatcha et al. [22] reported adaptive genetic algorithm (AGA) for solving sugarcane loading stations with multi-facility services problem, Jamrus et al. [11] reported discrete particle swarm optimization (PSO) approaches and extended priority based-HGA for solving multistage production distribution under uncertainty demands

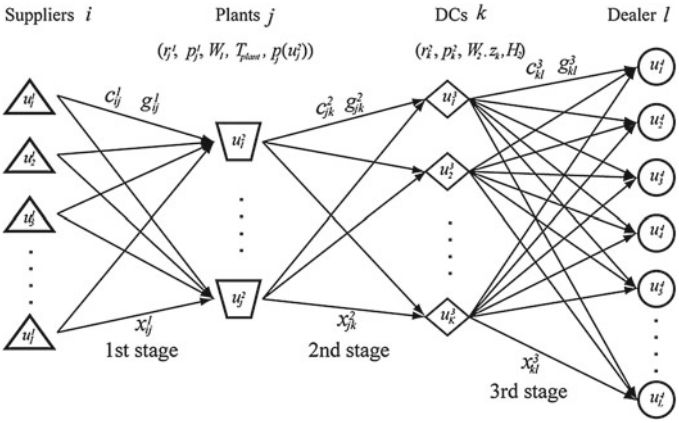

Fig. 7. 3-stages logistics model

and Lee et al. [13] reported multi-objective hybrid genetic algorithm (MoGA) to minimize the total cost and delivery tardiness in a reverse logistics.

Lin and Gen [15] proposed a random key-based genetic algorithm (rk-GA) for solving AGV (automatic guided vehicle) dispatching problem in flexible manufacturing system (FMS). Now we are going to use it for multistage logistics system with inventory. Now we define the following example of the cost matrix:

$$[e_{ij}] = \begin{bmatrix} 67 & 71 & 32 \\ 44 & 51 & 62 \\ 50 & 59 & 35 \\ 38 & 78 & 56 \\ 53 & 65 & 74 \end{bmatrix} \qquad [f_{jk}] = \begin{bmatrix} 39 & 32 & 62 & 44 \\ 45 & 56 & 53 & 59 \\ 47 & 35 & 41 & 38 \end{bmatrix}$$

$$[g_{kl}] = \begin{bmatrix} 48 & 62 & 74 & 20 & 92 & 83 & 32 \\ 51 & 26 & 89 & 17 & 35 & 59 & 86 \\ 34 & 50 & 56 & 65 & 50 & 53 & 47 \\ 71 & 44 & 23 & 38 & 11 & 62 & 29 \end{bmatrix}$$

Fig. 8. Sample of cost matrix

The algorithm created using the rk-GA technique has three logistics stages. Figure 7 shows the third stage process. Figure 8 shows a sample cost matrix. Figure 9 shows a sample rk-GA chromosome.

Gen and Cheng successfully applied rk-GA encoding to the shortest path and project scheduling problems in 2000 [2]. For transportation problems, a chromosome consists of the priorities of sources and depots, which make up a transportation tree; its length is equal to the total number of sources m and depots n, i.e., m + n. The transportation tree corresponding to a given chromosome is generated by the sequential arc between sources and depots. At each

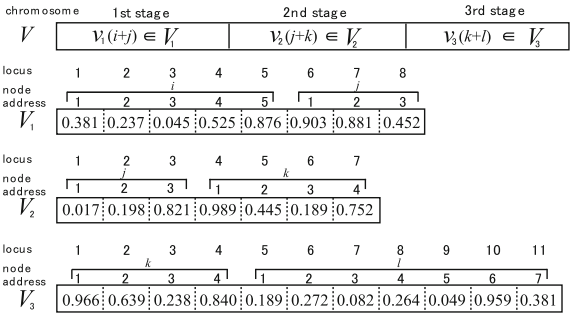

Fig. 9. Sample of rk-GA chromosome

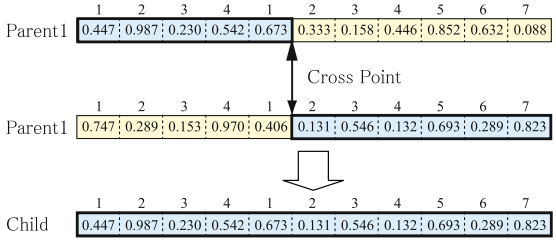

Fig. 10. Sample of one point crossover

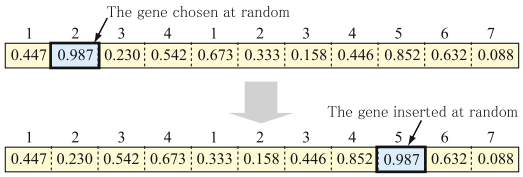

Fig. 11. Sample of insertion mutation

step, a single arc is added to a tree that selects a source (depot) with the highest priority and connects it to a depot (source) to minimize cost.

Figure 13 shows the brief decoding at each stage. Figure 14 shows the process of the m-logistics problem. In this study, we used the one-point crossover, which is the simplest method when using rk-GA. We used insertion and swap mutations. We used the roulette wheel approach, which selects the chromosome in ascending order of fitness. Examples of one-point crossover, insertion mutation, and swap mutation are shown in Figs. 10, 11 and 12, respectively.

The new generated chromosome is evaluated. It is selected in ascending order of fitness based on the number of *popSize* in the parent and newly generated chromosomes. The order of fitness then helps determine the next generation of chromosomes.

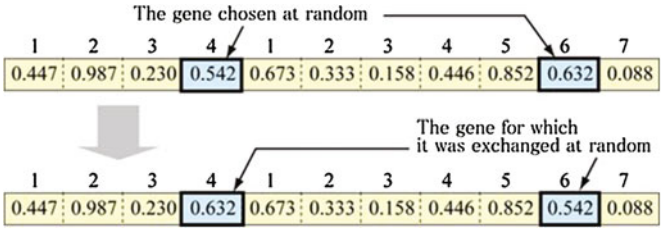

Fig. 12. Sample of swap mutation

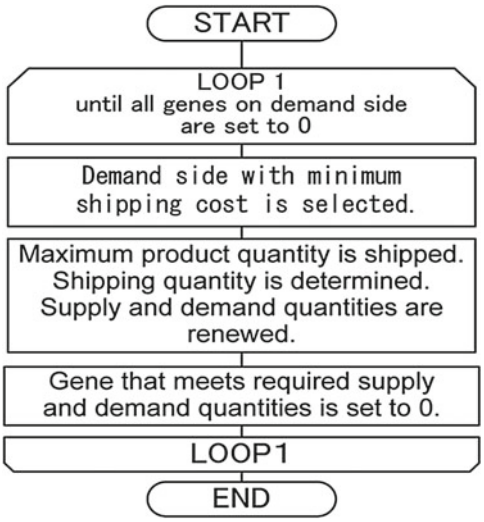

Fig. 13. Brief decoding of each stage

Inversion and displacement mutations are used in st-GA. The inversion mutation select two positions within a chromosome at random and then inverts the substring between these two positions. The displacement mutation selects a substring at random and inserts it in a random position.

4.2 Total Logistics Process

The total logistics system process can be explained as follows. Figure 15 shows the whole logistics system process in logistics cycle periods. The safety inventory is given first each cycle period. The inventory quantity is renewed last in each cycle period. The plant product shipping quantity is based on past demand, because of the production time in plants. In this study, this is called the number of delays for plant production N_{delay} where the plant production time is the number of order times. This is shown below.

```
procedure total m-logistics:
input: shipping data, GA parameter
output: minimum m-logistics cost
begin
    t ← 1;
    initialize P(t) by Random Key-based encoding routine;
    fitness eval(P) by total decoding routine;
    while (not terminating condition) do
        crossover P(t) to yield C(t) by partially matched crossover;
        mutation P(t) to yield C(t) by insertion and swap mutation;
        fitness eval(C) by total decoding routine;
        select P(t+1) from P(t) and C(t) by roulette wheel selection;
        t ← t + 1;
     end
output best m-logistics cost
end;
```

Fig. 14. m-Logistics Process

$$N_{\text{dalay}} = \frac{T_{\text{plant}}}{T_{\text{cycle}}}. \tag{29}$$

Here, we use an example for the shipping products when the customer's total demand quantity changes from 600 before delay cycle times (N_{delay}) to 1200. A pull-type demand quantity is applied to DC-customer and supplier-plant product distributes based on the demand quantity at the time. The shipping quantity is 1200. A push-type demand quantity is applied to plant-DC product distributions based on the demand quantity before the number of delay times for plant production (N_{delay}). The shipping quantity is 600.

The load of the customer's demand is shared by the total inventory quantity in DCs and total production in plants and suppliers. An example process is shown in Fig. 16. Step 1 shows a renewal of the demand quantity (u_l^4) and back order quantity (B_l). Step 2 calculates the planned order quantity ($u_j^2(t)$). Step 3 calculates the order quantity in suppliers ($u_i^1(t)$). Step 4 calculates the planned shipping quantity ($R_j^2(t)$). Step 5 calculates the quantity of cargo received in DCs ($R_k^3(t)$).

```
procedure total m-logistics in cycles period:
input: shipping data, inventory data, GA parameter
output: minimum total cost
begin
  t ← 1;
  z_k = ασ√(T_cycle + G_1);                    // initialize inventory quantity
  while (not termination condition) do
    initial setting routine;      //initial setting process
    m-logistics routine;
    t ← t + 1;
   end
output best total cost;
end;
```

Fig. 15. Total m-logistics process in a cycles period

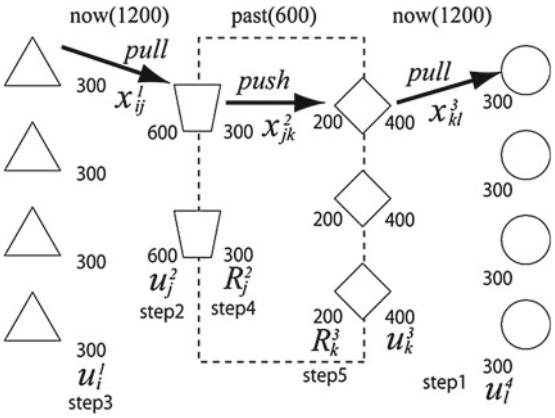

Fig. 16. Initial process

4.3 rk-GA with Distributed Environment

The immigration scheme is an independent genetic operation where each island is made up of several divided populations for the same generation. The basic concept of immigration is shown in Fig. 17. Immigration is an information exchange that is performed continuously for a group of chromosomes. The ratio to the number of immigration populations is called the immigration ratio, and the generation interval during which immigration occurs is called the immigration interval. In 2008, Lin et al. [14] solved the shortest route problem for cost and time using a priority-based GA with immigration, and reported their result.

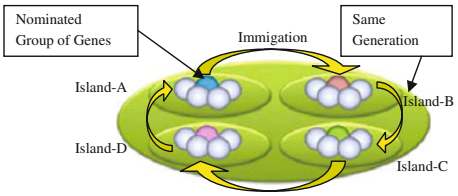

Fig. 17. Basic Concept of Immigration

In this study, we propose rk-GA with a distributed environment scheme (des-rkGA) that changes the crossover and mutation rates for each island. Miki et al. [21] proposed multi-objective genetic algorithms with a distributed environment scheme (MoGA-DES). We adopted an immigration rate of 0.1; we selected other islands randomly and changed the chromosome asynchronously. There are two methods for parallel processing [27]. The multithread method performed by one programming using time sharing. The multi-task method performs several tasks at the same time [21]. In this study, we adopted the multi-task method because

it is easy to disperse processing to several PCs. There are also two processing methods for GA. The synchronous method synchronized the time for each generation. The asynchronous method does not do so. We adopted the asynchronous method because we would need a wait time for the slowest island if we adopted the synchronous method. We created nine islands with crossover probability (PC) and lower mutation probability (PM) than the center island. Two of nine islands were chosen at random with the generation timing of the center island. The direction of immigration was decided at random, and the populations immigrated at a 0.1 immigration rate. Next, 10% of each island's worst chromosomes were destroyed, and 10% of other islands' best chromosomes were adopted as immigrations. The concrete values of PC and PM used by this study are discussed in detail in the next section. We performed the experiment using parallel processing with three PCs.

5 Numerical Experiments

We performed prior experiments to determine PC and PM for the center island and obtained the solution is shown in Table 1. We adopted $P_C = 0.4$ and $P_M = 0.6$ as the center island values for st-GA and $P_C = 0.6$ and $P_M = 0.4$ as the center island values for rk-GA because they provided the best solutions.

We adopted $P_C = 0.6$ and $P_M = 0.4$ as the center island values for des-rkGA, because it is based on rk-GA. We created 9 islands with $P_C = (0.4, 0.6, 0.8)$ and $P_M = (0.2, 0.4, 0.6)$ for the experiments.

popSize denotes the population size. maxGen is the maximum generation size used as the terminating condition for the experiments. We performed experiments repeated 20 times using maxGen = 5000 and popSize = $(20, 50, 100)$. However, we reported maxGen = 1000 as sufficient for the evolutive process because no more improvement was detected after 1000 generations. Table 2 shows the evolutive processes for the best value by each method. The best value Z_1 was at gen = $(300, 500, 1000)$.

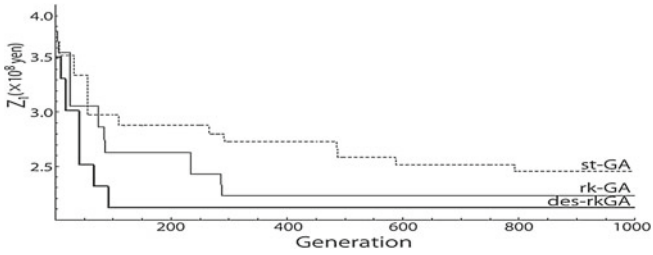

Fig. 18. Evolutive process for each method

Figure 18 shows the evolutive processes for each method when popSize was 100. The proposed Des-rkGA produced the best final result because its evolutive

Table 1. Solution at each P_C and P_M (unit: yen)

	P_M	P_C			
		0.2	0.4	0.6	0.8
st-GA	0.2	451, 461, 818	442, 315, 078	447, 088, 045	454, 269, 696
	0.4	445, 044, 804	440, 918, 667	444, 501, 564	444, 782, 298
	0.6	442, 653, 511	440, 602, 712	445, 820, 848	452, 372, 968
	0.8	445, 865, 777	443, 723, 937	451, 882, 261	454, 711, 777
rk-GA	0.2	412, 925, 351	406, 037, 962	408, 355, 139	413, 251, 110
	0.4	406, 280, 823	405, 783, 627	405, 476, 817	405, 988, 745
	0.6	408, 604, 249	406, 269, 725	405, 763, 570	413, 679, 127
	0.8	412, 120, 741	406, 777, 058	411, 116, 273	413, 073, 894

Table 2. Evolutive process for each popSize (unit: yen)

	PopSize	Gen		
		300	500	1000
st-GA	20	551, 429, 175	547, 095, 287	541, 769, 426
	50	499, 464, 718	485, 094, 520	482, 185, 269
	100	447, 629, 126	423, 064, 081	402, 412, 188
rk-GA	20	467, 164, 557	463, 804, 677	433, 653, 257
	50	452, 496, 499	434, 436, 525	394, 435, 691
	100	366, 335, 225	365, 780, 289	364, 869, 285
des-rkGA	20	418, 809, 097	396, 053, 541	389, 092, 924
	50	353, 999, 460	346, 130, 700	346, 030, 174
	100	348, 536, 438	345, 863, 047	345, 695, 925

Table 3. Evolution of each island by des-rkGA (unit: yen)

Island	Gen			
	10	20	50	70
A	590, 914, 974	505, 586, 248	442, 219, 078	407, 869, 638
B	545, 289, 874	519, 883, 420	431, 958, 888	411, 047, 134
C	596, 086, 519	530, 215, 824	413, 252, 989	399, 855, 446
D	589, 080, 440	496, 206, 544	433, 337, 814	386, 073, 858
E	569, 117, 032	535, 673, 932	419, 654, 636	412, 582, 256
F	556, 063, 137	497, 222, 107	454, 229, 946	381, 231, 373
G	582, 061, 836	535, 413, 007	412, 034, 067	417, 752, 697
H	548, 377, 594	520, 688, 854	446, 742, 473	378, 950, 050
I	572, 220, 316	493, 015, 494	415, 259, 267	382, 084, 372
des-rkGA	545, 289, 874	493, 015, 494	412, 034, 067	378, 950, 050

speed was faster than the other compared methods. As shown in Table 2, Z_1 was (402, 412, 188) when st-GA was used. It was (364, 869, 285) with rk-GA and (345, 695, 925) with des-rkGA. des-rkGA showed 16.41% and 5.55% improvements compared to st-GA and rk-GA, respectively. des-rkGA was confirmed to provide stable results because the standard deviation was only 3210 compared to 20, 338 with st-GA and 7780 with rk-GA. Table 3 shows the evolutive process of each island when popSize was 100 as in Table 2. The highlight shows the best solution for each generation, of which the best was with des-rkGA. In a prior experiment, the PC and PM for island E were the best combination. However, in the evolutive process for each island, many of the best solutions were produced at other islands. Table 4 shows the solution at each immigration rate, and the best solution by rk-GA is shown as reference. The experimental results show that if the immigration rate surpassed 50%, the results become bad. Figure 19 was created from Table 4; the best value was produced at 10% migration rate.

Table 4. Immigration Rate of des-rkGA

Immigration rate (%)	Z_1 (yen)
0	352, 543, 849
5	349, 209, 452
10	346, 035, 987
15	349, 468, 738
20	353, 121, 891
50	373, 634, 700
rk-GA	364, 769, 003

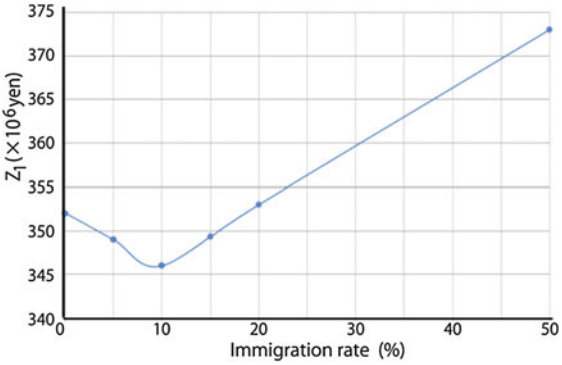

Fig. 19. Immigration Rate of des-rkGA

Table 5 summarizes the experimental results in terms of the des-rkGA improvement rate and standard deviation for Z_1 of maxGen when popSize

Table 5. Experimental results

	st-GA	rk-GA	des-rkGA
Z_1 (yen)	402, 412, 188	364, 869, 285	345, 695, 925
Improvement rate	16.41%	5.55%	-
standard deviation	20, 338	7, 780	3, 210

Table 6. Experimental results

	st-GA	rk-GA	des-rkGA	
			Each PC	Total PCs
maxGen	55.56	29.96	32.47	97.41
For 1d	35.78	19.98	20.45	61.35
Time to arrive at the maxGen value	55.56	11.07	5.74	17.22
Time to arrive at the rk-GA value	-	-	3.15	9.45

was 100. As shown in Table 2, there were differences when the number of generations was small.

In this experiment, we used three PCs of the same kind dual core AMD1212 2.0 GHz/2 MB; the memory size was 2 GB, and the development language was C#.

The computational time is shown in Table 6. We experimented with the test data for 90 days. When converted to 1 day, the computation time was 35.78, 19.98, and 20.45 s using st-GA, rk-GA, and des-rkGA, respectively. The average generation number when the solution arrived at the maxGen value was 685, 467, and 78 using st-GA, rk-GA, and des-rkGA, respectively. The generation time is shown in Table 6 as time to arrive at the maxGen value. The time was 55.56, 11.07, and 5.74 s using st-GA, rk-GA, and des-rkGA, respectively. There was no drastic improvement because the total CPU processing time was 9.45 s, but the CPU processing time could be distributed. Moreover, the average of the generation numbers when the des-rkGA value became better than maxGen of rk-GA was 165, and it took 3.15 s. We also confirmed the advantage of des-rkGA in terms of computer time. This was due to using parallel processing with 3 PCs of the same kind. We believe that the PC environment affects the solutions.

6 Conclusions

Using date of an actual automobile company in this study, we proposed random key-based genetic algorithm with distributed environment scheme (des-rkGA), for a multi-stage logistics system that calculates inventory values for many different cases. We proposed a logistics system that keeps the safety inventories only in DCs and that can cope with the location allocation problem. We performed numerical experiments with test data that were based on the disclosed data of a

certain automobile company. The proposed des-rkGA showed improvements of 16.25% and 5.43% compared to the performances of st-GA and rk-GA, respectively. The des-rkGA also showed a small dispersion of solutions compared with the other methods. For future work, we intend to investigate the effectiveness of the logistics system in actual conditions while continuing to advance research and analysis of case studies.

Acknowledgements. This work is partly supported by the Grant-in-Aid for Scientific Research (C) of Japan Society of Promotion of Science (JSPS): No. 15K00357.

References

1. Altiparmak F, Gen M, Lin L (2004) A priority-based genetic algorithm for supply chain design. Comput Ind Eng 13:22–25
2. Cauty R (2000) Genetic algorithms and engineering optimization. Wiley, Hoboken
3. Gen M, Cheng R (1997) Genetic algorithms and engineering design. Wiley, New York
4. Dobos I (2001) Production strategies under environmental constraints: continuous-time model with concave costs. Int J Prod Econ 71(1–3):323–330
5. Dobos I (2005) The effects of emission trading on production and inventories in the Arrow-Karlin model. Int J Prod Econ 93–94(1):301–308
6. Dobos I (2007) Tradable permits and production-inventory strategies of the firm. Int J Prod Econ 108(1–2):329–333
7. Gen M, Lin L (2009) Genetic algorithms. In: Wah B (ed) Wiley encyclopedia of computer science and engineering. Wiley, pp 1367–1381
8. Graves SC, Willems SP (2003) Supply chain design: safety stock placement and supply chain configuration. Handbooks Oper Res Manage Sci 11:95–132
9. Inoue H (2008) Utilization of genetic algorithm for transportation planning improvement in multi-objective logistics system. J Soc Plant Eng Jpn 19:252–259 (in Japanese)
10. Inoue H, Gen M (2012) A multistage logistics system design problem with inventory considering demand change by hybrid genetic algorithm. IEEJ Trans Electron Inf Syst 95(5):56–65
11. Jamrus T, Chien CF et al (2015) Multistage production distribution under uncertain demands with integrated discrete particle swarm optimization and extended priority-based hybrid genetic algorithm. Fuzzy Optim Decis Making 14(3):265–287
12. Lagodimos AG, Koukoumialos S (2008) Service performance of two-echelon supply chains under linear rationing. Int J Prod Econ 112(2):869–884
13. Lee JE, Chung KY et al (2015) A multi-objective hybrid genetic algorithm to minimize the total cost and delivery tardiness in a reverse logistics. Multimedia Tools Appl 74(20):9067–9085
14. Lin L, Gen M (2008) An effective evolutionary approach for bicriteria shortest path routing problems. IEEJ Trans Electron Inf Syst 128(3):416–423
15. Lin L, Gen M (2009) A random key-based genetic algorithm for agv dispatching in FMS. Int J Manufact Technol Manage 16(1):58–75
16. Mahadevan B, Pyke DF, Fleischmann M (2003) Periodic review, push inventory policies for remanufacturing. Soc Sci Electron Publ 151(3):536–551
17. Min H, Zhou G (2002) Supply chain modeling: past, present and future. Comput Ind Eng 43:231–249

18. Minner S (2003) Multiple-supplier inventory models in supply chain management: a review. Int J Prod Econ 81–82(02):265–279
19. Minner S, Silver EA (2005) Evaluation of two simple extreme transshipment strategies. Int J Prod Econ 93–94(1):1–11
20. Miranda PA, Garrido RA (2004) Incorporating inventory control decisions into a strategic distribution network design model with stochastic demand. Transp Res Part E Logistics Transp Rev 40(3):183–207
21. Miyata S, Kudo K et al (2003) Mass simulations based design approach and its environment: multi-objective optimization of diesel engine with distributed genetic algorithm using iSIGHT MOGADES and HIDECS. Parallel computational fluid dynamics, pp 499–506
22. Neungmatcha W, Sethanan K et al (2013) Adaptive genetic algorithm for solving sugarcane loading stations with multi-facility services problem. Comput Electron Agric 98(7):85–99
23. Porter ME (1985) Competitive advantage: creating and sustaining superior performance, pp 33–69
24. Rieksts BQ, Ventura JA, Herer YT (2009) Power-of-two policies for single-warehouse multi-retailer inventory systems with order frequency discounts. Comput Oper Res 36(7):2286–2294
25. Saul AT et al (1993) Numerical recipes in C. Cambridge University Press, Camridge, pp 293–295
26. Syarif A, Gen M (2003) Double spanning tree-based genetic algorithm for two stage transportation problem. Knowl Based Intell Eng Syst 7(4):214–221
27. Taniar D, Leung C et al (2008) High performance parallel database processing and grid databases. Wiley, New York
28. Thangam A, Uthayakumar R (2008) A two-level supply chain with partial backordering and approximated poisson demand. Eur J Oper Res 187(1):228–242
29. Zhou G, Min H, Gen M (2002) The balanced allocation of customers to multiple distribution centers in the supply chain network: a genetic algorithm approach. Comput Ind Eng 43:251–261