# Transshipment Vehicle Routing with Pickup and Delivery for Cross-Filling 

Suk-Chul Rim (i) and Hang T. T. Vu (D)<br>Department of Industrial Engineering, Ajou University, Suwon 16499, Republic of Korea<br>Correspondence should be addressed to Suk-Chul Rim; scrim@ajou.ac.kr

Received 11 November 2020; Revised 25 December 2020; Accepted 31 December 2020; Published 18 January 2021
Academic Editor: Kin Keung Lai
Copyright © 2021 Suk-Chul Rim and Hang T. T. Vu. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.


#### Abstract

Distribution centers (DCs) typically receive orders from the customers (mostly retail stores) located in their vicinity and deliver the ordered goods the next day morning. To maintain high item fill rate, DCs have to hold a high level of inventory, which will increase inventory cost. As an alternative, cross-filling is that, after closing the daily order receipt, DCs exchange surplus items during the night to reduce the shortage. The economic justification of such cross-filling will depend on the tradeoff between extra transshipment and handling cost versus saved shortage cost. In this paper, as an extension of Rim and Jiang, 2019, vehicles are allowed to drop and pick up items at the intermediate DCs in the route. We present a genetic algorithm to determine the routes and amount to pick up/drop at each DC to minimize the total cost.


## 1. Problem Description

The landscape of the logistics industry has been rapidly changing in recent years due to the evolution of the information technology. Quick delivery of goods to the consumers is becoming the key of the fast logistics business. Factories, distribution centers (DCs), and retailers are physical entities of the most manufacturers' supply chain. Moreover, online commerce also employs DCs as an important element of its supply chain for efficient storage and order fulfillment for fast delivery of goods. Order fulfillment has been one of the critical performance indicators in most DCs [1]. Besides, the shortage at DCs will very likely lead to the shortage of the item at the retail stores, which will result in sales loss and negative impact on the consumers' satisfaction.

To avoid shortage, DCs are forced to hold high level of inventory, which will incur significant cost. As an alternative, cross-filling has been studied in academia and practiced in business for decades. Cross-filling is, after closing the daily order receipt, to ship the surplus items to other DCs that lack the same item to avoid shortage with relatively lower level of inventory. Such a lateral transshipment is also referred to as
lateral resupply, reallocation of stock, and stock transfer [2]. Since extra trucking and handling during the night will clearly incur cost, a major concern is whether such a crossfilling can be economical or not, which will depend on the cost parameters such as shortage penalty, trucking, and handling cost. Also, we need to be able to determine the optimal transshipment routes of trucks within a very short length of time. This paper presents a viable solution procedure for this problem.

The problem we address in this paper is as follows: consider a set of DCs of a company and multiple stores. Each DC supplies multiple items to multiple retail stores located in its own geographical area. Each DC receives orders from its retail stores during a day, closes the receipt of orders at a certain time in the late afternoon, processes the received orders, and tries to fulfill the orders using its own inventory. For shortage items, DCs try to find a surplus of the same item from other DCs. To minimize the total cost, we need to determine how many of which items to transship from which DC to which DC in what sequence. Trucks transship the items among the DCs during the night and deliver the items to the retail stores the next morning.

In determining the transshipment routes, we assume that a truck may depart from any DC, but must return to its starting DC. Except the starting DCs, each intermediate DC in the routes can be visited by at most one truck, mainly due to the short length of transshipment time window. Also, trucks can not only drop some items but also pick up some items at the intermediate DCs, which makes the problem more complicated, but will improve the performance of the cross-filling. Transshipment must be completed within a certain time window so that transshipped items can be used for delivery to the retail stores along with the inventory stored in the DC.

Before the transshipment trucks depart, each DC places orders to the factory for replenishment of items, which arrive the next day morning. Each DC may use various replenishment policies. In this paper, we assume that all DCs use base stock policy, that is, to order up to a certain level for each item every day since it is known to be optimal [3]. Figure 1 shows an example of the resulting routes among ten DCs for cross-filling, where shaded DCs 3,5 , and 9 represent the DCs the transshipment trucks depart from. Recall that the truck departing from DC 3 can pick up some items at DC 10 and drop some of them at DC 8,1 , or 3 .

Unlike most existing research on this problem where transshipment cost is considered to be proportional to the number of transshipped items, to be more realistic, we assume that the transshipment cost comprises two elements; trucking cost is proportional to the travel distance of trucks, regardless of the quantity loaded in them; and material handling cost is proportional to the number of transshipped items. We assume that each DC is visited at most once, and only one truck can depart from any DC because moving more than one truckload a day for cross-filling seems unrealistic.

We name this problem as the transshipment vehicle routing with simultaneous pickup and delivery (TVRSPD) problem. In this paper, we propose a genetic algorithm, in which a chromosome defines a set of routes; and we present an algorithm to determine the optimal amount to pick up/ drop each item at each intermediate DC for each route so as to minimize the total cost for a day. Since the system regenerates every day, the optimal decision for one day will result in the optimal average performance over the infinite horizon [3].

This paper is organized as follows: Section 2 reviews the related research in the literature.Section 3 describes a genetic algorithm-based solution procedure, along with illustrative numerical examples.Section 4 reports the average performance of the proposed method. Conclusion and future research are given in Section 5.

## 2. Related Literature

To the best of our knowledge, the problem we address in this paper has not been presented in the literature. Our problem TVRSPD is positioned at the overlapping area of the two well-known problems: vehicle routing problem (VRP) and the transshipment problem. VRP has long been studied over six decades since the first work of Dantzig and Ramser [4].


Figure 1: An example of three transshipment routes among ten DCs.

For a comprehensive review on VRP including its variants, formulations, and solution methods, readers may refer to Toth and Vigo [5]. Among many variations, VRP with simultaneous pickup and delivery (VRPSPD) was introduced by Min [6]. Its comprehensive review is available in Berbeglia et al. [7] and Koc et al. [8]. TVRSPD has clear distinctions from VRPSPD as follows: (1) for the objective function, most of the VRP and its variants try to minimize the total travel distance. However, from the business viewpoint, we believe that the total travel distance of vehicles does not appropriately represent the total cost. Instead, to be more realistic, TVRSPD considers the total cost comprising shortage cost and handling cost, in addition to the trucking cost proportional to the travel distance of vehicles. (2) In VRPSPD, items are supplied from depot to the customers, while in TVRSPD, no depot is involved, and items are supplied from some DCs which have surplus. (3) In VRPSPD, vehicles must bring all the pickup amount to the depot, while in TVRSPD, vehicles are allowed to reallocate items, that is, to drop the items that are picked up from the DCs visited earlier at any DC to visit later. (4) In VRPSPD, all pickup and delivery demands at each customer must be met as constraints, while in TVRSPD, it is not a constraint but only the objective function of minimizing the total cost will determine how many of which items to pick up and drop at where.

TVRSPD is different from multiple depot VRP $[9,10]$ in which the locations of the multiple depot are predetermined, whereas in TVRSPD, any DC can be selected as the starting node of the route. TVRSPD is also different from the lo-cation-routing problem [11] which involves simultaneously locating a number of facilities among candidate sites and establishing delivery routes to a set of users in such a way that the total system cost is minimized.

Among many related research studies, Zhang et al. [12] studied a real-world problem found in fast fashion business with a warehouse and multiple retail outlets, where trucks depart from the warehouse and visit a sequence of outlets to drop and pick up items for possible reallocation to other outlets. Tasan and Gen [13] proposed a genetic algorithm approach to VRPSPD. The volume feasibility is guaranteed with a proper decoding procedure, and routes are determined due to the vehicle capacity. Nagy and Salhi [10] proposed insertion-based heuristics for VRPSPD. They
introduced the idea of weak and strong feasibility and extended the algorithm to the problems with multiple depots.

The VRP with backhauls (VRPB) is a related problem with TVRSPD, where a vehicle picks up goods after finishing all deliveries, and both actions are in one route [14]. VRPB can be separated as two capacitated vehicle routing problems, in which the demands of each customer must be fulfilled within the vehicle's capacity. Customers who require deliveries from the depot are known as line hauls, and customers who require pick-ups or send goods are backhauls [15]. Several studies related to VRPB have been conducted in the literature [16, 17]. The mixing of line hauls and backhauls has proven to be very beneficial to the industry. The "milk run" is a well-known concept that is obtained from the achievements with VRPB. By applying milk runs, transportation costs and total travel distance can be significantly reduced, and the vehicle loading rate is increased [18].

The other avenue related with TVRSPD is the transshipment problem. Transshipment can be classified by many criteria. One of the classifications is that it is either a centralized or decentralized problem. Centralized transshipment problem is to minimize the total cost incurred on all locations, while the decentralized one is that each location tries to maximize its own profit by determining the quantity and price of items for transshipment. Another classification is that transshipment can be either proactive or reactive. Proactive transshipment takes place at fixed points in time before observing the demand and periodically rebalances the whole system's stock levels. For example, Van der Heide and Roodbergen [19] addressed the problem of transshipping and rebalancing books in libraries. They presented a stochastic dynamic programming formulation and heuristic algorithm. Contrarily, reactive transshipment responds to the stocking points which face a shortage, while another has surplus stock on hand [2]. The transshipment can be classified as either complete pooling or partial pooling. While DCs are willing to share all of their stock in the complete pooling [2] or complete network [20], partial pooling transship only when its inventory level is above a fixed threshold [21].

Among many research results, Wong et al. [22] proposed a simple and efficient solution procedure to obtain a nearoptimal solution with lateral transshipments for the singleitem problem. The model is further extended to the case with multiple locations. Bhatnagar and Lin [23] developed two heuristic policies for the multilocation production/inventory system problem with the joint transshipment and production control policies, in which items are manufactured and stocked at each location to meet the demand. Wei et al. [24] explored the impacts of lateral transshipments on the stability, bullwhip effect, and other performance measurements in the context of a two-tiered supply chain system composed of one supplier and two retailers. Zhao and Sen [25] presented a comparison of sample-path-based simulation and stochastic decomposition for multilocation transshipment problems considering one supplier and $N$ nonidentical retailers who face uncertain customer demands, as proposed by Herer et al. [3]. Archibald et al. [26] addressed a multiperiod periodic review model of a pair of retailers in which
lateral transshipments can occur during the period. Smirnov and Gerchak [27] presented the single-period circular unidirectional chain problem, assuming that a location cannot both receive and ship items. They analyzed the optimality of the problem and compared its performance to that of no pooling and "pooling among all nodes" (namely, "complete pooling"). Axsäter [28] addressed the unidirectional lateral transshipment problem where transshipment is allowed only in one direction. Tagaras [29] and Herer et al. [30] addressed the problem of group configuration where transshipment is allowed only within groups. Kukreja and Schmidt [31] presented a model for lumpy demand parts in a multisite inventory system with feeders using analytical and simulation techniques.

Wong et al. [32] proposed an integer programming model with a nonlinear objective function and nonlinear constraints for multi-item, multilocation spare part systems. Rim and Jiang [33] proposed a linear programming model for the simplified multi-item, multi-DC problem where transshipment is allowed only between pairs of DCs. The proposed LP model determines the optimal number of items and number of trucks to transship. The model also allows the "simultaneous chain transshipment," which enables distant locations to supply surplus to the location that faces shortage. This simultaneous chain transshipment of items can be a practical tool for the cases where surplus is not available in its vicinity, but surplus from far distant sites can reach the needing site by simultaneously moving the items in a supply chain.

## 3. Solution Procedure

In this section, we will present a genetic algorithm as a solution procedure for the TVRSPD. We do not attempt to develop a mathematical formulation for the TVRSPD since it will not be able to obtain the optimal solution within an acceptable length of time for the problems of realistic size, as TVRSPD is NP-hard. Among many, we chose genetic algorithm (GA) since one of the features of our problem that each truck must return to its original location can be suitably modeled by designing the chromosomes in GA, as mentioned in the first paragraph of Section 3.2.
3.1. Framework of the Genetic Algorithm. Genetic algorithm is a heuristic search based on the theory of biological evolution, Darwinism. By continuously generating the candidate solutions, evaluating how well the solutions fit with the desired objective, and refining the best solutions, GA obtains a good answer to the large search space problems. Depending on the problem type, the chromosome is designed to operate in the most appropriate way for the GA engine. At the crossover operator, two chromosomes are randomly selected and mated at each generation; and at the mutation operator, one of the best chromosomes in the current population is selected and mutated. The two parent chromosomes generate two child chromosomes using the crossover operator, and then only the best chromosomes enter the next generation. To boost the diversity of the
population, a mutation procedure is applied. Only the following three individuals advance to the next iteration: one current best, one from crossover, and one from mutation. The rest of $N-3$ chromosomes are randomly generated to form the next population of size $N$. The GA engine stops when $M$ generations have been iterated or any stopping rule applies.

Let $n^{\circ}$ and $n$ denote the population size and the number of feasible chromosomes in the population, respectively. The following steps represent the basic framework of the proposed genetic algorithm:

$$
n=0
$$

Step 1: if $n=n^{\circ}$, go to Step 3; else, randomly generate a chromosome.

Step 2: determine the optimal pickup/delivery amount at each DC for all routes in the chromosome. If volumewise and distance-wise feasible, add it to the population, $n=n+1$; else, discard it.
Go to Step 1.
Step 3: randomly select two parent chromosomes from the population.
Step 4: execute the crossover to create two child chromosomes; choose the better one (c1), and discard the other. Select the best chromosome (c2) in the current $n^{\circ}$ chromosomes, and execute the mutation to obtain a new chromosome (c3). Update c2 as the current best chromosome.
Step 5: if any stopping rule applies, stop; else, initialize the population, and put $\mathrm{c} 1, \mathrm{c} 2$, and c 3 in the population. Set $n=3$; go to Step 1 .

Stopping rules include the number of iterations or computation time limit.
3.2. Generating a Chromosome. In Step 1, to randomly generate a chromosome which represents a set of routes, we first select a set of promising DCs as the "starting" DCs so that the remaining DCs are defined as the intermediate DCs and arrange the selected starting DCs in an arbitrary sequence. Since trucks must return to their starting DC, we duplicate each starting DC and randomly assign the intermediate DCs in the space between the pair of identical starting DCs. Figure 2 shows an example of a chromosome that represents the routes shown in Figure 1, in which DCs 3, 5,7 , and 9 are selected as the starting DCs. Note that some starting DCs may have no intermediate DCs, as DC 7 in Figure 2.

To describe the heuristic algorithm to select a set of starting DCs, we will use the notations given in Table 1.

Figure 3 illustrates the meaning of $I$ and $S$ in the daily operation schedule of DCs [33].

In selecting the starting DCs, it is desirable to select DCs that have larger value of weighted maximum possible transshipment. Let $\mathbf{X}^{-}=\left\{x_{i k}^{-}\right\}$denote the inventory-shortage matrix before transshipment, where $x_{i k}^{-}=i_{i k}^{0}-d_{i k}$. Note that $x_{i k}^{-}=, I_{i k}^{-}$if $x_{i k}^{-}>0$, and $x_{i k}^{-}=-S_{i k}^{-}$if $x_{i k}^{-} \leq 0$. Similarly, let $\mathbf{X}^{+}=\left\{x_{i k}^{+}\right\}$denote the inventory-shortage matrix after


Figure 2: An example of a chromosome representing three routes shown in Figure 1.
transshipment is completed. We propose a heuristic algorithm to select a set of starting DCs as follows:
(1) Compute $\mathbf{X}^{-}=\mathbf{I}^{0}-\mathbf{D}$
(2) Compute the maximum possible transshipment matrix $\mathbf{Z}=\left\{z_{i k}\right\}$, where $z_{i k}=\min \left\{I_{i k}^{-}, \sum_{j \neq i} S_{j k}^{-}\right\} \quad$ if $x_{i k}^{-}>0$; or 0 if $x_{i k}^{-} \leq 0$
(3) Compute $\mathbf{W}=\left\{w_{i}\right\}$, where $w_{i}=\sum_{k} p_{k} z_{i k}$
(4) Select $m$ largest $w_{i}$ values to determine the corresponding DCs as the set of starting DCs

We suggest that the value $m$ can be arbitrarily selected as an integer close to $0.4 Y$. Table 2 shows an example of ten DCs and seven items to illustrate the above procedure. For $m=4$, DCs $3,5,7$, and 9 are selected as the starting DCs, as already shown in Figures 1 and 2.
3.3. Crossover. We randomly select two parent chromosomes from the current population to generate two child chromosomes by the crossover operator. We use the partial mapped crossover (PMX, [34]) method for the crossover operator. It constructs the offspring by randomly selecting two crossing sites. The two points are used to define the mapping relationship. The numbers that follow the relationship represent which numbers are swapped in the parent chromosomes. The example shown in the following will help to explain our special PMX feature in detail. Unlike the conventional way of PMX, we allow "null" as a gene. The proposed crossover operator is as follows:
(1) Randomly select two blocks of adjacent intermediate DCs of any length ( $0,1,2, \ldots$ ) from each parent chromosome. Note that the lengths of the two blocks do not have to be equal. A length of zero means that no intermediate DC is selected.
(2) Exchange these two blocks to create two offsprings.
(3) Determine the mapping relationship of the two offsprings.
(4) Validate the two offsprings.
(a) If a gene appears twice in a chromosome, then change one of the two genes according to the mapping relationship
(b) If a gene is missing (this happens when two blocks of unequal length are selected in Step (1)), then insert the missing gene at a random location in the chromosome according to the mapping relationship
To illustrate case (a), in Figure 4, Step 1 selects $(8,6)$ and $(4,1)$ in parents 1 and 2 , respectively. Step 2 creates children 1 and 2 . Step 3 determines the mapping relationship, where number 8 is exchanged with 4 and 6 with 1 . As the genes of child 1 in Step 2 are examined, numbers 4 and 1 are found to

Table 1: Notations.

| $k$ | Item $(k=1,2, \ldots, K)$ |
| :--- | :---: |
| $i, j$ | Distribution center $(i, j=1,2, \ldots, Y)$ |
| $U_{k}$ | Order-up-to-level of item $k$ at all DCs |
| $\mathbf{I}^{0}=\left\{i_{i k}^{0}\right\}$ | Initial inventory matrix of a day |
| $\mathbf{D}=\left\{d_{i k}\right\}$ | Daily demand matrix of a day |
| $p_{k}$ | Unit price of item $k$ |
| $I_{i k}^{-}, S_{i k}^{-}$ | Inventory and shortage, respectively, of item $k$ at DC $i$ before transshipment after assigning on-hand inventory to the daily |
| $I_{i k}^{+}, S_{i k}^{+}$ | demand |



Figure 3: Daily operation schedule and corresponding inventory/shortage notations.

Table 2: An example of selecting a set of starting DCs out of 10 DCs with 7 items.

| $k$ | $v_{k}$ | $p_{k}$ | 1 | 2 | 3 | 4 | 5 | DC | 6 | 7 | 8 | 9 |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
| 1 | 0.3 | 300 | -20 | 15 | 0 | -5 | -3 | 10 | -5 | -5 | 4 | -14 |
| 2 | 0.8 | 500 | 0 | 3 | 6 | 10 | 15 | -4 | 8 | 20 | -2 | 3 |
| 3 | 0.04 | 50 | -28 | 4 | 64 | 1 | 49 | -6 | 29 | 6 | 25 | -25 |
| 4 | 0.5 | 50 | -10 | -10 | 20 | 31 | 15 | -6 | 27 | 15 | 49 | -38 |
| 5 | 0.35 | 200 | -22 | -10 | 28 | -2 | 15 | -12 | 59 | -38 | 43 | 30 |
| 6 | 0.2 | 100 | 0 | 2 | -10 | 23 | 10 | 14 | 42 | 15 | 40 | 20 |
| 7 | 0.4 | 80 | 10 | 25 | -5 | 38 | 54 | 54 | 8 | -17 | -10 | 14 |
|  |  |  | 0 | 15 | 0 | 0 | 0 | 10 | 0 | 0 | 4 | 0 |
|  |  | 0 | 3 | 6 | 0 | 15 | 0 | 8 | 10 | 0 | 3 |  |
|  |  | 0 | 4 | 59 | 1 | 49 | 0 | 29 | 6 | 25 | 0 |  |
|  |  | 0 | 0 | 20 | 31 | 15 | 0 | 27 | 15 | 49 | 0 | $\mathbf{Z}$ |
|  |  | 0 | 0 | 28 | 0 | 15 | 0 | 59 | 0 | 43 | 30 |  |
|  |  | 0 | 2 | 0 | 10 | 10 | 10 | 10 | 10 | 10 | 10 |  |

appear twice. So, in Step 4, we swap genes 4 and 8 and genes 1 and 6 to obtain two valid children.

To illustrate case (b), in Figure 5, Step 1 selects "null" gene between two 7 s and $(4,1)$ from parents 1 and 2 , respectively. Step 2 creates children 1 and 2 . Step 3 determines the mapping relationship, where genes 4 and 1 are exchanged with the null gene, respectively. In Step 4, as the genes of child 1 are examined, genes 4 and 1 are found to appear twice, so we change numbers 4 and 1 to the null gene. In child 2 , genes 4 and 1 are missing, so we insert genes 4 and 1 at random locations.
3.4. Mutation. Among many ways to implement mutation (see Soni and Kumar [35] for comprehensive reference), we use the following approach, as shown in Figure 6:
(1) Select the current best chromosome in the population
(2) Randomly select two genes in the chromosome, which can be either intermediate DC or "null" between the two adjacent identical starting DCs
(3) Swap the two selected genes to form a new chromosome


Figure 4: An example of validating two offsprings when a number appears twice.


Step 2

| 3 | 3 | 5 | 10 | 2 | 5 | 7 | 8 | 7 | 9 | 6 | 9 |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |
| Child 2 |  |  |  |  |  |  |  |  |  |  |  |


$4 \leftrightarrow$ "null"
Step 3


| 3 | 10 | 3 | 5 | 8 | 6 | 5 | 7 | 4 | 1 | 7 | 9 | 2 | 9 | Child 1 |
| :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- |



FIGURE 5: An example of validating two offsprings when some genes are missing.


Figure 6: An example of the mutation operator (current best chromosome, new chromosome).
3.5. Optimal Pickup/Delivery Amount of a Route. In order to evaluate the fitness of a chromosome, we present an algorithm to determine the optimal pickup/drop quantity of each item at every visit in each route defined in a chromosome as follows. Let $v_{0}$ and $v_{k}$ denote the volume capacity of a truck and unit volume of item $k$, respectively. A chromosome is volume-wise feasible if the trucks of all routes defined in the chromosome satisfy the volume constraint at all times during the trip since pickup is allowed during the trip. To illustrate, consider the first route 3-10-8-1-3 in Figure 2. The corresponding seven columns of $\mathbf{X}^{-}$are shown in Table 3. Let $\mathbf{Q}^{*}=\left\{q_{i k}^{*}\right\}$ denote the quantity of item $k$ in the truck of a route after finishing drop and pickup at DC $i$. Once $\mathbf{Q}^{*}$ is obtained, the chromosome is volume-wise feasible if $v_{r}=$ $\sum_{k} q_{r k}^{*} v_{k} \leq v_{0}$ for all DCs $r$ for $1 \leq r \leq R$ (where $R$ is the number of DCs in the route including the ending DC) and for all routes defined in the chromosome. Table 3 shows an example where route $3-10-8-1-3$ is volume-wise feasible, when $v_{0}=32 \mathrm{~m}^{3}$.

Before the numerical example in Table 3, the flowchart in Figure 7 will help understand the logic of the algorithm. In computing $\mathbf{Q}^{*}$, we need a temporary matrix $\mathbf{Q}$ for the backward computation. Once $\mathbf{Q}$ is obtained by the heuristic algorithm described in Figure 7, then $\mathbf{Q}$ is forward adjusted to obtain valid $\mathbf{Q}^{*}$. Recall that $R$ denotes the number of DCs in a route including the ending DC. Let $g$ denote the first index such that $x_{g k}^{-}>0$ for $g=1,2, \ldots, R$. Since excessive quantity loaded in the truck will unnecessarily increase the handling cost, each truck must have no leftover in it when it returns to its starting DC, that is, $q_{R k}=0$ for all $k$, for all routes.

Starting from the ending node, $q_{i k}$ is calculated backwardly as $q_{i k}=\max \left(0, q_{i+1, k}-x_{i+1, k}^{-}\right)$. To calculate the total volume of items in the truck at each DC, we consider each item's status for all DCs in the route. In order to not carry too much from the starting DC to transship (it may violate the volume constraint), we only load enough amount of each item that is already calculated based on the next DCs' status. The following cases illustrate how $\mathbf{Q}^{*}$ is determined using $\mathbf{X}^{-}$and $\mathbf{Q}$ under various situations.
(1) Item 1: $q_{i 1}^{*}=0$ for all $i$ since $x_{i 1}^{-} \leq 0$ for all $i$
(2) Item 2: $q_{i 2}^{*}=0$ for all $i$ since $x_{i 2}^{-}>0$ for all $i$
(3) Item 3: $q_{53}=0, q_{43}=\max \left(0, q_{53}-x_{53}^{-}\right)=\max (0$, $0-64)=0, q_{33}=\max \left(0, q_{43}-x_{43}^{-}\right)=\max (0,0+28)=$ 28, $q_{23}=\max \left(0, q_{33}-x_{33}^{-}\right)=\max (0,28-6)=22$, and

Table 3: An illustrative example of route 3-10-8-1-3.

$q_{13}=\max \left(0, q_{23}-x_{23}^{-}\right)=\max (0,22+25)=47<x_{13}^{-}$, so it is valid; we set $q_{i 3}^{*}=q_{i 3}$ for all $i$
(4) Item 4: calculate backwardly as same as item 3, and we found that $q_{14}=\max \left(0, q_{24}-x_{24}^{-}\right)=\max (0$, $0+38)=38>x_{14}^{-}$and $x_{14}^{-}>0$ and $q_{14}=-x_{24}^{-}=38$, so it is invalid. Then, we only change $q_{14}^{*}=x_{14}^{-}=20$ and keep $q_{i 4}^{*}=q_{i 4}$ for $2 \leq i \leq R$.
(5) Item 5: calculate backwardly, and we have $q_{15}=\max$ $\left(0, q_{25}-x_{25}^{-}\right)=\max (0,60-30)=30>x_{15}^{-}$and $x_{15}^{-}>0$ and $q_{15} \neq-x_{25}^{-}$, so it is invalid. Using forward adjustment $\mathrm{A}, h=4, q_{15}^{*}=x_{15}^{-}=28, q_{25}^{*}=\max \quad(0$, $\left.q_{15}^{*}+x_{25}^{-}\right)=\max \quad(0,28+30)=58, \quad q_{35}^{*}=\max \quad(0$, $\left.q_{25}^{*}+x_{35}^{-}\right)=\max (0,58-38)=20$, and $q_{45}^{*}=\max (0$, $\left.q_{35}^{*}+x_{45}^{-}\right)=\max (0,20-22)=0$, and keep $q_{i 5}^{*}=q_{i 5}$ for $h<i \leq R$.
(6) Item 6: $q_{16}=0$ since $x_{16}^{-}<0$ and $g=2$ and $q_{26}<x_{26}^{-}$, so no further changes are needed.
(7) Item 7: $q_{17}=3$ since $x_{17}^{-}<0$ and $g=2$ and $q_{27}>x_{27}^{-}$ ( $17>14$ ), so it is invalid. Using forward adjustment B, set $q_{17}^{*}=0$ (all values before the first positive value of $x_{i 7}^{-}$are set to be zero) $, b=2, q_{27}^{*}=\max (0$, $\left.q_{17}^{*}+x_{27}^{-}\right)=\max (0,0+14)=14$, and $q_{37}^{*}=\max (0$, $\left.q_{27}^{*}+x_{37}^{-}\right)=\max (0,14-17)=0$, and keep the other $q_{i 7}^{*}$ for $(b+g) \leq i \leq R$.
The pseudo-codes of forward adjustment A and B are given in Appendix. Using $\mathbf{v}=\left\{v_{k}\right\}$, the array of unit volume of item $k$, we obtain $\mathbf{A}=\left\{a_{i}\right\}$, the array of the total volume at $\mathrm{DC} i$ after pickup and drop as $\mathbf{A}=\mathbf{v} \mathbf{Q}^{*}$. For example, for $\mathbf{v}=\{0.3,0.8,0.04,0.5,0.35,0.2,0.4\}, a_{i} \leq v_{0}=32 \mathrm{~m}^{3}$ for all $i$, as shown in Table 3; the route in Table 3 is volume-wise feasible. If all routes defined in a chromosome are volumewise feasible, then the chromosome is volume-wise feasible. If not, the infeasible chromosome is discarded.

Regarding the distance constraint of a route, the total distance, $d$, of a given route can be readily computed by adding all Euclidean (straight line) distances between adjacent DCs in the route, given the coordinates of DCs. Let $d_{0}$ denote the distance limit of the routes due to the limited time for transshipment. Ignoring the handling time, a route is distance-wise feasible if $d \leq d_{0}$. A chromosome is distancewise feasible if all its routes are distance-wise feasible.
3.6. Fitness Function. To define the fitness function for a feasible chromosome, we use the cost parameters given in Table 4.

Shortage cost is the sum of shortage after the transshipment is completed ( $S_{i k}^{+}$) multiplied by its price and shortage cost rate. Trucking cost incurs proportionally to the total distance of the route, regardless of the loaded quantity. Extra material handling for transshipment includes picking, loading, and unloading. We assume that the unit handling $\operatorname{cost} \theta$ for any item at any DC is equal. Note that $\theta$ includes both one loading and one unloading operation, as a loaded item will be eventually unloaded somewhere in the route. Let $\mathbf{H}=\left\{h_{i k}\right\}$ denote the quantity of item $k$ additionally loaded in the truck at DC $i$ (see Table 3). Then, $h_{1 k}=q_{1 k}^{*}$; and for $2 \leq i \leq R, h_{i k}=\max \left(0, q_{i k}^{*}-q_{(i-1) k}^{*}\right)$, for all $k$. Then, the total handling cost is $\theta \sum_{i} \sum_{k} h_{i k}$.

Let $t_{1}$ and $t_{2}$ denote the number of routes (having at least one intermediate DC) and isolated DCs (having no intermediate DC ), respectively. Let $R_{s}$ and $d_{s}$ denote the number of DCs and total travel distance of route $s$, respectively. Now, we define the fitness function $Z$ of a chromosome as equation (1). Note that the last term representing inventory cost is a constant and can be eliminated.

$$
\begin{align*}
\mathrm{Z}= & \sum_{s=1}^{t_{1}}\left(\theta \sum_{i=1}^{R_{s}} \sum_{k=1}^{K} h_{i k}+\delta d_{s}\right)+\sum_{s=1}^{t_{1}+t_{2}}\left(\gamma \sum_{i=1}^{R_{s}} \sum_{k=1}^{K} p_{k} S_{i k}^{+}\right)  \tag{1}\\
& +\beta Y \sum_{k=1}^{K} p_{k} U_{k} .
\end{align*}
$$

## 4. Performance Evaluation

4.1. Parameters. To evaluate the average performance of the proposed approach by simulation, we use the realistic cost parameters as shown in Table 5. Demand for item $k$ of all DCs is assumed to follow normal distribution $N\left(\mu_{k}, \sigma_{k}^{2}\right)$, where $\mu_{k}$ follows uniform $(10,100)$ and $\sigma_{k}=0.3 \mu_{k}$. The order-up-to-level for item $k$ is determined as $U_{k}=\mu_{k}+z_{\alpha} \sigma_{k}$, where $z_{\alpha}$ is the coefficient from the standard normal distribution table with in-stock probability $\alpha$.
4.2. Average Performance. For a given inventory level, we compare the average total cost when transshipment is not applied against the one when transshipment is applied. The former will suffer substantial shortage cost, and the latter will have less shortage cost but additional trucking and handling cost. Using target in-stock probability $\alpha$ and corresponding $U_{k}=\mu k+\sigma k z_{\alpha}$, daily demand and resulting $\mathbf{X}^{-}$are generated. With a population size of 100 , stopping rule of 500 iterations, and 20 replications, Figures $8-10$ show dramatic average cost savings by using the proposed approach. The cost saving is larger as $\alpha$ is smaller, $\gamma$ is larger, and $\delta$ is smaller. For a practical value of $\alpha=0.95$, the average total cost is saved by $30.8 \%$ to $63.3 \%$. Even for very high $\alpha=99 \%$, the average total cost saving ranges between $9.7 \%$ and $28.68 \%$, which is significant. The item fill rate is $100 \%$ in most cases as transshipment is applied.
4.3. Computation Time. In order for the proposed GA to be practical, the computation time has to be within a reasonable range. Using MATLAB R2019a on a personal computer with 16.0 GB RAM and base speed 3.60 GHz , Table 6 shows that, for 10 DCs and selected 100 items for transshipment, it takes less than 6 minutes, which can be further reduced if the number of iterations, population size, and/or the number of items are reduced.


Figure 7: Flowchart to determine the optimal pickup/delivery amount.

Table 4: Cost parameter notations.

| $\beta$ | Inventory cost rate |
| :--- | :---: |
| $\gamma$ | Shortage cost rate |
| $\delta$ | Trucking cost per unit distance |
| $\theta$ | Handling cost to load and unload an item |

Table 5: Parameters used for simulation with 10 DCs and 10 items.

|  | $k$ |  |  |  |  |  |  |  |  |  |
| :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: | :---: |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| $v_{k}$ | 0.3 | 0.8 | 0.4 | 0.04 | 0.05 | 0.2 | 0.5 | 0.3 | 0.45 | 0.7 |
| $p_{k}$ | 300 | 500 | 200 | 50 | 300 | 150 | 50 | 400 | 100 | 200 |
|  | DC |  |  |  |  |  |  |  |  |  |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Coordinate (units of 5 km ) | $(2,5)$ | $(6,4)$ | $(4,3)$ | $(8,1.5)$ | $(5,6)$ | $(8,5)$ | $(10,3.5)$ | $(1,3)$ | $(2,0)$ | (5.5, 1.5) |
| $p_{k}$ | $U(\$ 10 \sim \$ 500)$ |  |  |  |  |  |  |  |  |  |
| $v_{k}$ | $U\left(0.03 \mathrm{~m}^{3} \sim 1 \mathrm{~m}^{3}\right)$ |  |  |  |  |  |  |  |  |  |
| $v_{0}$ | $32 \mathrm{~m}^{3}$ |  |  |  |  |  |  |  |  |  |
| $\beta$ | $0.25 / 365=0.00068$ per day |  |  |  |  |  |  |  |  |  |
| $\theta$ | \$0.5 |  |  |  |  |  |  |  |  |  |
| $d_{0}$ | 100 km |  |  |  |  |  |  |  |  |  |
| $\gamma$ | $0.1,0.2$, and 0.3 per shortage |  |  |  |  |  |  |  |  |  |
| $\delta$ | \$0.5, \$1, and \$2 per km |  |  |  |  |  |  |  |  |  |



Figure 8: The average total cost saving by transshipment when $\delta=\$ 0.5 / \mathrm{km}$.


Figure 9: The average total cost saving by transshipment when $\delta=\$ 1.0 / \mathrm{km}$.


Figure 10: The average total cost saving by transshipment when $\delta=\$ 2 / \mathrm{km}$.

Table 6: Sample computation times.

| Number of items | Computation time (sec) | Number of items | Computation time (sec) |
| :--- | :---: | :---: | :---: |
| 10 | 87.55 | 80 | 337.10 |
| 20 | 147.77 | 90 | 369.74 |
| 30 | 185.73 | 100 | 407.62 |
| 40 | 206.00 | 200 | 802.87 |
| 50 | 233.58 | 300 | 1123.45 |
| 60 | 266.63 | 400 | 1291.46 |
| 70 | 300.84 | 500 | 1825.83 |

## 5. Conclusion

In this paper, we proposed a solution procedure for the transshipment vehicle routing problem for cross-filling multiple items among multiple DCs, where simultaneous pickup and delivery is allowed in the routes (TVRSPD). Any DC can be selected as a starting DC, and trucks must return to their starting DC. We present an algorithm to determine the optimal amount of each item to be picked up and/or delivered at each DC in the route. Volume feasibility of the truck is considered in the algorithm. Distance feasibility is also considered in the stage of generating chromosomes. Trucking cost is assumed to be proportional to the travel distance of the truck.

Instead of maintaining higher level of inventory to avoid shortage, cross-filling effectively reduces the shortage with relatively lower level of inventory. For practical parameter values, the proposed approach obtains significant saving of 9.7\% to $63.3 \%$ in the average total cost, which includes the shortage cost and extra trucking and handling cost. Computation time required for the proposed algorithm is acceptable for the problem of reasonable size such as 10 DCs and 100 items.

Despite its successful GA modeling, our approach has the following limitations: the number of items for transshipment is limited to about one hundred major items due to computational time; truck drivers may not be available around the midnight for transshipment; or DCs may be too dispersed in the rural area to apply this model.

In the near future, advanced transportation devices such as drones and driverless trucks will make our model practically more useful. Furthermore, a new distribution channel called microfulfillment center, which is relatively small but automated distribution centers densely located in the urban area to serve consumers in shorter response time, will promote implementing our model to exchange the shortage items among the microfulfillment centers. For further research, one may allow each DC to be visited more than once by trucks from more than one starting DC.

## Appendix

## A. Pseudo-Code of Forward Adjustment A

$$
\begin{aligned}
& \text { for } i=2 \text { to } R-1 \\
& q_{1 k}^{*}=x_{1 k}^{-} \\
& \text {if } x_{i-1, k}^{-}<0 \text { and } x_{i k}^{-}>0 \text { and } x_{i+1, k}^{-}<0 \\
& q_{i k}^{*}=\min \left(\left|x_{i+1, k}^{-}\right|, x_{i k}^{-}\right)
\end{aligned}
$$

$$
q_{i+1, k}^{*}=\max \left(0, q_{i k}^{*}+x_{i+1, k}^{-}\right)
$$

else
Find the smallest $h$ where $q_{h k}=0$

$$
\text { for } j=2 \text { to } h
$$

$$
q_{j k}^{*}=\max \left(0, q_{j-1}^{*}+x_{j k}^{-}\right)
$$

end for
end if
Find the largest $s$ where $x_{s k}^{-}<0$; set $q_{s k}^{*}=0$.
end for

## B. Pseudo-Code of Forward Adjustment B

$q_{i k}^{*}=0$ for $i=1, \ldots, g-1$
For $b=g, \ldots, R$, find the smallest $b$ such that $q_{b k}=0$
for $i=1$ to $b$
$q_{i+g-1, k}^{*}=\max \left(0, q_{i+g-2, k}^{*}+x_{i+g-1, k}^{-}\right)$
end for

## Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

## Conflicts of Interest

The authors declare that they have no conflicts of interest.

## Acknowledgments

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) Funded by the Ministry of Education (2017R1D1A1B03032829). The authors are grateful to Dr. Si-Hyeon Paik for his advices in designing the genetic algorithm.

## References

[1] S.-C. Rim and I.-S. Park, "Order picking plan to maximize the order fill rate," Computers \& Industrial Engineering, vol. 55, no. 3, pp. 557-566, 2008.
[2] C. Paterson, G. Kiesmüller, R. Teunter, and K. Glazebrook, "Inventory models with lateral transshipments: a review," European Journal of Operational Research, vol. 210, no. 2, pp. 125-136, 2011.
[3] Y. T. Herer, M. Tzur, and E. Yücesan, "The multilocation transshipment problem," IIE Transactions, vol. 38, no. 3, pp. 185-200, 2006.
[4] G. B. Dantzig and J. H. Ramser, "The truck dispatching problem," Management Science, vol. 6, no. 1, pp. 80-91, 1959.
[5] P. Toth and D. Vigo, "The vehicle routing problem," SIAM Monographs on Discrete Mathematics and Applications, SIAM, Philadelphia, PA, USA, 2002.
[6] H. Min, "The multiple vehicle routing problem with simultaneous delivery and pick-up points," Transportation Research Part A: General, vol. 23, no. 5, pp. 377-386, 1989.
[7] G. Berbeglia, J.-F. Cordeau, I. Gribkovskaia, and G. Laporte, "Static pickup and delivery problems: a classification scheme and survey," Top, vol. 15, no. 1, pp. 1-31, 2007.
[8] C. Koc, G. Laporte, and I. Tukenmez, "A review of vehicle routing with simultaneous pickup and delivery," Computers and Operations Research, vol. 122, Article ID 104987, 2020.
[9] H. Min, J. Current, and D. Schilling, "The multiple depot vehicle routing problem with backhauling," Journal of Business Logistics, vol. 13, no. 1, pp. 259-271, 1992.
[10] G. Nagy and S. Salhi, "Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries," European Journal of Operational Research, vol. 162, no. 1, pp. 126-141, 2005.
[11] G. Laporte, "Location-routing problems," Technical report, MU/HEC-8701:MICROLOG-88-00022, Montreal Univ., Ecole des Hautes Etudes Commerciales, Centre de Recherche sur les Transports, Montreal, Canada, 1987.
[12] Z. Zhang, B. Cheang, C. Li, and A. Lim, "Multi-commodity demand fulfillment via simultaneous pickup and delivery for a fast fashion retailer," Computers \& Operations Research, vol. 103, pp. 81-96, 2019.
[13] A. S. Tasan and M. Gen, "A genetic algorithm based approach to vehicle routing problem with simultaneous pick-up and deliveries," Computers \& Industrial Engineering, vol. 62, no. 3, pp. 755-761, 2012.
[14] L. Pradenas, B. Oportus, and V. Parada, "Mitigation of greenhouse gas emissions in vehicle routing problems with backhauling," Expert Systems with Applications, vol. 40, no. 8, pp. 2985-2991, 2013.
[15] N. Wassan and G. Nagy, "Vehicle routing problem with deliveries and pickups: modelling issues and meta-heuristics solution approaches," International Journal of Transportation, vol. 2, no. 1, pp. 95-110, 2014.
[16] S. Ropke and D. Pisinger, "A unified heuristic for a large class of vehicle routing problems with backhauls," European Journal of Operational Research, vol. 171, no. 3, pp. 750-775, 2006.
[17] S.-C. Liu and C.-H. Chung, "A heuristic method for the vehicle routing problem with backhauls and inventory," Journal of Intelligent Manufacturing, vol. 20, no. 1, pp. 29-42, 2009.
[18] G. S. Brar and G. Saini, "Milk run logistics: literature review and directions," in Proceedings of the World Congress on Engineering, vol. 1, pp. 797-801, London, UK, July 2011.
[19] G. Van der Heide and K. J. Roodbergen, "Transshipment and rebalancing policies for library books," European Journal of Operational Research, vol. 228, no. 2, pp. 447-456, 2013.
[20] R. W. Lien, S. M. R. Iravani, K. Smilowitz, and M. Tzur, "An efficient and robust design for transshipment networks," Production and Operations Management, vol. 20, no. 5, pp. 699-713, 2011.
[21] T. W. Archibald, D. Black, and K. D. Glazebrook, "An index heuristic for transshipment decisions in multi-location
inventory systems based on a pairwise decomposition," European Journal of Operational Research, vol. 192, no. 1, pp. 69-78, 2009.
[22] H. Wong, D. Cattrysse, and D. Van Oudheusden, "Inventory pooling of repairable spare parts with non-zero lateral transshipment time and delayed lateral transshipments," European Journal of Operational Research, vol. 165, no. 1, pp. 207-218, 2005.
[23] R. Bhatnagar and B. Lin, "The joint transshipment and production control policies for multi-location production/ inventory systems," European Journal of Operational Research, vol. 275, no. 3, pp. 957-970, 2019.
[24] Y. Wei, F. Chen, and F. Xiong, "Dynamic complexities in a supply chain system with lateral transshipments," Complexity, vol. 2018, Article ID 3959141, 15 pages, 2018.
[25] L. Zhao and S. Sen, "A comparison of sample-path-based simulation-optimization and stochastic decomposition for multi-location transshipment problems," in Proceedings of the 38th Conference on Winter Simulation, pp. 238-245, Monterey, CA, USA, December 2006.
[26] T. W. Archibald, S. A. E. Sassen, and L. C. Thomas, "An optimal policy for a two depot inventory problem with stock transfer," Management Science, vol. 43, no. 2, pp. 173-183, 1997.
[27] D. Smirnov and Y. Gerchak, "Inventory sharing via circular unidirectional chaining," European Journal of Operational Research, vol. 237, no. 2, pp. 474-486, 2014.
[28] S. Axsäter, "Evaluation of unidirectional lateral transshipments and substitutions in inventory systems," European Journal of Operational Research, vol. 149, no. 2, pp. 438-447, 2003.
[29] G. Tagaras, "Pooling in multi-location periodic inventory distribution systems," Omega, vol. 27, no. 1, pp. 39-59, 1999.
[30] Y. T. Herer, M. Tzur, and E. Yücesan, "Transshipments: an emerging inventory recourse to achieve supply chain leagility," International Journal of Production Economics, vol. 80, no. 3, pp. 201-212, 2002.
[31] A. Kukreja and C. P. Schmidt, "A model for lumpy demand parts in a multi-location inventory system with transshipments," Computers \& Operations Research, vol. 32, no. 8, pp. 2059-2075, 2005.
[32] H. Wong, G. J. van Houtum, D. Cattrysse, and D. V. Oudheusden, "Multi-item spare parts systems with lateral transshipments and waiting time constraints," European Journal of Operational Research, vol. 171, no. 3, pp. 1071-1093, 2006.
[33] S. C. Rim and J. J. Jiang, "Multi-item, multi-location transshipment model for cross-filling," in Proceedings of the International Conference on Industrial Engineering and Operations Management, Bangkok, Thailand, March 2019.
[34] D. Goldberg and R. Lingle, "Alleles, Loci and the traveling salesman problem," in Proceedings of the 1st International Conference on Genetic Algorithms and their Applications, pp. 154-159, Los Angeles, CA, USA, July 1985.
[35] N. Soni and T. Kumar, "Study of various mutation operators in genetic algorithms," International Journal of Computer Science and Information Technologies, vol. 5, no. 3, pp. 4519-4521, 2014.

