Hindawi
Complexity
Volume 2018, Article ID 2726070, 17 pages
https://doi.org/10.1155/2018/2726070

Research Article

A Nonlinear Integer Programming Model for Integrated Location, Inventory, and Routing Decisions in a Closed-Loop Supply Chain

Hao Guo,[1] **Congdong Li**,[1] **Ying Zhang**,[2] **Chunnan Zhang**,[3] **and Yu Wang**[4]

[1]*School of Management, Jinan University, Guangzhou, Guangdong 510632, China*
[2]*Department of Industrial and Systems Engineering, State University of New York at Buffalo, Buffalo, NY 14260, USA*
[3]*Department of Accounting, Temple University, Philadelphia, PA 19122, USA*
[4]*International Business School, Jinan University, Zhuhai, Guangdong 519070, China*

Correspondence should be addressed to Congdong Li; licd@jnu.edu.cn and Ying Zhang; yz29@buffalo.edu

Received 21 October 2017; Revised 29 May 2018; Accepted 10 June 2018; Published 22 July 2018

Academic Editor: Marek Reformat

Facility location, inventory management, and vehicle routing are three important decisions in supply chain management, and location-inventory-routing problems consider them jointly to improve the performance and efficiency of today's supply chain networks. In this paper, we study a location-inventory-routing problem to minimize the total cost in a closed-loop supply chain that has forward and reverse logistics flows. First, we formulate this problem as a nonlinear integer programming model to optimize facility location, inventory control, and vehicle routing decisions simultaneously in such a system. Second, we develop a novel heuristic approach that incorporates simulated annealing into adaptive genetic algorithm to solve the model efficiently. Last, numerical analysis is presented to validate our solution approach, and it also provides meaningful managerial insight into how to improve the closed-loop supply chain under study.

1. Introduction

Supply chain management is critical for many business organizations to gain advantage in a competitive environment, and its impact has increased steadily in the past decades [1]. Although most practices in supply chain management were focused on forward logistics in early days, reverse logistics flows that are caused by consumer returns have gained a lot of attention recently and hence are considered by many firms to improve their business. According to a National Retail Federation report, the total merchandise returns accounted for $260.5 billion and $28.3 billion for the loss of the U.S. retailers and Canadian retailers in 2015, respectively [2]. Consumer returns also have a significant impact on e-commerce, and it is shown that at least 30% of all the products ordered online are returned as compared to 8.89% in traditional offline stores [3]. Particularly, for fashion products such as fashion apparel, the return rate can be as high as 75% [4]. Therefore, consumer returns represent a growing financial and operational concern for many firms in different industries, and they also have a significant impact on their supply chains.

Closed-loop supply chains (CLSCs) [5, 6] are an emerging topic in supply chain management because of the growing concern about consumer returns and environmental sustainability. Unlike traditional supply chains that only consider forward logistics flows directed from manufacturers to consumers, CLSCs also consist of reverse flows of new or used products that are directed from consumers to manufacturers. In practice, business managers need to make many strategic, tactical, and operational decisions such as facility locations, inventory control, and vehicle routing decisions to improve the efficiency and sustainability of their supply chains. In this paper, we study a location-inventory-routing problem (LIRP) that integrates those three decisions in a multiechelon closed-loop supply chain network for a manufacturer. This network comprises a manufacturing factory, multiple hybrid distribution-collection centers

(HDCCs), and several retailers, where HDCCs will operate as warehouses and collection centers in the forward and reverse flows, respectively. From a practical perspective, the research questions that motivate this study are summarized as follows:

(1) For a manufacturer, how to decide HDCC locations in a supply chain network when forward and reverse logistics flows are both considered, and how to use those HDCCs to fulfill the demands and collect returns from retailers?

(2) What is the optimal stock replenishment policy for those HDCCs?

(3) How to optimize vehicle routes in the forward and reverse flows when retailers are served by those HDCCs?

The rest of this paper is organized as follows: Section 2 reviews the related literature. Section 3 describes the research problem under study and formulates it as a nonlinear integer programming model. Section 4 proposes an adaptive hybrid simulated annealing genetic algorithm (AHSAGA) to solve the model efficiently. Section 4.1 presents the numerical study and computational results. Section 5 concludes this paper and provides directions for future research.

2. Literature Review

The location-inventory-routing problem (LIRP) comprises three subproblems: facility location, inventory control, and vehicle routing. Since they are highly correlated in the real-world business, many research efforts have been conducted to study those problems jointly. The location inventory problem (LIP) is the integrated form of the first two problems, and they are first proposed by Daskin et al. [7] and Shen et al. [8]. LIPs have been extended with many business scenarios such as lateral transshipment [9], perishable products [10], correlated demands [11], disruption risk [12], and inventory control strategies [13], and most of those works are reviewed by Farahani et al. [14]. Recently, LIPs are also studied by incorporating CLSCs. For example, Diabat et al. [15] study a LIP by considering spare parts in a closed-loop system, Guo et al. [16] study location-inventory decisions for closed-loop supply chain management with secondary market consideration, and Li et al. [17] present an important and meaningful work by studying LIP and CLSC with third-party logistics (3PL) because it is a fundamental logistics strategy that has been adopted by many firms in practice. The location routing problems (LRPs) integrate facility location and vehicle routing problems but ignore inventory management decisions. Min et al. [18] and Nagy and Salhi [19] review the research works related to LRPs in early days, and Schneider and Drexl [20] examine the most recent works that are published in the literature since the survey by Nagy and Salhi [19].

LIRPs incorporate all three decisions above, and hence they are a more comprehensive form. In early days, Shen and Qi [21] develop a location-allocation model which approximates routing costs according to the locations of opened depots, and then Javid and Azad [22] study such a problem without any approximation. Moreover, LIRPs are extensively studied under many practical settings such as perishable products [23, 24], deterministic or stochastic demand [25–27], and disruption risks [28]. Closed-loop supply chains (CLSCs) have attracted considerable attention from researchers and practitioners because of the significant impact of consumer returns, and it is emergent to study LIRPs under a CLSC setting. For example, Li et al. [29] study a LIRP by considering returns in an electronic supply chain environment, and Deng et al. [30] develop and solve a model when returned products can be either defective or nondefective. From the perspective of sustainability, Zhalechian et al. [31] design a closed-loop system with location routing inventory decisions under mixed uncertainty.

In this paper, a nonlinear integer program model is formulated to study a LIRP in a closed-loop logistics system by considering many real-world business scenarios such as vehicle capacity and the disposal of different types of returned products. To solve this model efficiently, we develop a novel solution approach that extends the power of the adaptive genetic algorithm by incorporating simulated annealing, and numerical study shows that it is more powerful and efficient than other similar heuristics in the literature.

3. The Model

3.1. Problem Description. In this paper, we study a closed-loop supply chain network that comprises a manufacturing factory, multiple hybrid distribution-collection centers (HDCCs), and several retailers. This network can be represented by a directed graph in which vertices are the factory, HDCCs, and retailers, and the edges can be directed from the factory to retailers via HDCCs, or vice versa. More specifically, in the forward flow, new products are first shipped from the factory to HDCCs and then from HDCCs to retailers by vehicles on certain routes. In the reverse flow, returned products are sent from retailers to HDCCs for inspection first. A returned product will be disposed immediately at a HDCC if it cannot be refurbished. Otherwise, it will be sent from HDCCs to the factory for repair. In this system, HDCCs operate as warehouses and return collection centers on working days in the forward and reverse flows, respectively. Vehicles are used to deliver new products from HDCCs to retailers as well as to collect returned products from retailers to HDCCs, and a vehicle must return to the same HDCC after it visits all retailers on a route. Figure 1 illustrates the closed-loop supply chain network under study.

For simplicity, we consider a single type of products and vehicles and assume that a retailer will be assigned to a same HDCC in the forward and reverse flows. Given the locations of the factory and retailers, HDCCs will be built at selected locations, and a HDCC will order new products from the factory and serve at least one retailer in the forward and reverse flows. To minimize the total cost in this system, the following decisions will be optimized:

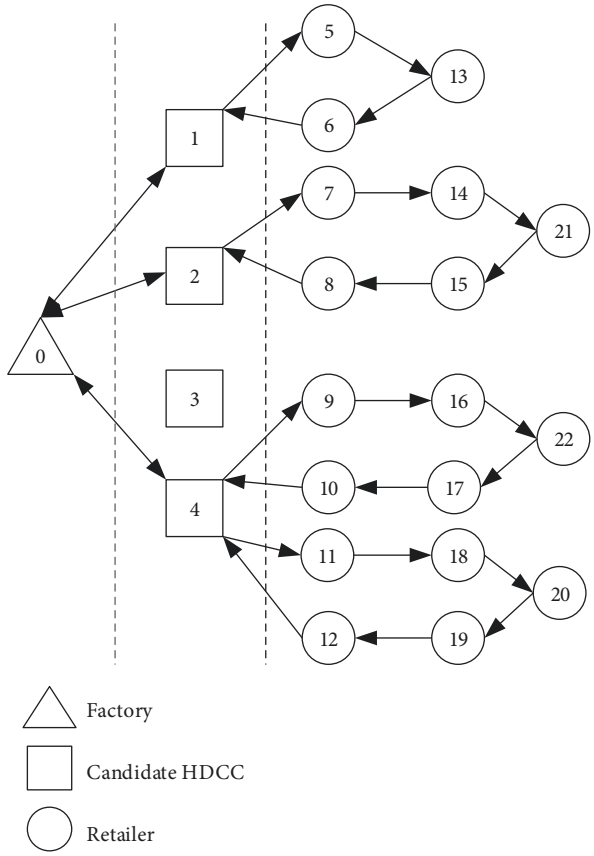


Figure 1: A closed-loop supply chain network.

(1) HDCC location and retailer assignment: selecting locations to build HDCCs and assigning retailers to those HDCCs

(2) Inventory replenishment: deciding the optimal order frequency and quantity for each HDCC

(3) Vehicle routing: designing circular vehicle routes starting from and ended by each HDCC

3.2. Objective Function. In the closed-loop supply chain under study, the total cost is composed by the following: (1) location cost which is the fixed cost of building and operating HDCCs; (2) working inventory cost including order, holding, and shipping costs; (3) routing cost between HDCCs and retailers; (4) return cost. The individual costs per year are calculated as follows:

(1) Location cost: $C_{\mathrm{LOC}} = \sum_{r\in R} a_r W_r$

(2) Working inventory cost

The working inventory cost comprises three individual terms. The first term is the order cost that is incurred when placing orders to the factory at HDCCs, the second term is the holding cost of new products in inventory, and the third term is the shipping cost of new products from the factory to HDCCs. Similar to [22], we adopt a (Q, r) inventory model with type I service to manage inventories at HDCCs, and the holding cost is adapted from the standard form in the economic order quantity (EOQ) model. Obviously, the order frequency and quantity at a HDCC is determined by the expected demands of the retailers that are served by the HDCC. Therefore, the individual terms of the working inventory cost can be written as follows:

(i) Order cost: $\sum_{r\in R} f_r N_r$

(ii) Holding cost of new products: $\sum_{r\in R}\sum_{i\in S}(h\lambda d_i X_{ir}/2N_r)$

(iii) Shipping cost from the factory to HDCCs: $\sum_{r\in R} e_r N_r + \sum_{r\in R}\sum_{i\in S} b_r \lambda d_i X_{ir}$

Consequently, the total working inventory cost per year is given as follows:

$$C_{\mathrm{INV}} = \sum_{r\in R}(f_r + e_r)N_r + \lambda \sum_{r\in R}\sum_{i\in S}\left(\frac{hd_i}{2N_r} + b_r d_i\right)X_{ir}. \tag{1}$$

(3) Vehicle routing cost

(i) Forward logistics: $\sum_{r\in R}\sum_{v\in V}\sum_{i\in S} u\lambda d_i s_{ri} Y_{irv}$

(ii) Reverse logistics: $\sum_{r\in R}\sum_{v\in V}\sum_{i\in S} u\lambda q_i t_{ir} Y_{irv}$

Therefore, the total annual routing cost is given as follows:

$$C_{\mathrm{VRT}} = \lambda u \sum_{r\in R}\sum_{v\in V}\sum_{i\in S}(s_{ri}d_i + t_{ir}q_i)Y_{irv}. \tag{2}$$

(4) Return cost

(i) Inspection cost: $\sum_{r\in R}\sum_{i\in S} p_r \lambda q_i X_{ir}$

(ii) Disposal cost at HDCCs: $\sum_{r\in R}\sum_{i\in S} g_r \theta \lambda q_i X_{ir}$

(iii) Cost of refurbish returned products at the factory: $\sum_{i\in S} m(1-\theta)\lambda q_i$

(iv) Shipment cost from HDCCs to the factory: $\sum_{r\in R}\sum_{i\in S} b_r \lambda(1-\theta) q_i X_{ir}$

(v) Holding cost of returned products: $\sum_{r\in R}\sum_{i\in S} k \lambda q_i X_{ir}$

For simplicity, we assume that the holding cost of a returned product is independent of how long it stays in inventory. Therefore, the total return cost per year is given as follows:

$$C_{\mathrm{RET}} = \sum_{i\in S} m(1-\theta)\lambda q_i + \lambda \sum_{r\in R}\sum_{i\in S}(p_r + g_r\theta + b_r(1-\theta))q_i X_{ir}. \tag{3}$$

According to the individual costs above, the total annual cost in the CLSC is calculated as follows:

$$\begin{aligned} C = {} & \sum_{r\in R} a_r W_r + \lambda u \sum_{r\in R}\sum_{v\in V}\sum_{i\in S}(s_{ri}d_i + t_{ir}q_i)Y_{irv} + \sum_{r\in R}(f_r + e_r)N_r \\ & + \lambda \sum_{r\in R}\sum_{i\in S}\left(\frac{hd_i}{2N_r} + b_r d_i X_{ir}\right) + \sum_{i\in S} m(1-\theta)\lambda q_i \\ & + \lambda \sum_{r\in R}\sum_{i\in S}(p_r + g_r\theta + b_r(1-\theta) + k)q_i X_{ir}. \end{aligned} \tag{4}$$

Therefore, the location-inventory-routing problem under study can be formulated as follows:

$$\min \quad C,$$

$$\text{subject to} \quad \sum_{r\in R} W_r \geq 1, \tag{5}$$

$$\sum_{r\in R} U_{rv} \leq 1 \quad \forall v \in V, \tag{6}$$

$$\sum_{r\in R} X_{ir} = 1 \quad \forall i \in S, \tag{7}$$

$$\sum_{r\in R}\sum_{v\in V} Y_{irv} = 1 \quad \forall i \in S, \tag{8}$$

$$\sum_{r\in R}\sum_{i\in S} d_i Y_{irv} \leq c \quad \forall v \in V, \tag{9}$$

$$U_{rv} \leq W_r \quad \forall r \in R, \forall v \in V, \tag{10}$$

$$\sum_{v\in V} U_{rv} \geq W_r \quad \forall r \in R, \tag{11}$$

$$X_{ir} \leq W_r \quad \forall i \in S, \forall r \in R, \tag{12}$$

$$\sum_{i\in S} X_{ir} \geq W_r \quad \forall r \in R, \tag{13}$$

$$\sum_{v\in V} U_{rv} \leq \sum_{i\in S} X_{ir} \quad \forall r \in R, \tag{14}$$

$$Y_{irv} \leq U_{rv} \quad \forall i \in S, \forall r \in R, \forall v \in V, \tag{15}$$

$$\sum_{i\in S} Y_{irv} \geq U_{rv} \quad \forall r \in R, \forall v \in V, \tag{16}$$

$$Y_{irv} \leq X_{ir} \quad \forall i \in S, \forall r \in R, \forall v \in V, \tag{17}$$

$$\sum_{v\in V} Y_{irv} = X_{ir} \quad \forall i \in S, \forall r \in R, \tag{18}$$

$$\sum_{j\in L} Z_{ijrv} = Y_{irv} \quad \forall i \in S, \forall r \in R, \forall v \in V, \tag{19}$$

$$\sum_{i\in L} Z_{ijrv} = Y_{jrv} \quad \forall j \in S, \forall r \in R, \forall v \in V, \tag{20}$$

$$\sum_{j\in S} Z_{rjrv} \leq \sum_{i\in S} Y_{irv} \quad \forall r \in R, \forall v \in V, \tag{21}$$

$$\sum_{j\in S} Z_{jrrv} \leq \sum_{i\in S} Y_{irv} \quad \forall r \in R, \forall v \in V, \tag{22}$$

$$M\sum_{j\in S} Z_{rjrv} \geq \sum_{i\in S} Y_{irv} \quad \forall r \in R, \forall v \in V, M \text{ is a big number}, \tag{23}$$

$$M\sum_{j\in S} Z_{jrrv} \geq \sum_{i\in S} Y_{irv}, \quad \forall r \in R, \forall v \in V, M \text{ is a big number}, \tag{24}$$

$$\sum_{j\in S} Z_{rjrv} \leq 1 \quad \forall r \in R, \forall v \in V, \tag{25}$$

$$\sum_{j\in S} Z_{jrrv} \leq 1 \quad \forall r \in R, \forall v \in V, \tag{26}$$

$$\sum_{i\in L} Z_{ikrv} - \sum_{j\in L} Z_{kjrv} = 0 \quad \forall k \in L, \forall r \in R, \forall v \in V, \tag{27}$$

$$Z_{ijrv} = 0 \quad \forall i \in S, \forall j, r \in R, j \neq r, \forall v \in V, \tag{28}$$

$$Z_{ijrv} + Z_{jirv} = 1 \quad \forall i, j \in S, \forall r \in R, \forall v \in V, \tag{29}$$

$$W_r \in \{0,1\} \quad \forall r \in R, \tag{30}$$

$$U_{ir} \in \{0,1\} \quad \forall r \in R, \forall v \in V, \tag{31}$$

$$X_{ir} \in \{0,1\} \quad \forall i \in S, \forall r \in R, \tag{32}$$

$$Y_{irv} \in \{0,1\} \quad \forall i \in S, \forall r \in R, \forall v \in V, \tag{33}$$

$$Z_{ijrv} \in \{0,1\} \quad \forall i \in W, \forall j \in W, \forall r \in R, \forall v \in V. \tag{34}$$

The constraints of this model are explained as follows. Constraint (5) means that at least one HDCC will be built. Constraint (6) means that a vehicle can be assigned to at most one HDCC. Constraint (7) means that a retailer will be served by exactly one HDCC. Constraint (8) means that a retailer will be placed in exactly one route. Constraint (9) is the vehicle capacity constraint which means that the total demand in a route cannot exceed the capacity of a vehicle. Constraint (10) means that vehicles can be assigned to a HDCC only if the HDCC has been built. Constraint (11) means that at least one vehicle will be assigned to a HDCC. Constraint (12) means that retailers can be served by a HDCC only if it has been built. Constraint (13) means that at least one retailer will be served by a HDCC. Constraint (14) means that for each HDCC, the number of vehicles or routes is less than the number of retailers. Constraints (15) and (16) enforce the relationship between U_{rv} and Y_{irv}, and they mean that a retailer can be placed in a route only if the route exists as well as that at least one retailer will be included in a route. Constraints (17) and (18) enforce the relationship between X_{ir} and Y_{irv}, and they mean that a retailer will be placed in exactly one route which belongs to a HDCC if and only if it is served by the same HDCC. Constraints (19) and (20) enforce the relationship between Y_{irv} and Z_{ijrv} for retailers, and they mean that a retailer cannot have neighbor locations in a route if it is not in the route as well as that no closed subloop will present in a route. Constraints (21), (22), (23), and (24) enforce the relationship between Y_{irv} and Z_{ijrv} for HDCCs, and they mean that exactly one HDCC

will be placed in each route. Constraints (25) and (26) mean that in a route, a HDCC can be directed to/from at most one retailer. Constraint (27) is the flow conservation constraint, and it means that a vehicle must leave a retailer after it arrives at this retailer and hence the route is circular. Constraint (28) means that a route cannot be directed from a retailer to a HDCC if the retailer is not served by this HDCC. Constraint (29) guarantees that a route will be in one direction but not in two directions. Constraints (30), (31), (32), (33), and (34) specify that W_r, U_{rv}, X_{ir}, Y_{irv}, and Z_{ijrv} are binary variables.

It is obvious that the objective function is convex with respect to N_r. To calculate the optimal number of orders placed at HDCC r annually, let $\partial C/\partial N_r = 0$, then we have

$$N_r^* = \sqrt{\frac{\sum_{v\in V}\sum_{i\in S}\lambda h d_i Y_{irv}}{2(e_r + f_r)}}. \tag{35}$$

By substituting N_r^* into (4), the objective function can be rewritten as

$$\begin{aligned} C' = {} & \sum_{r\in R} a_r W_r + \lambda u \sum_{r\in R}\sum_{v\in V}\sum_{i\in S}(s_{ri}d_i + t_{ir}q_i)Y_{irv} \\ & + \sqrt{2\lambda h \sum_{r\in R}\sum_{i\in S}(f_r + e_r)d_i X_{ir}} + \lambda \sum_{r\in R}\sum_{i\in S} b_r d_i X_{ir} \\ & + \sum_{i\in S} m(1-\theta)\lambda q_i + \lambda \sum_{r\in R}\sum_{i\in S}(p_r + g_r\theta + b_r(1-\theta) + k)q_i X_{ir}. \end{aligned} \tag{36}$$

4. Solution Approach

Facility location and vehicle routing problems are NP-hard in general [22], and LIRPs can be more complex to solve because of the integration of those problems. In this paper, we propose a two-phase heuristic method that incorporates simulated annealing (SA) into adaptive genetic algorithm (AGA). More specifically, facility location and vehicle routing decisions will be encoded as chromosomes in AGA, and then the two decisions will be optimized by an evolution process. Thereafter, the optimal inventory replenishment decision will be determined accordingly.

GA is a popular search technique to solve optimization problems based on the principles of natural selection and genetics [32]. In practice, a GA process may converge prematurely or do not converge at all, both of which will lead to bad solutions, and hence adaptive coefficients are usually used to compensate those shortcomings. SA is a probabilistic method which was first proposed to find the global minimum of a cost function that may possess several local minima [33], and it has been widely used to solve many research problems. In this paper, we propose a novel heuristic algorithm, that is, adaptive hybrid simulated annealing genetic algorithm (AHSAGA), to solve the nonlinear integer programming model presented in Section 3. AHSAGA is an improved form of traditional AGAs by adopting the great local search capacity of SA, and our numerical experiments show that it is an effective approach in terms of both solution accuracy and time efficiency.

4.1. Basis

4.1.1. Encoding and Decoding. When GA is applied to solve an optimization problem, chromosomes are usually used to represent the candidate solutions to this problem, and they will evolve to better solutions iteratively. In this study, the solutions to the location and routing problems will be first encoded as chromosomes and then solved by AHSAGA. Once the location and routing problems are solved, inventory decisions can be easily optimized by solving (35).

The length of a chromosome is $R + S$, where R and S are the number of candidate HDCC locations and retailers, respectively. Let N be the population size, then an initial population can be created by randomly choosing N chromosomes that satisfy (5), (6), (7), (8), (9), (10), (11), (12), (13), (14), (15), (16), (17), (18), (19), (20), (21), (22), (23), (24), (25), (26), (27), (28), (29), (30), (31), (32), (33), and (34). In a chromosome, a HDCC and its subsequent retailers comprise a circular route. Therefore, each chromosome will start with a candidate HDCC location. If a chromosome starts with a retailer, then the first allele will be replaced by a candidate HDCC location because the retailers before the first candidate HDCC location in the chromosome will not be assigned to any HDCC. Moreover, if there are consecutive candidate HDCC locations in a chromosome, then a HDCC will be built at the location represented by the last allele of this string.

4.1.2. Fitness Function and Selection Method. Once a population is created, chromosomes or candidate solutions will be evaluated by their fitness to decide whether they will be kept in its offspring population. In this study, the fitness of an individual is measured as follows:

$$f_k = \frac{1}{C'}, \quad 1 \le k \le M, \tag{37}$$

where C' is the objective function given by (36).

In AHSAGA, roulette-wheel selection is adopted to select and copy solutions with higher fitness values into new populations. Let N be the population size, then chromosome i will be reproduced in the next generation if it satisfies the equation below:

$$\frac{\sum_{k=1}^{i-1} f_k}{\sum_{k=1}^{N} f_k} < \xi_i \le \frac{\sum_{k=1}^{i} f_k}{\sum_{k=1}^{N} f_k}, \tag{38}$$

where f_k is the fitness value of chromosome k, $\xi_i \in [0, 1]$ is a random number that follows the uniform distribution.

4.1.3. Crossover Operator. In general, a GA process will start with an initial population that is generated randomly, and the fitness of solutions will be improved iteratively by applying selection, crossover, mutation, and replacement operators. In AHSAGA, crossover operator will be applied in an iteration by the following three steps to recombine individuals for a better offspring:

(1) Choose two parents from a population randomly and decide two crossover points arbitrarily.

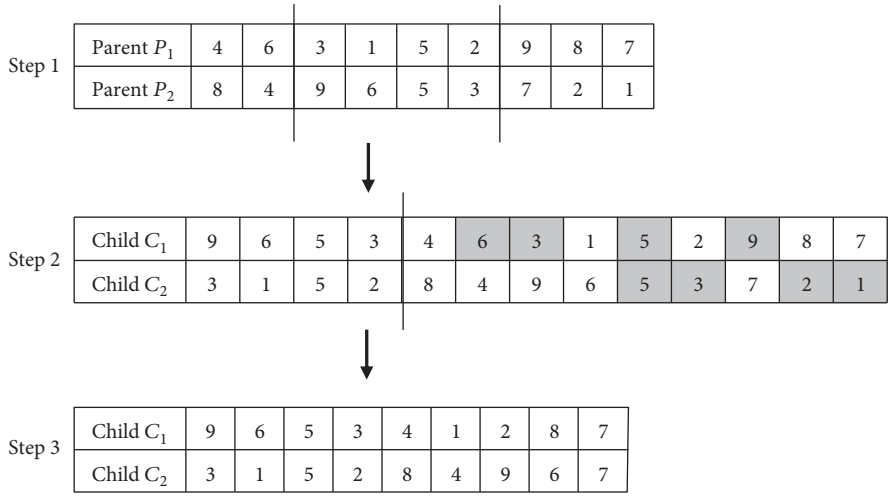


Figure 2: An example of crossover operation.

(2) Generate two intermediate chromosomes by moving all the alleles positioned between the crossover lines in a parent to the beginning of the other.

(3) In each intermediate chromosome, remove the same alleles which appear in the string moved from the other parent.

An example of this procedure is illustrated in Figure 2.

Usually, fitness values of chromosomes will be significantly different at the beginning of a GA process, and hence crossover is greatly beneficial to speed up the evolution. In AHSAGA, the probability of crossover is given by (39), which is similar to [34] in spirit.

$$p_c = p_{c0} + \alpha_c \frac{\left(f_{avg}\right)^{n_c}}{(f_{max} - f_{min})^{n_c} + \left(f_{avg}\right)^{n_c}}, \quad (39)$$

where p_{c0} is the initial probability of crossover, α_c and n_c are the two adaptive coefficients, f_{max}, f_{avg}, and f_{min} are the maximal, average, and minimal fitness values in a population, respectively.

4.1.4. Mutation Operator. Crossover operators cannot work effectively if individuals have similar fitness values in a population. For example, in some cases, new chromosomes cannot be generated by crossover if two parents have the same allele at a given gene. To solve this problem, mutation is designed to add diversity to the population and make it possible to explore the entire search space [32]. AHSAGA uses an inverse function as the mutation operator to select two points in a parent chromosome randomly and invert the order of the alleles between the two points. For example, if (7 8 5 |3 6 9 1| 4 10 2) is a parent, then the first and second split points are located after the third and seventh genes, respectively, and hence the offspring will be (7 8 5|1 9 6 3|4 10 2) after inverse. Mutation will occur randomly, and the probability of mutation is given as follows:

$$p_m = p_{m0} + \alpha_m \frac{(f_{max} - f_{min})^{n_m}}{(f_{max} - f_{min})^{n_m} + \left(f_{avg}\right)^{n_m}}, \quad (40)$$

where p_{m0} is the initial probability of mutation, α_m and n_m are the two adaptive coefficients, f_{max}, f_{avg}, and f_{min} are the maximal, average, and minimal fitness values in a population, respectively.

If a chromosome starts with a retailer, then the initial allele will be inverted with the first allele that represents a candidate HDCC location.

4.1.5. Simulated Annealing and Individual Replacement. In AGAs, individuals will be replaced by new ones for evolution. AHSAGA adopts SA as the steady-state technique [32], and the probability that a chromosome will be replaced is given as follows:

$$p = \begin{cases} 1, & \text{if } f_{new} \geq f_{old}, \\ \exp\left(-\dfrac{(1/f_{new})/(1/f_{old})}{T}\right), & \text{if } f_{new} < f_{old}, \end{cases} \quad (41)$$

where f_{new} and f_{old} are the fitness values of the new and old individuals, respectively, and T is the temperature given as follows:

$$T_{t+1} = \alpha T_t, \quad t \geq 0, 0 < \alpha < 1, \quad (42)$$

where T_t is the temperature at time t, and α is the change rate of temperature.

4.2. Algorithm. The pseudocode of AHSAGA is shown in Algorithm 1, and the steps in this algorithm are briefly explained as follows:

Step 1. Initialize parameters such as population size *pop_size*, iteration number M, crossover factors p_{c0}, α_c, n_c, mutation

Input: Parameters in Section 4.1
Output: Optimal location-route decisions
Begin
Choose population size *pop_size*;
Create an initial population *pop*(0) randomly;
for ($i = 1$ to *pop_size*) {
Calculate fitness f_i for individual i in population *pop(0)*;
}
$f_{\max}(0) = \max\{f_i\}$;
$f_{\min}(0) = \min\{f_i\}$;
$f_{avg}(0) = sum(f_i)/pop\_size$;
$f_{best} = f_{\max}(0)$;
pop_{best} = individual with f_{best};
Choose M ($M > 0$) as the number of iterations;
$m = 1$;
while ($m \le M$) {
$n = 1$;
while ($n \le pop\_size$) {
Apply select operator to create the mating pool;
Choose two chromosomes (parents) from the mating pool randomly;
Generate random number r_c;
Calculate crossover probability p_c;
if ($r_c \le p_c$) {
Apply crossover operator;
}
Generate random number r_m;
Calculate mutation probability p_m;
if ($r_m \le p_m$) {
Apply mutation operator;
}
}
for ($j = 1$ to *pop_size*) {
Calculate fitness f_j for individual j in population *pop (m)*;
}
$f_{\max}(m) = \max\{f_j\}$;
$f_{\min}(m) = \min\{f_j\}$;
if ($f_{\max}(m) \ge f_{\max}(m-1)$) then {
$f_{best} = f_{\max}(m)$;
pop_{best} = individual with f_{best};
}
else {
$\Delta = 1/f_{\max}(m) - 1/f_{\max}(m-1)$;
$z = e^{-\Delta/T}$;
Generate random number r_z;
If ($z \le r_z$) then {
$f_{best} = f_{\max}(m)$;
pop_{best} = individual with f_{best};
}
else {
Replace the individual with $f_{\min}(m)$ by pop_{best} in population *pop* (m);
}
$T = \alpha T$;
}
$m = m + 1$;
}
end

Algorithm 1: AHSAGA.

factors p_{m0}, α_m, n_m, initial temperature T_0, and cooling factor α.

Step 2. Create an initial population randomly.

Step 3. Generate an offspring population by applying selection, crossover, and mutation operators.

Step 4. Calculate the fitness values of the individuals in a new population, and identify those with the maximal and minimal fitness values.

Step 5. Check whether the maximal fitness value in an offspring population is greater than that in its parent population. If yes, go to Step 6. Otherwise, apply SA to decide whether the best solution in the parent population will be introduced into the offspring population, then update temperature in SA.

Step 6. Check whether the termination condition is satisfied. If yes, return the chromosome with the maximal fitness value. Otherwise, go to Step 3.

5. Numerical Study

In this study, AHSAGA is implemented by Matlab R2014a and all numerical experiments are conducted on a workstation equipped with an Intel Core i7-4790 CPU at 3.60 GHz and 8.0 GB of RAM under Windows 7.

To validate its performance, AHSAGA has been tested on five data sets that are adapted from LRP files provided by the University of Aveiro [35] for locations, fixed costs, and demands. For example, the input data adapted from the Gaskell67-21 × 5 files for HDCCs and retailers are shown in Tables 1 and 2, respectively. All other parameters are provided in Table 3.

5.1. Sensitivity Analysis. Since the performance of AHSAGA can be affected significantly by its parameters, a sensitivity analysis is conducted on the parameters shown in Table 4 to identify the optimal setting in this study. To eliminate the excessive number of combinations, other parameters will be set to their median values when a parameter is tested. The numerical results are shown in Figure 3, where red and blue lines represent the mean objective values and average computational times, respectively. Figure 3 shows that solution accuracy and computational times are both affected by those parameters, and the best result can be archived when $M = 1000$, $N = 100$, $p_{c0} = 0.8$, $p_{m0} = 0.25$, $\alpha_c = 0.09$, $\alpha_m = 0.175$, $n_c = 3$, $n_m = 2$, $T_0 = 200$, and $\alpha = 0.98$. The optimal setting is presented under the "Experiment setting" column in Table 4, and it will be used in the subsequent experiments.

From Figure 3, we can see that AHSAGA can be affected by those parameters in the following way:

(1) M and N

When M or N increases, the optimal value and computational time will decrease and increase, respectively. This indicates that more iterations or greater population diversity will lead to a better optimal solution with an additional time cost.

(2) p_{c0} and p_{m0}

The optimal value will always decrease when p_{m0} increases, but it will not always decrease when p_{c0} increases. This indicates that a larger probability of mutation is always helpful to get a better optimal solution, but the probability of crossover should be moderately large. Moreover, we can see that when p_{c0} and p_{m0} increase, the computational time will increase and decrease, respectively. This indicates that a larger p_{c0} and p_{m0} will slow down and speed up the convergence of AHSAGA, respectively.

(3) a_c and a_m

Table 1: Gaskell67-21 × 5 (retailer).

Retailer	Coordinates	Demand
i_1	(151,264)	55
i_2	(159,261)	35
i_3	(130,254)	40
i_4	(128,252)	70
i_5	(163,247)	105
i_6	(146,246)	20
i_7	(161,242)	40
i_8	(142,239)	5
i_9	(163,236)	25
i_{10}	(148,232)	30
i_{11}	(128,231)	60
i_{12}	(156,217)	65
i_{13}	(129,214)	65
i_{14}	(146,208)	15
i_{15}	(164,208)	45
i_{16}	(141,206)	105
i_{17}	(147,193)	50
i_{18}	(164,193)	45
i_{19}	(129,189)	125
i_{20}	(155,185)	90
i_{21}	(139,182)	35

Remark: in this table, the daily demands are adapted by dividing the original quantities by 20 due to the vehicle capacity parameter used in this study.

Table 2: Gaskell67-21 × 5 (HDCC).

Depot	Coordinates	Fixed cost
1	(136,194)	50
2	(143,237)	50
3	(136,216)	50
4	(137,204)	50
5	(128,197)	50

Table 3: LIRP parameters.

Parameter	Description	Value
b_r	Shipping cost per unit of product between a manufacturing plant and HDCC r	U [6, 10]
c	Vehicle capacity	1500
e_r	Fixed cost per shipment from a plant to HDCC r	U [21, 25]
f_r	Fixed administrative and handling cost of placing an order to a plant at HDCC r	U [16, 20]
g_r	Disposal cost per unit of returned product which cannot be refurbished at HDCC r	2
h	Holding cost per unit of new product per year at HDCC r	2
k	Holding cost per unit of returned product at HDCC r	1
m	Fixed cost of repairing and repacking one unit of returned product at a manufacturing plant	2
p_r	Inspection cost per unit of returned product at HDCC r	1
q_i	Daily returns from retailer i	U [1, 5]
u	Shipping cost per unit of product and distance	5
θ	Probability that a returned product cannot be refurbished	0.3
λ	Working days per year	300

Table 4: AHSAGA parameters.

Parameter	Description	Sensitivity analysis		Experimental setting
		Range	Median	
M	Number of iterations	{200, 400, 600, 800, 1000}	600	600
N	Population size	{20, 40, 60, 80, 100}	60	60
p_{c0}	Initial crossover probability	{0.5, 0.6, 0.7, 0.8, 0.9}	0.7	0.8
p_{m0}	Initial mutation probability	{0.05, 0.1, 0.15, 0.2, 0.25}	0.15	0.25
α_c	Adaptive coefficient	{0.08, 0.09, 0.1, 0.11, 0.12}	0.1	0.09
α_m	Adaptive coefficient	{0.1, 0.125, 0.15, 0.175, 0.2}	0.15	0.175
n_c	Adaptive coefficient	{1, 2, 3, 4, 5}	3	3
n_m	Adaptive coefficient	{1, 2, 3, 4, 5}	3	2
T_0	Initial temperature	{50, 100, 150, 200, 250}	150	200
α	Cooling rate	{0.95, 0.96, 0.97, 0.98, 0.99}	0.97	0.98

When a_c increases, the optimal solution can be always improved with an additional time cost. But a_m should have a moderate value to achieve the best performance in terms of optimal solution and computational time.

(4) n_c and n_m

n_c needs to be moderate to get the best optimal solution, but a larger n_c can always improve the convergence and reduce the computational time. When n_m increases, the optimal solution can always be improved with an additional time cost.

(5) T_0 and α

A higher initial temperature T_0 in SA can always reduce computational time, but it is not always helpful to get a better optimal solution. However, a larger cooling factor α will always improve the optimal solution and computational time.

5.2. Illustrative Example. In this section, Gaskell67-21 × 5 files [35] are used as an example to show the application and performance of AHSAGA. To get started, an initial chromosome is generated randomly, which is {5, 9, 10, 8, 26, 7, 14, 21, 15, 12, 13, 25, 1, 2, 11, 16, 3, 4, 18, 20, 6, 17, 24, 22, 19, 23}. According to the encoding-decoding scheme in Section 4.1, HDCC locations and vehicle routes can be decoded as that in Table 5, and then the corresponding optimal number of orders per year can be calculated by (36). When the algorithm is executed iteratively, objective values and adaptive probabilities change monotonically as shown in Figures 4 and 5, respectively.

To validate its performance, AHSAGA is compared with other two heuristics in the literature, which are adaptive annealing genetic algorithm (IAGA) [34] and hybrid genetic simulated annealing algorithm (HGSAA) [29]. To avoid any bias, each algorithm is replicated 50 times by using a same data set, and the mean objective values and computational times are compared. To illustrate the stochastic nature of

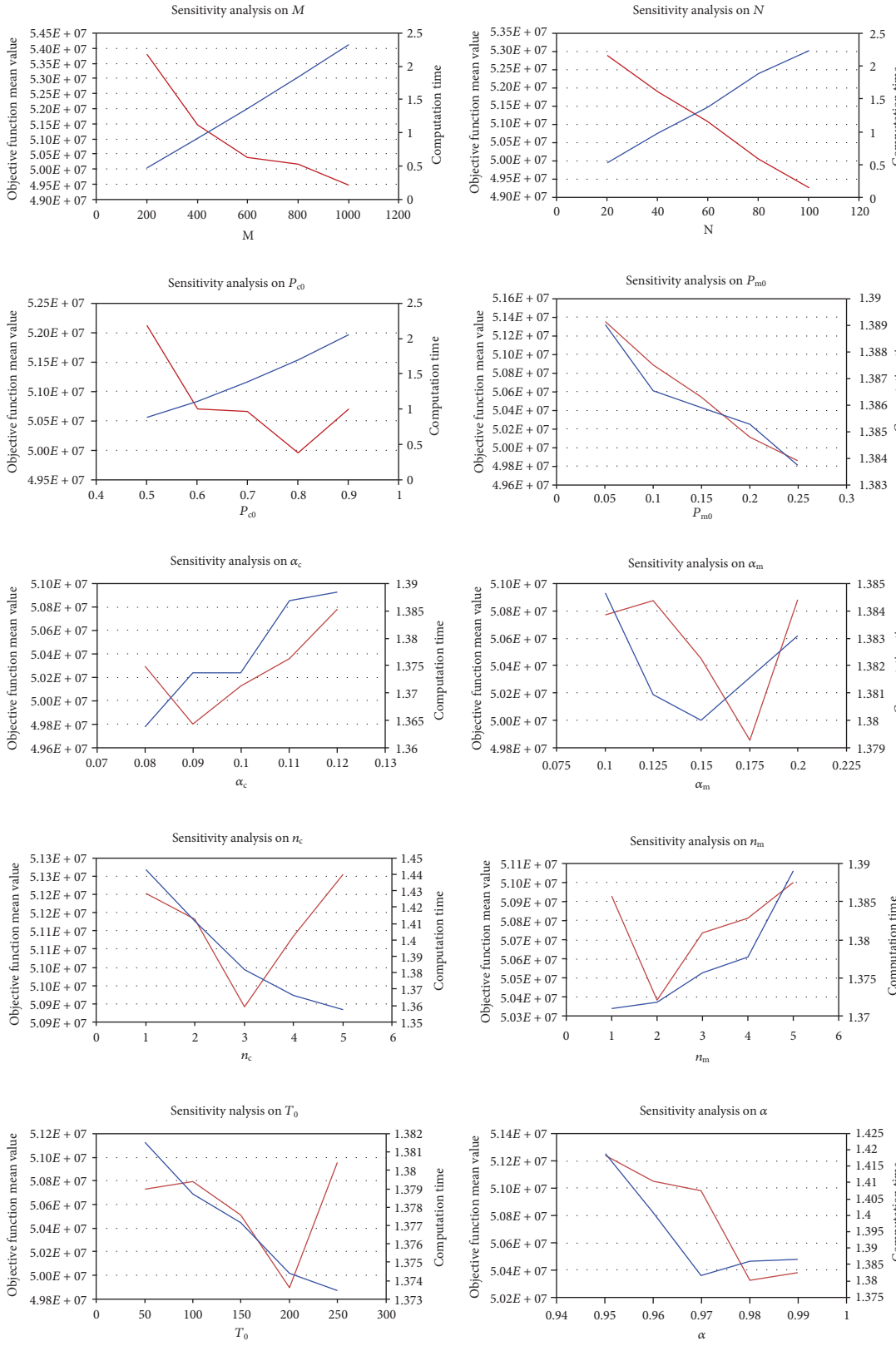


Figure 3: Sensitivity analysis on AHSAGA parameters.

the three algorithms, Figure 6 shows the 50 objective values from AHSAGA, HGSAA, and IAGA in a descending order by using the data adapted from Gaskell67-21 × 5 files, and Table 6 presents a more thorough comparison between the three algorithms on this data set, which shows that AHSAGA is more effective than HGSAA and IAGA

Table 5: Initial HDCC locations and vehicle routes.

HDCC number	Vehicle number	Route	Number of orders
2	1	11-16	24
4	2	18-20-6-17	62
	3	24-22-19-23	
5	4	9-10-8-26-7-14	65
	5	21-15-12-13-25	

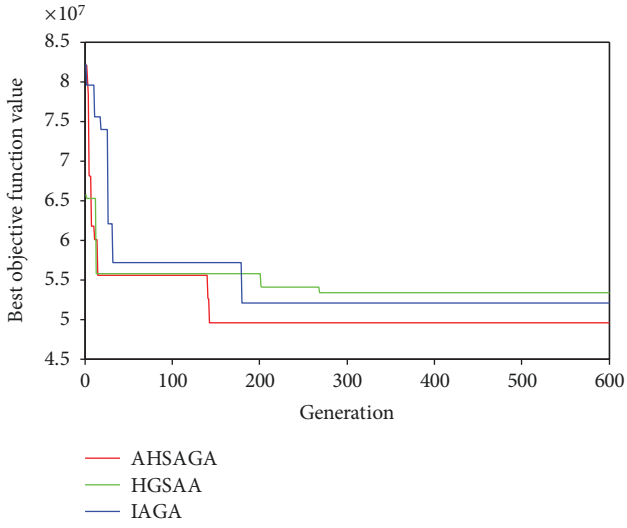
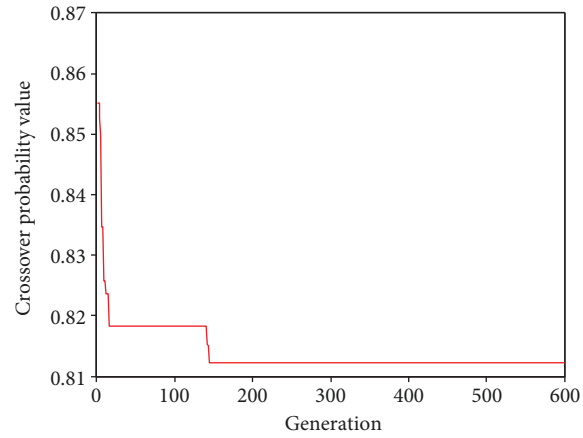
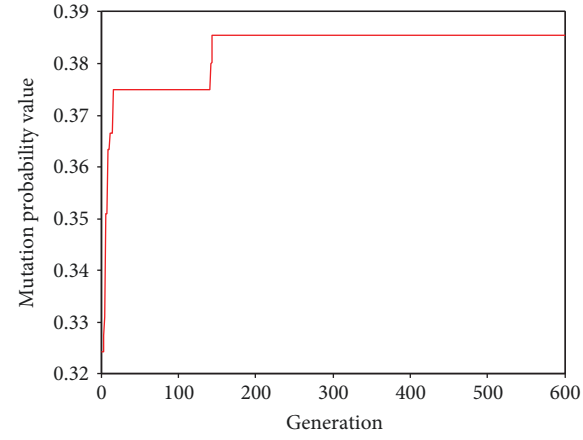


Figure 4: Trend in objective values.



(a) Crossover probability



(b) Mutation probability

Figure 5: Trend in adaptive probabilities.

from the perspectives of robustness, solution quality, and time efficiency.

5.3. Performance Comparison. The section presents a comprehensive comparison between AHSAGA, HGSAA, and IAGA on three types of problems by the number of retailers. More specifically, the number of retailers is less than 50 in small-size problems, between 50 and 100 in medium-size problems, and more than 100 in large-size problems. The numerical results on small-size, medium-size, and large-size problems are shown in Tables 7–9, respectively, from which we can make the following conclusions:

(i) The mean objective values from AHSAGA are significantly lower than those from IAGA and HGSAA for most problems. This indicates that AHSAGA has a great capability to search global optimums and hence can provide better solutions.

(ii) AHSAGA takes less computational times and convergence generations to find the optimal solution than IAGA and HGSAA for all problems. This indicates that AHSAGA is the most efficient approach.

(iii) The variation of the optimal values from AHSAGA, which is measured by the coefficient of variation, is lower than that from IAGA and HGSAA for all problems. This indicates that AHSAGA is more robust and consistent than the other two algorithms.

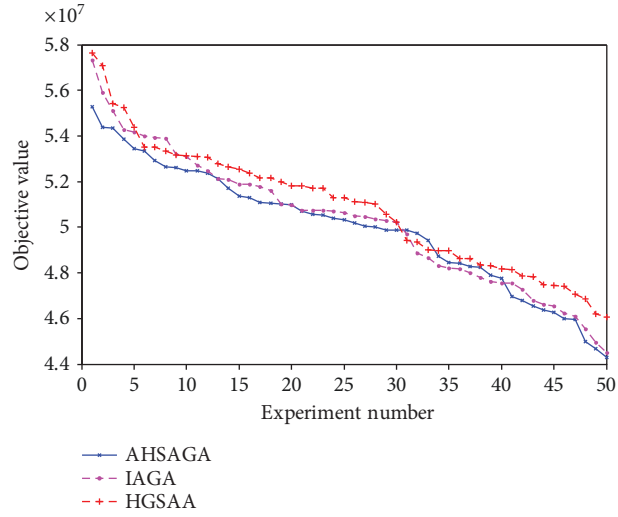


Figure 6: Objective values from AHSAGA, HGSAA, and IAGA (Gaskell67-21 × 5).

Table 6: Computational results on Gaskell67-21 × 5.

Instance name		HGSAA			IAGA			AHSAGA				
		M1	SD	CV	M2	SD	CV	M3	SD	CV	$\frac{(M1-M3)}{M1}$	$\frac{(M2-M3)}{M2}$
Gaskell67-21 × 5	Computational time	2.09	0.02	0.01	1.85	0.03	0.02	1.76	0.02	0.01	15.59%	4.80%
	Convergence generation	221.22	121.62	0.55	211.02	112.90	0.54	202.18	107.53	0.53	8.61%	4.19%
	Total cost	50860938.99	2778602.09	0.05	50276017.05	2991285.98	0.06	49977998.24	2709926.51	0.05	1.74%	0.59%

Remark: in Table 6, M1, M2, and M3 represent the mean objective values obtained by HGSAA, IAGA, and AHSAGA, respectively. SD means standard deviation, and CV means coefficient of variation. The same convention will also be used in the subsequent tables.

Table 7: Computational results for small-size problems.

Instance name		HGSAA			IAGA			AHSAGA			$\frac{(M1-M3)}{M1}$	$\frac{(M2-M3)}{M2}$
		M1	SD	CV	M2	SD	CV	M3	SD	CV		
Srivastava86-8 × 2	Computational time	1.30	0.02	0.02	1.09	0.01	0.01	1.10	0.01	0.01	15.73%	−0.43%
	Convergence generation	201.66	116.19	0.58	203.34	115.63	0.57	195.14	108.22	0.55	3.23%	4.03%
	Total cost	58357282.29	3318088.97	0.06	58972563.90	3408120.16	0.06	58080565.56	3150230.79	0.05	0.47%	1.51%
Perl83-12 × 2	Computational time	1.39	0.02	0.01	1.20	0.02	0.01	1.19	0.01	0.01	14.24%	1.01%
	Convergence generation	197.06	109.60	0.56	199.30	133.01	0.67	187.28	116.90	0.62	4.96%	6.03%
	Total cost	9879584.53	514520.39	0.05	9866462.28	494570.55	0.05	9857315.61	490727.85	0.05	0.23%	0.09%
Gaskell67-22 × 5	Computational time	2.01	0.04	0.02	1.71	0.03	0.02	1.71	0.03	0.01	14.99%	0.11%
	Convergence generation	206.52	115.21	0.56	217.36	123.96	0.57	198.64	109.48	0.55	3.82%	8.61%
	Total cost	90392692.15	4533650.78	0.05	90140435.61	5993545.10	0.07	89430743.81	4280679.66	0.05	1.06%	0.79%
Min92-27 × 5	Computational time	2.37	0.01	0.01	1.99	0.02	0.01	1.99	0.02	0.01	16.38%	0.00%
	Convergence generation	175.82	106.58	0.61	188.10	103.53	0.55	175.68	94.84	0.54	0.08%	6.60%
	Total cost	363157783.99	25693304.31	0.07	367201239.25	27220166.35	0.07	356654161.93	24405194.09	0.07	1.79%	2.87%
Gaskell67-29 × 5	Computational time	2.51	0.03	0.01	2.08	0.03	0.01	2.07	0.02	0.01	17.74%	0.87%
	Convergence generation	212.40	117.98	0.56	208.40	134.58	0.65	200.86	101.74	0.51	5.43%	3.62%
	Total cost	85966292.78	4976794.74	0.06	85879276.51	5292913.64	0.06	85053221.84	4861829.37	0.06	1.06%	0.96%
Gaskell67-32 × 5	Computational time	2.62	0.04	0.02	2.28	0.03	0.01	2.24	0.03	0.01	14.48%	1.55%
	Convergence generation	202.96	119.95	0.59	201.96	119.24	0.59	197.00	113.53	0.58	2.94%	2.46%
	Total cost	142731259.91	7358411.37	0.05	142691917.33	7239503.55	0.05	140407259.50	7051863.18	0.05	1.63%	1.60%
Gaskell67-36 × 5	Computational time	2.86	0.04	0.01	2.45	0.04	0.02	2.45	0.03	0.01	14.33%	0.03%
	Convergence generation	192.58	113.85	0.59	205.52	122.26	0.59	189.00	111.51	0.59	1.86%	8.04%
	Total cost	51830112.18	3370888.67	0.07	52045789.31	2998029.49	0.06	51240375.41	2951876.85	0.06	1.14%	1.55%

Table 8: Computational results for medium-size problems.

Instance name		HGSAA			IAGA			AHSAGA				
		M1	SD	CV	M2	SD	CV	M3	SD	CV	$\frac{(M1-M3)}{M1}$	$\frac{(M2-M3)}{M2}$
Christofides69-50 × 5	Computational time	3.52	0.04	0.01	2.96	0.05	0.02	2.97	0.04	0.01	15.60%	−0.14%
	Convergence generation	185.52	118.83	0.64	179.58	114.49	0.64	173.90	106.46	0.61	6.26%	3.16%
	Total cost	64622673.96	4896193.44	0.08	65120367.91	4880581.60	0.07	63410536.90	4447087.25	0.07	1.88%	2.63%
Perl83-55 × 15	Computational time	4.55	0.04	0.01	4.14	0.06	0.01	4.09	0.05	0.01	10.01%	1.27%
	Convergence generation	202.68	120.89	0.60	199.34	117.33	0.59	194.64	114.00	0.59	3.97%	2.36%
	Total cost	75608999.15	4218565.18	0.06	75488435.46	4821373.57	0.06	74318631.82	3372606.19	0.05	1.71%	1.55%
Christofides69-75 × 10	Computational time	5.53	0.06	0.01	4.76	0.09	0.02\	4.80	0.05	0.01	13.24%	−0.73%
	Convergence generation	203.52	119.05	0.58	201.04	115.46	0.57	193.18	110.48	0.57	5.08%	3.91%
	Total cost	140932062.97	10696126.94	0.08	137838194.63	13260464.21	0.10	137205765.38	10092485.60	0.07	2.64%	0.46%
Perl83-85 × 7	Computational time	6.03	0.04	0.01	5.21	0.11	0.02	5.21	0.10	0.02	13.48%	−0.11%
	Convergence generation	211.78	119.39	0.56	186.32	113.21	0.61	182.14	102.34	0.56	14.00%	2.24%
	Total cost	407102768.27	10738366.84	0.03	404838974.84	12150698.81	0.03	404614214.54	9236869.46	0.02	0.61%	0.06%
Daskin95-88 × 8	Computational time	6.23	0.04	0.01	5.47	0.10	0.02	5.43	0.08	0.02	12.84%	0.75%
	Convergence generation	214.16	125.73	0.59	206.94	112.33	0.54	206.64	110.85	0.54	3.51%	0.14%
	Total cost	95326029.80	6322807.23	0.07	94077056.19	5903902.59	0.06	94002405.72	5837160.63	0.06	1.39%	0.08%

Table 9: Computational results for large-size problems.

Instance name		HGSAA			IAGA			AHSAGA				
		M1	SD	CV	M2	SD	CV	M3	SD	CV	$\frac{(M1-M3)}{M1}$	$\frac{(M2-M3)}{M2}$
Christofides69-100 × 10	Computational time	7.19	0.08	0.01	6.55	0.14	0.02	6.48	0.07	0.01	9.97%	1.18%
	Convergence generation	190.72	114.09	0.60	191.74	116.64	0.61	188.22	106.69	0.57	1.31%	1.84%
	Total cost	229586535.05	13462701.82	0.06	231296925.10	15532578.99	0.07	227749748.70	10551026.83	0.05	0.80%	1.53%
Or76-117 × 14	Computational time	8.70	0.07	0.01	7.71	0.20	0.03	7.71	0.10	0.01	11.34%	−0.08%
	Convergence generation	209.78	126.74	0.60	198.64	119.05	0.60	188.28	108.04	0.57	10.25%	5.22%
	Total cost	5062462547.35	255122955.85	0.05	5079364715.42	256027550.46	0.05	5010781647.96	244879865.55	0.05	1.02%	1.35%
Min92-134 × 8	Computational time	9.57	0.10	0.01	8.53	0.17	0.02	8.48	0.10	0.01	11.40%	0.61%
	Convergence generation	191.82	133.67	0.70	188.68	111.54	0.59	184.32	106.78	0.58	3.91%	2.31%
	Total cost	3727217265.35	200966165.84	0.05	3750482276.62	201259019.17	0.05	3721752794.82	166680491.26	0.04	0.15%	0.77%
Daskin95-150 × 10	Computational time	10.95	0.10	0.01	9.07	0.19	0.02	9.07	0.13	0.01	17.22%	0.05%
	Convergence generation	223.24	112.92	0.51	204.74	107.05	0.52	195.22	95.69	0.49	12.55%	4.65%
	Total cost	17531706417.34	1394570344.68	0.08	14896666385.68	1140033562.24	0.08	14688680807.96	1050208955.68	0.07	16.22%	1.40%
Perl83-318 × 4	Computational time	9.50	0.10	0.01	8.40	0.18	0.02	8.39	0.08	0.01	11.64%	0.03%
	Convergence generation	195.98	113.32	0.58	189.50	105.69	0.56	187.38	102.74	0.55	4.39%	1.12%
	Total cost	3758638540.59	174594029.53	0.05	3764348485.23	174118187.75	0.05	3718507764.20	166085079.53	0.04	1.07%	1.22%

6. Conclusions and Future Study

Closed-loop supply chains are an emerging and important topic due to the tremendous economic and environmental impact of consumer returns. In this paper, we study a location-inventory-routing problem in a closed-loop supply chain by formulating it as a nonlinear integer programming model. Since the problem is NP-hard, we also design a novel adaptive genetic algorithm by incorporating simulated annealing to solve this model efficiently. To make this study more practical, many real-world business scenarios such as vehicle capacity and the disposal of different types of returned products are also considered and modeled precisely.

This study can be extended in several directions in the future: first, this problem will be more practical and flexible if some assumptions are relaxed. For example, it will be flexible to allow a many-to-many relationship between vehicles and retailers, and it will also be more practical to relax the assumption that a retailer will be visited by a vehicle every working day. Second, since secondary markets have become an important channel to sell used products, it will be greatly beneficial to study LIRPs in a CLSC by considering those markets. Third, our model will be more valuable if it incorporates more business scenarios such as supply risk and multiple sourcing.

Sets

R: set of candidate HDCC locations, where $r \in R$
V: set of vehicles, where $v \in V$
S: set of retailers, where $i, j \in S$
L: set of locations, which is the union of HDCCs and retailers (i.e., $L = R \cup S$).

Parameters

a_r: fixed cost of building and operating a HDCC at location r
b_r: shipping cost per unit of product between the factory and HDCC r
c: vehicle capacity
d_i: daily demand of retailer i
e_r: fixed cost per shipment from the factory to HDCC r
f_r: fixed administrative and handling cost of placing an order to the factory from HDCC r
g_r: disposal cost per unit of returned product which cannot be refurbished at HDCC r
h: holding cost per unit of new product per year at HDCC r
k: holding cost per unit of returned product at HDCC r
m: fixed cost of repairing and repacking one unit of returned product at the factory
p_r: inspection cost per unit of returned product at HDCC r
q_i: daily returns from retailer i, where $q_i < d_i$
s_{ri}: distance from HDCC r to retailer i in a route (forward logistics)
t_{ir}: distance from retailer i to HDCC r in a route (reverse logistics)
u: shipping cost per unit of product and distance
θ: probability that a returned product cannot be refurbished
λ: workdays per year (remark: similar to [21], we assume that a retailer will be visited by a vehicle every workday. Hence, λ is also the number of road trips for a vehicle per year).

Decision Variables

N_r: number of orders placed at HDCC r per year

$$W_r = \begin{cases} 1, & \text{if a HDCC is built at location } r, \\ 0, & \text{otherwise.} \end{cases}$$

$$U_{rv} = \begin{cases} 1, & \text{if vehicle } v \text{ is operated by the HDCC at location } r, \\ 0, & \text{otherwise.} \end{cases}$$

$$X_{ir} = \begin{cases} 1, & \text{if retailer } i \text{ is served by the HDCC at location } r, \\ 0, & \text{otherwise.} \end{cases}$$

$$Y_{irv} = \begin{cases} 1, & \text{if the logistics flows between retailer } i \text{ and the HDCC at location } r \text{ are carried by vehicle } v, \\ 0, & \text{otherwise.} \end{cases}$$

$$Z_{ijrv} = \begin{cases} 1, & \text{if vehicle } v \text{ is directed from retailer } i \text{ to } j \text{ on a route that belongs to the HDCC at location } r, \\ 0, & \text{otherwise.} \end{cases}$$

Conflicts of Interest

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This research was supported by the National Natural Science Foundation of China under Grant nos. 71672074 and 71772075.

References

[1] T. P. Harrison, "Principles for the strategic design of supply chains," in *The practice of supply chain management: Where theory and application converge*, pp. 3–12, Springer, Boston, MA, USA, 2004.

[2] National Retail Federation, "2015 consumer returns in the retail industry," April 2018, https://nrf.com/sites/default/files/Images/Media%20Center/NRF%20Retail%20Return%20Fraud%20Final_0.pdf.

[3] "E-commerce product return rate - statistics and trends," May 2018, https://www.invespcro.com/blog/ecommerce-product-return-rate-statistics/.

[4] J. Mostard and R. Teunter, "The newsboy problem with resalable returns: a single period model and case study," *European Journal of Operational Research*, vol. 169, no. 1, pp. 81–96, 2006.

[5] M. Al-Salem, A. Diabat, D. Dalalah, and M. Alrefaei, "A closed-loop supply chain management problem: reformulation and piecewise linearization," *Journal of Manufacturing Systems*, vol. 40, pp. 1–8, 2016.

[6] Y. C. Tsao, V. T. Linh, and J. C. Lu, "Closed-loop supply chain network designs considering RFID adoption," *Computers & Industrial Engineering*, vol. 113, pp. 716–726, 2017.

[7] M. S. Daskin, C. R. Coullard, and Z. J. M. Shen, "An inventory-location model: formulation, solution algorithm and computational results," *Annals of Operations Research*, vol. 110, no. 1/4, pp. 83–106, 2002.

[8] Z.-J. M. Shen, C. Coullard, and M. S. Daskin, "A joint location-inventory model," *Transportation Science*, vol. 37, no. 1, pp. 40–55, 2003.

[9] J. Meissner and O. V. Senicheva, "Approximate dynamic programming for lateral transshipment problems in multi-location inventory systems," *European Journal of Operational Research*, vol. 265, no. 1, pp. 49–64, 2018.

[10] Z. Dai, F. Aqlan, X. Zheng, and K. Gao, "A location-inventory supply chain network model using two heuristic algorithms for perishable products with fuzzy constraints," *Computers & Industrial Engineering*, vol. 119, pp. 338–352, 2018.

[11] B. Vahdani, M. Soltani, M. Yazdani, and S. Meysam Mousavi, "A three level joint location-inventory problem with correlated demand, shortages and periodic review system: robust meta-heuristics," *Computers & Industrial Engineering*, vol. 109, pp. 113–129, 2017.

[12] M. Farahani, H. Shavandi, and D. Rahmani, "A location-inventory model considering a strategy to mitigate disruption risk in supply chain by substitutable products," *Computers & Industrial Engineering*, vol. 108, pp. 213–224, 2017.

[13] S. J. Sadjadi, A. Makui, E. Dehghani, and M. Pourmohammad, "Applying queuing approach for a stochastic location-inventory problem with two different mean inventory considerations," *Applied Mathematical Modelling*, vol. 40, no. 1, pp. 578–596, 2016.

[14] R. Z. Farahani, H. Rashidi Bajgan, B. Fahimnia, and M. Kaviani, "Location-inventory problem in supply chains: a modelling review," *International Journal of Production Research*, vol. 53, no. 12, pp. 3769–3788, 2015.

[15] A. Diabat, T. Abdallah, and A. Henschel, "A closed-loop location-inventory problem with spare parts consideration," *Computers & Operations Research*, vol. 54, pp. 245–256, 2015.

[16] H. Guo, C. Li, Y. Zhang, C. Zhang, and M. Lu, "A location-inventory problem in a closed-loop supply chain with secondary market consideration," *Sustainability*, vol. 10, no. 6, p. 1891, 2018.

[17] Y. Li, H. Guo, and Y. Zhang, "An integrated location-inventory problem in a closed-loop supply chain with third-party logistics," *International Journal of Production Research*, vol. 56, no. 10, pp. 3462–3481, 2018.

[18] H. Min, V. Jayaraman, and R. Srivastava, "Combined location-routing problems: a synthesis and future research directions," *European Journal of Operational Research*, vol. 108, no. 1, pp. 1–15, 1998.

[19] G. Nagy and S. Salhi, "Location-routing: issues, models and methods," *European Journal of Operational Research*, vol. 177, no. 2, pp. 649–672, 2007.

[20] M. Schneider and M. Drexl, "A survey of the standard location-routing problem," *Annals of Operations Research*, vol. 259, no. 1-2, pp. 389–414, 2017.

[21] Z.-J. Max Shen and L. Qi, "Incorporating inventory and routing costs in strategic location models," *European Journal of Operational Research*, vol. 179, no. 2, pp. 372–389, 2007.

[22] A. Ahmadi Javid and N. Azad, "Incorporating location, routing and inventory decisions in supply chain network design," *Transportation Research Part E: Logistics and Transportation Review*, vol. 46, no. 5, pp. 582–597, 2010.

[23] A. Hiassat, A. Diabat, and I. Rahwan, "A genetic algorithm approach for location-inventory-routing problem with perishable products," *Journal of Manufacturing Systems*, vol. 42, pp. 93–103, 2017.

[24] Z. Rafie-Majd, S. H. R. Pasandideh, and B. Naderi, "Modelling and solving the integrated inventory-location-routing problem in a multi-period and multi-perishable product supply chain with uncertainty: Lagrangian relaxation algorithm," *Computers & Chemical Engineering*, vol. 109, pp. 9–22, 2018.

[25] W. J. Guerrero, C. Prodhon, N. Velasco, and C. A. Amaya, "Hybrid heuristic for the inventory location-routing problem with deterministic demand," *International Journal of Production Economics*, vol. 146, no. 1, pp. 359–370, 2013.

[26] Y. Zhang, M. Qi, L. Miao, and E. Liu, "Hybrid metaheuristic solutions to inventory location routing problem," *Transportation Research Part E: Logistics and Transportation Review*, vol. 70, pp. 305–323, 2014.

[27] N. Nekooghadirli, R. Tavakkoli-Moghaddam, V. R. Ghezavati, and S. Javanmard, "Solving a new bi-objective location-routing-inventory problem in a distribution network by meta-heuristics," *Computers & Industrial Engineering*, vol. 76, pp. 204–221, 2014.

[28] F. Rayat, M. M. Musavi, and A. Bozorgi-Amiri, "Bi-objective reliable location-inventory-routing problem with partial back-ordering under disruption risks: a modified amosa approach," *Applied Soft Computing*, vol. 59, pp. 622–643, 2017.

[29] Y. Li, H. Guo, L. Wang, and J. Fu, "A hybrid genetic-simulated annealing algorithm for the location-inventory-routing problem considering returns under e-supply chain environment," *The Scientific World Journal*, vol. 2013, Article ID 125893, 10 pages, 2013.

[30] S. Deng, Y. Li, H. Guo, and B. Liu, "Solving a closed-loop location-inventory-routing problem with mixed quality defects returns in e-commerce by hybrid ant colony optimization algorithm," *Discrete Dynamics in Nature and Society*, vol. 2016, Article ID 6467812, 12 pages, 2016.

[31] M. Zhalechian, R. Tavakkoli-Moghaddam, B. Zahiri, and M. Mohammadi, "Sustainable design of a closed-loop location-routing-inventory supply chain network under mixed uncertainty," *Transportation Research Part E: Logistics and Transportation Review*, vol. 89, pp. 182–214, 2016.

[32] K. Sastry, D. Goldberg, and G. Kendall, "Genetic algorithms," in *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*, pp. 97–125, Springer, 2005.

[33] D. Bertsimas and J. Tsitsiklis, "Simulated annealing," *Statistical Science*, vol. 8, no. 1, pp. 10–15, 1993.

[34] L. Wang and D. Tang, "An improved adaptive genetic algorithm based on hormone modulation mechanism for job-shop scheduling problem," *Expert Systems with Applications*, vol. 38, no. 6, pp. 7243–7250, 2011.

[35] S. S. Barreto, "Location-routing problems (LRP)," http://sweet.ua.pt/~iscf143/_private/SergioBarretoHomePage.htm.