Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

Two-stage stochastic programming for the inventory routing problem with stochastic demands in fuel delivery

Zhenping Li[a] and Pengbo Jiao[a*]

[a]*School of Information, Beijing Wuzi University, Beijing, Republic of China*

C H R O N I C L E

Article history:
Received April 3 2022
Received in Revised Format May 20 2022
Accepted July 27 2022
Available online July, 28 2022

Keywords:
Inventory routing
Fuel delivery
Two-stage stochastic programming
Benders decomposition
Two-phase heuristic

A B S T R A C T

The inventory routing problem (IRP) arises in the joint practices of vendor-managed inventory (VMI) and vehicle routing problem (VRP), aiming to simultaneously optimize the distribution, inventory and vehicle routes. This paper studies the multi-vehicle multi-compartment inventory routing problem with stochastic demands (MCIRPSD) in the context of fuel delivery. The problem with maximum-to-level (ML) replenishment policy is modeled as a two-stage stochastic programming model with the purpose of minimizing the total cost, in which the inventory management and routing decisions are made in the first stage while the corresponding recourse actions are implemented in the second stage. An acceleration strategy is incorporated into the exact single-cut Benders decomposition algorithm and its multi-cut version respectively to solve the MCIRPSD on the small instances. Two-phase heuristic approaches based on the single-cut decomposition algorithm and its multi-cut version are developed to deal with the MCIRPSD on the medium and large-scale instances. Comparing the performance of the proposed algorithms with the Gurobi solver within limited time, the average objective value obtained by the proposed algorithm has decreased more than 7.30% for the medium and large instances, which demonstrates the effectiveness of our algorithms. The impacts of the instance features on the results are further analyzed, and some managerial insights are concluded for the manager.

1. Introduction

This paper focuses on a multi-vehicle multi-compartment inventory routing problem with stochastic demands that occurs, for instance, in the background of petrol station replenishment. This problem is also denoted as the petrol station replenishment problem, fuel replenishment (delivery) problem, or refined oil secondary distribution problem (Cornillier et al., 2008; Chowmali & Sukto, 2020; Li & Zhang, 2020). This problem is always considered as a vehicle routing problem due to vehicle distribution cost accounting for more than 60% of the total cost of the current refined oil logistics system (Wang et al., 2020). However, one of the best applications now for collaboration between refined oil depots and petrol stations is a vendor-managed inventory (VMI) system in refined oil supply chain management. Under VMI, the oil supplier is responsible for managing the inventories from all petrol stations. Hence, the oil distributor is then faced with solving an integrated problem of determining replenishment quantities and timing for petrol stations and organizing vehicle routes for those replenishments simultaneously. Such an integrated problem in the literature is well known as the Inventory Routing Problem (IRP) (Coelho et al., 2014).

Generally, a vehicle is divided into several identical compartments, and the liquid oil in one compartment must be delivered completely to a single customer for the consideration of transportation safety. In addition, vehicles loaded with hazardous chemicals such as refined oil are forbidden to perform replenishment tasks during the day, and can only occur during 0 am to 5 am. Petrol stations or other customers are replenished only once by a fleet of heterogeneous vehicles from a single oil depot during the working time.

* Corresponding author
E-mail: 15735172173@163.com (P. Jiao)

2022 Growing Science Ltd.
doi: 10.5267/j.ijiec.2022.7.004

Most of the literature on such a problem assumes a constant consuming rate at each customer, a fleet of identical vehicles at the distributor, and focuses on developing solution methods that design proper quantities and vehicle route schedules. In fact, the actual demand at each customer is often stochastic, vehicles at the supplier are usually heterogeneous, each type vehicle might be equipped with a different number of compartments. After the actual demand of each customer is observed at the end of the period, the recourse cost can be calculated by the supplier. Failing to take these facts into account would certainly result in more cost of the collaboration system such as VMI.

Our problem then could be seen as a combination of two classical variants of the inventory routing problem (IRP): an inventory routing problem with multi-vehicle multi-compartment (MCIRP) and an inventory routing problem with stochastic demands (IRPSD). For the convenience of expression, our problem is defined as the multi-compartment inventory routing problem with stochastic demands (MCIRPSD).

The remainder of this paper is organized as follows: Section 2 provides a review of the literature related to this paper. Section 3 describes our problem, introduces assumptions and notation, and then gives two formal mathematical models. After that, the solution methodology is developed in Section 4 and computational results, based on MCIRPSD instances, are discussed in Section 5. Finally, conclusions and future directions for further research are presented in Section 6.

2. Literature review

The inventory routing problem (IRP) has received more and more attention in recent decades. The most recent surveys of the IRPs are provided by Andersson et al. (2010), Mosca et al. (2019), Raa and Aghezzaf (2009), Roldán et al. (2017). In addition, Coelho et al. (2014) and Adulyasak et al. (2015) present a comprehensive literature review on the IRP, based on the main characteristics e.g., variants, model, and solution approaches.

There are many variants of the IRP, one of the important features is whether to consider the type of vehicle and the number of compartments. The majority of the literature only considers a fleet of homogeneous vehicles consisting of a single compartment, heuristic approaches and exact algorithms are developed for this single-vehicle single-compartment IRP with various constraints (Gruler et al., 2020; Hiassat et al., 2017; Wang et al., 2018). Several IRP literatures about multi-vehicle or multi-compartment can be found in green inventory routing problems (Cheng et al., 2017; Micheli & Mantella, 2018), livestock collection problems (Oppen et al., 2010), and fuel replenishment problems (Popović et al., 2012).

Another essential stream of the IRP discusses two fundamental replenishment policies. The first replenishment policy introduced by Archetti et al. (2012) is called order-up-to-level policy (OU), once the customer is visited, the replenishment volume delivered will restore the inventory level to maximum storage capacity. The second replenishment policy called the maximum-to-level policy (ML) is a relaxation of the OU policy, which relaxes the requirement that the quantities received must restore the maximum inventory capacity at the customer. Obviously, the second replenishment policy has more flexibility and complexity, which results in higher challenges. The multi-vehicle multi-product IRP with the ML policy was proposed by Nikzad et al. (2019), who adopt a matheuristic algorithm. Mirzapour et al. (2019) developed a hybrid algorithm based on the L-shaped method for the multi-objective IRP with multi-vehicle multi-product and OU policy described above. More research for the OU and ML policies has been investigated in Su et al. (2020) and Zhang et al. (2021).

Special cases of the continuous-time IRP were also studied. Usually, researchers discretize the infinite or finite time and add an extra constraint on the maximum working time of the vehicle during each period. The work studied by Raa and Aghezzaf (2009) belongs to the continuous-time version. Several papers related to this research have presented heuristic solution approaches (Chitsaz et al., 2016; Hemmati et al., 2016; Raa & Dullaert 2017). For the first cyclic IRP with stochastic demands, additional safety stock, and expedited replenishments are used by Raa and Aouam (2021) to buffer the demand variability, the algorithm incorporates the local-search procedure into a metaheuristic.

Studies on the stochastic demand IRP reformulate the problem as a two-stage stochastic programming model (Agra et al., 2018; Cho et al., 2018). The modeling idea we adopt in this article belongs to this method. Li et al. (2020) constructed a two-stage stochastic model for the IRP in the context of the fuel distribution with multi-depot multi-vehicle, where determining the routes and quantities lies in the first stage and deciding the recourse actions is in the second stage. The reformulated deterministic equivalent model was solved by a CPLEX solver. For the single-vehicle allocation problem of the limited medical reserves, Zhang et al. (2021) regarded the donated supplies as the recourse action, the objective is to minimize the allocation cost and the expected recourse cost. The optimal solutions are obtained using Gurobi solver. This modeling approach can be found in other applications, such as transportation network protection problems (Liu et al., 2009), biodiesel supply chains problems (Marufuzzaman et al., 2014), emergency system management problems (Moreno et al., 2018; Wang 2020). However, the multi-vehicle multi-compartment IRP mentioned above focuses on the situation where the demands of the customers are deterministic. Moreover, the existing literature on the IRP with stochastic demands rarely considers both continuous-time during a single period and the multi-vehicle multi-compartment. According to the real situation, all features must be taken into account in constructing the distribution scheme. To the best of our knowledge, Li et al. (2020) studied the multi-vehicle multi-compartment IRP under stochastic demands. In order to reduce the complexity of the problem, they only consider the direct distribution mode (one vehicle only visits a single customer) in their paper without organizing optimal routes. Also, maximum working time for each vehicle during a period is not considered. The purpose of this paper is to fill this gap in the research by modeling the MCIRPSD when the time is continuous and developing a powerful solution method.

The main contributions of this research are summarized as follows:

First, a comprehensive variant of the IRP is studied, considering multi-compartment, stochastic demand, a fleet of heterogeneous vehicles and their maximum working time during a single period in fuel delivery. This problem of a single period can be further extended to the case of an infinite period.

Second, a two-stage stochastic programming model is proposed to design the quantities delivered to customers, the number of employed vehicles and corresponding vehicle routes. The objective of the model is to minimize the total cost consisting of distribution cost (fixed and variable costs for vehicles, loading costs for compartment) and expected recourse cost (shortage costs and inventory costs).

Finally, the accelerated Benders decomposition algorithms including single-cut algorithm and multi-cut version are developed to solve the two-stage model of smaller instances. For larger instances, the accelerated decomposition approaches are treated as the second-phase to improve the performance of the two-phase heuristic algorithms. Computational and managerial insights can be obtained based on the numerical experiments and the results.

3. Problem definition and mathematical model

In this section, the MCIRPSD is formally defined and the two-stage stochastic programming model is given. The MCIRPSD problem is considered where customers are replenished with a single refined oil product by a fleet of heterogeneous vehicles from a supplier depot (e.g., refined oil depot) at the beginning of a single period. Every customer's daily demand (sales volume) is stochastic. The oil depot has sufficient available supply (Raa& Aouam, 2021). Fig. 1 presents the timing of the task we assume in this study.

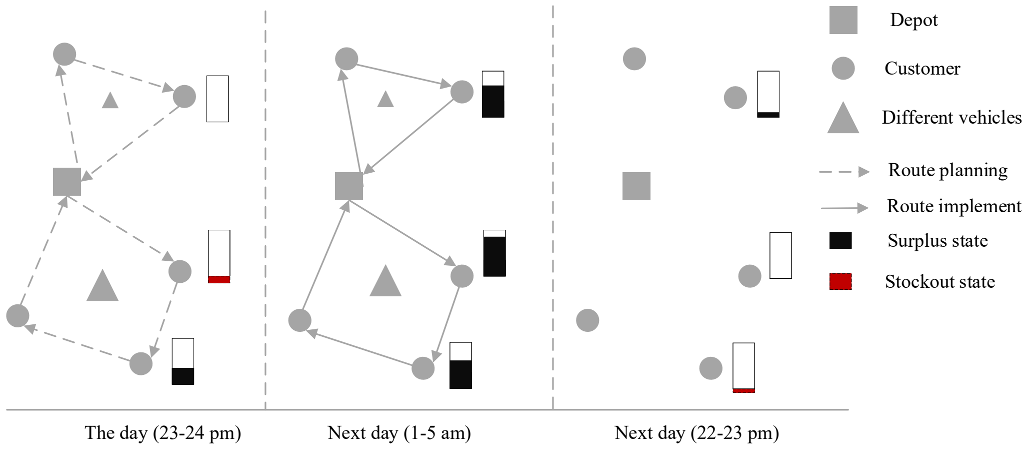

Fig. 1. Timing of the task in the MCIRPSD

At the end of each day (e.g., from 23 pm to 24 pm), the supply depot makes a distribution plan for the next day according to each customer's inventory level and limited storage capacity, the deliveries are performed before the beginning of the next daily sales task (e.g., from 1 am to 5 am), and the recourse costs are calculated for all customers at the end of the next day (e.g., from 22 pm to 23 pm). There are several types of vehicles available at the supplier depot, the number of compartments in any type vehicle differs from others. Each customer can be replenished at most once during maximum working time for every vehicle per period. Each vehicle takes the same time to unload oil products at any of the customers. The customer's storage capacity and replenishment times are limited while demand is uncertain per period, then three cases might happen at the end of each period (ⅰ) customer's demand is larger than the sum of replenishment and initial inventory (inventory level from previous period), this means that the customer will be in a stockout state; (ⅱ) the demand is less than the sum of replenishment and initial inventory, in this case the customer will be in a surplus state; (ⅲ) the sum of replenishment and initial inventory satisfies the daily demand accurately. As mentioned above, the MCIRPSD could be formulated as a two-stage stochastic programming model. In the first stage, before the realization of the stochastic demand, the model decides (1) when and how much oil product to deliver to the customer; (2) how to organize customers into different routes during a single period; (3) how to assign each route to a proper vehicle. In the second stage, while the stochastic information has been revealed, the model decides (1) how much shortage costs will be paid due to case (ⅰ); or (2) how much inventory costs will be paid for case (ⅱ) or other costs to pay for dealing with the surplus inventory.

The objective of the MCIRPSD is to minimize the sum of the first-stage cost and the recourse cost (second-stage cost) per period. The first-stage cost consists of fixed costs for vehicles, loading costs for oil products and routing costs. The recourse cost includes shortage costs for the stockout and inventory costs for the surplus inventory. In addition, the ML policy is adopted as a replenishment policy because of the high flexibility.

3.1 Assumptions

There are several necessary assumptions itemized:

(1) Each vehicle capacity and customer replenishment are integer multiples of a single compartment's capacity.
(2) Before the start of the daily sales task, each customer's inventory level is equal to the previous period and remains unchanged until it is replenished.
(3) The daily stochastic demand of each customer follows a Poisson distribution with identical known parameters (Li et al., 2020).

3.2 Notation

To formulate the MCIRPSD model, we define the following notations.

Sets

$N=\{0,n+1\}\cup C$: set of nodes, consisting of an oil depot 0 ($n+1$) and a set of customers $C=\{1,2,...,n\}$;
$V=\{1,2,...,v\}$: set of vehicle types;
$K=\{1,2,...,k\}$: set of vehicles for each type.

Parameters

d_{ij}: distance or travel time from $i\in N$ to $j\in N$;
Q_v: capacity of vehicle type $v\in V$, the capacity is an even multiple of the compartment capacity;
f_v : fixed cost per period for vehicle type $v\in V$;
c_v : fixed cost for loading oil in one compartment of vehicle type $v\in V$;
u_i : fixed service time at customer $i\in C$;
R_i : storage capacity at customer $i\in C$;
I_i : initial inventory level at customer $i\in C$;
r_i : shortage cost per unit at customer $i\in C$;
h_i : inventory holding cost per unit at customer $i\in C$;
E : maximum working time per period for each vehicle type $v\in V$;
M : an enough large number.

Stochastic parameters
ξ : stochastic scenario corresponding to all customers' demands;
$S=\{\zeta_1,\zeta_2,...,\zeta_{|S|}\}$: set of all realizations of scenario ξ;
$q_i(\xi)$: actual demand of customer i under scenario ξ.

Decision variables
e_{vk}^{i} : the time when vehicle $k\in K$ of type $v\in V$ arrives at customer $i\in C$;
x_{vk}^{i} : non-negative integer variable corresponding to delivery quantity when customer $i\in C$ is replenished by vehicle $k\in K$ of type $v\in V$ (integral multiple of a single compartment's capacity);
y_{vk}^{ij} : 1 if $i\in N$ and $j\in N$ are visited via route (i,j) by vehicle $k\in K$ of type $v\in V$, 0 otherwise;
z_{vk} : 1 if vehicle $k\in K$ of type $v\in V$ is being used, 0 otherwise;
$w_i^+(\xi)$: non-negative continuous variable corresponding to surplus inventory level at customer i when ξ is realized;
$w_i^-(\xi)$: non-negative continuous variable corresponding to amount of shortage at customer i when ξ is realized.

3.3 Two-stage stochastic programming model

The first-stage mathematical model of the MCIRPSD is as follows:

$$\min \sum_{v\in V}\sum_{k\in K}(f_v z_{vk} + \sum_{i\in N} c_v x_{vk}^i + \sum_{i\in N}\sum_{j\in N} d_{ij} y_{vk}^{ij}) + E_\xi(Q(x,\xi)) \tag{1}$$

subject to

$$\sum_{j\in C} y_{vk}^{0j} = z_{vk},\ \forall v\in V,\ k\in K \tag{2}$$

$$\sum_{i\in C} y_{vk}^{in+1} = z_{vk},\ \forall v\in V,\ k\in K \tag{3}$$

$$\sum_{i\in N} y_{vk}^{ij} = \sum_{i\in N} y_{vk}^{ji}, \forall j\in C, i\neq j, v\in V, k\in V \tag{4}$$

$$\sum_{i\in C} x_{vk}^i \le Q_v, \forall v\in V, k\in K \tag{5}$$

$$x_{vk}^i \le \sum_{j\in N} Q_v y_{vk}^{ij}, \forall i\in C, i\neq j, v\in V, k\in K \tag{6}$$

$$\sum_{j\in N}\sum_{v\in V}\sum_{k\in K} y_{vk}^{ij} \le 1, \forall i\in C, i\neq j \tag{7}$$

$$e_{vk}^i + u_i + d_{ij} \le e_{vk}^j + M(1-y_{vk}^{ij}), \forall i,j\in N, i\neq j, v\in V, k\in K \tag{8}$$

$$0\le e_{vk}^i \le E, \forall i\in N, v\in V, k\in K \tag{9}$$

$$0\le x_{vk}^i \le R_i - I_i, \forall i\in C, v\in V, k\in K \tag{10}$$

$$y_{vk}^{ij}\in\{0,1\}, \forall i,j\in N, i\neq j,\ v\in V,\ k\in K \tag{11}$$

$$z_{vk}\in\{0,1\}, \forall v\in V, k\in K \tag{12}$$

The second-stage mathematical model is:

$$Q(x,\xi) = \min\left(\sum_{i\in C} r_i w_i^-(\xi) + \sum_{i\in C} h_i w_i^+(\xi)\right) \tag{13}$$

subject to

$$I_i + \sum_{v\in V}\sum_{k\in K} x_{vk}^i + w_i^-(\xi) - w_i^+(\xi) = q_i(\xi), \forall i\in C \tag{14}$$

$$w_i^-(\xi)\ge 0, \forall i\in C \tag{15}$$

$$w_i^+(\xi)\ge 0, \forall i\in C \tag{16}$$

The first-stage problem determines the routes of vehicles, the delivery quantity and arriving time to each customer before demand is revealed. The objective function (1) minimizes the distribution cost and recourse cost per period, consisting of the fixed costs for vehicles, the loading costs for compartments and the route costs, where $E_\xi(Q(x,\xi))$ denotes the expected recourse cost for the second-stage problem. Constraints (2) and (3) imply each vehicle being used starts from and returns to the oil depot respectively. Constraints (4) represent flow balance. Constraints (5) indicate that the total delivery quantity from a single route cannot exceed the vehicle's capacity. Constraints (6) impose the delivery quantity constraint on a visited customer. Constraints (7) ensure that every customer is replenished at most once during one period. Constraints (8) define the arrival time constraint of the vehicle between two adjacent customers. Constraints (9)-(12) describe the domains of the first-stage decision variables.

After the demand is known, the second-stage problem minimizes the recourse cost based on the first-stage variable x, as stated in formulation (13). Constraints (14) design the relationship between the customer's initial inventory, replenishment, shortage, surplus, and actual demand. The rest constraints describe the non-negativity of the second-stage variables.

3.4 Deterministic equivalent mixed-integer programming model

This study focuses on the two-stage stochastic programming under the realizations of finite scenarios $S=\{\zeta_1,\zeta_2,...,\zeta_{|S|}\}$. Lets $s\in S$ to be the realization of the stochastic demand under scenario ξ. The probability of occurrence for any realization $s\in S$ is regarded as p_s, where $\sum_{s\in S} p_s = 1$. For brevity, $w_{is}^+, w_{is}^-, \forall s\in S$ denote the amount of surplus and shortage at customer i under realization s, while $q_{is}, \forall s\in S$ is the actual demand level at customer i under realization s. As discussed above, the two-stage stochastic programming model of the MCIRPSD can be specified as an equivalent deterministic mixed-integer formulation:

$$\min \sum_{v\in V}\sum_{k\in K}(f_v z_{vk} + \sum_{i\in N} c_v x_{vk}^i + \sum_{i\in N}\sum_{j\in N} d_{ij} y_{vk}^{ij}) + \sum_{s\in S} p_s\left(\sum_{i\in C} r_i w_{is}^- + \sum_{i\in C} h_i w_{is}^+\right) \tag{17}$$

subject to (2)-(12)

$$I_i + \sum_{v \in V} \sum_{k \in K} x_{vk}^i + w_{is}^- - w_{is}^+ = q_{is}, \forall i \in C, s \in S \tag{18}$$

$$w_{is}^- \geq 0, \forall i \in C, s \in S \tag{19}$$

$$w_{is}^+ \geq 0, \forall i \in C, s \in S \tag{20}$$

Note that the two-stage stochastic programming model is a complete recourse problem due to its second-stage problem being always feasible for any given first-stage decision variables values (Birge & Louveaux 2011).

4. Solution approach

The two-phase heuristic approaches are based on accelerated Benders decomposition algorithms, whose primal version called classical Benders decomposition was formally proposed by Benders (1962). Basically, the classical Benders decomposition is a procedure in which all decision variables are divided into distinctive subsets so that the corresponding solutions of each subset are not processed considering all constraints simultaneously. This decomposition approach was later extended and applied by Van Slyke and Wets (1969), who introduced the so-called L-shaped method.

This approach decomposes the primal problem into two relatively simpler formulations, called master problem (BMP) and subproblem (BSP), respectively (Zhang et al., 2021). For two-stage stochastic programming, the first-stage problem is modeled as a master problem and each realization of the second-stage problem is regarded as a subproblem naturally. The master problem is solved as a relaxation of the primal problem and provides first-stage variables values for the subproblem to get a feasible solution. At each iteration, the Benders cuts (i.e., feasibility and optimality cuts) are generated and added to the master problem to improve the feasible solution obtained in the last iteration, such cuts could be seen as new constraints that must be satisfied by the master problem. Both problems are solved iteratively until the optimal solution to the primal problem is found.

4.1 Classic Benders decomposition

In this section, we designed the classical Benders decomposition method to solve our two-stage stochastic programming model. Naturally, the Benders master problem (BMP) is as follows:

$$\text{(BMP)} \ \min \sum_{v \in V} \sum_{k \in K} (f_v z_{vk} + \sum_{i \in N} c_v x_{vk}^i + \sum_{i \in N} \sum_{j \in N} d_{ij} y_{vk}^{ij}) + E_\xi (Q(x, \xi)) \tag{21}$$

subject to (2)-(12)

The Benders subproblem (BSP) is given by

$$\text{(BSP)} \ \ Q(x,s) = \min p_s \left(\sum_{i \in C} r_i w_{is}^- + \sum_{i \in C} h_i w_{is}^+ \right) \tag{22}$$

subject to (19)-(20)

$$w_{is}^- - w_{is}^+ = q_{is} - I_i - \sum_{v \in V} \sum_{k \in K} x_{vk}^i, \forall i \in C, s \in S \tag{23}$$

Firstly, the classical decomposition algorithm is designed based on the single-cut, which aggregates the dual information of the linear programming subproblems corresponding to all realizations of the stochastic scenarios to find optimality cuts. Then the single-cut based algorithm is extended to the multi-cut version decomposition algorithm. The multi-cut approach makes full use of the dual information associated with each subproblem of the random realization to generate the optimality cuts without losing any information. Furthermore, since the stochastic programming model proposed in Section 3.4 is a complete recourse problem, the feasibility cuts need not be considered (Noyan, 2012).

4.1.1 Single-cut based Benders decomposition algorithm

The BMP is a relaxation of the primal problem, its optimal value provides a lower bound for the primal optimal function value. In addition, the single-cut decomposition algorithm will solve $|S|$ subproblems associated with all realizations of the stochastic demands. The subproblems' total objective value is equal to the sum of the expected function values over all subproblems. Then, the Benders master problem (BMP 1) in single-cut algorithm can be reformulated as follows:

$$\text{(BMP 1)} \ \ \min \sum_{v \in V} \sum_{k \in K} (f_v z_{vk} + \sum_{i \in N} c_v x_{vk}^i + \sum_{i \in N} \sum_{j \in N} d_{ij} y_{vk}^{ij}) + \theta \tag{24}$$

subject to (2)-(12)

$$\theta \geq 0 \tag{25}$$

where θ is the lower bound of subproblems' total objective function value, while the upper bound is denoted by $\bar{\theta} = \sum_{s \in S} Q(x,s)$.

The Benders subproblem (BSP 1) in single-cut algorithm is:

$$\text{(BSP 1)} \quad Q(x,s) = \min p_s \left(\sum_{i\in C} r_i w_{is}^- + \sum_{i\in C} h_i w_{is}^+ \right) \tag{22}$$

subject to (19)-(20)

$$w_{is}^- - w_{is}^+ = q_{is} - I_i - \sum_{v\in V}\sum_{k\in K} x_{vk}^i, \forall i \in C \tag{23}$$

Algorithm 1. Single-cut based Benders decomposition algorithm (SC).

```
1: Initialize iterative counter $v = 0$, set the upper bound of the subproblems' total objective value $\overline{\theta} = +\infty$.
2: If $v = 0$, $\theta$ in the objective function is 0. Otherwise, add the actual value of $\theta$ to the objective function.
3: Solve the BMP 1 to optimality. Let ($x^*, \theta^*$) be the optimal solution from the BMP 1.
4: Solve all BSPs 1 after adding the optimal solution $x^*$ to each BSP 1, and then let $\pi_s^T$ denotes the dual vector corresponding to the $s$ th BSP, $\forall s \in S$. Update the total function value of the subproblems $\overline{\theta} = \sum_{s\in S} p_s \pi_s^T (e_s - T_s x^*)$.
5: If $\overline{\theta} > \theta^*$, initialize the optimality cut with $\pi_s^T$ and add the cut $\sum_{s\in S} p_s \pi_s^T (e_s - T_s x) \le \theta$ to BMP 1. $v=v+1$, then return to Step 2.
6: If $\overline{\theta} = \theta^*$, then stop and return the optimal BMP 1 solution $x^*$.
```

The stopping criterion $\overline{\theta} = \theta^*$ (Step 6) here need to be relaxed to guarantee that the algorithm can eventually converge and terminate with an acceptable optimality gap ε. For the single-cut decomposition algorithm, $\overline{\theta}$ is the upper bound of the subproblems' total cost and θ^* is the lower bound. Then, the optimal termination criterion for the single-cut algorithm can be written as follows:

$$\frac{\overline{\theta} - \theta^*}{\overline{\theta}} \le \varepsilon$$

The algorithm is terminated when the optimal stopping criterion is achieved, where the optimality gap ε is a given stopping tolerance value (Noyan 2012, Sikora 2021).

Note that in Step 5, $e_s = \begin{bmatrix} q_{1s} - I_1 \\ q_{2s} - I_2 \\ \vdots \\ q_{Cs} - I_C \end{bmatrix}_{C\times 1}$ and the coefficient matrix $T_s = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}_{C\times C}$.

4.1.2 Multi-cut based Benders decomposition algorithm

To design the multi-cut based Benders composition algorithm, we introduce extra variables $\theta_s, \forall s \in S$ associated with the lower bound of each subproblem's expected function value to reconstruct the master problem's objective function.

Then, the BMP 2 in multi-cut algorithm can be reformulated as follows:

$$\text{(BMP 2)} \quad \min \sum_{v\in V}\sum_{k\in K} (f_v z_{vk} + \sum_{i\in N} c_v x_{vk}^i + \sum_{i\in N}\sum_{j\in N} d_{ij} y_{vk}^{ij}) + \sum_{s\in S} \theta_s \tag{26}$$

subject to (2)-(12)

$$\theta_s \ge 0, \forall s \in S \tag{27}$$

Similarly, the upper bound of each subproblem's expected objective function value is denoted by $\overline{\theta}_s = EQ(x,s)$.
The Benders subproblem (BSP 2) in multi-cut algorithm is:

$$\text{(BSP 2)} \quad Q(x,s) = \min p_s \left(\sum_{i\in C} r_i w_{is}^- + \sum_{i\in C} h_i w_{is}^+ \right) \tag{22}$$

subject to (19)-(20)

$$w_{is}^- - w_{is}^+ = q_{is} - I_i - \sum_{v\in V}\sum_{k\in K} x_{vk}^i, \forall i \in C \tag{23}$$

Algorithm 2. Multi-cut based Benders decomposition algorithm (MC).

```
1: Initialize iterative counter $v = 0$, set the upper bound of each subproblem's expected objective value $\bar{\theta}_s = +\infty, \forall s \in S$.
2: If $v = 0$, $\theta_s, \forall s \in S$ in the objective function is 0. Otherwise, add the actual value of $\theta_s$ to the objective function.
3: Solve the BMP 2 to optimality and get the optimal solution ($x^*, \theta_s^*, \forall s \in S$).
4: Solve all BSPs 2 after adding the optimal solution $x^*$ to each BSP 2, and then let $\pi_s^T$ denote the dual vector corresponding to the $s$ th BSP, $\forall s \in S$.
5: If $\exists \bar{\theta}_s > \theta_s^*$, generate the optimality cut with $\pi_s^T$ and add all the cuts $p_s \pi_s^T (e_s - T_s x) \le \theta, \forall s \in S$ to (26). $v = v + 1$, then return to Step 2.
6: If $\bar{\theta}_s = \theta_s^*, \forall s \in S$, then stop and return the optimal BMP 2 solution $x^*$.
```

It is clear that the $e_s, \forall s \in S$ and matrix $T_s, \forall s \in S$ in Step 5 are the same as those in the single-cut algorithm.

Furthermore, a similar termination criterion for the multi-cut algorithm can be represented as follows:

$$\frac{\bar{\theta} - \theta^*}{\bar{\theta}} = \frac{\sum_{s \in S} (\bar{\theta}_s - \theta_s^*)}{\sum_{s \in S} \bar{\theta}_s} \le \varepsilon$$

The algorithm should be terminated when the optimal stopping criterion is achieved, where the optimality gap ε is the tolerance value adopted in the single-cut version.

4.2 Acceleration strategy

The classical decomposition algorithm SC or MC can be applied to solve the two-stage stochastic programming model directly, in which the first-stage problem contains plenty of integer variables. That means the algorithm has to solve a mixed-integer programming formulation. In other words, the algorithm would deal with a huge computational challenge, because more CPU time will be consumed at each iteration to obtain the master problem's optimal solution.

In this study, a simple strategy is incorporated into Step 3 to improve the efficiency of the decomposition algorithms. This strategy is to restrict the solution time of Step 3 to reduce each BMP's computing time. At each iteration, this strategy allows the optimal solution of the BMP to be replaced by a feasible solution obtained within a restricted computing time. Regardless of the feasible solution's quality, the algorithm accepts the feasible solution and executes subsequent steps such as constructing optimality cuts and inserting them to the BMP. Such optimality cuts can always continuously optimize the solution space of the BMP in each iteration, may be at the cost of more iterations.

Other acceleration strategies have been embedded into the decomposition algorithm to generate optimal (or near-optimal) solutions to the Benders master problem more efficiently (Laporte & Louveaux 1993). The multi-cut decomposition algorithm with acceleration strategy and the single-cut version are called "A-MC" and "A-SC", respectively.

4.3 Two-phase heuristic algorithm

Although the above-mentioned acceleration technology can speed up the convergence of the decomposition algorithms. It is still hard for the improved algorithms to solve more complicated instances, because relative low-quality solutions obtained in limited time are quite difficult to produce satisfactory optimality cuts, especially for our MCIRPSD problem. As indicated above, the number of iterations and computation time required by the algorithms may still be unacceptable in practice.

To remedy this issue, inspired by different distribution modes (i.e., the matching relationship between vehicle capacity and customer maximum demand), two-phase heuristic approaches based on the accelerated Benders decomposition algorithms are developed. In the first phase, the original customers set is divided into two subsets (i.e., the customers set using direct distribution mode and another set) and then the former set's total cost is calculated. In the second phase, the improved decomposition algorithms are respectively used to produce another set's distribution schemes. The two-phase heuristic algorithm can be summarized as Algorithm 3.

Algorithm 3
The two-phase heuristic algorithm

```
Input: Data on the customers, vehicles, costs, and stochastic parameters;
Output: The optimal solutions $x^*, y^*, z^*$, and the optimal total cost of
        primal problem $total\_cost$;
1: Initialize the customers set with direct delivery $D = \varnothing$, the
   corresponding number of large-capacity vehicles $direct\_vehicle = 0$,
   and the total cost of direct delivery $direct\_total = 0$;
2: Calculate each customer's maximum demand $mq_i$,
    $mq_i \leftarrow \max\{q_{is}\}_{s\in S}$;
3: Obtain the replenishment quantity $mq_i$ for the customer $i$.
4: **for** $i \in C$ **do**
5:   **if** $mq_i > Q_{max}/2$ **then**
6:     $D \leftarrow D \bigcup \{i\}$, $C \leftarrow C \setminus \{i\}$, $direct\_vehicle \leftarrow direct\_vehicle{+}1$;
7:   **else**
8:     $C \leftarrow C(i)$;
9:   **end if**
10:  $direct\_total \leftarrow direct\_total(D)$;
11: **end for**
12: Calculate the total quantity of vehicles $K = direct\_vehicle + \lfloor C/2 \rfloor$;
13: Call A-MC (or A-SC) to solve $C$, and get the solution and the optimal cost $optimize\_total$;
14: $total\_cost \leftarrow direct\_total + optimize\_total$;
15: **return** $x^*, y^*, z^*$, $total\_cost$
```

The details of the two-phase heuristic algorithm are listed as follows:

Stage 1. Divide the customers set into two subsets, and compute the total cost $direct\_total$ of the direct distribution scheme.

1. According to the known random demand scenario q_s, obtain each customer's maximum demand mq:

$$mq = \max\{q_s\}_{s\in S}$$

The maximum demand value is an integer.

2. By the first assumption, the replenishment quantity for each customer must be an integer multiple of a single compartment capacity. The maximum demand mq obtained in Step 1 is used as the replenishment quantity represented by the number of compartment. The advantage of this operation is that the quantity delivered to each customer will cover the demand fully during this period after completing a replenishment.

3. Assign the customer whose replenishment mq exceeds $Q_{max}/2$ to a group, and then assign the corresponding number ($direct\_vehicle$) of large-capacity vehicles to serve each customer directly without designing the optimal routes.

4. According to all the relevant parameters (i.e., distance, replenishment, and various costs) and the objective function, compute the total cost $direct\_total$ under the direct delivery mode.

5. The number of remaining customers denotes as $C - direct\_vehicle$ and the upper bound of the replenishment for such customers is $Q_{max}/2$. Determine the total quantity of vehicles K required to serve all customers:

$$K = direct\_vehicle + \left\lfloor \frac{(C - direct\_vehicle) * Q_{max}/2}{Q_{max}} \right\rfloor$$

Stage 2. For another group with the remaining customers (calculated as $C - direct\_vehicle$), the scale of this instance relative to the primal instance has been reduced. We can solve this instance associated with fewer customers and obtain the total cost $optimize\_total$ by the algorithms introduced in Section 4.2.

The total cost of the primal problem is denoted as follows:

$$total\_cost = direct\_total + optimize\_total$$

For simplicity, the two-phase algorithms based on the improved Benders decomposition algorithms are represented by "A-MC*" and "A-SC*".

5. Experiments

In this section, the computational results for some problem instances are presented to illustrate the effectiveness and performance of the methods described in Section 4.2 and 4.3. The optimization model and all the algorithms are coded in Python 3.8 and executed on a 64-bit workstation equipped with an Intel(R) Xeon(R) CPU E5-2650 clocked at 2.20 GHz and 24 GB RAM, running on a Linux operating system with 2 twelve-core. Gurobi 9.1.2 is used as the BMP problem solver and BSP problem solver in decomposition and two-phase algorithms. For the deterministic equivalent mixed-integer programming model proposed in Section 3.4, a time limit of 2500 seconds is imposed on the Gurobi solver per instance.

5.1 Problem instances

For the computational experiments, we adopt a subset of the R101 benchmark instance created by Solomon (1987) to test our algorithms. In this section, instances are designed with the following features:

(1). The instance size is between 10 and 50 customers, small instances are constructed with 10 and 20 customers (C=10, 20), medium instances are generated with 30 and 40 customers (C=30, 40), and the largest instances are developed with 50 customers (C=50).

(2). The vehicle type is set to 2 ($V = \{1,2\}$), the single compartment capacity is fixed at 5 tons. The vehicle capacity of the two types is 10 tons and 20 tons ($Q_v = 2,4, \forall v \in V$) respectively.

(3). Although there are a large number of scenarios for all customers, the demand scenarios from different customers are dependent, and all scenarios can be divided according to weather conditions such as extreme weather or not, and so on. As a result, it is reasonable to merge most similar scenarios into fewer ones (Li et al., 2020). Then, the number of scenarios with stochastic demand is between 2 and 5 (S=2,3,4,5).

(4). The inventory holding cost and shortage cost are 40 and 1000 yuan per ton per period ($h_i = 1000, r_i = 40, \forall i \in C$). The single compartment loading cost for all vehicles is 10 ($c_v = 10, \forall v \in V$).

(5). Each customer storage capacity is limited to 25 tons ($R_i = 5, \forall i \in C$). It is assumed that the initial inventory level at the beginning of each period is 0 ($I_i = 0, \forall i \in C$).

(6). The maximum working time for each vehicle is limited to 3 hours per period ($E = 180$). Loading time for all the vehicles is not considered in this study, while each vehicle takes 30 min ($u_i = 30, \forall i \in C$) to serve a customer.

(7). The fixed costs of two different vehicles are 100 and 200 yuan per period.

For both decomposition algorithms, the optimality gap ε is set to 0.05. For the route costs, Euclidean distance is used as the distance between two locations, then the sum of the length for all routes is regarded as the route costs. For the extension to the instance of random demand, we generate stochastic demands according to Poisson distribution with parameter λ=2 .

5.2 Performance of the algorithms

In the first experiment, the improved decomposition algorithms are validated by benchmarking them against classical algorithms. This experiment is conducted on 4 instances with 8 customers. Table 1 reports the results obtained on MCIPRSD instances. For each method, Table 1 gives the CPU time ("t") of the instances solved to optimality and the improvement percentage of the best solution time relative to the worst time ("imp"). The following naming convention is used to uniquely remark each instance: C-S, where C denotes the number of customers, S the number of scenarios. The computational results reported can be summarized as follows:

(1). All methods could solve the instances to optimality in 780.00 seconds.

(2). For the classical decomposition algorithms, the MC outperforms SC, and it is obvious that the SC has the worst performance in all instances. The maximum CPU time (minimum CPU time) of MC for all instances is 453.22 (108.95) seconds, while the CPU time of SC is 779.38 (300.39) seconds respectively.

(3). For the improved decomposition algorithms, A-MC is also better than A-SC, the maximum time (minimum time) of A-MC for all instances is 82.69 (59.63) seconds, while the CPU time of A-SC is 161.66 (68.38) seconds. The efficiency of both algorithms has been improved more than the classical versions. The minimum improvement percentage of the CPU time (imp) are 30.16% and 75.66% respectively.

Table 1
CPU time on the MCIRPSD with decomposition algorithms

C-S	MC	A-MC		SC	A-SC	
	t	t	imp(%)	t	t	imp(%)
8-2	290.59	59.63	79.48	300.39	68.38	77.24
8-3	108.95	76.09	30.16	495.04	100.28	79.74
8-4	178.50	71.99	59.67	779.38	127.31	83.67
8-5	453.22	82.69	81.75	664.13	161.66	75.66

5.2.1 Decomposition algorithms for small instances

The next experiment measures the performance of the improved decomposition algorithms on small instances. In this experiment, we solved 8 small instances with the Gurobi solver and improved algorithms respectively. As described above, running time is limited to 2500 seconds on the solver for the MCIRPSD model per instance. The computing time of 50 seconds is imposed on the problem BMP of the improved algorithms.

Detailed results for each method and instance are given in Table 2. The first column of the table gives the information of the instance. For each method, this table gives the best (optimal or near-optimal) objective function values ("best", "best$_M$", "best$_S$") computed within the limited computing time, the solution time ("t") in seconds, and the percentage gap between the algorithm's best cost and the solver's, calculated as $gap1 = (best_M - best) / best_M$ and $gap2 = (best_S - best) / best_S$.

For the instances of 10 customers, the solver solves all to optimality, the same optimal solutions for 3 instances (10-3, 10-4, 10-5) are also found by the A-SC approach, the method A-MC only finds the optimal solution for the 10-3 instance. Generally, the high-quality solutions (e.g., the highest gap is only 3.19%) are obtained by our algorithms for all instances. From the perspective of solution time, the running time of the solver is no more than 100 seconds, and the time does not show a strict increasing trend with the increase of the instance size. This trend is also reflected in the solution time of the A-MC algorithm. However, the CPU time of the A-SC algorithm gradually increases with the size of the instance increase.

For the instances of 20 customers, the solver can only get the near-optimal total cost (also called upper bound in other literature) in the limited time. Similarly, none algorithm obtained the optimal solutions, the percentage gap1 is between 1.22% and 2.84% while the gap2 is between 0.43% and 3.51%. The last row of this table presents the average value of all computational results. The average total cost value of the solver is the best among the three methods, although the A-MC method has poor performance, its average gap is only 1.40%. When comparing the solution time among the three methods, we observe that the A-MC significantly outperforms the Gurobi solver and A-SC algorithm.

Table 2
Detailed results on the small instances

C-S	Gurobi		A-MC			A-SC		
	best	t	best_M	t	gap1(%)	best_S	t	gap2(%)
10-2	2469.31	49.55	2496.21	67.43	1.08	2516.21	78.56	1.86
10-3	3017.43	17.45	3017.43	57.75	0.00	3017.70	175.24	0.00
10-4	3018.18	59.46	3035.86	109.34	0.58	3018.18	1047.77	0.00
10-5	3228.60	96.50	3335.00	178.56	3.19	3228.60	1151.34	0.00
20-2	4862.98	2500	4911.33	1505.83	0.98	4879.72	3352.41	0.34
20-3	5582.66	2500	5657.27	1913.05	1.32	5686.21	3459.84	1.82
20-4	5157.88	2500	5221.61	1614.32	1.22	5345.75	3209.68	3.51
20-5	5699.41	2500	5866.12	1564.91	2.84	5768.18	3910.48	1.19
Avg	4612.43	1527.87	4679.60	876.40	1.40	4667.53	2217.22	1.09

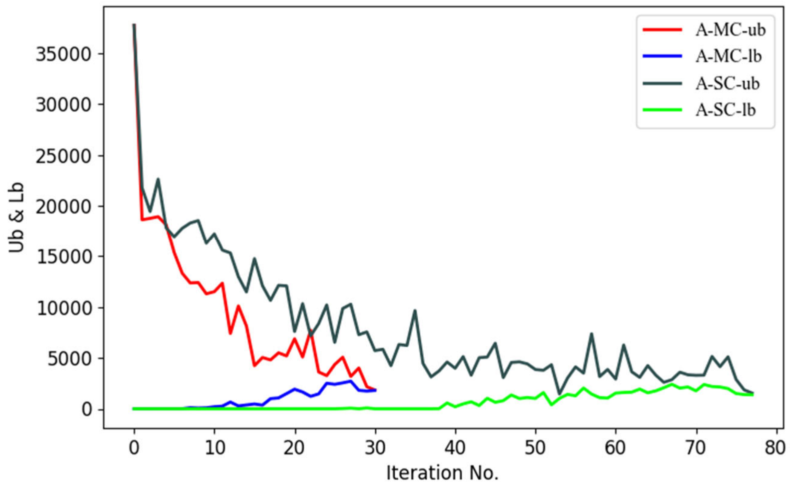

Fig. 2. Iterative process of the upper and lower bounds for subproblem of the 20-5 instance

Taking the 20-5 instance as the example, the upper bound $\overline{\theta}$ and lower bound θ^* for the problem BSP of both algorithms are shown in Fig. 2. The method A-MC requires 31 iterations while the A-SC algorithm needs 78 iterations. It is clear that the problem BSP's lower bound is continuously tight, but the upper bound always fluctuates up and down instead of falling steadily. Since the upper bound $\overline{\theta}$ associated with the first-stage feasible solution x^* needs to be obtained indirectly by solving the problem BMP, which is an MVMCIRP problem, the quality of the feasible solution obtained within the imposed time limit cannot be controlled. This is why the upper bound preforms such an unusual characteristic. However, from the overall trend, the feasible solution is approximating the optimal solution with the iteration of each algorithm.

5.2.2 Two-phase algorithms for medium and large instances

This experiment evaluates the two-phase heuristic algorithms for instances with more than 20 customers. We solved all 12 instances under increasing numbers of the customer, i.e., $C = 30, 40, 50$. In this section, the same solution time settings were adopted for the solver and the problem BMP in the decomposition algorithms. Table 3 shows the results of these methods on each instance, together with the average value and difference of these results. The columns "gap1(%)" and "gap2(%)" in Table 3 have a slight difference (i.e., added symbol "-") compared to the columns in Table 2 because these two methods (A-MC* and A-SC*) perform better than the solver in some medium instances (i.e., 30-2, 40-2, 40-4, 40-5) and all large instances.

When the number of customers increases, our two-phase algorithms are more and more powerful than the solver, particularly well on the 50-2 instance (the percentage gap reaching -47.45%). On the other hand, for those instances with the same customers, it is obvious that the computing time goes down when the number of scenarios increases. With larger customer and scenario numbers, this trend becomes more apparent, ranging from 4244.80 to 7.30 (6342.04 to 6.93) seconds for the large instances. These results in Table 3 confirm that better performance can be obtained by visiting some customers directly with more demand and without organizing the routes. This increases the actual number of vehicles required and fixed costs, but such a method decreases the customers who need replenishment and route design by the algorithms.

Table 3
Detailed results on the medium and large instances

C-S	Gurobi		A-MC*			A-SC*		
	best	t	$best_M$	t	gap1(%)	$best_S$	t	gap2(%)
30-2	8287.48	2500	8047.95	457.92	-2.89	8047.95	252.78	-2.89
30-3	8350.38	2500	8535.28	45.29	2.21	8535.28	32.49	2.21
30-4	9309.98	2500	9390.28	0.11	0.86	9506.28	0.15	2.11
30-5	8810.04	2500	9002.76	80.21	2.19	9002.76	116.67	2.19
40-2	10523.02	2500	10290.38	1002.14	-2.21	10255.84	1252.74	-2.54
40-3	10713.82	2500	11716.76	751.54	9.36	11716.93	900.32	9.36
40-4	12721.94	2500	12676.61	10.13	-0.36	12676.61	10.55	-0.36
40-5	15236.60	2500	13342.41	0.28	-12.43	13342.41	0.30	-12.43
50-2	34645.26	2500	18206.29	4244.80	-47.45	18811.57	6342.04	-45.70
50-3	16536.20	2500	14889.99	952.04	-9.96	14889.99	1253.78	-9.96
50-4	18424.75	2500	15825.07	842.32	-14.11	15825.07	889.34	-14.11
50-5	18926.13	2500	16013.62	7.30	-15.39	16004.50	6.93	-15.44
Avg	14373.80	2500	12328.17	699.51	-7.52	12384.60	921.51	-7.30

Next, Table 4 reports more detailed costs obtained by the three methods described above about the instances with 40 and 50 customers. Note that the direct delivery is applied to serve the customer whose maximum demand exceeds $Q_{max}/2$ (more than 2 compartments in our settings) in the first-phase algorithm. Firstly, we get the sum of the vehicle costs, compartment loading costs and routing costs, called the first-stage objective function value from the Gurobi solver and this cost value is denoted by "$cost_f$", and then remark the recourse cost consisting of shortage costs and inventory holding costs as "$cost_r$". In the following, the cost of direct delivery is denoted by "$cost_{Md}$" ("$cost_{Sd}$") in Table 4, including vehicle fixed costs, compartment loading costs, and routing costs in the first phase. Similarly, the sum of the vehicle fixed costs, loading costs and routing costs obtained from the second-phase algorithms (i.e., A-MC and A-SC methods) is denoted by "$cost_{Mg}$" ("$cost_{Sg}$"). Finally, we record the recourse costs obtained from both algorithms as "$cost_{Mr}$" and "$cost_{Sr}$". With the number of customers and scenarios increases, higher demand for a customer will appear more frequently and the number of customers who need direct service increases, resulting in a significant increase in the direct delivery costs "$cost_{Md}$" and "$cost_{Sd}$". Conversely, fewer customers need the second-phase algorithms to arrange their optimal replenishment and routes, which leads to lower "$cost_{Mg}$" and "$cost_{Sg}$". In addition, it can be seen that the recourse costs "$cost_r$" in our two-phase algorithms are equal or smaller than the solver with increasing sizes of instances.

Table 4
Detailed costs of the three methods

C-S	Gurobi		A-MC*			A-SC*		
	$cost_f$	$cost_r$	$cost_{Md}$	$cost_{Mg}$	$cost_{Mr}$	$cost_{Sd}$	$cost_{Sg}$	$cost_{Sr}$
40-2	9403.02	1120.00	7210.14	1960.24	1120.00	7210.14	1925.70	1120.00
40-3	8587.42	2126.40	8675.38	1295.78	1745.60	8675.38	1318.35	1723.20
40-4	9920.34	2801.60	10027.32	770.09	1879.20	10027.32	770.09	1879.20
40-5	100810.00	5155.60	10620.26	456.15	2266.00	10620.26	456.15	2266.00
50-2	9399.26	25246.00	9864.54	1444.55	6897.20	9864.54	1457.03	7490.00
50-3	12387.80	4148.40	11546.90	1289.09	2054.00	11546.90	1289.09	2054.00
50-4	13077.15	5347.60	12124.12	1241.35	2459.60	12124.12	1241.35	2459.60
50-5	12107.33	6818.80	13025.82	628.60	2359.20	13025.82	619.48	2359.20

Furthermore, in order to compare intuitively the first-stage costs and the recourse costs obtained by the three methods, Table 5 shows the improvement percentage gap of the costs difference between the algorithms and solver. The gap formulations here are the improvement percentage of the algorithms costs relative to the solver costs: $imp1 = (cost_f - cost_{Mf}) / cost_f$ and $imp2 = (cost_f - cost_{Sf}) / cost_f$, where the first-stage cost "$cost_{Mf}$" can be calculated as $cost_{Mf} = cost_{Md} + cost_{Mg}$ and the cost "$cost_{Sf}$" is calculated as $cost_{Sf} = cost_{Sd} + cost_{Sg}$.

The differences for the first-stage costs "$cost_f$", "$cost_{Mf}$" and "$cost_{Sf}$" are relatively limited for most instances, but it is the second-stage cost that cause all the difference. The second-phase algorithms (A-MC and A-SC) do provide sufficient advantages for our two-phase algorithms, which lead to less recourse costs. At the same time, this effect is becoming more apparent with larger customer numbers, i.e., the highest improvement gap is 72.68%.

Table 5
Detailed results of the cost variability of the three methods

C-S	A-MC*			A-SC*		
	$cost_{Mf}$	imp1(%)	imp2(%)	$cost_{Sf}$	imp1(%)	imp2(%)
40-2	9170.38	2.47%	0.00%	9135.84	2.84%	0.00%
40-3	9971.16	-16.11%	17.91%	9993.73	-16.38%	18.96%
40-4	10797.41	-8.84%	32.92%	10797.41	-8.84%	32.92%
40-5	11076.41	-9.87%	56.05%	11076.41	-9.87%	56.05%
50-2	11309.09	-20.32%	72.68%	11321.57	-20.45%	70.33%
50-3	12835.99	-3.62%	50.49%	12835.99	-3.62%	50.49%
50-4	13365.47	-2.20%	54.01%	13365.47	-2.20%	54.01%
50-5	13654.42	-12.78%	65.40%	13645.30	-12.70%	65.40%

5.3 Sensitivity analysis of instance features

The influences of two features to the results are analyzed in this subsection, i.e., the number of employed vehicles and the ratio of fixed costs between the large-capacity vehicle and the small-capacity vehicle. The medium instance 30-5 with 26 vehicles is regarded as an appropriate example. Let the number of used vehicles vary from 24 to 28 while fixing other parameters. The total cost and running time corresponding to different numbers of vehicles are obtained by our two-phase algorithms. Fig. 3 presents the impact of the number of employed vehicles on the total cost. The total costs from both algorithms are almost the same affected by the number of employed vehicles. We can find that the total costs decrease significantly as the number increases from 24 to 25. When increasing this number up to 26, its impact on the total cost is very small. The total cost is no longer affected until the number exceeds 26.

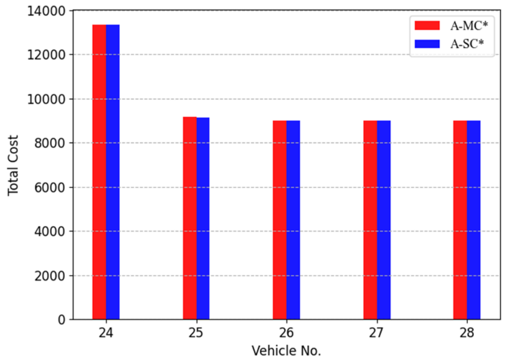

Fig. 3. Influence of different number of vehicles on total cost

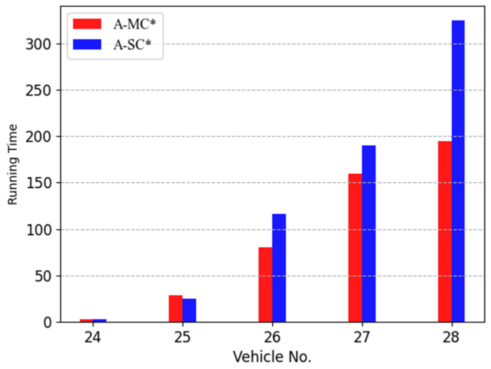

Fig. 4. Influence of different number of vehicles on running time

Fig. 4 illustrates how the running time of two-phase algorithms increases as the number of employed vehicles increases. According to the results detailed in Fig. 4, we can claim that the running time increases with the number of vehicles and the increased running time is relatively limited as the number of vehicles increases from 24 to 25. However, we can also make such a claim for the number of vehicles, i.e., a larger number (more than 25) result in a higher running time, especially for the method A-SC*.

Based on the above analysis, a larger number of vehicles avoids the occurrence of the higher total cost but hence leads to a longer running time to formulate the distribution schedule. As a trade-off, an appropriate amount can not only effectively decrease the total cost but also actually reduce the formulation time of the distribution policies, which illustrates the importance of an outstanding method to determine the vehicles' number.

For the medium instance 30-5 with 26 vehicles in the previous section, the fixed cost ratio is 2 (200 yuan for the large-capacity vehicle and 100 yuan for the smaller one). Let the ratio vary from 1 to 5 while fixing the cost of the small-capacity vehicle.

The total cost and the number of vehicles employed corresponding to different ratios are obtained by the two-phase algorithms.

Fig. 5 presents the influence of the fixed cost ratio on the total cost. The total costs from both algorithms are almost the same affected under the same ratio of the fixed cost. We can find that the total costs increase significantly with the ratio increases from 1 to 5. Obviously, a higher fixed cost ratio (higher fixed cost for the large-capacity vehicle) results in more total cost.

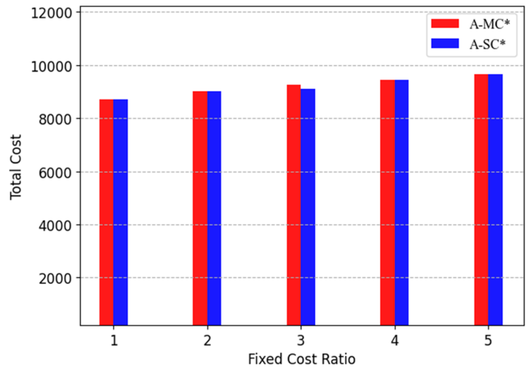

Fig. 5. Influence of different fixed cost ratios on total cost

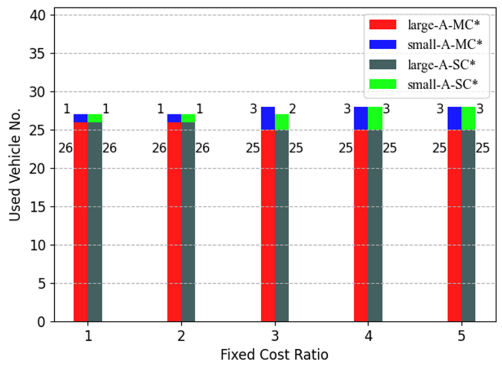

Fig. 6. Influence of different fixed cost ratios on the number of vehicles used

Fig. 6 presents the influence of the different fixed cost ratios on the number of vehicles used, including large-capacity vehicles and small-capacity ones. The number of the vehicles used by both algorithms are the same under the same ratio, except for the case where the ratio is 3. Since the fixed cost of the large-capacity vehicle is too large (the fixed cost of the large-capacity vehicle is at least 3 times of the small-capacity vehicle), which results in more fixed cost on each compartment of oil, so it cost-effective to use more small-capacity vehicles instead of one larger vehicle. Particularly, when the ratio exceeds 2, two additional small-capacity vehicles are used in most instances, and the number of the large-capacity vehicles used is decreased by 1. In addition, for instance 30-5 with 27 vehicles, when the ratio varies from 1 to 5, the corresponding number of the vehicles used is 27-0, 26-1, 24-4, 24-4, 24-4. It can be concluded that when the ratio is no more than 3, the larger the fixed cost ratio, the more small-capacity vehicles are used in the instance. As mentioned before, these experiments confirm that our two-phase solution approaches are quite extraordinary in designing routes, delivering replenishment, and employing applicable vehicles to search for solutions that provide a better total cost for the MCIRPSD.

6. Conclusions

This paper studies a new inventory routing problem in fuel delivery. Specifically, we consider a multi-vehicle multi-compartment inventory routing problem with stochastic demand and ML policy (MCIRPSD) in a single period. Before obtaining the accurate demand from the customers, the appropriate replenishment and routes are determined for all customers by balancing the distribution cost of vehicles and routes with the expected recourse cost that result from surplus and stocking out. To describe such a problem, a two-stage stochastic programming model is proposed and its deterministic equivalent MILP model is formulated.

The MCIRPSD consists of determining, in the first-stage, the delivery quantities and arriving time, the vehicles and routes that will be used in the single period. Therefore, the first-stage problem is a continuous-time, multi-vehicle multi-compartment inventory routing problem with ML policy, which contains integer variables and continuous variables and is NP-hard. In order to deal with the two-stage problem, improved decomposition algorithms and two-phase heuristic algorithms based on the former are presented. The proposed problem and model can be solved for different scale instances, and then different distribution strategies could be evaluated by the decision-makers. From the computational results, we can observe that the improved decomposition approaches can find the optimal and near-optimal solutions compared to the equivalent MILP model using Gurobi solver on instances with up to 20 customers. Moreover, our two-phase algorithms are quite powerful in obtaining good solutions for different medium and large instances with up to 50 customers.

In addition, sensitivity analysis is conducted to assess the impacts of the number of vehicles and the fixed cost ratio on the results. More vehicles obtain relatively stable total cost but result in unacceptable running time, less numbers get more total cost but satisfactory running time. According to the trade-off and preference of total cost and waiting time, a more appropriate number of vehicles can be employed based on the method in this paper. Once the fixed cost of the both vehicles is known and the fixed cost ratio is equal to 1, it is better to employ enough large-capacity vehicles for the distributor. When this ratio is more than 2, relatively more small-capacity vehicles and less large vehicles should be employed by the manager.

In terms of future research, we could consider many relevant extensions of the MCIRPSD. So far, we assume that the oil depot always has enough available inventory at the beginning of each period. Then considering a limited supply at the oil depot is the first extension. Another interesting extension for further research is the improvement of the decomposition algorithms. The solution time is limited for the problem BMP, but it is worthwhile to develop other more effective heuristic or exact approaches to accelerate the computing time for the BMP.

Acknowledges

This research was jointly supported by the National Natural Science Foundation of China (NSFC) under Project No. 71771028, Beijing Natural Science Foundation under Project No. 9212004 and No. Z180005.

Conflicts of interest/Competing interests

These authors contributed equally to this work and should be considered co-first authors. On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

Andersson, H., Hoff, A., Christiansen, M., Hasle, G., & Lokketangen, A. (2010). Industrial aspects and literature survey: Combined inventory management and routing. *Computers & Operations Research, 37*(9), 1515-1536.

Adulyasak, Y., Cordeau, J. F., & Jans, R. (2015). The production routing problem: A review of formulations and solution algorithms. *Computers & Operations Research, 55*, 141-152.

Archetti, C., Bertazzi, L., Hertz, A., & Speranza, M. G. (2012). A hybrid heuristic for an inventory routing problem. *INFORMS Journal on Computing, 24*(1), 101-116.

Agra, A., Requejo, C., & Rodrigues, F. (2018). An adjustable sample average approximation algorithm for the stochastic production-inventory-routing problem. *Networks, 72*(1), 5-24.

Birge, J. R., & Louveaux, F. (2011). Introduction to stochastic programming. Springer Science & Business Media.

Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische mathematik, 4*(1), 238-252.

Cornillier, F., Boctor, F. F., Laporte, G., & Renaud, J. (2008). A heuristic for the multi-period petrol station replenishment problem. *European Journal of Operational Research, 191*(2), 295-305.

Chowmali, W., & Sukto, S. (2020). A novel two-phase approach for solving the multi-compartment vehicle routing problem with a heterogeneous fleet of vehicles: a case study on fuel delivery. *Decision Science Letters*, *9*(1), 77-90.

Coelho, L. C., Cordeau, J. F., & Laporte, G. (2014). Thirty years of inventory routing. *Transportation Science, 48*(1), 1-19.

Cheng, C., Yang, P., Qi, M. Y., & Rousseau, L. M. (2017). Modeling a green inventory routing problem with a heterogeneous fleet. *Transportation Research Part E-Logistics and Transportation Review, 97*, 97-112.

Chitsaz, M., Divsalar, A., & Vansteenwegen, P. (2016). A two-phase algorithm for the cyclic inventory routing problem. *European Journal of Operational Research, 254*(2), 410-426.

Cho, J., Lim, G. J., Kim, S. J., & Biobaku, T. (2018). Liquefied natural gas inventory routing problem under uncertain weather conditions. *International Journal of Production Economics, 204*, 18-29.

Gruler, A., Panadero, J., de Armas, J., Perez, J. A. M., & Juan, A. A. (2020). A variable neighborhood search simheuristic for the multiperiod inventory routing problem with stochastic demands. *International Transactions in Operational Research, 27*(1), 314-335.

Hiassat, A., Diabat, A., & Rahwan, I. (2017). A genetic algorithm approach for location-inventory-routing problem with perishable products. *Journal of Manufacturing Systems, 42*, 93-103.

Hemmati, A., Hvattum, L. M., Christiansen, M., & Laporte, G. (2016). An iterative two-phase hybrid matheuristic for a multi-product short sea inventory-routing problem. *European Journal of Operational Research, 252*(3), 775-788.

Li, Z. P., Zhang, Y. W., & Zhang, G. W. (2020). Two-stage stochastic programming for the refined oil secondary distribution with uncertain demand and limited inventory capacity. *IEEE Access, 8*, 119487-119500.

Liu, C. Z., Fan, Y. Y., & Ordonez, F. (2009). A two-stage stochastic programming model for transportation network protection. *Computers & Operations Research, 36*(5), 1582-1590.

Laporte, G., & Louveaux, F. V. (1993). The integer L-shaped method for stochastic integer programs with complete recourse. *Operations Research Letters, 13*(3), 133-142.

Mosca, A., Vidyarthi, N., & Satir, A. (2019). Integrated transportation-inventory models: A review. *Operations Research Perspectives, 6*, 100101.

Micheli, G. J. L., & Mantella, F. (2018). Modelling an environmentally-extended inventory routing problem with demand uncertainty and a heterogeneous fleet under carbon control policies. *International Journal of Production Economics, 204*, 316-327.

Mirzapour Al-e-hashem, S. M. J., Rekik, Y., & Mohammadi Hoseinhajlou, E. (2019). A hybrid L-shaped method to solve a bi-objective stochastic transshipment-enabled inventory routing problem. *International Journal of Production Economics, 209*, 381-398.

Marufuzzaman, M., Eksioglu, S. D., & Huang, Y. (2014). Two-stage stochastic programming supply chain model for biodiesel production via wastewater treatment. *Computers & Operations Research, 49*, 1-17.

Moreno, A., Alem, D., Ferreira, D., & Clark, A. (2018). An effective two-stage stochastic multi-trip location-transportation model with social concerns in relief supply chains. *European Journal of Operational Research, 269*(3), 1050-1071.

Nikzad, E., Bashiri, M., & Oliveira, F. (2019). Two-stage stochastic programming approach for the medical drug inventory routing problem under uncertainty. *Computers & Industrial Engineering, 128*, 358-370.

Noyan, N. (2012). Risk-averse two-stage stochastic programming with an application to disaster management. *Computers & Operations Research, 39*(3), 541-559.

Oppen, J., Lokketangen, A., & Desrosiers, J. (2010). Solving a rich vehicle routing and inventory problem using column generation. *Computers & Operations Research, 37*(7), 1308-1317.

Popović, D., Vidović, M., & Radivojević, G. (2012). Variable neighborhood search heuristic for the inventory routing problem in fuel delivery. *Expert Systems with Applications, 39*(18), 13390-13398.

Raa, B., & Aghezzaf, E. H. (2009). A practical solution approach for the cyclic inventory routing problem. *European Journal of Operational Research, 192*(2), 429-441.

Raa, B., & Dullaert, W. (2017). Route and fleet design for cyclic inventory routing. *European Journal of Operational Research, 256*(2), 404-411.

Raa, B., & Aouam, T. (2021). Multi-vehicle stochastic cyclic inventory routing with guaranteed replenishments. *International Journal of Production Economics, 234*, 108059.

Roldán, R. F., Basagoiti, R., & Coelho, L. C. (2017). A survey on the inventory-routing problem with stochastic lead times and demands. *Journal of Applied Logic, 24*, 15-24.

Su, Z. X., Lu, Z. P., Wang, Z., Qi, Y. M., & Benlic, U. (2020). A matheuristic algorithm for the inventory routing problem. *Transportation Science, 54*(2), 330-354.

Sikora, C. G. S. (2021). Benders' decomposition for the balancing of assembly lines with stochastic demand. *European Journal of Operational Research, 292*(1), 108-124.

Solomon, M. M. (1987). Algorithms for the vehicle-routing and scheduling problems with time window constraints. *Operations Research, 35*(2), 254-265.

Vanslyke, R. M., & Wets, R. (1969). L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM journal on applied mathematics, 17*(4), 638-663.

Wang, L., Kinable, J., & van Woensel, T. (2020). The fuel replenishment problem: A split-delivery multi-compartment vehicle routing problem with multiple trips. *Computers & Operations Research, 118*, 104904.

Wang, S. Y., Tao, F. M., & Shi, Y. H. (2018). Optimization of inventory routing problem in refined oil logistics with the perspective of carbon tax. *Energies, 11*(6), 1437.

Wang, L. (2020). A two-stage stochastic programming framework for evacuation planning in disaster responses. *Computers & Industrial Engineering, 145*, 106458.

Zhang, Z. Z., Luo, Z. X., Baldacci, R., & Lim, A. (2021). A Benders decomposition approach for the multivehicle production routing problem with order-up-to-level policy. *Transportation Science, 55*(1), 160-178.

Zhang, Y. W., Li, Z. P., Jiao, P. B., & Zhu, S. (2021). Two-stage stochastic programming approach for limited medical reserves allocation under uncertainties. *Complex & Intelligent Systems, 7*(6), 3003-3013.