Transportation Science

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Cyclic Delivery Schedules for an Inventory Routing Problem

Ali Ekici, Okan Örsan Özener, Gültekin Kuyzu

TRANSPORTATION SCIENCE
Vol. 49, No. 4, November 2015, pp. 817–829
ISSN 0041-1655 (print) | ISSN 1526-5447 (online)

http://dx.doi.org/10.1287/trsc.2014.0538
© 2015 INFORMS

Cyclic Delivery Schedules for an Inventory Routing Problem

Ali Ekici, Okan Örsan Özener
Department of Industrial Engineering, Ozyegin University, Istanbul, 34794 Turkey
{ali.ekici@ozyegin.edu.tr, orsan.ozener@ozyegin.edu.tr}

Gültekin Kuyzu
Department of Industrial Engineering, TOBB University of Economics and Technology, Ankara, 06560 Turkey,
gkuyzu@etu.edu.tr

We consider an inventory routing problem where a common vendor is responsible for replenishing the inventories of several customers over a perpetual time horizon. The objective of the vendor is to minimize the total cost of transportation of a single product from a single depot to a set of customers with deterministic and stationary consumption rates over a planning horizon while avoiding stock-outs at the customer locations. We focus on constructing a repeatable (cyclic) delivery schedule for the product delivery. We propose a novel algorithm, called the *Iterative Clustering-Based Constructive Heuristic Algorithm*, to solve the problem in two stages: (i) clustering, and (ii) delivery schedule generation. To test the performance of the proposed algorithm in terms of solution quality and computational efficiency, we perform a computational study on both randomly generated instances and real-life instances provided by an industrial gases manufacturer. We also compare the performance of the proposed algorithm against an algorithm developed for general routing problems.

Keywords: inventory routing problem; iterative clustering; cyclic delivery schedules; heuristic algorithm
History: Received: June 2012; revisions received: April 2013; February 2014; accepted: April 2014. Published online in *Articles in Advance* August 25, 2014.

1. Introduction

The *Inventory Routing Problem* (IRP) arises from the situations where a common vendor is responsible for replenishing the warehouse(s) of one or more customers through a set of vehicle routes over a planning horizon. In this vendor-managed inventory setting, the objective of the vendor is to construct a *delivery schedule* that minimizes the system-wide total inventory and transportation costs of serving the customers over the planning horizon. A delivery schedule specifies a set of delivery routes to execute, their execution times, the customers on each route, and the delivery volume to each customer. The feasibility of a delivery schedule means allowing no shortage of product at any customer location while not violating the warehouse or vehicle capacities. The challenge about this distribution setting is to coordinate the routing decisions with the inventory decisions, and the vendor is expected to make decisions about the vehicle routes that serve the customers, the frequencies of the routes, and the volume delivered to each customer in these routes.

Unlike the *Vehicle Routing Problem* (VRP), in the IRP the decisions are made over a planning horizon, and the decisions in one period affect the decisions in the following periods. The objective is to construct feasible and efficient vehicle routes with specified delivery volumes to each customer while keeping the inventory levels at customers below the capacity limit and avoiding product shortage over the planning horizon. As the vendor has the flexibility to decide when to serve, on which route to visit, how much to deliver to each customer, the options become practically endless, and determining the best solution among them presents a formidable challenge.

Besides the planning horizon, there is a list of factors which affect the vendor's routing and delivery volume decisions such as the locations of the depot(s) and the customers, the size of the vehicle fleet, the capacities of the vehicles, the capacities of the warehouses, the consumption rate of each customer, and the time window restrictions on the deliveries. The presence of this many factors results in a variety of problem settings for the IRP, and hence, a number of solution methodologies can be found in the literature to handle each particular setting. However, in most cases, it is quite challenging to find the optimal solution to the IRP even for very small instances (Campbell et al. 1998).

In the IRP setting we consider, a common vendor supplies a single product from a single depot to several customers using a homogenous fleet of capacitated vehicles. We assume that the inventory holding costs are incurred by the customers. Thus, from the perspective of the vendor, the inventory management component affects the feasibility of the delivery schedule rather than its cost. We also assume that the consumption rates of the customers are deterministic and stationary over time. Moreover, customers deplete inventory on a continuous time basis which makes the exact timing of the delivery during a day important. As a result, the problem is reduced to minimizing the total cost of transportation of a single product from a single depot to a set of customers with deterministic and stationary consumption rates over a very long finite or infinite time planning horizon.

In this paper, we focus on a simplified version of the problem where each customer location is visited by only one vehicle. In real-life distribution systems, this is a common practice as it eliminates the unnecessary coordination effort for timing of the deliveries (Jung and Mathur 2007; Li, Chu, and Chen 2011; Zhao, Chen, and Zang 2008). Thus, in our problem setting, each customer is assigned to one vehicle only. Note that a customer can still be served by different delivery routes performed by the same vehicle. Finally, we focus on periodic routing strategies in which a schedule for a "short" planning horizon is constructed and this schedule is assumed to be repeated over the entire planning horizon (Anily and Federgruen 1990; Raa and Aghezzaf 2009; Viswanathan and Mathur 1997). This is a reasonable assumption for a real-life distribution strategy as having repeated delivery schedules may be desirable by the vendor and customers for planning and coordination purposes. A cyclic approach provides a more stable and predictable replenishment and delivery plan.

We propose a novel two-phase solution approach, called the *Iterative Clustering-Based Constructive Heuristic Algorithm* (IC-CH), for solving the IRP. In the first phase of our solution approach, we partition the customer set into clusters each of which is assigned to a vehicle. Moreover, each vehicle serves only a single cluster of customers. In the second phase of the proposed solution approach, we develop feasible and cost-efficient delivery schedules for each cluster. The main idea is using the algorithm proposed for the second phase of our solution approach in an iterative manner to find a "good" clustering of the customers. For the first phase, we propose a procedure called *Iterative Clustering* (IC). IC utilizes the *Approximate Stability Assessment* procedure (Özener, Ergun, and Savelsbergh 2013). For the second phase, we develop an algorithm, called the *Constructive Heuristic Algorithm* (CH), based on a constructive heuristic idea. CH finds a feasible solution using a construction subroutine and then improves the solution using an exact mixed integer programming formulation. Although our proposed solution approach solves the problem in two phases, we take an unusual approach by implementing it in an integrated manner in the sense that we also determine the delivery schedule while forming the clusters since we use the algorithm proposed for the second phase as a subroutine for the first phase.

The rest of the paper is organized as follows. In §2, we thoroughly review the related work in the literature. In §3, we provide a formal statement of the IRP and list our assumptions about the problem setting. We also present a simple example to illustrate the problem structure. In §4, we discuss our two-phase solution approach. We first describe the algorithm used in the clustering phase. Next, we discuss the algorithm proposed for constructing the delivery schedules. In §5, we computationally demonstrate how our algorithm performs in comparison to a lower bound and an algorithm developed for general routing problems in the literature. Concluding remarks are provided in §6.

2. Literature Review

The IRP has been studied by several authors in maritime transportation and ground transportation contexts. In maritime transportation, the problem is called the *Maritime Inventory Routing Problem* (MIRP), and it focuses on developing delivery plans for the transportation of chemical and petrochemical products between refineries and manufacturers. We refer the reader to Christiansen and Fagerholt (2009) and Ronen (2002) for a recent survey about the MIRP. In this section, we mainly focus on the prior work conducted in ground/truck transportation settings, and simply use the term *Inventory Routing Problem* (IRP) to refer to the work in this field.

Several variants of the IRP have been studied in the literature since the early works by Bell et al. (1983), Federgruen and Zipkin (1984), and Golden, Assad, and Dahl (1984). Recent surveys for the IRP are provided by Andersson et al. (2010), Bertazzi and Speranza (2012, 2013), and Moin and Salhi (2007). Although the main motivation behind the IRP is coordinating the inventory control and vehicle routing decisions, the models developed in the literature vary greatly in several aspects. The most important differences between several variants of the IRP are the objective function, the replenishment strategy, and the nature of the demand:

- *Objective Function*: A typical objective function includes vehicle routing and inventory related costs. Depending on the application, vehicle routing costs may include (i) a fixed cost that is incurred every time a vehicle is dispatched (Chan, Federgruen, and Simchi-Levi 1998; Zhao, Chen, and Zang 2008), (ii) variable routing cost that is based on the distance traveled (Bertazzi, Paletta, and Speranza 2002; Campbell and Savelsbergh 2004), (iii) fixed cost per stop at a customer location (Qu, Bookbinder, and Iyogun 1999;

Raa and Aghezzaf 2009), and (iv) fixed cost of a vehicle if the fleet size is a decision (Raa and Aghezzaf 2008). Inventory-related costs may include (i) inventory holding cost at the customer location (Anily and Federgruen 1990; Archetti et al. 2012) and/or central depot (Anily and Federgruen 1993; Archetti et al. 2012), (ii) fixed order/production cost if the product is procured from an external supplier or produced internally based on demand (Jung and Mathur 2007; Zhao, Chen, and Zang 2008), and (iii) shortage cost (Federgruen and Zipkin 1984; Kleywegt, Nori, and Savelsbergh 2004).

- *Replenishment Strategy*: Since the structure of the optimal solution in the IRP is very complex, most authors focus on specific classes of strategies and analyze the problem under these strategies: (i) *fixed partition policy* (FPP), where the set of customers is partitioned into a number of clusters such that each cluster is served independently (Bramel and Simchi-Levi 1995; Chan and Simchi-Levi 1998; Li, Chu, and Chen 2011), (ii) *zero inventory ordering policy*, where a customer is replenished if and only if its inventory is zero (Bertazzi, Chan, and Speranza 2007; Chan, Federgruen, and Simchi-Levi 1998), (iii) *power-of-two policy*, where each customer is replenished at a constant reorder interval which is a power-of-two multiple of a base planning period (Jung and Mathur 2007; Viswanathan and Mathur 1997), and (iv) *order-up-to-level policy*, where the customer's inventory is raised to its maximum level whenever visited (Archetti et al. 2012; Solyali and Sural 2011).
- *Nature of Demand*: First, variants of the IRP can be classified into two according to demand characteristics: (i) stochastic demand (Adelman 2004; Kleywegt, Nori, and Savelsbergh 2002), and (ii) deterministic demand (Archetti et al. 2012; Jung and Mathur 2007). Most of the papers in the literature assume that the demand in each period is satisfied instantly (Bertazzi, Paletta, and Speranza 2002; Solyali and Sural 2011). On the other hand, some authors assume that the demand is realized on a continuous time basis (Campbell and Savelsbergh 2004; Song and Savelsbergh 2007). Although the continuous demand assumption may be a more realistic assumption in certain settings, it complicates the problem since the *Traveling Salesman Problem* tour, which is the optimal solution for a given set of customers with discrete demand, may not be feasible for the continuous demand case.

In addition to the objective function, the replenishment strategy and the nature of the demand, differentiating aspects of variants of the IRP include (i) the length of the planning horizon (single versus multiple periods and rolling horizon framework versus cyclic policy), (ii) the shipping strategy (direct shipping versus multi-stop tours and single versus multiple daily visits), (iii) the availability of product (unlimited versus outsourcing/internal production), (iv) the number of products (single versus multiple), (v) the characteristics of the fleet (capacitated versus uncapacitated vehicles, single versus multiple vehicles and homogeneous versus heterogeneous vehicles), and (vi) the storage capacities (capacitated versus uncapacitated vendor/customers). Bertazzi and Speranza (2012, 2013) provide a very thorough introduction to the IRP with examples, present a similar classification of the IRP according to various characteristics including shipping times, planning horizon, replenishment strategy, and objective function, and provide a detailed discussion of direct shipping and multi-stop shipping strategies.

In the literature, most of the proposed algorithms decompose the problem into two stages: (i) inventory control (determining the delivery amounts), and (ii) vehicle routing (Campbell and Savelsbergh 2004; Dror, Ball, and Golden 1985/6; Federgruen and Zipkin 1984; Qu, Bookbinder, and Iyogun 1999). In some papers, the overall solution is found by iterating between these two problems (Federgruen and Zipkin 1984; Qu, Bookbinder, and Iyogun 1999).

In this paper, we study a variant of the IRP where a central depot with an unlimited supply of product replenishes multiple capacitated customers with a single product. We assume that there are multiple capacitated homogeneous vehicles and the demand/consumption rate at each customer is *continuous* and *deterministic*. The objective is to find a cyclic replenishment and delivery schedule that minimizes the transportation cost while avoiding stock-outs at the customers. We develop an FPP where the customers are partitioned into clusters, a single vehicle is assigned to each cluster, and each vehicle is assigned to a single cluster. In addition to reduced problem size, FPP is more practical since it is easy to implement from a managerial point of view. We allow multiple tours to visit the same customer, and we do not impose any frequency restrictions for the visits to a customer. Finally, each vehicle can make multiple tours in a day. In our solution approach, we handle the inventory and routing decisions in an integrated manner.

Among the variants of the IRP, the problem most related to ours is the one studied by Campbell and Savelsbergh (2004). Campbell and Savelsbergh (2004) analyze the same problem in a rolling planning horizon setting. They develop a two-phase solution approach in which different length planning horizons are used. In the first phase, they use a coarse approximation of the problem and assign customer deliveries to days in the longer planning horizon. To reduce the number of routes considered in this phase, they also propose an FPP. First, they generate a large set of possible clusters based on truck capacity utilization and proximity of customers. The cost of serving each cluster is estimated by an approximate integer program. Then, they solve a set partitioning problem over the generated clusters with the estimated costs. In the second phase, they solve a sequence of VRPs with time windows to construct the delivery routes for a shorter planning horizon and adjust the delivery amounts determined in the first phase as necessary. They utilize a *Greedy Randomized Adaptive Search Procedure*-based customized insertion heuristic to solve the VRP. They measure the quality of the solutions with statistics such as volume delivered per mile and truck utilization. Although these are good performance indicators, they are not absolute due to several factors such as the storage capacities, the locations, and the consumption rates of the customers. Song and Savelsbergh (2007) analyze the same problem and develop a linear programming-based method to calculate lower bounds that can be used to evaluate the effectiveness of delivery strategies.

3. Problem Definition

In this section, we provide a formal definition of the IRP setting under consideration. We define the problem on a Euclidean graph $G = (V_0, E)$, where $V_0 = \{0, 1, 2, \ldots, n\}$ is the set of customer locations ($V = \{1, 2, \ldots, n\}$), and the depot (0), E is the set of edges connecting the nodes in V_0. The cost of traveling along an edge (i, j) is denoted by c_{ij}, and the travel time (in hours) along the same edge is denoted by t_{ij}. The cost of traveling between two locations is equal to the distance between the locations. The customers are served a single product from a single depot with unlimited supply and storage capacity over an infinite time planning horizon. Customers operate K hours per day and continuously deplete inventory at a rate of q_i units per day during their hours of operation. Customer i has a storage capacity C_i, which cannot be exceeded any time, and customer stock-outs are not allowed.

There are M identical vehicles, each of which has a capacity of Q units, available to deliver the product to the customers. The vehicles make deliveries to the customers via routes that start and end at the depot, and they operate according to the same schedule (for a time period of length K hours per day) as the customers. The vehicles must return to the depot before the end of the daily operating hours. Hence, the total duration of the routes of a vehicle on a given day cannot exceed K hours. We assume that the vehicles move at a constant speed of p distance units per hour; i.e., $t_{ij} = c_{ij}/p$.

We are interested in developing a periodic/cyclic delivery schedule repeated every T days, where customer i starts with initial inventory $\mathcal{I}_i^0$ and ends with the same amount of inventory at the end of this T-day cycle. In our setting, T is given, and our objective is to identify a cyclic and no stock-out delivery schedule under an FPP with minimum daily transportation cost. Note that a feasible solution for the problem where the length of the planning horizon is a submultiple of T is also a feasible solution for the original problem. Hence, we run our algorithm for both T and submultiples of T and report the best solution.

During the clustering stage, we generate "good" clusters and select a subset of these clusters to form a partition of the customer set. We identify the clusters to be selected by estimating the cost of each cluster and solving a set partitioning problem. To obtain estimates of the cluster costs in a reasonable amount of time, we limit the number of customers per cluster to *seven* even though this may result in suboptimal solutions. (We also tested our proposed algorithm without this limitation, but the results worsen in that case.)

To further explain the problem setting, we provide a simple example with two customers as illustrated in Figure 1. In this example, we assume that there is a single vehicle with a capacity of 2,000 and a constant speed of 10 distance units per hour initially located at the depot. The daily operating hours are assumed to be 8 A.M. to 6 P.M. The daily usage rate, initial inventory amount, and the capacity of each customer are provided in Figure 1. For this simple example we assume that the length of the planning cycle (T) is two days, and we present a feasible delivery schedule to explain how the inventory levels change and the transportation costs are calculated. At the beginning of Day 1 (at 8 A.M.), the vehicle leaves the depot with a full truckload, visits Customers 1 and 2 in this order, and returns to the depot at 5 P.M. We deliver 800 units to Customer 1 and 1,200 units to Customer 2. On Day 1, the vehicle visits Customer 1 at 10 A.M., and the inventory of Customer 1 decreases to 120 just before the delivery. After delivering 800 units to Customer 1, its inventory level increases to 920. Then, the vehicle visits Customer 2 at 2 P.M. and delivers 1,200 units. Customer 2 has 100 units in the inventory just before the delivery, and its inventory increases to 1,300 after the delivery. At the end of Day 1 (at 6 P.M.), Customers 1 and 2 have 600 and 700 units in the inventory, respectively. On Day 2, the vehicle leaves the depot at 8 A.M., visits Customer 2 at 11 A.M., and delivers 1,800 units. Customer 2 has 250 and 2,050 units of inventory just before and after the delivery, respectively. The vehicle returns to the

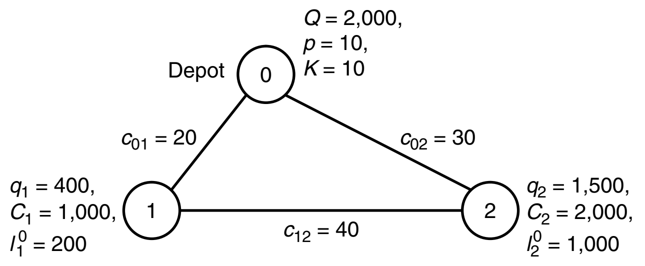

Figure 1 A Simple Example Provided to Illustrate the Problem Structure

depot at 2 p.m., and stays at the depot for the rest of the day. At the end of Day 2 (and at the beginning of Day 3 accordingly), Customers 1 and 2 have 200 and 1,000 units of inventory. This delivery schedule is repeated every two days. The cost of delivery on Day 1 is 90 ($=20+40+30$), and the cost of delivery on Day 2 is 60 ($=2\times 30$). The long-run average daily cost of this delivery schedule is 75.

4. A Two-Phase Solution Approach

In this section, we describe our two-phase solution approach, the *Iterative Clustering-Based Constructive Heuristic Algorithm* (IC-CH), for solving the IRP. As mentioned before, we first cluster the customers using the *Iterative Clustering* (IC) procedure, and then determine the delivery schedule for each cluster using the *Constructive Heuristic Algorithm* (CH).

4.1. Cluster Generation: Iterative Clustering

In *Iterative Clustering* (IC), we first implement a modified version of the *Approximate Stability Assessment* procedure developed by Özener, Ergun, and Savelsbergh (2013) to generate a pre-specified number of random, but potentially good, clusters and then use a lower bounding method for calculating the approximate cost of serving these clusters. In the second part of this clustering algorithm, we solve a series of set partitioning problems to choose a set of clusters among the generated clusters. At the very beginning, we use the approximate cost of serving each cluster while solving the set partitioning problem. Then, every time we choose a set of clusters after solving a set partitioning problem, we use the algorithm proposed for the second phase to calculate a better approximation, an upper bound, for the cost of serving these clusters instead of a lower bound. Then, after updating the "cost-to-serve" estimates of these clusters with the upper bounds, we solve a set partitioning problem again to select the clusters. This is repeated for a pre-specified number of iterations or until the solution of the set partitioning problem does not change any more. The steps of the algorithm are provided in Algorithm 1.

Algorithm 1 (*IC* procedure)

```
    input: An IRP instance with a $D \times D$ map of
           customers and the depot
    output: A partition of the customer set $V$
 1: repeat
 2:     Pick a random point $b$, the base point, on
          the map.
 3:     Calculate $c_{bi}$ for all $i \in V$.
 4:     Set the probability $p_i$ of selecting customer $i$
          for the cluster to be generated as
          $p_i \leftarrow \begin{cases} 1-\sqrt{\frac{c_{bi}}{D}}, & \text{if } c_{bi} \le \frac{D}{4}, \\ 0, & \text{otherwise.} \end{cases}$
 5:     Generate the cluster by selecting customers
          based on their respective probabilities.
          That is, customer $i$ will be in the generated
          cluster with probability $p_i$.
 6: until A termination condition is reached.
 7: Calculate the cost of serving each generated
      cluster using PSLP (Song and Savelsbergh 2007).
 8: repeat
 9:     Solve the set partitioning problem to select
          a subset of the generated clusters.
10:     For the selected clusters, update the cost of
          serving with an upper bound found by the
          algorithm proposed for the second phase
          (see §4.2).
11: until A pre-specified number of iterations is
        reached or the selected set of clusters
        consists of clusters with previously
        updated costs.
```

If we were to generate all possible clusters, calculate the exact cost of each cluster, and finally solve the resulting set partitioning problem, the result would be the optimal solution to the IRP. However, this procedure is not computationally efficient due to the exponential ($O(2^n)$) number of clusters to be generated and the difficulty of finding the optimal delivery schedule for each cluster. Hence, we use approximate procedures that limit the number of clusters generated and under/over estimate the cost of the optimal delivery schedule.

First of all, we would like to generate "good" clusters for the second phase. As mentioned in Özener, Ergun, and Savelsbergh (2013), it is not straightforward to generate good clusters as this depends on many factors such as the customer locations, the usage rates, the storage capacities, the vehicle capacity, and the interactions between these factors. On the other hand, we can still claim that clusters with geographically close customers are potentially good clusters since our objective is to minimize transportation costs. Hence, we employ Steps 1–6 of Algorithm 1 to randomly generate such clusters.

As we would like to have potentially good clusters, we assign high probabilities to the customers close to the *base point*. Note that the algorithm may terminate before generating the pre-specified number of clusters due to the selection of these probabilities.

In generating the random clusters, when an already-generated cluster is generated or the number of customers in the generated cluster is more than seven, then we call this an *unsuccessful* cluster-generation trial and discard the generated cluster. The algorithm is terminated after either 10,000 successive or 200,000 cumulative unsuccessful cluster-generation trials. The cluster-generation subroutine is terminated when a pre-specified number of clusters is generated or one of the above-mentioned termination criteria is met.

After the cluster-generation subroutine is terminated, in Step 7 we calculate the lower bound on the cost of serving each cluster using the *Pattern Selection Linear Program* (PSLP) (Song and Savelsbergh 2007) which is based on both base pattern/route generation and linear programming.

Finally, using the lower bound results for the generated clusters as initial estimates of "costs-to-serve" we solve a series of set partitioning problems to select the clusters while updating the "costs-to-serve" estimates of the selected clusters with an upper bound at each iteration (Steps 8–11). The motivation behind this algorithm is the fact that after a certain number of iterations, we would select the previously selected cluster set in the set partitioning problem even though their costs are based on an upper bound and the costs of others are based on a lower bound. The reason is that the selected subset of clusters is so cost-efficient that their updated total costs are lower than any other potential set of clusters' costs. Note that we may not end up with selecting a set of clusters with all updated costs due to a limited number of iterations. In that case, we select the current set of clusters as the cluster set for the second phase.

In general, in IC any algorithm producing a feasible delivery schedule can be used to find an upper bound on the serving costs of a given cluster (see Step 10 of Algorithm 1). In the proposed solution approach, we employ a time-restricted version of the CH to find a feasible delivery schedule.

4.2. Delivery Schedule Generation: Constructive Heuristic Algorithm

In this section, assuming we determined the clusters for each vehicle, we focus on the selection and the scheduling of the delivery routes for each vehicle. For this purpose, we propose a construction-based heuristic algorithm, called the *Constructive Heuristic Algorithm* (CH), employing operators such as insertion (Solomon 1987) and delivery volume per mile (Song and Savelsbergh 2007) and some new ideas. We use $\tilde{V}$ to denote the set of customers in the cluster under consideration.

The CH has two stages: (i) construction stage, and (ii) post-processing stage. In the first stage, we construct the delivery schedule (delivery routes and their execution times), and then in the second stage, we try to improve this solution by modifying the structures and the execution times of the delivery routes using a mixed integer programming formulation. The main idea behind the first stage is as follows: Starting from time zero, first we construct a delivery route using an insertion subroutine, and then accept or reject this route based on the total amount delivered in the route. We set a *threshold value* (τ) for the vehicle capacity utilization in a route, and accept the constructed route if the vehicle capacity utilization in this route is greater than or equal to this threshold. If the route is accepted, we start constructing another route after the vehicle returns to the depot. If not, we increase the time by λ while keeping the vehicle idle at the depot for λ amount of time. In this case, the proposed route-generation procedure cannot find a route with vehicle capacity utilization greater than or equal to the threshold value. This is mainly because of the inventory levels of the customers. By sliding the time by λ, we allow customers to consume more and open more space for the incoming units which results in a route with a higher vehicle capacity utilization. We continue constructing routes until the end of the planning horizon. In constructing the routes, we use a *delivery volume criterion* to decide which customer to insert into the current route and to determine the delivery amount to each customer. Delivery volume criterion is basically a weighted summation of the delivery amounts to each customer, where the weights are determined based on "urgency."

Next, we explain the main components of the algorithm. We start with how we determine the delivery amount to each customer in a given route and how we calculate the *delivery volume criterion*.

4.2.1. Determining the Delivery Amounts. We calculate the delivery amount to each customer in a given route Γ which starts and ends at the depot after visiting a subset of customers in the cluster. First, we provide the notation used. We use $|\Gamma|$ to denote the size of the route, i.e., the number of customers visited in route Γ.

Γ = Route under consideration,
Γ_j = Customer visited in the jth position in route Γ $\forall j \in \{1, \ldots, |\Gamma|\}$,
t = Current day,
k = Current time of current day (also start time of route),
I_i^t = Inventory level of customer i at time k on day t,
v_i = Visiting time of customer i in route Γ,
d_i = Delivery amount to customer i in route Γ,
d_i^l = Portion of the delivery amount to customer i that can be used to satisfy the demand on day l $\forall l \in \{t, t+1, \ldots, T\}$.

We calculate v_i values as follows: $v_{\Gamma_1} = t_{0,\Gamma_1}$, and $v_{\Gamma_j} = v_{\Gamma_{j-1}} + t_{\Gamma_{j-1},\Gamma_j}$ for all $j \in \{2, 3, \ldots, |\Gamma|\}$. Finally, d_i^l values are calculated as follows:

$$d_i^l = \max\left\{0, \min\left\{q_i\left(l-t+1-\frac{k}{K}\right) - \sum_{z=t}^{l-1} d_i^z - I_i^t,\right.\right.$$
$$\left.\left. C_i - \max\left\{0, I_i^t - \frac{q_i v_i}{K}\right\} - \sum_{z=t}^{l-1} d_i^z\right\}\right\}. \quad (1)$$

In summary, assuming a first-in first-out policy for the inventory, we determine how much of the volume delivered in the current route would be used to satisfy a specific future day's (day l) demand. In Equation (1), d_i^l is calculated as the minimum of (i) the amount of day l's demand that is not satisfied by the available inventory ($q_i(l-t+1-k/K) - \sum_{z=t}^{l-1} d_i^z - I_i^t$), and (ii) the available storage capacity at the delivery time that can be utilized to store day l's delivery amount ($C_i - \max\{0, I_i^t - q_i v_i/K\} - \sum_{z=t}^{l-1} d_i^z$). In this equation, $q_i(l-t+1-k/K)$ is the total demand until the end of day l; $\sum_{z=t}^{l-1} d_i^z$ is the total demand until the end of day $l-1$ that is satisfied by the current delivery; and $\max\{0, I_i^t - q_i v_i/K\}$ is the inventory level just before the delivery. Using these d_i^l values, we determine the delivery amounts to each customer by solving the following linear program to determine the amount delivered (u_i^l) to satisfy a certain day's demand

$$\text{DV-LP:} \quad \max \left\{ \sum_{i\in\Gamma} \sum_{l=t}^{T} w_l u_i^l \right\} \tag{2}$$

$$\text{s.t.} \quad \sum_{i\in\Gamma} \sum_{l=t}^{T} u_i^l \le Q; \tag{3}$$

$$u_i^l \le d_i^l, \quad \forall i \in \Gamma, l \in \{t, t+1, \ldots, T\}; \tag{4}$$

$$u_i^l \ge 0, \quad \forall i \in \Gamma, l \in \{t, t+1, \ldots, T\}; \tag{5}$$

where w_l is the weight of the delivery amount for day l. DV-LP assigns the total capacity of the vehicle to the customers while maximizing the weighted summation of demand satisfied for each day. Using the optimal solution ($\bar{u}_i^l$) of DV-LP, one can calculate the total amount delivered to customer i as follows:

$$d_i = \sum_{l=t}^{T} \bar{u}_i^l.$$

For a given route, we call the optimal objective function value of DV-LP the *delivery volume criterion* of this route. Note that DV-LP can be solved easily by considering the weight w_l values for each day. First, we start with sorting w_l values in a nonincreasing order. Assuming that l_j is the day with the jth largest weight, one can find an optimal solution by allocating the vehicle capacity starting from day l_1 with customer Γ_1. If all $\bar{u}_i^{l_j}$'s are equal to $d_i^{l_j}$, we move to day l_{j+1} and repeat the same steps until the entire vehicle capacity is assigned, or all $\bar{u}_i^l$ values are equal to d_i^l. If there are alternative optimal solutions, we give higher priority to the customers that are visited earlier. That is, if both $\bar{u}_{\Gamma_i}^l$ and $\bar{u}_{\Gamma_j}^l$ are nonzero for some $i<j$, then we make sure that $(\bar{u}_{\Gamma_i}^l - d_{\Gamma_i}^l)\bar{u}_{\Gamma_j}^l = 0$. In calculating the delivery volume criterion and delivery amount to each customer by solving DV-LP, an important parameter is the weight w_l. Among several alternatives tested, we decided to use the following four: (i) $w_l = 1$, (ii) $w_l = T-l+1$, (iii) $w_l = 1/2^{l-t}$, and (iv) $w_l = (T-l+1)^2$. For the first alternative, there is no priority between different u_i^l. Thus, in this case, we make maximum delivery to the earlier visited customer. For the other alternatives, we give higher weight to the delivery amounts consumed early.

The subroutine DeliveryVolume( ) for calculating the delivery volume criterion and delivery amounts is provided in Algorithm 2. The input Ω determines the weight structure to be used. That is, (i) $w_l = 1$ if $\Omega = 1$, (ii) $w_l = T-l+1$ if $\Omega = 2$, (iii) $w_l = 1/2^{l-t}$ if $\Omega = 3$, and (iv) $w_l = (T-l+1)^2$ if $\Omega = 4$.

Algorithm 2 (DeliveryVolume( ) function)

```
input: Day $t$, hour $k \in [0, K]$, delivery route $\Gamma$, $w_l$
       option $\Omega$, current inventory vector $I^t$
output: Delivery volumes vector $d$, and delivery
        volume criterion $VC$
1: function DeliveryVolume($t, k, \Gamma, \Omega, I^t$)
2: Calculate $d_i^l$ values for each $i \in \Gamma$ and $l \ge t$.
3: Determine $\bar{u}_i^l$ values by solving DV-LP and
     set $VC = \sum_{i\in\Gamma} \sum_{l=t}^{T} w_l \bar{u}_i^l$.
4: Calculate the delivery volume for each customer:
     $d_i = \sum_{l=t}^{T} \bar{u}_i^l$.
5: return ($d$, $VC$).
```

4.2.2. Constructing the Delivery Routes. We use ConstructRoutes( ) function for generating the routes on a given day. The pseudocode for this function is provided in Algorithm 3. This function uses the above-explained DeliveryVolume( ) function while determining the delivery volumes to customers for a given route and the delivery volume criterion of this route. We start with an empty route (0–0), insert the customer which yields the highest positive improvement in delivery volume criterion per unit traveling cost of the route, and continue until the delivery volume criterion per unit traveling cost of the route cannot be improved any more. Then, the route is accepted if the vehicle capacity utilization is greater than or equal to the threshold value (τ). Otherwise, time is increased by λ.

Algorithm 3 (ConstructRoutes( ) function)

```
input: Day $t$, threshold value $\tau$ for vehicle capacity
       utilization per route, time increment $\lambda$, $w_l$
       option $\Omega$, starting inventory vector $I^t$
output: $P_t$ := Set of routes for day $t$, and
        $TC_t$ := Total cost associated with $P_t$
1: function ConstructRoutes($t, \tau, \lambda, \Omega, I^t$)
2: $s \leftarrow 0$
3: $P_t \leftarrow \varnothing$
4: $TC_t \leftarrow 0$
5: while $s \le K$ do
6:     $\Gamma \leftarrow \varnothing$
7:     $V' \leftarrow \tilde{V}$
8:     while $V' \ne \varnothing$ do
```

```
 9:        $(i^*, j^*) \leftarrow \underset{i \in V', j \in \{1,\ldots,|\Gamma|+1\}}{\arg\max} \left\{ \frac{VC^{\Gamma_{i,j}}}{\text{Cost}(\Gamma_{i,j})} : \Gamma_{i,j} = \Gamma \cup i$ with $i$ at position $j$, $(d^{\Gamma_{i,j}}, VC^{\Gamma_{i,j}}) \leftarrow$ DeliveryVolume$(t, k, \Gamma_{i,j}, \Omega, I^t) \right\}$
10:        if $(VC^{\Gamma_{i^*,j^*}})/\text{Cost}(\Gamma_{i^*,j^*}) > (VC^{\Gamma})/\text{Cost}(\Gamma)$ then
11:            $\Gamma \leftarrow \Gamma_{i^*,j^*}$
12:            $V' \leftarrow V' \setminus \{i^*\}$
13:        else
14:            break
15:    if $\Gamma \neq \varnothing$ and $\sum_{i \in \Gamma} d_i^{\Gamma} \geq \tau \times Q$ then
16:        if Route $\Gamma$ with depart time $s$ is feasible then ▹ i.e., no stockouts when executed
17:            $P_t \leftarrow P_t \cup \Gamma$
18:            $TC_t \leftarrow TC_t + \text{Cost}(\Gamma)$
19:            $s \leftarrow s + \text{Time}(\Gamma)$
20:            $I_i^t \leftarrow I_i^t + d_i^{\Gamma} - \text{Time}(\Gamma)/K \times q_i \ \forall\, i \in \tilde{V}$
21:        else
22:            return $(\varnothing, \infty)$
23:    else
24:            $s \leftarrow s + \lambda$
25:            $I_i^t \leftarrow I_i^t - \lambda/K \times q_i \ \forall\, i \in \tilde{V}$
26: return $(P_t, TC_t)$.
```

4.2.3. Determining the Best Solution. The pseudocode for the *Constructive Heuristic Algorithm* utilizing the above-discussed functions is provided in Algorithm 4. The main idea is to construct a delivery schedule for each threshold value (τ) and weight alternative (w_l) for DV-LP, and choose the best solution found. After trying several alternatives for λ, we decided to set it to 1% of the total daily operating hours K.

Algorithm 4 (*CH*)

```
input: An IRP instance and a cluster $\tilde{V}$ in this instance
output: Set of routes $P_{\text{best}} = \{1, \ldots, L\}$, and the associated total cost $TC_{\text{best}}$
 1: function CH($\tilde{V}$)
 2:     $\lambda \leftarrow K/100$
 3:     $TC_{\text{best}} \leftarrow \infty$
 4:     $P_{\text{best}} \leftarrow \varnothing$
 5:     for all $\tau \in \{0.0, 0.1, 0.2, \ldots, 0.9\}$ do
 6:         for all $\Omega \in \{1, 2, 3, 4\}$ do
 7:             $TC_{\tau,\Omega} \leftarrow 0$
 8:             $P_{\tau,\Omega} \leftarrow \varnothing$
 9:             for all $t \in \{1, 2, \ldots, T\}$ do
10:                 $(P_t, TC_t) \leftarrow$ ConstructRoutes$(t, \tau, \lambda, \Omega, I^t)$
11:                 if $P_t \neq \varnothing$ then
12:                     $P_{\tau,\Omega} \leftarrow P_{\tau,\Omega} \cup P_t$
13:                     $TC_{\tau,\Omega} \leftarrow TC_{\tau,\Omega} + TC_t$
14:                 else
15:                     $P_{\tau,\Omega} \leftarrow \varnothing$
16:                     $TC_{\tau,\Omega} \leftarrow \infty$
17:             if $TC_{\tau,\Omega} < TC_{\text{best}}$ then
18:                 $TC_{\text{best}} \leftarrow TC_{\tau,\Omega}$
19:                 $P_{\text{best}} \leftarrow P_{\tau,\Omega}$
20:     return $(P_{\text{best}}, TC_{\text{best}})$.
```

4.2.4. Post-Processing Stage. At this stage, we try to improve the delivery schedule developed in the first stage. While keeping the execution order of the routes in the initial plan fixed, we consider (i) eliminating one or more of the routes, (ii) eliminating one or more of the customers from the already-constructed routes, and (iii) changing the execution times of the routes and delivery volume to each customer. We use the following mixed integer programming formulation (CH-MIP) to achieve this. We start with introducing some notation. We use L to denote the total number of routes in the delivery schedule constructed in the first stage for T days, and assign numbers to the routes in the order they are performed. We use Γ^r to denote the customers in the rth route in this cluster, and v_{ir} to denote the original visiting time of customer i in route r. We use the following decision variables in our formulation:

S_r = Start time of route r $\forall\, r \in \{1, 2, \ldots, L\}$,
E_r = End time of route r $\forall\, r \in \{1, 2, \ldots, L\}$,
P_{ir} = Partial cost of route r until customer i $\forall\, i \in \tilde{V}$, $r \in \{1, 2, \ldots, L\}$,
R_r = Cost of route r $\forall\, r \in \{1, 2, \ldots, L\}$,
I_{ir} = Inventory at customer i right after the visit in route $r-1$ $\forall\, i \in \tilde{V}$, $r \in \{1, 2, \ldots, L+1\}$ (with a slight abuse of definition, we use I_{i1} to denote the starting inventory of customer i),
m_r = Day route r is executed $\forall\, r \in \{1, 2, \ldots, L\}$,
d_{ir} = Delivery volume to customer i in route r $\forall\, i \in \tilde{V}$, $r \in \{1, 2, \ldots, L\}$,
f_{ir} = Visiting time of customer i in route r $\forall\, i \in \tilde{V}$, $r \in \{1, 2, \ldots, L\}$,

$$x_r = \begin{cases} 1, & \text{if route } r \text{ is performed,} \\ 0, & \text{otherwise,} \end{cases} \quad \forall\, r \in \{1, 2, \ldots, L\},$$

$$y_{ir} = \begin{cases} 1, & \text{if customer } i \text{ is visited in route } r, \\ 0, & \text{otherwise,} \end{cases} \quad \forall\, i \in \tilde{V},\ r \in \{1, 2, \ldots, L\}.$$

We set d_{ir} and y_{ir} to zero for the customers that are not visited in route r originally. The corresponding f_{ir} and I_{ir} values are left flexible so that f_{ir} corresponds to any time between the start and end time of route r, and I_{ir} is the inventory level at this point in time which is used to check the feasibility of the delivery schedule in terms of demand. Using these variables, we minimize the average daily cost of the delivery schedule as follows:

$$\text{CH-MIP:} \ \min \left\{ \frac{1}{T} \sum_{r \in L} R_r \right\} \tag{6}$$

$$\text{s.t.} \ K(m_r - 1) - S_r \leq 0, \ \forall\, r \in \{1,2,\ldots,L\}; \tag{7}$$

$$Km_r - E_r \geq 0, \quad \forall r \in \{1,2,\ldots,L\}; \tag{8}$$

$$E_{r-1} - S_r \leq 0, \quad \forall r \in \{1,2,\ldots,L\}; \tag{9}$$

$$E_r - y_{ir}t_{i0} - f_{ir} \geq 0, \quad \forall i \in \tilde{V},\ r \in \{1,2,\ldots,L\}; \tag{10}$$

$$f_{ir} - S_r - y_{ir}t_{0i} \geq 0, \quad \forall i \in \tilde{V},\ r \in \{1,2,\ldots,L\}; \tag{11}$$

$$f_{ir} - f_{pr} - y_{ir}t_{pi} + 2TK(1 - y_{pr}) \geq 0, \quad \forall i,p \in \Gamma^r,\ r \in \{1,2,\ldots,L\},\ v_{ir} > v_{pr}; \tag{12}$$

$$P_{ir} - y_{ir}c_{0i} \geq 0, \quad \forall i \in \tilde{V},\ r \in \{1,2,\ldots,L\}; \tag{13}$$

$$P_{ir} - P_{pr} - y_{ir}c_{pi} + A(1 - y_{pr}) \geq 0, \quad \forall i,p \in \Gamma^r,\ r \in \{1,2,\ldots,L\},\ v_{ir} > v_{pr}; \tag{14}$$

$$R_r - y_{ir}c_{i0} - P_{ir} \geq 0, \quad \forall i \in \tilde{V},\ r \in \{1,2,\ldots,L\}; \tag{15}$$

$$y_{ir} - x_r \leq 0, \quad \forall i \in \tilde{V},\ r \in \{1,2,\ldots,L\}; \tag{16}$$

$$d_{ir} - Qy_{ir} \leq 0, \quad \forall i \in \tilde{V},\ r \in \{1,2,\ldots,L\}; \tag{17}$$

$$\sum_{i \in \tilde{V}} d_{ir} - Qx_r \leq 0, \quad \forall r \in \{1,2,\ldots,L\}; \tag{18}$$

$$I_{ir} - (f_{ir} - f_{i,r-1})\frac{q_i}{K} \geq 0, \quad \forall i \in \tilde{V},\ r \in \{1,2,\ldots,L\}; \tag{19}$$

$$I_{i,r+1} \leq C_i, \quad \forall i \in \tilde{V},\ r \in \{1,2,\ldots,L\}; \tag{20}$$

$$I_{ir} + d_{ir} - (f_{ir} - f_{i,r-1})\frac{q_i}{K} - I_{i,r+1} = 0, \quad \forall i \in \tilde{V},\ r \in \{1,2,\ldots,L\}; \tag{21}$$

$$I_{i,L+1} - (TK - f_{iL})\frac{q_i}{K} - I_{i1} = 0, \quad \forall i \in \tilde{V}; \tag{22}$$

$$I_{i1} = \mathcal{I}_i^0, \quad \forall i \in \tilde{V}; \tag{23}$$

$$d_{ir}, y_{ir} = 0, \quad \forall i \in \tilde{V}/\Gamma^r,\ r \in \{1,2,\ldots,L\}; \tag{24}$$

$$f_{i0} = 0, \quad \forall i \in \tilde{V}; \tag{25}$$

$$S_r, E_r, R_r \geq 0, \quad \forall r \in \{1,2,\ldots,L\}; \tag{26}$$

$$m_r \in \{1,2,\ldots,T\}, \quad \forall r \in \{1,2,\ldots,L\}; \tag{27}$$

$$P_{ir}, I_{ir}, f_{ir}, d_{ir} \geq 0, \quad \forall i \in \tilde{V},\ r \in \{1,2,\ldots,L\}; \tag{28}$$

$$x_r \in \{0,1\}, \quad \forall r \in \{1,2,\ldots,L\}; \tag{29}$$

$$y_{ir} \in \{0,1\}, \quad \forall i \in \tilde{V},\ r \in \{1,2,\ldots,L\}. \tag{30}$$

Equation (6) is the objective function that is the average transportation cost per day. Constraints (7)–(8) make sure that each route starts and ends on the same day. Constraints (9) ensure that a route cannot start before the completion of the previous route. The end times of the routes are determined by Constraints (10). Similarly, the visiting times of the customers in each route are determined by Constraints (11)–(12). Constraints (13)–(14) calculate the partial cost of the routes after a customer is visited. Here, A is a large enough number. Then, these partial costs are used to determine the total cost of the route in Constraints (15). Constraints (16) determine whether a route is performed or not. The amount delivered to a customer can be nonzero only if it is visited, and the total amount delivered on a route cannot be more than the capacity of the vehicle. These are guaranteed by Constraints (17)–(18). Constraints (19)–(20) make sure that customers do not have stock-outs and excess inventory. Inventory flow-balance equations are represented by Constraints (21). Constraints (22) make sure that the ending inventory at the end of the planning cycle is equal to the inventory at the beginning of the cycle. Finally, Constraints (23)–(25) set the initial values, and Constraints (26)–(30) impose the nonnegativity, integrality, and binary restrictions.

The main advantages of the *Constructive Heuristic Algorithm* are its low computation time and the large number of alternatives checked for different threshold values and weight alternatives for determining the delivery amounts and delivery volume criterion. The large number of alternatives allows considering multiple solutions and choosing the best one. On the other hand, the disadvantage of this algorithm is its greedy nature. Although we consider different weights for delivery volume calculations, the reactive (rather than proactive) nature of the algorithm still means it cannot avoid potential stock-outs. If a stock-out occurs while constructing the routes, the algorithm simply moves to another set of parameters (threshold value and weight alternative) instead of anticipating/handling these stock-outs ahead of time.

5. Computational Study

We performed a computational study on randomly generated and real-life instances to test the performance of the proposed algorithm in terms of solution quality and computational efficiency. We generated instances with 50 customers randomly located over a square map of size 1,000 × 1,000, which includes dense regions of customers to represent metropolitan areas as well as remote customers. The instances contain two types of customers: *low storage capacity* and *high storage capacity* customers. Low storage capacity customers have storage capacity values between 75% and 125% of a *base storage capacity value*, whereas high storage capacity customers have storage capacity values between 175% and 225% of the same base storage capacity value (Özener, Ergun, and Savelsbergh 2013). We take the *base storage capacity value* as 1,000. Similarly, the instances contain *low consumption rate* and *high consumption rate* customers with equal probability of being either. Low consumption rate customers have daily consumption rates between 15% and 35% of their storage capacities, whereas high consumption rate customers have daily consumption rates between 45% and 65% of their storage capacities. The travel costs between nodes are based on the Euclidean distances between nodes and we assume constant speed for the vehicles ($p = 600$);

hence, the travel times (in hours) are calculated by dividing the travel distances by a constant.

To test the algorithms on instances with different characteristics, we generated five base instances with 50 customers. For each base instance, we calculate the average number of single-customer (direct) delivery trips that can be made in a day by a single vehicle as $nK/\sum_{i\in V}(2t_{0i})$. We then compute a *base delivery capacity value* by dividing the total daily consumption of the customers by the average number of single-customer delivery trips by a single vehicle: $\sum_{i\in V} q_i/(nK/\sum_{i\in V}(2t_{0i}))$. We multiply the base delivery capacity value by a predetermined *delivery capacity factor* and obtain the *total delivery capacity value*, which is made equal to the total vehicle capacities. We use delivery capacity factors of 2, 2.5, and 3, which give us *restricted*, *normal*, and *relaxed* feasibility instances, respectively. Moreover, to see the effect of lower consumption rates, using *normal* instances we create *lower consumption* instances where the consumption rates of the customers are halved. Finally, for each of these *restricted*, *normal*, *relaxed*, and *lower consumption* instances, we experiment with two different fleet types: (i) *large fleet* of small capacity vehicles where the vehicle capacity is equal to half of the base storage capacity value, and (ii) *small fleet* of large capacity vehicles where the vehicle capacity is equal to the base storage capacity value. For each fleet type, we determine the number of vehicles available by dividing the total delivery capacity value by the corresponding vehicle capacity. In total, 40 instances were created for the computational experiments.

As mentioned before, our algorithm constructs a feasible delivery schedule for a planning cycle of T days and assumes that this schedule is repeated over the planning horizon. Our objective in the computational study is to minimize the average delivery cost per day for an infinite horizon problem even though the planning horizon can be finite as well. In the computational study, we assume biweekly planning cycles; i.e., T is equal to 14 days. We run our algorithm for both T and submultiples of T (specifically, 2 and 7) and choose the best solution that returns the lowest average daily delivery cost. We assume that the customer locations operate 10 hours per day and the initial inventory at customer i is equal to the daily consumption rate (q_i).

We test the performance of the proposed algorithm in terms of both solution quality and computation time. The solution quality refers to the optimality gap of the solution provided by the algorithm. We use the following simple lower bound mentioned by Song and Savelsbergh (2007) to calculate the optimality gaps of the solutions found

$$\sum_{i\in V}\frac{Tq_i}{Q}2c_{i0}.$$

This lower bound simply ignores the storage capacities of the customers and assumes that we can make direct full truckload deliveries to each customer.

In addition to a lower bound, we compare the performance of the proposed algorithm against that of a well-known solution method developed by Bramel and Simchi-Levi (1995). Bramel and Simchi-Levi (1995) propose a new heuristic, called the *Location-Based Heuristic* (LBH), to partition the customer set into clusters for general routing problems. They implement the proposed heuristic to VRP and a variant of the IRP. The main idea in the LBH is to formulate the routing problem as a location problem called the *Capacitated Concentrator Location Problem*. The IRP variant studied by the authors is much different from ours in several aspects including the inventory holding cost, customer capacities, fixed order cost, and unlimited number of vehicles. However, one can still use the main idea to cluster the customers. Hence, we compare the performances of the IC-CH and the modified version of the LBH, where we utilize the LBH in the first phase of our two-phase solution approach instead of the IC and then determine the delivery schedule for each cluster using the CH. We denote this modified version of the LBH by the LBH-CH. Next, we explain how we modify the LBH for our setting. First, since we have M vehicles in our setting, we limit the number of concentrators to be located in the LBH by M. In the variant of the IRP studied by Bramel and Simchi-Levi (1995), the deliveries to the customers are assumed to be done instantly at the beginning of each period and the demand is discrete; i.e., the demand in each period is satisfied instantly. Thus, the travel time to each customer and the amount of demand realized until the vehicle visits the customer are not taken into account in their model. To provide a fair comparison of the IC-CH and the LBH-CH, we modify the LBH by adding a limit on the total direct distance of the customers assigned to a concentrator. We tried different values (100%, 75%, and 50% of the maximum distance that can be traveled by a vehicle in a day, which is calculated as pK) for this limit and reported the best solution found.

All of the computational experiments are carried out on a 64-bit Windows Server with two 2.4 GHz Intel Xeon CPU's and 24 GB RAM. The algorithms are implemented using C++ and CPLEX 12.4 with Concert Technology. In implementing IC, we generate 2,000 random clusters in addition to all single- and two-customer clusters. Then, in the second part of IC we perform 10 iterations. We set time limits on the search operation of CPLEX using built-in parameters of CPLEX. Since CH-MIP is a relatively simple problem, we set the solution time to 15 seconds in the clustering phase of the IC-CH and to 60 seconds in the final step. In most of the cases, we find the optimal solution in a couple of seconds.

Table 1 **Results for the Instances with Relaxed, Normal, Restricted Feasibility, and Lower Consumption**

	Setting									
	Relaxed		Normal		Restricted		Lower cons.		Overall	
Algorithm	#Fea	OG	#Fea	OG	#Fea	OG	#Fea	OG	#Fea	OG
LBH-CH	8	2.51	8	2.51	8	2.49	9	4.21	33	2.97
IC-CH	10	2.52	10	2.56	9	2.72	10	3.69	39	2.87

5.1. Randomly Generated Instances

In evaluating the performance of the IC-CH and comparing it against the LBH-CH, our performance measures are the number of instances for which a feasible solution is found (#*Fea* column) and the average optimality gap of the feasible solutions (*OG* column).

To evaluate the impact of the total delivery capacity values and lower consumption rates, we partition the instances into four groups according to the base capacity multiplier and consumption rates and present the results in Table 1. We observe that the IC-CH finds a feasible solution for 39 of 40 instances, and the average optimality gap of the solutions found is less than 3%. The average optimality gap of the solutions for the instances with lower consumption rates is slightly higher than that of instances with higher consumption rates. We observe that the base capacity multiplier does not have a significant impact on the performance of the IC-CH. However, the IC-CH fails to find a feasible solution for one of the instances with restricted feasibility due to tight total delivery capacity. On the other hand, although the LBH-CH finds a feasible solution for the instance that cannot be solved by the IC-CH, the LBH-CH cannot find a feasible solution for seven of 40 instances in total. The average optimality gap of the solutions found by the LBH-CH is slightly worse than that of the solutions found by the IC-CH, but the main drawback of the LBH-CH is that it cannot find a feasible solution for 17.5% of the instances. Similar to the observation for IC-CH, the performance of the LBH-CH slightly deteriorates for the instances with lower consumption rates.

Table 2 breaks down the results into two groups: instances with a small fleet and a large fleet. Even though both types of instances have the same total capacity values, in the former set of instances we have a fewer number of vehicles with higher capacities. We observe that the IC-CH performs better on the instances with a large fleet of vehicles by finding a feasible solution for all of the instances with an average optimality gap of 1.99%. On the other hand, it fails to identify a feasible solution for one of the instances with a small fleet and the average optimality gap of the solutions provided that a feasible solution can be found is around 3.8%. Intuitively, the superior performance of the IC-CH on the instances with a large fleet is due to the fact that there exists more clustering options for customers, and hence, better utilization of the vehicles. A similar observation about the average optimality gaps can be made for the solutions found by the LBH-CH. However, the LBH-CH cannot find a feasible solution for 20% of the instances with a large fleet.

Table 2 **Results for the Instances with a Small and a Large Fleet**

	Fleet setting			
	Small fleet		Large fleet	
Algorithm	#Fea	OG	#Fea	OG
LBH-CH	17	4.11	16	1.75
IC-CH	19	3.81	20	1.99

Table 3 **CPU Time (in Seconds) of the Second Phase (*Constructive Heuristic Algorithm*) for Different Planning Cycle Lengths**

Algorithm	2-day	7-day	14-day	Total
LBH-CH	1.8	341.6	671.5	1,014.8
IC-CH	2.3	192.1	588.9	783.3

Finally, we report on the computational time of the proposed algorithm for different planning cycle length values. For the IC-CH, it takes around 500 seconds to generate and select clusters, whereas the LBH-CH finds the clusters in less than a second. Note that in the clustering phase of the IC-CH, we implement a time-restricted version of the CH. After selecting the clusters, we determine the delivery schedule for each cluster using the full version of the CH. In Table 3, we present the average CPU times for the second phase of the LBH-CH and the IC-CH for different planning cycle length (T) values. Since we solve for 2-day, 7-day, and 14-day planning cycles and report the best available result, we also list the total CPU time.

We observe that the CPU time is significantly affected by the length of the planning cycle. The proposed algorithm runs extremely fast for a 2-day planning cycle. However, its run time is about 100 times and 250 times higher for 7-day and 14-day planning cycles, respectively. On average, it takes around 13 minutes to find the best feasible solution for different planning cycle length values.

5.2. Real-Life Instances

The real-life instances are provided by an industrial gases manufacturer. The locations of the depot and the customers on three test instances with 26, 70, and 80 customers are shown in Figure 2.

Despite the poor performance of the LBH-CH on randomly generated instances, we observe that it can find a feasible solution for all three real-life instances

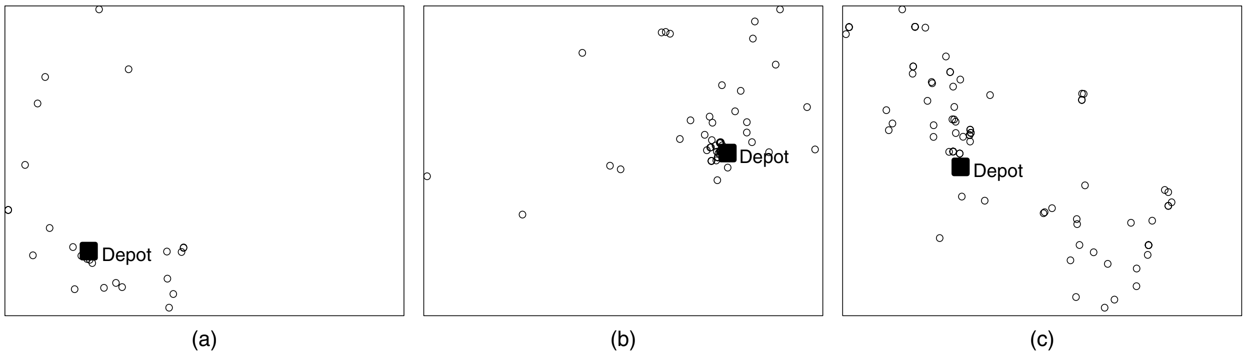

Figure 2 Real-Life Instances with (a) 26, (b) 70, and (c) 80 Customers

Table 4 Optimality Gaps of the Solutions Found for Real-Life Instances

Instance #	LBH-CH	IC-CH
1	1.62	2.37
2	4.95	7.52
3	0.99	1.38

even with an optimality gap 1% smaller than that of the IC-CH on average (see Table 4). The average optimality gap of the solutions found by the IC-CH is 3.76% for real-life instances. The optimality gaps of the solutions found by both the IC-CH and the LBH-CH are slightly higher for the second instance because it presents a higher challenge due to the highly varied truck capacity over storage capacity ratios of the customers.

6. Conclusion

The *Inventory Routing Problem* (IRP) is a challenging problem due to the considerable number of possible delivery schedules and many other factors affecting the feasibility and cost-effectiveness of these schedules such as depot and customer locations, vehicle capacity, storage capacities, usage rates, and the timing of the deliveries. In this paper, we propose a novel two-phase solution approach, called the *Iterative Clustering-Based Constructive Heuristic Algorithm* (IC-CH), for solving the IRP. For the first phase, we propose an iterative procedure (*Iterative Clustering*) which partitions the customer set into clusters by choosing among a set of randomly generated "good" clusters based on the estimated serving cost of each cluster. For the second phase, we propose a delivery schedule generation algorithm called the *Constructive Heuristic Algorithm*. Based on our computational study on randomly generated and real-life instances, we observe that the IC-CH yields high-quality solutions with optimality gap values around 3%. Since we use a lower bound to calculate the optimality gaps, this is a conservative estimate about the performance of the algorithm.

Acknowledgments

The authors would like to thank the editor-in-chief, the associate editor, and the two anonymous referees for their constructive comments.

References

Adelman D (2004) A price-directed approach to stochastic inventory/routing. *Oper. Res.* 52(4):499–514.

Andersson H, Hoff A, Christiansen M, Hasle G, Lokketangen A (2010) Industrial aspects and literature survey: Combined inventory management and routing. *Comput. Oper. Res.* 37(9):1515–1536.

Anily S, Federgruen A (1990) One warehouse multiple retailer systems with vehicle routing costs. *Management Sci.* 36(1):92–114.

Anily S, Federgruen A (1993) Two-echelon distribution systems with vehicle routing costs and central inventories. *Oper. Res.* 41(1):37–47.

Archetti C, Bertazzi L, Hertz A, Speranza MG (2012) A hybrid heuristic for an inventory routing problem. *INFORMS J. Comput.* 24(1):101–116.

Bell WJ, Dalberto LM, Fisher ML, Greenfield AJ, Jaikumar R, Kedia P, Mack RG, Prutzman PJ (1983) Improving the distribution of industrial gases with an on-line computerized routing and scheduling optimizer. *Interfaces* 13(6):4–23.

Bertazzi L, Speranza MG (2012) Inventory routing problems: An introduction. *EURO J. Transportation Logist.* 1(4):307–326.

Bertazzi L, Speranza MG (2013) Inventory routing problems with multiple customers. *EURO J. Transportation Logist.* 2(3):255–275.

Bertazzi L, Chan LMA, Speranza MG (2007) Analysis of practical policies for a single link distribution system. *Naval Res. Logist.* 54(5):497–509.

Bertazzi L, Paletta G, Speranza MG (2002) Deterministic order-up-to level policies in an inventory routing problem. *Transportation Sci.* 36(1):119–132.

Bramel J, Simchi-Levi D (1995) A location-based heuristic for general routing problems. *Oper. Res.* 43(4):649–660.

Campbell A, Clarke L, Kleywegt A, Savelsbergh M (1998) The inventory routing problem. Crainic TG, Laporte G, eds. *Fleet Management and Logistics* (Kluwer Academic Publishers, Norwell, MA), 95–113.

Campbell AM, Savelsbergh MWP (2004) A decomposition approach for the inventory-routing problem. *Transportation Sci.* 38(4): 488–502.

Chan LMA, Simchi-Levi D (1998) Probabilistic analyses and algorithms for three-level distribution systems. *Management Sci.* 44(11):1562–1576.

Chan LMA, Federgruen A, Simchi-Levi D (1998) Probabilistic analyses and practical algorithms for inventory-routing models. *Oper. Res.* 46(1):96–106.

Christiansen M, Fagerholt K (2009) Maritime inventory routing problems. Floudas CA, Pardalos PM, eds. *Encyclopedia of Optimization*, 2nd ed. (Springer-Verlag, New York), 1947–1955.

Dror M, Ball M, Golden B (1985/6) A computational comparison of the algorithms for the inventory routing problem. *Ann. Oper. Res.* 4(1):3–23.

Federgruen A, Zipkin P (1984) A combined vehicle routing and inventory allocation problem. *Oper. Res.* 32(5):1019–1037.

Golden B, Assad A, Dahl R (1984) Analysis of a large scale vehicle routing problem with an inventory component. *Large Scale Systems* 7(2–3):181–190.

Jung J, Mathur K (2007) An efficient heuristic algorithm for a two-echelon joint inventory and routing problem. *Transportation Sci.* 41(1):55–73.

Kleywegt AJ, Nori VS, Savelsbergh MWP (2002) The stochastic inventory routing problem with direct deliveries. *Transportation Sci.* 36(1):94–118.

Kleywegt AJ, Nori VS, Savelsbergh MWP (2004) Dynamic programming approximations for a stochastic inventory routing problem. *Transportation Sci.* 38(1):42–70.

Li J, Chu F, Chen H (2011) A solution approach to the inventory routing problem in a three-level distribution system. *Eur. J. Oper. Res.* 210(3):736–744.

Moin NH, Salhi S (2007) Inventory routing problems: A logistical overview. *J. Oper. Res. Soc.* 58(9):1185–1194.

Özener OÖ, Ergun Ö, Savelsbergh M (2013) Allocating cost of service to customers in inventory routing. *Oper. Res.* 61(1):112–125.

Qu WW, Bookbinder JH, Iyogun P (1999) An integrated inventory-transportation system with modified periodic policy for multiple products. *Eur. J. Oper. Res.* 115(2):254–269.

Raa B, Aghezzaf EH (2008) Designing distribution patterns for long-term inventory routing with constant demand rates. *Internat. J. Production Econom.* 112(1):255–263.

Raa B, Aghezzaf EH (2009) A practical solution approach for the cyclic inventory routing problem. *Eur. J. Oper. Res.* 192(2):429–441.

Ronen D (2002) Marine inventory routing: Shipments planning. *J. Oper. Res. Soc.* 53(1):108–114.

Solomon M (1987) Algorithms for the vehicle routing and scheduling problems with time window constraints. *Oper. Res.* 35(2):254–265.

Solyali O, Sural H (2011) A branch-and-cut algorithm using a strong formulation and an a priori tour-based heuristic for an inventory-routing problem. *Transportation Sci.* 45(3):335–345.

Song J-H, Savelsbergh M (2007) Performance measurement for inventory routing. *Transportation Sci.* 41(1):44–54.

Viswanathan S, Mathur K (1997) Integrating routing and inventory decisions in one-warehouse multiretailer multiproduct distribution systems. *Management Sci.* 43(3):294–312.

Zhao Q-H, Chen S, Zang C-X (2008) Model and algorithm for inventory/routing decision in a three-echelon logistics system. *Eur. J. Oper. Res.* 191(3):623–635.