SORT 50 (2) July-December 2026, 1-38 DOI:

A column generation-based hybrid optimization algorithm for two-echelon swap-body distribution

Yao-Na Li[1] and Zhi-Hua Hu[1,*]

Abstract

This study investigates a two-echelon swap-body distribution problem in urban logistics, in which goods are delivered from a central depot to customers via dynamically selected intermediate swap locations. A mathematical model and five hybrid algorithms are developed to jointly optimize routing decisions and swap-location selection. Numerical experiments using an ideal dataset and benchmark datasets validate the effectiveness of the proposed models and algorithms. Results show that the first-echelon cost is dominant, and the proposed method reduces computation time by up to 89.7% for small-scale instances while maintaining solution quality. In large-scale instances, it reduces computational time by 79.9% while reporting gaps below 3%. Furthermore, sensitivity analyses demonstrate that the proposed approach adapts well to demand variations, achieving average cost savings of 94.1%. These findings provide practical insights and theoretical contributions for improving the efficiency of urban distribution systems.

MSC: *90B06, 90B10, 90C57, 90B50.*

Keywords: *Two-echelon distribution, column generation, genetic algorithm, swap-body distribution system, urban logistics.*

1. Introduction

Swap-body distribution (SBD) is an innovative approach to urban logistics that enables cargo transfer between vehicles via detachable swap-body containers. At the heart of this system is a three-step transfer mechanism (as illustrated in Figure 1) that enables seamless container movement between vehicles: transfer arms position the container, unlock it, lift it, and then securely place it onto the receiving vehicle. This fully automated process eliminates the need for manual handling and fixed infrastructure, distinguishing

* Corresponding author. Email: zhhu@shmtu.edu.cn

[1] Logistics Research Center, Shanghai Maritime University, Shanghai, China.

Received: August 2025

Accepted: July 2026

it from conventional logistics systems. We are seeing real-world validation of this technology's potential through industry adoption. A prime example is JD Logistics, which has successfully implemented IoT (Internet of Things) and big data analytics to optimize container sharing in swap-body operations, achieving notable improvements in delivery efficiency and cost reduction. The strategic importance of such systems is further reinforced by the inclusion in China's 14th Five-Year Plan for standardized logistics development.

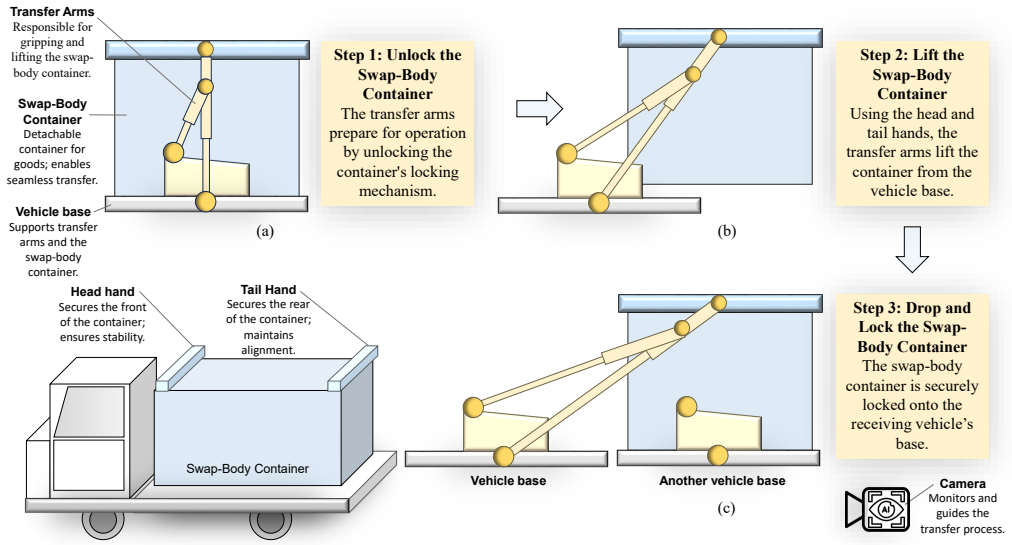


Figure 1. *Three-step process for transferring a swap-body container between two vehicles: (a) Preparation phase, where the transfer arms are positioned, and the container is unlocked. (b) Lifting phase, where the container is raised by the transfer mechanism. (c) Locking phase, where the container is dropped onto the receiving vehicle and secured in place.*

In this study, we investigate the novel two-echelon swap-body distribution (2E-SBD) problem, characterized by three key distinguishing features. First, it enables fully automated container transfers between vehicles without requiring additional infrastructure or personnel. Second, it operates within a two-echelon distribution structure. Third, it utilizes distinct vehicle types in adjacent echelons that intersect at dynamically optimized swap locations. Existing two-echelon distribution networks face limitations on fixed infrastructure and homogeneous fleets, leading to rigid capacity allocation, persistent load imbalances, and limited demand responsiveness (Dellaert et al., 2019; Lan et al., 2022). Under the modeling assumptions adopted in this study, the proposed 2E-SBD network represents swap-body operations without dedicated transfer stations or permanent intermediate facilities. This assumption enables dynamic reconfiguration of swap locations and capacity allocation within the optimization framework. The resulting dynamic optimization capability significantly reduces capital expenditures while enhancing operational flexibility to accommodate stochastic demand patterns, demonstrating advan-

tages over static network designs. However, the 2E-SBD problem requires simultaneous optimization of swap locations, customer assignments, and routing across both echelons, making it a generalized NP-hard problem.

Existing research has developed a spectrum of methods ranging from exact algorithms to heuristic and hybrid techniques. For modest-sized problems, approaches such as branch-and-price (Dellaert et al., 2019) and neighborhood-search-enhanced branch-and-cut (Jepsen et al., 2013) have shown promising results. As problem scales increase, the limitations of exact methods become apparent, leading to the development of innovative hybrid approaches that combine heuristic and intelligent techniques. Particularly noteworthy are several successful integrations: Contardo et al. (2012) achieved faster solutions by combining adaptive large neighborhood search with branch-and-cut, while Contardo et al. (2012) showed how genetic algorithms could enhance column generation for spare parts distribution. Other creative integrations include Wang et al. (2021) column generation for multi-depot problems, Xue et al. (2022) cutting-plane-GA hybrid, and the machine learning-enhanced column generation approach developed by Ramírez-Villamil et al. (2023).

Building on these methodological advances, this work presents a novel hybrid algorithm that combines column generation (CG) with genetic algorithms (GA). The proposed CG-GA approach reformulates the two-echelon location routing problem (2E-SBD) within a CG framework, decomposing it into a restricted master problem and a specialized pricing subproblem. The solution process must address interdependencies between first- and second-echelon routes at dynamically optimized swap locations, which pose significant challenges for reduced cost computation during column generation. As the pure exact methods have limitations in large-scale applications, the CG-GA framework strategically combines both approaches to balance solution quality with computational tractability. Specifically, the CG component establishes rigorous lower bounds for optimal solutions, while the GA component enables efficient exploration of diverse solution spaces.

The main contributions are summarized as follows: (i) The 2E-SBD problem is introduced, which is a novel urban logistics challenge that leverages swap-body technology. (ii) A MILP model is formulated to optimize swap locations, customer assignments, and distribution routes while considering the two-echelon structure. (iii) Solution algorithms integrating CG and GA are proposed, demonstrating efficiency in solving large-scale instances. (iv) The effects of new customer integration and parameter variations on the 2E-SBD model are analyzed, offering insights into distribution cost dynamics and optimization sensitivity.

The remainder of this paper is organized as follows: Section 2 reviews the research relevant to the 2E-SBD problem. Section 3 presents the MILP formulation for the 2E-SBD problem. Section 4 proposes hybrid solution approaches. Section 5 reports computational results. Section 6 discusses practical implications, and Section 7 concludes with the main contributions and future research opportunities.

2. Literature review

The 2E-SBD problem exhibits conceptual parallels with several location-routing problems (LRPs), including truck-and-trailer distribution (TTD) and two-echelon LRP (2E-LRP). TTD and 2E-SBD share a fundamental characteristic: the employment of interchangeable container units to optimize distribution efficiency. As demonstrated by Chao (2002) and Accorsi and Vigo (2020), TTD systems typically involve tractors transporting trailers between central depots and local distribution points, requiring physical trailer swaps at fixed locations. This operational paradigm bears conceptual similarity to 2E-SBD's use of swap locations, though with distinct technical implementations. Accorsi and Vigo (2020) addresses the practical constraints that arise in distribution problems, including limited mobility space, restrictions on large vehicles, and road conditions, which result in some customers needing to be served by a pure truck (Chao, 2002). In the TTD, solutions typically include three different types of routes. (1) Pure Truck Route (PTR): PTR is a route completed by a single truck without a trailer, suitable for customers with space constraints or other reasons that require truck-only service (Derigs et al., 2013; Scheuerer, 2006). (2) Pure Vehicle Route (PVR): PVR is a route completed by a truck and trailer combination, suitable for customers who can be served by both simultaneously (Accorsi and Vigo, 2020). Due to its greater transport capacity, PVR is typically used by customers with higher demand (Li et al., 2015). (3) Complete Vehicle Route (CVR): In TTD, a CVR consists of a main route with both truck and trailer, and sub-routes where only the truck operates. The trailer is detached at a transshipment point before the sub-route and reattached later for continued service or return to the depot (Lin et al., 2009, 2010). The CVR in TTD resembles swap bodies in SB-2E-LRP. However, trailers cannot switch vehicles, requiring the tractor to return, reconnect the trailer, and continue the delivery, which adds time and distance.

The 2E-LRP extends the Vehicle Routing Problem (VRP) by introducing an intermediate distribution tier, enabling vehicles to depart directly from a central warehouse or distribution center to serve all customers. This distribution network involves two echelons: commodities are first transported from the central warehouse to intermediate facilities (such as satellite or transfer stations), where they are consolidated, repacked, and subsequently delivered to customers. This hierarchical structure offers three key advantages: enhanced regulatory compliance, greater operational flexibility, and improved economies of scale (Sluijk et al., 2023). Both 2E-LRP and 2E-SBD incorporate a multi-echelon network structure. However, in 2E-LRP, vehicle fleets are predefined and assigned to each echelon (Dellaert et al., 2019; Lan et al., 2022), whereas in 2E-SBD, vehicle assignments are not predetermined. Therefore, unlike 2E-VRP, 2E-SBD does not segregate by echelons but rather by vehicle flexibility, allowing vehicles to travel on all network arcs. In the 2E-LRP literature, some studies allow customers to be served directly from the central depot by first-echelon vehicles (Anderluh et al., 2021; Bayliss et al., 2023; Macrina et al., 2020). However, few works consider the

Table 1. *The solution methods of the studies on TTD, 2E-LRP, and SBD.*

Study	Problem	Model	Algorithm
Tian and Hu (2023)	2E-LRP	MILP	BP
Sutrisno and Yang (2023)	2E-LRP	MILP	NS
Lan et al. (2022)	2E-LRP	MILP	NS
Escobar-Vargas and Crainic (2024)	2E-LRP	MILP	HA
Li et al. (2024)	2E-LRP	MILP	PSO
Rothenbächer et al. (2018)	TTD	MILP	CG
Alves da Cruz and Salles da Cunha (2023)	TTD	MILP	BC
Accorsi and Vigo (2020)	TTD	MILP	HA
Belenguer et al. (2016)	TTD	MILP	BC
Todosijević et al. (2017)	SBD	MILP	NS
Miranda-Bront et al. (2017)	SBD	MILP	GRAS
Huber et al. (2020)	SBD	MILP	CG; NS
Toffolo et al. (2018)	SBD	MILP	HA
Absi et al. (2017)	SBD	MILP	GA
This paper	2E-SBD	MILP	CG; GA

Note: BC = Branch-and-Cut; BP = Branch-and-Price; CG = Column Generation; GA = Genetic Algorithm; GRASP = Greedy Randomized Adaptive Search; HA = Heuristics algorithm; NS = Neighborhood search; PSO = Particle Swarm Optimization.

swap-body technology. Instead, they primarily focus on loading/unloading operations at intermediate facilities. Consequently, customers must be pre-classified to determine whether they will be served by first- or second-echelon vehicles. The 2E-LRP models assume dedicated fleets for each echelon: first-echelon vehicles transport goods from central depots to intermediate transfer points, while second-echelon vehicles handle final-mile deliveries (Dellaert et al., 2019; Lan et al., 2022). The 2E-SBD extends this framework by incorporating automated swap-body technology, enhancing operational flexibility through dynamic reconfiguration of swap locations in response to fluctuating demand. However, this added flexibility introduces greater complexity, as the problem now requires joint optimization of intermediate facility selection and multi-echelon vehicle routing.

To better understand the solution methods of the studies on TTD, 2E-LRP, and 2E-SBD, Table 1 summarizes the key studies in urban logistics contexts. The "Model" column reveals that most of these studies adopt MILP formulations. Tian and Hu (2023) formulate a MILP model for a 2E-LRP with recommended satellites (2E-LRPRS) to address the inflexibility of fixed relay stations in urban logistics, developing a column-generation-based branch-and-pricing algorithm that solves the MILP more efficiently than Cplex. Alves da Cruz and Salles da Cunha (2023) formulate a MILP model for the profitable single-truck-and-trailer routing problem with time windows (PSTTRPTW), where trailers can be decoupled at satellite depots to enable truck sub-tours. Huber et al.

(2020) develop a MILP formulation for the swap-body vehicle routing problem, in which vehicles dynamically reconfigure at intermediate locations via trailer decoupling and swap-body exchanges.

Due to the computational difficulty of solving directly, researchers have predominantly employed heuristic solution approaches. Sutrisno and Yang (2023) developed a clustering-based simultaneous neighborhood search (CSNS) heuristic for 2E-LRP with mobile satellites (2E-LRP-MS) that outperforms existing methods and Cplex by leveraging k-means clustering and a probabilistic mechanism. Accorsi and Vigo (2020) developed an innovative hybrid metaheuristic combining iterative route-pooling and set-partitioning post-optimization for the Extended Single Truck and Trailer Routing Problem (XSTTRP). Toffolo et al. (2018) proposed a stochastic local search algorithm with general and problem-specific heuristics, subproblem optimization, and a learning mechanism. However, these methods often struggle to guarantee global optimality and may become trapped in local optima. Conversely, math-heuristic approaches such as branch-and-cut (BC) and branch-and-price (BP) demonstrate stronger performance in finding global optima, though their computational efficiency tends to decrease substantially as problem instances grow in size. Tian and Hu (2023) studied the 2E-LRPRS problem and demonstrated that B&P remains computationally tractable only for small instances ($\leq$ 15 nodes). Alves da Cruz and Salles da Cunha (2023) developed a reinforced B&C algorithm for the PSTTRPTW, achieving 25% tighter LP relaxations through preprocessing and valid inequalities, yet remaining computationally tractable only for instances up to 30 nodes.

This study advances two-echelon distribution research through three key contributions: (1) developing a MILP model that simultaneously optimizes swap location selection, customer assignment, and routing decisions for the 2E-SBD problem; (2) proposing a hybrid solution framework that integrates exact methods with heuristic approaches; (3) conducting operational experiments to analyze model performance and key parameters, deriving actionable managerial insights.

Compared with the existing TTD, SBD, and 2E-LRP studies summarized in Table 1, the proposed 2E-SBD problem differs in several important aspects. First, unlike TTD and conventional SBD problems, where container exchanges are usually performed at predefined locations or require vehicles to return for trailer reconnection, the proposed 2E-SBD allows dynamically selected swap locations without fixed infrastructure. Second, although 2E-LRP considers multi-echelon distribution networks, it generally assumes dedicated vehicle fleets for different echelons and fixed intermediate facilities, whereas the proposed 2E-SBD jointly optimizes swap-location selection, customer assignment, and routing decisions under a flexible vehicle operation mechanism. Third, existing SBD studies mainly focus on single-echelon routing or container exchange operations, while this study integrates swap-body technology into a two-echelon distribution framework and develops corresponding optimization models and solution algorithms. Therefore, the proposed 2E-SBD problem represents a distinct and more flexible urban logistics optimization problem that extends existing two-echelon and swap-body distribution research.

3. Problem description and formulation

3.1. Problem description

In the proposed 2E-SBD system, packages are transported from a central depot to customers via two echelons using swap-body container-equipped vehicles. Figure 2 presents an example of a 2E-SBD network comprising one global depot, four swap locations, and 16 customers. In the first echelon, a vehicle transports a swap-body container from the global depot to selected swap locations. At the swap locations, the containers are transferred to a second-echelon vehicle for delivery to customers. The automated swapping process streamlines container transfers, eliminating manual handling and reducing reliance on operators. In the proposed model, swap locations are treated as temporary transfer points that can be dynamically selected based on demand patterns. We assume that these locations do not require dedicated operators or specialized permanent facilities, and their infrastructure costs are negligible compared with transportation costs. This assumption simplifies the optimization problem by focusing on routing and location decisions. In practice, however, deploying swap-body systems may still involve land availability, safety requirements, equipment maintenance, and regulatory considerations beyond the scope of this study.

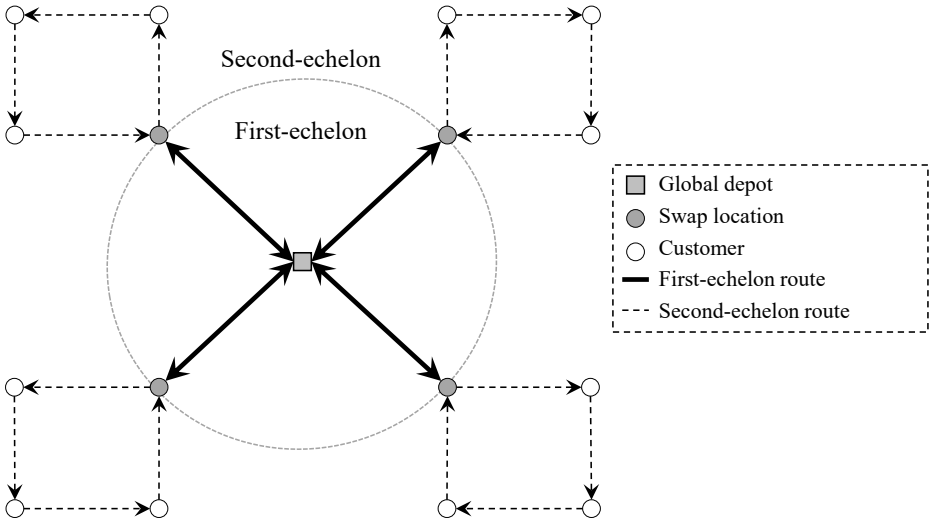


Figure 2. *A demonstration of delivery routes for the 2E-SBD.*

3.2. Sets, parameters, and variables

The related sets, parameters, and decision variables are listed in Table 2. The 2E-SBD involves three key decisions: (1) selecting a subset of customers as swap locations,

denoted by x_k, where $x_k = 1$ if customer k is chosen as a swap location and $x_k = 0$ otherwise; (2) assigning customers to each swap location k, represented by y^A_{ik}, where $y^A_{ik} = 1$ if vehicle k serves customer i and 0 otherwise; and (3) determining the sequence of customer visits for each vehicle departing from k, where $x_{ijk} = 1$ indicates that the vehicle k visits customer j immediately after customer i and $y_{ijk} = 0$ otherwise.

Table 2. *The sets, parameters, and decision variables of the 2E-SBD model.*

Sets	**Description**
$\{Source, Sink\}$	Dummy nodes representing the global depot
C	A set of customers, indexed by $i, j \in C$
K	The set of candidate swap locations. In this study, every customer node can serve as a swap location; therefore, $K = C$, and $k \in K$.
V	The set $V = \{Source, Sink\} \cup N$, indexed by $i, j, k \in V$. Here, *Source* and *Sink* are dummy nodes representing the global depot. *Source* denotes the departure point of first-echelon vehicles, while *Sink* denotes their return point after completing delivery operations.
Parameters	**Description**
$C_{ij} \geq 0$	The distance from customer $i \in V$ to customer $j \in V$
Q_i	The demand of customer $i \in C$
$\bar{Q} \geq 0$	The capacities of the first- and second-echelon vehicles
C^1, C^2	The per-mile distribution costs for first- and second-echelon vehicles, respectively
Variables	**Description**
$x_k \in \{0,1\}$	$x_k = 1$ if the candidate swap location ($k \in K$) is selected as a first-echelon transfer point; otherwise $x_k = 0$.
$y_{ijk} \in \{0,1\}$	$y_{ijk} = 1$ if a second-echelon vehicle associated with candidate swap location $k \in K$ travels directly from customer $i \in C$ to customer $j \in C$.
$y^A_{ik} \in \{0,1\}$	Customer assignment variable, where $y^A_{ik} = 1$ if customer $i \in C$ is served by swap location $k \in K$.
$y^s_k \in \{0,1\}$	$y^s_k = 1$ if candidate swap location $k \in K$ serves only itself.
$y^n_k \in \{0,1\}$	$y^n_k = 1$ if candidate swap location $k \in K$ serves itself and at least one additional customer.
$q_{ik} \geq 0$	The cumulative delivered demand after servicing customer $i \in C$ on the second-echelon route originating from candidate swap location $k \in K$.

3.3. A MILP formulation

To minimize the total distribution cost z, the objective function is formulated as the sum of the first-echelon cost z^1 and the second-echelon cost z^2, as shown in Eq. (1). The first-echelon distribution cost z^1 includes the travel cost from the global depot (Source) to the swap location k (denoted as $C_{Source,k}$) as well as the return travel cost $C_{k,Sink}$ from swap location k back to the global depot (*Sink*), with C^1 serving as a cost coefficient. The second-echelon distribution cost z^2 represents the total travel costs from a swap location k to sequentially visit other customers, finally returning to the swap location, where C^2 is a cost coefficient.

$$[\text{OP}]\min \quad z = z^1 + z^2 \tag{1}$$

$$\text{where} \quad z^1 = C^1 \sum_{k \in N} (C_{Source,k} + C_{k,Sink}) x_k \tag{2}$$

$$z^2 = C^2 \sum_{k \in K} \sum_{i \in V} \sum_{j \in V; i \neq j} C_{ij} y_{ijk} \tag{3}$$

Constraints (4) and (5) ensure the flow conservation of second-echelon routes. Specifically, if customer i is assigned to swap location k, exactly one incoming and one outgoing arc associated with customer i must be selected in the corresponding second-echelon route. Constraint (6) links customer assignments with swap-location service patterns. If swap location k serves customer i ($i \neq k$), it must be classified as a swap location serving itself and additional customers, which is represented by $y_k^n = 1$. Constraint (7) ensures that a selected swap location either serves only itself or serves itself together with other customers. These two service patterns are mutually exclusive. Constraint (8) guarantees that a customer can act as a swap location only when it is selected. Constraint (9) ensures complete customer coverage: each customer is either selected as a swap location and served by itself, or assigned to another selected swap location. Constraint (10) imposes a capacity limit on second-echelon vehicles. For a selected swap location k, the total demand of customers assigned to it cannot exceed the available vehicle capacity after satisfying its own demand. Constraint (11) defines the cumulative delivered demand variable q_{ik} and eliminates infeasible repeated visits and subtours. When customer j is visited immediately after customer i on the route associated with swap location k, the cumulative delivered demand at j equals that at i plus the demand of customer i. Finally, Constraint (12) defines the domains of all decision variables.

$$\sum_{j \in N; j \neq i} y_{ijk} = y_{ik}^A, \forall i, k \in N,\ i \neq k \tag{4}$$

$$\sum_{j \in N; j \neq i} y_{jik} = y_{ik}^A, \forall i, k \in N,\ i \neq k \tag{5}$$

$$y_{ik}^A \leq y_k^n, \forall i, k \in N,\ i \neq k \tag{6}$$

$$(1-y_k^s)|N|^2 \geq \sum_{i\in N;i\neq k} y_{ik}^A, \forall k \in N \tag{7}$$

$$x_k \geq y_k^s + y_k^n, \forall k \in N \tag{8}$$

$$x_i + \sum_{k\in N;k\neq i} y_{ik}^A = 1, \forall i \in N \tag{9}$$

$$\sum_{i\in N;i\neq k} y_{ik}^A Q_i \leq (\bar{Q} - Q_k)x_k, \forall k \in N \tag{10}$$

$$q_{ik} + Q_i \leq q_{jk} + (\bar{Q} + Q_i)(1 - y_{ijk}), \forall i,j,k \in N,\ j \neq i,\ i \neq k,\ j \neq k \tag{11}$$

$$x_k, y_{ijk}, y_{ik}^A, y_k^s, y_k^n \in \{0,1\}, q_{ik} \geq 0, \forall i,j,k \in N \tag{12}$$

The complete model is structured as the following optimization problem: [OP] $\min Z$ subject to (4)-(12).

4. Solution methods

Optimizing swap locations and two-echelon routes in the 2E-SBD presents challenges. Commercially available MILP solvers, such as Cplex and Gurobi, can solve [OP] for small-scale instances (e.g., $|N| \leq 30$, see Section 5). To balance solution quality and computational efficiency, we propose a hybrid optimization algorithm that combines CG and GA.

4.1. A CG-based solution framework for 2E-SBD

4.1.1. Solution framework

We employ the Dantzig-Wolfe decomposition to reformulate the compact model [OP] defined by the objective function (1) and constraints (4)-(12) into a Master Problem (MP) and a Subproblem (SUB). The [MP] handles the linear relaxation of the original problem and coordinates feasibility across echelons. The [SUB] constitutes a 2E-ESPPRC that generates feasible routes while accounting for negative-weight cycles induced by dual pricing. To use CG algorithms, the two key principles are described below.

Principle 1. Restricted master problem (RMP). The master problem simplifies the original problem by considering only a subset of columns that represent the current decision variables. Since the original problem is an integer linear program, it can be linearly relaxed in the [RMP] by converting binary variables into continuous ones, thereby yielding solutions and dual variables.

Principle 2. Generating new columns. The master problem model [MP] includes all potential columns in the constraint matrix, representing every possible decision variable.

Generating new feasible columns is handled by an independent subproblem [SUB], which involves complex constraints. Solving the [SUB] identifies a new column that satisfies these constraints while minimizing the reduced cost. A new column with a negative reduced cost can improve the current solution to the master problem. When the pricing problem is solved exactly (i.e., [CG+MILP]), the absence of a negative reduced-cost column provides a valid optimality certificate for the linear relaxation. However, when the pricing problem is solved heuristically by GA (i.e., [CG+GAs] and [CG+GAg]), the absence of an identified negative reduced-cost column only indicates that no improving column was found by the heuristic search, rather than proving that no such column exists.

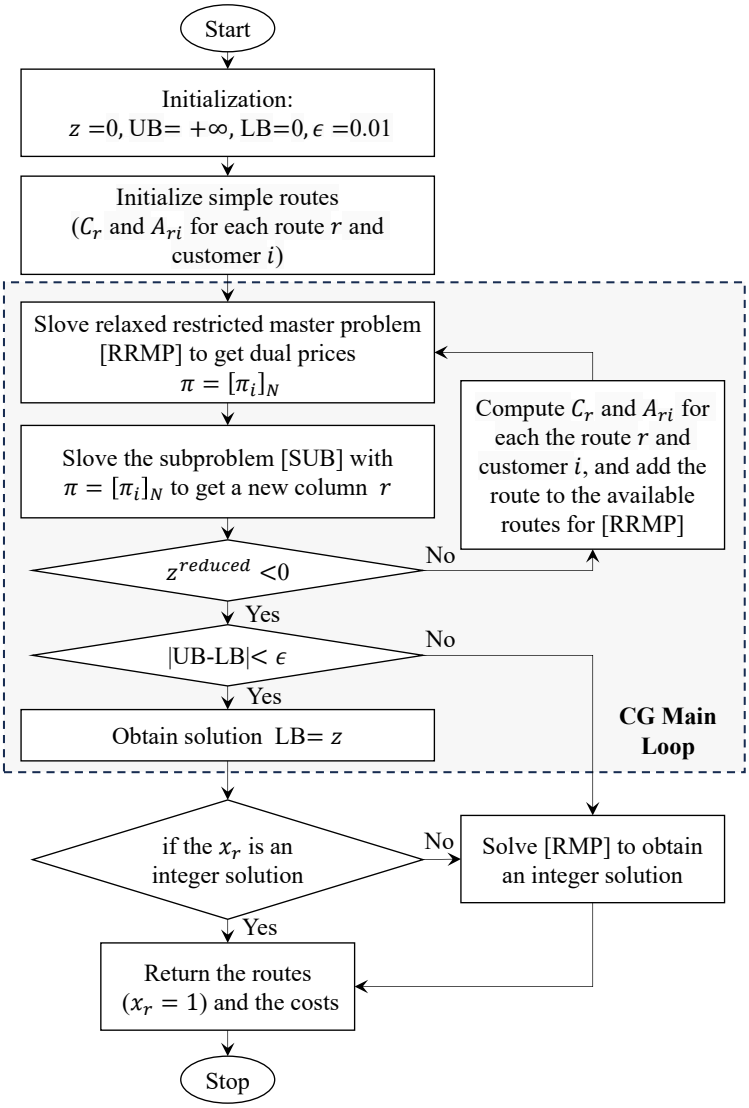


Figure 3. *A CG iteration framework for 2E-SBD.*

The iterative process of the CG-based solution framework for 2E-SBD is illustrated in Figure 3. In the first stage, by linearly relaxing [RMP] to [RRMP], a dual process corresponding to the customer constraints is obtained. The second stage involves solving the [SUB] of the 2E-ESPPRC, to minimize the reduced cost while incorporating the dual prices. Each CG iteration calculates dual prices using the [RRMP] and generates a new column with the [SUB], updating the available columns with routing costs. The process iterates until the gap between the upper bound (UB) and lower bound (LB) is less than a specified threshold ε. If an integer solution is not obtained, the integer program [RMP] can be solved as a post-processing step. Algorithm 1 details the procedures of the CG-based solution framework.

Algorithm 1 CG-based solution framework for [2E-SBD]

Require: N: a set of customers; $I = [c_{ij}, Q_i, \bar{Q}, C^1, C^2]$; [SUB]: a solver for the subproblem; ε: the convergence threshold.

Ensure: $Routes^*$: the optimal delivery routes (e.g., $\{\{1,2,\cdots\},\{3,4,\cdots\},\cdots\}$); z: the total distribution cost including first- and second-echelon costs.

Step 1: Initialize $Routes^* \leftarrow \emptyset$; $C_r \leftarrow \emptyset$; $A_{ri} \leftarrow \emptyset$

Step 2: for each node $i \in N$ **do**

  Step 2.1: $Routes^* \leftarrow Routes^* \cup \{Source,\ i,\ Sink\}$

  Step 2.2: $C_r \leftarrow C_r \cup \{C^1 \cdot (C_{Si} + C_{iE})\}$, where C_{Si} is the cost from *Source* to node i, and C_{iE} is the cost from node i to *Sink*.

  Step 2.3: Set $A_{ri} = 1$ and for all $j \neq i$, set $A_{rj} = 0$

end for

Step 3: Initialize $z \leftarrow 0$; $UB \leftarrow +\infty$; $LB \leftarrow 0$; $\varepsilon \leftarrow 0.01$

Step 4: while $UB - LB > \varepsilon$ **do**

  Step 4.1: Use C_r and A_{ri} to solve the [RRMP] and update dual prices.

  Step 4.2: Compute reduced cost $\bar{C}_r$ for the new column r.

  Step 4.3: Update the lower bound LB and upper bound UB based on the new C_r.

end while

Step 5: if the solution from [RRMP] is fractional **then**

  Step 5.1: Solve the [RMP] using A_{ri} and C_r.

  Step 5.2: Generate final $Routes^*$ from the integer solution.

end if

Step 6: return $Routes^*, z$

4.1.2. Master problem

In the restricted master problem [RMP], where the binary variable x_r denotes the selection of route r. Let Ω denote the feasible set of routes, indexed by $r \in \Omega$.

Define $P^r = \{P_1^r, P_2^r\} = \{\{S,k,E\}, \{k,\cdots,i,\cdots,k\} | k,i \in N\}$, where P_1^r and P_2^r represent the node sequences of the first- and second-echelon routes, respectively.

For each route r, A_{ri} is defined as:

$$A_{ri} = \begin{cases} 1, & i \in P_2^r, \\ 0, & \text{otherwise.} \end{cases}$$

The cost of route $r \in \Omega$, denoted as C_r, is given by:

$$C_r = C^1(C_{Sk} + C_{kE}) + C^2 \sum_{(i,j)\in P_2^r} C_{ij}.$$

$$\text{[RMP]} \quad \min \quad \sum_{r\in\Omega} C_r x_r \tag{13}$$

$$\text{s.t.} \quad \sum_{r\in\Omega} A_{ri} x_r = 1, \quad \forall i \in N \tag{14}$$

$$x_r \in \{0,1\}, \quad \forall r \in \Omega \tag{15}$$

The [RRMP] formulates the linear relaxation of [RMP], where the binary variable x_r is relaxed as a continuous variable. This reformulation allows the derivation of dual variables $\pi = [\pi_i]_{i\in N}$.

$$\text{[RRMP]} \quad \min \quad \sum_{r\in\Omega} C_r x_r$$

$$\text{s.t.} \quad \sum_{r\in\Omega} A_{ri} x_r = 1, \quad \forall i \in N$$

$$0 \le x_r \le 1, \quad \forall r \in \Omega \tag{16}$$

4.1.3. Subproblem

The subproblem is a two-echelon elementary shortest path problem with resource constraints (2E-ESPPRC) involving negative-weight cycles, as formulated in the [SUB] model. As depicted in Figure 4, a first-echelon vehicle departs from the global depot, travels to a designated swap location, transfers the swap-body container to a second-echelon vehicle, and then returns. Each swap location is served by a single first-echelon vehicle and functions as an intermediate facility for container transfer between the first- and second-echelon vehicles.

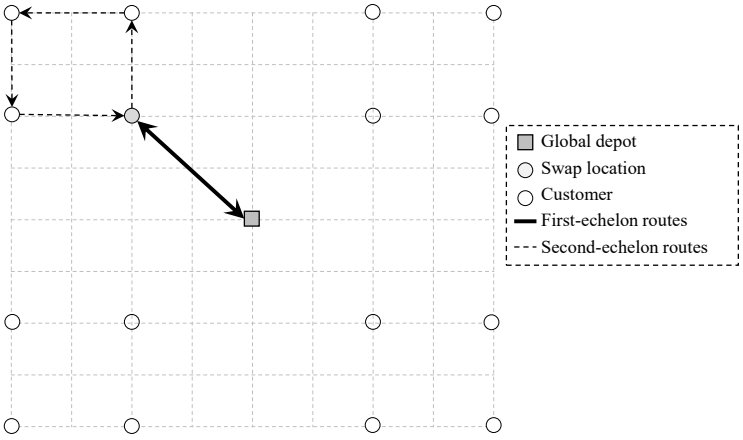


Figure 4. *A demonstration of delivery routes for 2E-ESPPRC.*

$$\text{[SUB]} \quad \min \quad z^{reduced} = z_1^{reduced} + z_2^{reduced} \tag{17}$$

$$\text{where} \quad z_1^{reduced} = C^1 \sum_{k \in N} x_k (C_{Sk} + C_{kE}) \tag{18}$$

$$z_2^{reduced} = \sum_{i,j,k \in N; i \neq j} y_{ijk} \bar{C}_{ij} \tag{19}$$

$$\bar{C}_{ij} = C^2 C_{ij} - \frac{\pi_i}{2} - \frac{\pi_j}{2}, \forall (i,j) \in P_2^r \tag{20}$$

$$\text{Subject to} \sum_{j \in N; j \neq i} y_{ijk} = y_{ik}^A, \forall i,k \in N \tag{21}$$

$$\sum_{j \in N; j \neq i} y_{jik} = y_{ik}^A, \forall i,k \in N \tag{22}$$

$$\sum_{i \in N; i \neq k} y_{kik} = x_k, \forall k \in N \tag{23}$$

$$\sum_{i \in N; i \neq k} y_{ikk} = x_k, \forall k \in N \tag{24}$$

$$x_k \geq y_{ik}^A, \forall i,k \in N \tag{25}$$

$$\sum_{i \in N} y_{ik}^A Q_i \leq \bar{Q} x_k, \forall k \in N \tag{26}$$

$$q_{ik} + Q_i \leq q_{jk} + (\bar{Q} + Q_i)(1 - y_{ijk}), \forall i,j,k \in N; j \neq i,\ i \neq k \tag{27}$$

$$\sum_{k \in N} x_k = 1 \tag{28}$$

$$x_k \in \{0,1\}, y_{ijk} \in \{0,1\}, y_{ik}^A \in \{0,1\}, q_{ik} \geq 0, \quad \forall i,j,k \in N \tag{29}$$

In Eq. (17), if $z^{reduced} \geq 0$, no improving variable is found; otherwise, the generated variables are added as new columns to the master problem. The modified arc cost $\bar{C}_{ij}$ represents the reduced cost of using arc (i, j), which is defined as:

$$\bar{C}_{ij} = C^2 C_{ij} - \frac{\pi_i}{2} - \frac{\pi_j}{2},$$

allowing $\bar{C}_{ij}$ to take either positive or non-positive values. Constraints (21) and (22) ensure the flow balance of second-echelon routes. Specifically, the number of incoming and outgoing arcs for each customer assigned to swap location k must be consistent with its assignment decision. Constraints (23) and (24) ensure the consistency between the selected swap locations and the customers they serve. If node k is selected as a swap location, it must serve at least one customer, including itself; otherwise, it cannot participate in the second-echelon routes. Constraint (25) establishes the relationship between customer assignment and swap-location selection. When customer i is assigned as a swap location, it can only be served by an activated swap location k. Constraint (26) imposes the maximum capacity limitation of swap locations. The total demand of customers assigned to swap location k cannot exceed the available vehicle capacity. Constraint (27) defines the cumulative demand variable q_{ik} and eliminates repeated visits and infeasible circulation loops in the routes. Constraint (28) guarantees that only one customer node can be selected as a swap location. Finally, the variable domains are specified in Constraint (29).

4.2. Using GA to generate columns

The CG-based solution framework with GA generating columns, denoted by [CG+GA] provides a GA framework to generate a set of second-echelon routes covering all customers for solving [2E-SBD]. Unlike [CG+MILP], the [CG+GA] variants employ heuristic pricing procedures. Therefore, they provide high-quality feasible solutions but do not guarantee exact convergence of the column generation process. In [CG+GA], a GA is devised to solve the subproblem [SUB] of [2E-SBD], thereby solving [2E-ESPPRC], while Algorithm 1 generates holistic solutions for [2E-SBD]. [CG+GA] searches for a candidate second-echelon route with minimal reduced cost. First, a new parameter of dual prices π is derived from solving [RRMP]. Second, the evaluation operator $Evaluate(\cdot)$ must be revised to accept π. Third, the decoding algorithm will search for only one route with minimal reduced cost.

In the CG algorithm for the 2E-SBD problem, the reduced cost calculation for a route $Route = \{Source, k_1, \cdots, k_n, Sink\}$ comprises three essential components. First, the first-echelon reduced cost $z_1^{reduced}$ captures the primary transportation expenses associated with the movement from the source node to the selected swap locations and back to the sink. Second, the second-echelon component $z_2^{reduced}$ aggregates the reduced costs of all arcs (i, j) in the customer visitation sequence. The total reduced cost is calculated as:

$$z^{reduced} = z_1^{reduced} + z_2^{reduced},$$

which serves as the criterion for route inclusion in the basis. A negative value of $z^{reduced}$ indicates a promising column that can improve the current restricted master problem solution. The complete computational procedure is specified in Algorithm 3.

Algorithm 2 A GA for solving [SUB] of [2E-SBD]

Require: N: set of customers; $I = [C_{ij}, Q_i, \bar{Q}, C^1, C^2]$; p_m: mutation probability; p_c: crossover probability; P: population size; G_{max}: maximum number of generations; π: dual price obtained from solving [RRMP].

Ensure: $Routes^*$: a set of best-found delivery routes; $z_{min}^{reduced}$: the minimum identified reduced cost.

Step 1: Initialize Population: *Population* $\leftarrow$ *Initialize(P, N, I)*
Step 2: Evaluate Initial Fitness: $Fitness \leftarrow Evaluate(Population, N, I, \pi)$
Step 3: for generation $= 1$ to G_{max} **do**
    Step 3.1: Select: select parents based on their fitness
    Parents $\leftarrow$ *Select(Population, Fitness)*
    Step 3.2: Crossover: create offspring by combining features of parents
    $Children \leftarrow Crossover(Parents, p_c)$
    Step 3.3: Mutate: introduce random changes in the offspring
    $Children \leftarrow Mutate(Children, p_m)$
    Step 3.4: Evaluate: evaluate the fitness of the new individuals
    $NewFitness \leftarrow Evaluate(Children, N, I, \pi)$
    Step 3.5: Replace: replace the population with the best individuals from Parents and Children
    Population $\leftarrow$ *Replace(Parents, Children, Fitness, NewFitness)*
    Step 3.6: Termination: check termination criteria (e.g., no improvement or maximum fitness reached)
    if Termination*(Fitness, NewFitness)* **then**
        Break
    end if
end for
Step 4: $Routes^* \leftarrow SelectBest(Population, Fitness)$
Step 5: Calculate total reduced cost $z^{reduced}$ according to Algorithm 3
Step 6: return $Routes^*, z_{min}^{reduced}$

Algorithm 3 Calculate the total reduced cost $z^{reduced}$

Require: *Route*: a decoded sequence of customers (e.g., $\{Source, k_1, \cdots, k_n, Sink\}$); C^1, C^2, C_{ij}: cost-related parameters; π: dual price obtained from solving [RRMP].

Ensure: $z^{reduced}$: the total reduced cost, including both first- and second-echelon contributions.

1: **Step 1:** Calculate the first-echelon reduced cost:

$$z_1^{reduced} = C^1 \cdot (C_{ij}[Sink, k] + C_{ij}[k, Source]),$$

where k is the first route node representing the swap location; set $z_2^{reduced} = 0$.

2: **Step 2: for** each edge (i, j) in $Route \setminus \{Source,\ Sink\}$ **do**

3: $z_2^{reduced} \leftarrow z_2^{reduced} + C^2 \cdot C_{ij}[i, j] - \frac{\pi_i}{2} - \frac{\pi_j}{2}$

4: **end for**

5: **Step 3:** Calculate the total reduced cost:

$$z^{reduced} = z_1^{reduced} + z_2^{reduced}$$

6: **Step 4: return** $z^{reduced}$

4.2.1. Sequence-based encoding

Following the canonical genetic algorithm framework (Baker and Ayechew, 2003), each individual in the population is represented as a chromosome, a sequence of genes. In the implementation, a chromosome p represents a permutation of customers, where each gene $p[i] \in N$ corresponds to a unique customer from the set N. This permutation-based encoding directly maps to vehicle routing decisions, with the sequence order determining the visitation schedule. The complete encoding scheme is described in Algorithm 4.

Algorithm 4 Sequence-based encoding

Require: N: set of customers; P: population size.

Ensure: *Population*: a set of chromosomes.

1: **Step 1:** Initialize $Population \leftarrow \emptyset$

2: **Step 2: for** $i = 0$ to P **do**

3: **Step 2.1:** Generate a random permutation of N

4: **Step 2.2:** Append Chromosome to the Population

5: **end for**

6: **Step 3: return** *Population*

Algorithm 5 Segmented decoding method

Require: N: a set of customers; $I = [C_{ij}, Q_i, \bar{Q}, C^1, C^2]$; R: a chromosome as a sequence of customer indices according to Algorithm 4; π: dual price obtained from solving [RRMP].

Ensure: $Route^*$: an optimal route with minimal reduced cost; $z_{min}^{reduced}$: minimum reduced cost.

Step 1: Initialize $Routes \leftarrow \emptyset$; $Route \leftarrow [Source]$; $RemainingCapacity \leftarrow \bar{Q}$
Step 2: for each $geneIndex$ in R **do**
  Step 2.1: $Customer \leftarrow R[geneIndex]$
  Step 2.2: $Demand \leftarrow Q[Customer]$
end for
Step 3: if $RemainingCapacity \geq Demand$ **then**
  Step 3.1: Append $Customer$ to $Route$
  Step 3.2: Update $RemainingCapacity \leftarrow RemainingCapacity - Demand$
else
  Step 3.3: Append $SwapLocation$ and $Sink$ to $Route$
  Step 3.4: Append $Route$ to $Routes$: $Routes \leftarrow Routes \cup \{Route\}$
  Step 3.5: $SwapLocation \leftarrow Customer$
  Step 3.6: $Route \leftarrow [Source, SwapLocation]$
  Step 3.7: Update $RemainingCapacity \leftarrow \bar{Q} - Demand$
end if
Step 4: Initialize $Route^* \leftarrow \emptyset$; $z_{min}^{reduced} \leftarrow +\infty$
Step 5: for each $Route$ in $Routes$ **do**
  Step 5.1: Calculate reduced cost for current route
  Step 5.2:

$$z^{reduced} \leftarrow f^{rdc}(Route; C^1, C^2, C_{ij}; \pi)$$

  where $f^{rdc}(\cdot)$ is the reduced cost function computed in Algorithm 3.
  Step 5.3: if $z^{reduced} < z_{min}^{reduced}$ **then**
    Step 5.3.1: Set $Route^* \leftarrow Route$
    Step 5.3.2: Update $z_{min}^{reduced} \leftarrow z^{reduced}$
  end if
end for
Step 6: return $Route^*, z_{min}^{reduced}$

4.2.2. Segmented decoding

Based on the hierarchical nature of 2E-SBD, we design a segmented decoding method that decomposes the gene sequence into multiple segments, constructing first- and second-echelon routes through sequential decoding while enforcing problem-specific constraints during the decoding process. As illustrated in Figure 5, a chromosome with $|N|$ genes, denoted by a vector R containing two dummy nodes *Source* and *Sink*, is segmented into multiple sub-routes. Specifically, the chromosome is decomposed into four route segments, represented as:

$$R[Source,3], R[3,i], R[i,i+1], R[i+1,Sink].$$

For example, the first sub-route $R[Source,3]$ starts from the *Source* node and includes customers 1, 2, and 3. The second sub-route $R[3,i]$ starts from customer 3 and continues to customer i, forming part of the second-echelon delivery route. Here, $R[a,b]$ represents the subsequence of customers corresponding to the sliced gene indices $a, a+1, \cdots, b-1$ in R, where *Source* and *Sink* are assigned the values 0 and $|N|+1$, respectively. The segmented decoding method, as implemented in Algorithm 5, transforms a customer sequence into a feasible 2E-ESPPRC route while minimizing the reduced cost. The dual values obtained from solving the [RRMP] are incorporated as critical pricing parameters. Algorithm 3 computes route costs based on the reduced cost metric rather than the original [2E-SBD] objective function. The decoder strategically returns only the most promising partial customer routes instead of complete solutions. The reduced cost of each candidate route is calculated using the specialized pricing procedure defined in Algorithm 2.

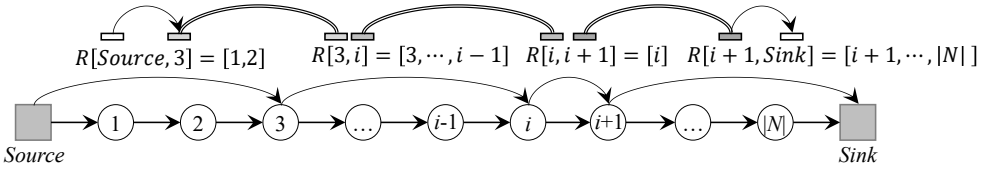


Figure 5. *The segmented decoding method.*

4.2.3. Global decoding

The gene sequence encodes multiple potential segments, many of which may satisfy the 2E-SBD constraints but are not initially selected as candidate routes. For example, the segment $R[i-1,i+1]$ may be feasible but overlooked during the initial decoding process. To maximize the utilization of genetic information, we propose a global decoding strategy that systematically explores all possible subsequences $[a,b]$ within the chromosome R. As illustrated in Figure 6, the chromosome R (comprising $|N|$ genes) is partitioned into a series of candidate routes. Algorithm 6 implements this strategy by exhaustively evaluating every possible segment $[a,b]$ and constructing the corresponding route $R[a,b]$ only if it satisfies the 2E-SBD constraints.

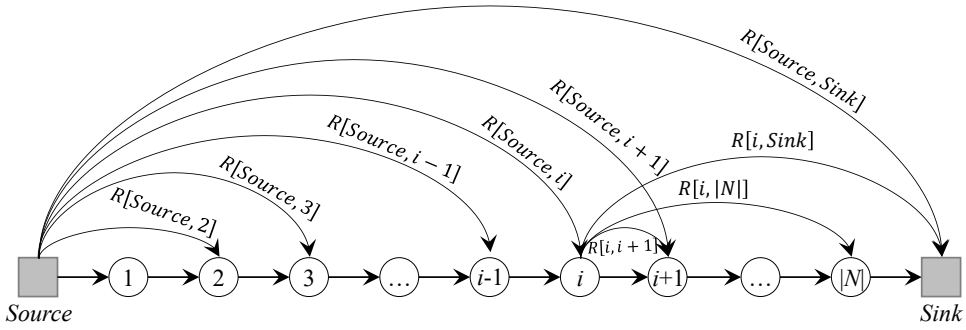


Figure 6. *The global decoding method.*

Algorithm 6 Global decoding method

Require: N: set of customers; $I = [C_{ij}, Q_i, \bar{Q}, C^1, C^2]$; R: a chromosome as a sequence of customer indices according to Algorithm 4; π: dual price obtained from solving [RRMP].

Ensure: $Route^*$: the optimal 2E-SBD route with minimal reduced cost; $z_{min}^{reduced}$: minimum reduced cost.

Step 1: Initialize $Routes \leftarrow \emptyset$

Step 2: for each pair $(geneIndex_a, geneIndex_b) \in [1,|R|] \times [1,|R|]$ **do**

  Step 2.1: if $geneIndex_a \geq geneIndex_b$ **then**

    continue

  end if

  Step 2.2: $Route \leftarrow R[geneIndex_a, geneIndex_b]$

  Step 2.3: $Demand \leftarrow \sum_{i \in Route} Q[i]$

  Step 2.4: if $Demand > \bar{Q}$ **then**

    continue

  end if

  Step 2.5: $Routes \leftarrow Routes \cup \{Route\}$

end for

Step 3:

$$Route^* = \arg\min_{Route \in Routes} \{f^{rdc}(Route; C^1, C^2, C_{ij}; \pi)\}$$

where $f^{rdc}(\cdot)$ is the reduced cost function computed in Algorithm 3.

Step 4: return $Route^*$

5. Numerical experiments

This section introduces ideal instances and benchmark datasets sourced from VRPLIB (http://vrp.atd-lab.inf.puc-rio.br/index.php/en/), which were used to conduct numerical experiments to evaluate and validate the effectiveness of the proposed models and algorithms. All models and algorithms were implemented in Python 3.8 and executed on an 11th Gen Intel Core i7-1165G7 CPU with 16 GB RAM and a 64-bit Windows 10 OS.

5.1. Instances and settings

5.1.1. Instances

We developed an ideal dataset, denoted as [IDEAL] that comprises one global depot, four swap locations, and sixteen customers. To evaluate the performance of the proposed models and algorithms, we also tested them on benchmark instances derived from the Capacitated Vehicle Routing Problem Library (CVRPLIB). The CVRPLIB datasets follow the format "a-ni-ks," where "a" is the dataset name, "i" represents the number of nodes (customers), and "s" denotes the number of vehicles. Based on the number of customers, we choose three groups of CVRPLIB datasets: Small (S) = [P-n16-k8, P-n23-k8], Medium (M) = [P-n40-k5, P-n45-k5, P-n50-k10, P-n55-k10, P-n60-k10], and Large (L) = [P-n65-k10, P-n70-k10, P-n76-k5, P-n101-k4].

We developed an ideal dataset, denoted as [IDEAL] that comprises one global depot, four swap locations, and sixteen customers. To evaluate the performance of the proposed models and algorithms, we also tested them on benchmark instances derived from the Capacitated Vehicle Routing Problem Library (CVRPLIB). The CVRPLIB datasets follow the format "a-ni-ks", where "a" is the dataset name, "i" represents the number of nodes (customers), and "s" denotes the number of vehicles. For example, "P-n16-k8" corresponds to dataset "P", with 16 nodes and 8 vehicles. In this study, the parameter "i" refers to the number of customers, while the number of swap locations, "s", is determined by the model's solution. Based on the number of customers, we choose three groups of CVRPLIB datasets: Small (S) = [P-n16-k8, P-n23-k8], Medium (M) = [P-n40-k5, P-n45-k5, P-n50-k10, P-n55-k10, P-n60-k10], and Large (L) = [P-n65-k10, P-n70-k10, P-n76-k5, P-n101-k4].

Since the original CVRPLIB instances are designed for classical capacitated vehicle routing problems, they were transformed into 2E-SBD instances according to the following rules. The original depot was designated the global depot in the first echelon, while each customer node was considered a candidate swap location. Customer demands and geographical coordinates were directly inherited from the original CVRPLIB instances. The vehicle capacity parameter was used as the swap-body container capacity in the proposed model. The number of active swap locations and the corresponding first- and second-echelon routes were determined by the optimization model rather than being predefined.

All optimization models were solved using IBM ILOG CPLEX Optimization Studio 22.1.1. The default CPLEX feasibility and optimality tolerances were adopted, with a relative MIP gap tolerance of 0.01% unless otherwise specified. The maximum computational time limit was set to 10,800 seconds for each instance.

5.1.2. Settings

To compare the computational performance of various algorithms for solving the 2E-SBD problem, the following definitions are established for the relevant solution approaches. Employing Cplex to directly solve the 2E-SBD problem through MILP is denoted as [2E-SBD]. Applying the GA to directly tackle the 2E-SBD problem is labeled as [GA]. Utilizing the CG framework to reformulate the original problem and subsequently solve the sub-problem [SUB] using Cplex is designated as [CG+MILP]. During the solution of the sub-problem [SUB], the approach that adopts the global decoding strategy of the genetic algorithm is [CG+GAg]. Similarly, the approach using the genetic algorithm's segmented decoding strategy to solve the sub-problem [SUB] is named [CG+GAs]. Table 3 outlines the default parameters. Table 4 provides GA parameter settings.

Table 3. *Parameter settings for default values.*

Parameters	**Default values**
First-echelon unit transportation cost, c_1	3 (RMB/km)
Second-echelon unit transportation cost, c_2	5 (RMB/km)
Delivery demand for customer i, q_i	1
The capacity of the swap-body container, Q	4
The computing time limit of the solver	10800 (seconds)
Convergence threshold, ε	0.001

Table 4. *Default parameter settings for GA.*

Parameters	**Default values**
Mutation probability (p_m)	0.3
Crossover probability (p_c)	0.8
Size of the population (P_{size})	40
Maximum iterations (G_{max})	100

To evaluate the performance of different solution approaches, three indicators are reported. Computational time (CT) denotes the CPU time required by each algorithm to obtain the final solution, measured in seconds. Win-times represents the number of test instances in which an algorithm achieves the best objective value among all compared methods. GAP represents the relative deviation between the obtained and reference

solutions. For exact methods, GAP corresponds to the solver's optimality gap. For heuristic approaches, GAP denotes the reported empirical gap calculated based on the available reference solution.

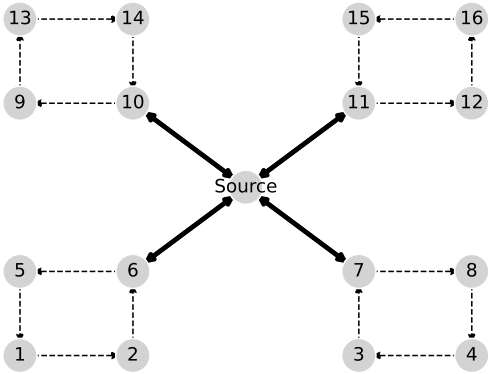


Figure 7. *A demonstrative diagram of 2E-SBD*

Table 5. *Comparative performance of solution methods on [IDEAL] instance.*

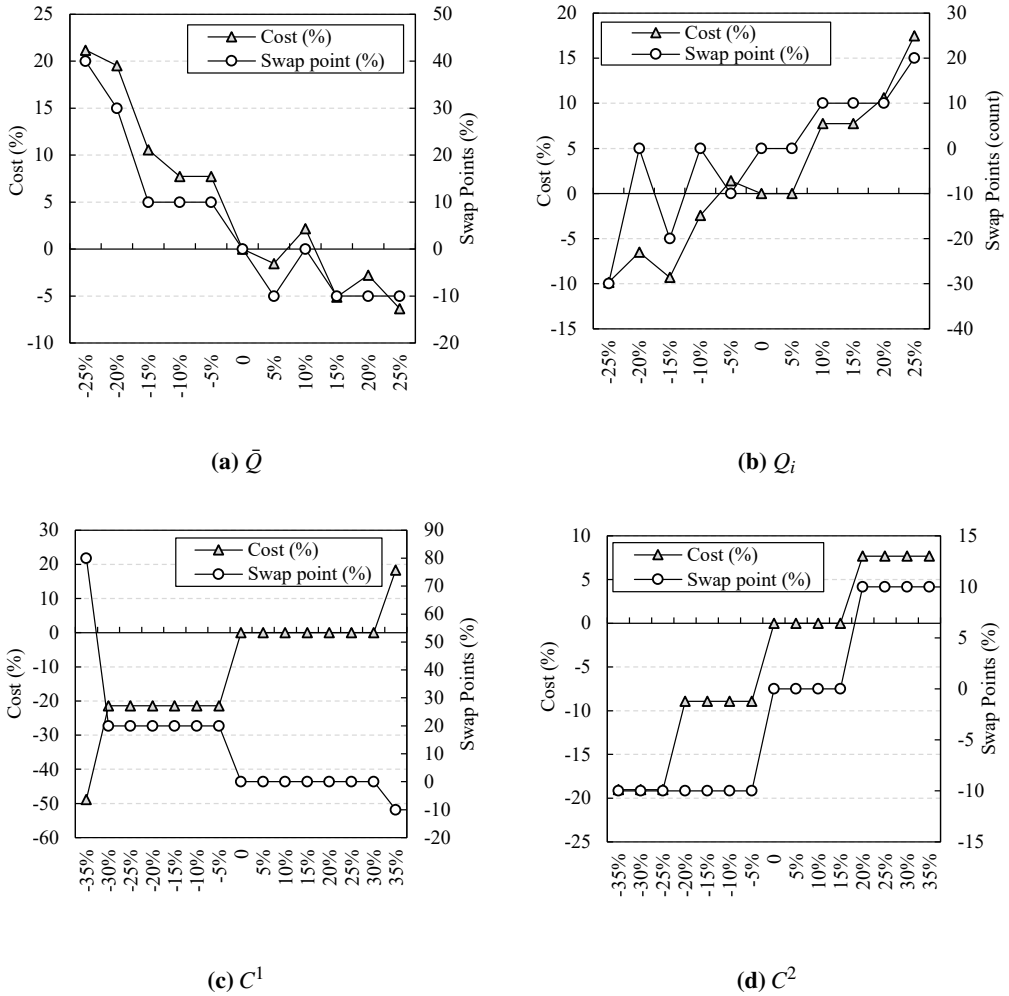
Instance	**Objective**[#] **z/RMB**	**[2E-SBD] CT/s**	**[GA]** CT^{avg}**/s**	**[CG+MILP] CT/s**	**[CG+GAg]** CT^{avg}**/s**	**[CG+GAs]** CT^{avg}**/s**
[IDEAL]	227.88	21.68	0.00	10.59	54.53	2.24

Note: CT = Computing time for single run; CT^{avg} = Average computing time across ten independent runs; Objective[#]: five solution methods all achieve the optima.

5.2. Results

5.2.1. Validation of model and algorithm efficacy

The developed models and algorithms are tested using the [IDEAL] dataset, yielding consistent results. As shown in Figure 7, four swap locations were chosen at nodes 6, 7, 10, and 11 based on capacity and demands. The results of five solution methods for solving [IDEAL] are shown in Table 5. For [GA], [CG+GAg], and [CG+GAs], the reported computing times are averaged over 10 independent runs. The optimal total distribution cost Z^* for the first and second echelons is 227.88 RMB, with all methods achieving the same optimal value. [GA] is dominantly superior. [CG+GAs] outperforms [2E-SBD] and [CG+MILP] in optimality and computing time. The average computing time for [CG+GAs] was 2.24 seconds, while [CG+MILP] incurred 10.59 seconds, [2E-SBD] reached 21.68 seconds, and [CG+GAg] was 54.53 seconds. [CG+GAs] reduces runtime by 89.7% compared to standard MILP (2.24 vs. 21.68 seconds) while preserving solution optimality. These results highlight the competitiveness of [CG+GAs] in both solution speed and effectiveness.



(a) $\bar{Q}$ **(b)** Q_i

(c) C^1 **(d)** C^2

Figure 8. *Sensitivities of the four data parameters* ($\bar{Q}, Q_i, C^1, C^2$).

5.2.2. Parameter sensitivities analysis

Using the [CG+MILP] approach with the P-n23-k8 dataset to analyze the impacts of the critical parameters in the 2E-SBD model, namely $\bar{Q}$, Q_i, C^1, and C^2 on total cost and swap location configurations, each parameter is systematically varied from -25% to $+25\%$ in 5% increments. The results depicted in Figure 8, reveal three key findings. First, the cost decreases with the variations in $\bar{Q}$, from negative (-25%) to positive (+25%), while it increases for Q_i, C^1, and C^2. Second, although the general trends are significant, the cost curves for $\bar{Q}$ and Q_i exhibit slight fluctuations. Third, the number of swap locations decreases for $\bar{Q}$ and C^1, but increases for Q_i and C^2, though these trends are not absolute, and show some fluctuations. Finally, the slopes of the cost variations are consistent for $\bar{Q}$, Q_i, and C^2, but contradictory for C^1.

Table 6. *Variance ranges of costs and swap locations of the four data parameters ($\bar{Q}$, Q_i, C^1 and C^2).*

Variances (from -25% to 25%)	$\bar{Q}$	Q_i	C^1	C^2
Cost ($\max z - \min z$)	27.54	27.31	67.14	26.70
Swap locations ($\max\{K\} - \min\{K\}$)	50.00	50.00	90.00	20.00

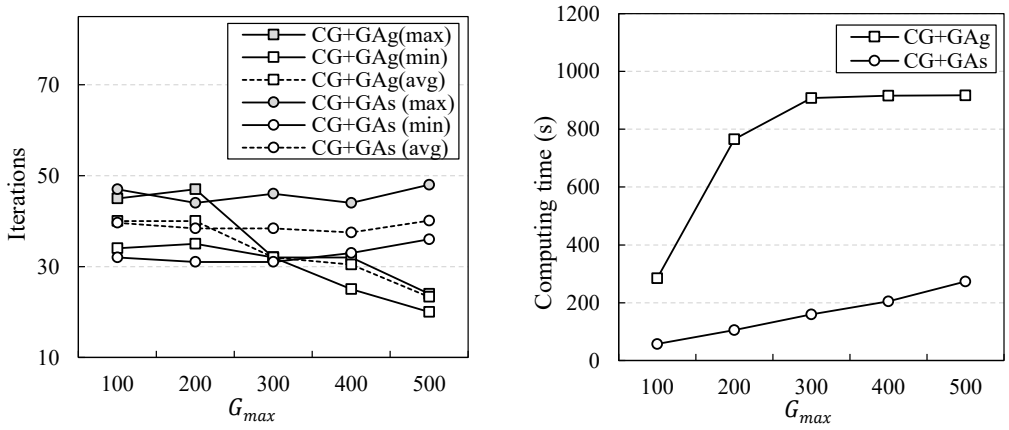
Note: $\{K\}$ is the set of chosen swap locations.

As summarized in Table 6, the cost and swap location ranges, adjusted from -25% to +25%, indicate that, in terms of cost, the parameter importance order is $C^1 > \bar{Q} > Q_i > C^2$, for swap location configuration the ordering is $C^1 > \bar{Q} = Q_i > C^2$, where the notion $a > b$ denotes parameter a with greater impact relative to b on the performance metric. Notably, C^1 dominates both metrics, with a 1-unit reduction yielding a 1.34% cost saving (67.14/50 = 1.34). While $\bar{Q}$ and Q_i exhibit comparable effects, the cost parameters (C^1, C^2) demonstrate stronger marginal impacts.

5.2.3. Generic parameter impacts

To assess the impacts of the maximum number of generations ($G_{\max}$) in [CG+GA] on solution optimality and computational time, experiments are conducted on P-n23-k8 using [CG+GAg] and [CG+GAs], with $G_{\max}$ varying from 100 to 500 in increments of 100. Results are shown in Figure 9. Optimality was evaluated based on the number of CG iterations required until [CG+GA] could no longer identify new columns with negative reduced costs, or when the gap between the upper and lower bounds fell below a predefined threshold. Figure 9(a) illustrates the relationship between $G_{\max}$ and optimality. For $G_{\max} \geq 300$, [CG+GAg] converges faster than [CG+GAs], as it considers all possible segments as candidate 2E-ESPPRC routes, leveraging more information. However, for $G_{\max} < 300$, the advantage of [CG+GAg] over [CG+GAs] diminishes. In early generations, chromosomes may be of suboptimal quality, and exploring additional segments yields minimal improvement. Overall, [CG+GAs] shows lower sensitivity to $G_{\max}$, suggesting that its genetic operator does not fully exploit its potential. Consequently, [CG+GAs] reaches its evolutionary objectives with fewer generations without sacrificing optimality. Figure 9(b) presents the computational time comparison between [CG+GAg] and [CG+GAs] on the P-n23-k8 instance, as $G_{\max}$ varies from 100 to 500. The results demonstrate a near-linear increase in computational time for [CG+GAs] with respect to $G_{\max}$. In contrast, [CG+GAg] exhibits significantly faster growth, reaching the 900-second time limit at higher $G_{\max}$ values. This pronounced difference stems from the computationally intensive decoding process inherent to [CG+GAg], which requires cost evaluation for all chromosome segments.

To further validate the robustness of this setting under GA stochasticity, we repeated the above configuration with 10 independent trials per generation setting, benchmarking each trial against the exact MILP solution via a Wilcoxon signed-rank test with a numer-



(a) *Iterations of convergence* **(b)** *Computing time*

Figure 9. *Impacts of G_{max} on [CG+GA] optimality and computing time.*

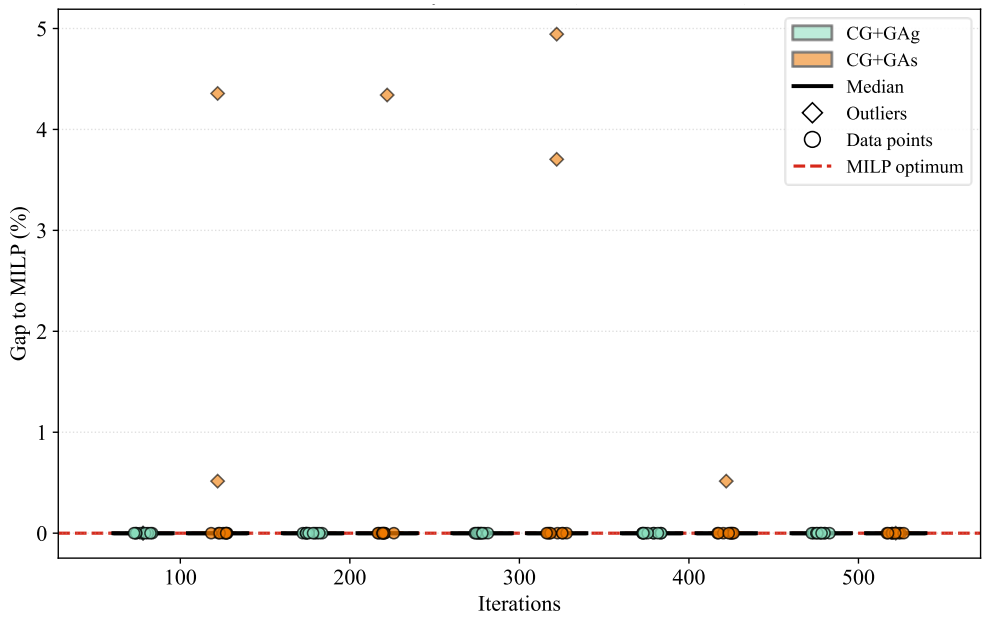
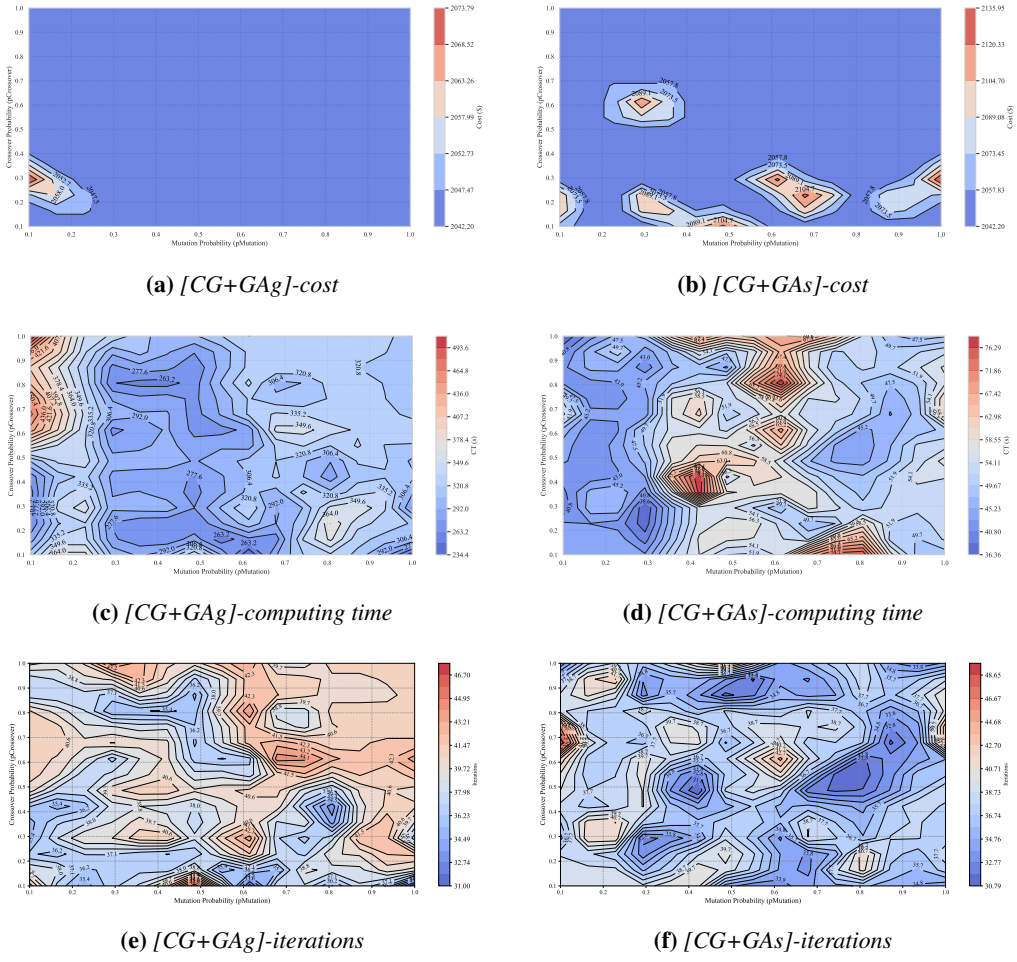


Figure 10. *Robustness of [CG+GA] solution quality across G_{max}.*

ical tolerance of 10^{-6} (García et al., 2009). Figure 10 shows that [CG+GAg] matched the MILP optimum in all runs across all generation settings, with zero deviation and no statistically significant difference from the benchmark. [CG+GAs] achieved a median gap of 0% at all settings but showed occasional deviations at lower generation counts, reaching a maximum gap of 13.1% when G_{max} equals 100, 4.9% at 300, and 3.4% at 400,

converging only when G_{max} reached 500. This confirms that [CG+GAg] is substantially more robust than [CG+GAs].



(a) *[CG+GAg]-cost* **(b)** *[CG+GAs]-cost*

(c) *[CG+GAg]-computing time* **(d)** *[CG+GAs]-computing time*

(e) *[CG+GAg]-iterations* **(f)** *[CG+GAs]-iterations*

Figure 11. *Cross-experiment contours of mutation probability and crossover probability.*

The cross-experiment analysis evaluates the impact of p_m and p_c on the performance of hybrid algorithms [CG+GAg] and [CG+GAs] using the P-n23-k8 dataset. Figure 11 presents contour plots systematically examining the parameter space with p_m and p_c, varying continuously from 0.1 to 1.0, to visualize the relationships between genetic operator probabilities and three critical performance metrics: solution cost, computational time, and termination iteration count. The results demonstrate that no single combination of p_m and p_c can simultaneously optimize all performance metrics. For minimizing solution cost, the optimal parameter range was found to be $p_m \in [0.1, 0.2]$ with $p_c \in [0.1, 0.3]$. When computational time is prioritized, parameter combinations of $p_m \in [0.4, 0.6]$ and $p_c \in [0.2, 0.4]$ demonstrated superior performance. For achieving faster

convergence (fewer iterations), the recommended ranges are $p_m \in [0.1, 0.2]$ with $p_c \in [0.1, 0.3]$. A balanced trade-off between solution quality, computational efficiency, and convergence rate can be achieved using $p_m \in [0.3, 0.4]$ combined with $p_c \in [0.2, 0.3]$.

Table 7. *Computing time, objective values, and gaps of the five solution methods.*

Instance	[2E-SBD]			[GA]			[CG+MILP]			[CG+GAg]			[CG+GAs]		
	z	GAP	CT/s	z	GAP	CT/s	z	GAP	CT/s	z	GAP	CT/s	z	GAP	CT/s
P-n16-k8	1632	0	1	1632	0	0	1674	3	19	1674	3	90	1674	3	23
P-n20-k2	1020	0	2073	1020	0	0	1057	4	920	1185	16	773	1337	31	7
P-n23-k8	2042	0	600	2157	6	0	2042	0	280	2042	0	240	2042	0	43
P-n40-k5	2614	0	10800	3543	36	0	4204	61	934	2643	1	906	3122	19	20
P-n45-k5	3281	0	10800	3994	22	0	5019	53	937	3375	3	923	3803	16	19
P-n50-k10	3681	27	10800	5153	40	0	5092	38	1095	3757	2	914	3681	0	93
P-n55-k10	3998	23	10800	5390	35	0	6311	58	968	4046	1	917	3998	0	88
P-n60-k10	4495	16	10800	6442	43	0	6558	46	1011	4628	3	903	4495	0	105
P-n65-k10	5069	21	10800	6998	38	0	8063	59	1029	5069	0	905	5216	3	106
P-n70-k10	5458	18	10800	7612	39	0	8049	47	1062	5810	6	902	5458	0	132
P-n76-k5	4946	0	10800	7421	50	0	8290	68	1105	6544	32	919	6913	40	18
P-n101-k4	9660	55	10800	9660	0	0	14968	55	443	10165	5	946	10176	5	55
Win	-	35	3	-	19	48	-	10	14	-	35	21	-	34	34

Note: Win = (GAP wins, CT wins).

5.2.4. Comparisons of computing time and optimality

Table 7 benchmarks five solution methods ([2E-SBD], [GA], [CG+MILP], [CG+GAg], [CG+GAs]) across twelve datasets using three evaluation metrics: (1) absolute objective values (z), (2) percentage reported gaps (GAP = $100\% \times \frac{z-z^*}{z^*}$), and (3) computational times. The comparative analysis uses a "win-times" metric to quantify relative dominance in terms of solution quality or speed. The results demonstrate that [2E-SBD] and [CG+GAs] achieve the best solution quality, each dominating 35 cases in empirical gap comparisons. However, their computational profiles differ substantially. While [2E-SBD] produces the minimum objective value in 8 instances, it fails to converge within the time limit in 9 cases. In contrast, [CG+GAs] maintains excellent solution quality, with gaps under 3% across 7 instances, while also delivering superior computational speed, achieving the fastest solutions in 9 cases and leading the speed rankings with 34 wins. Other methods show more limited performance. [CG+MILP] ranks third in solution quality with 19 wins but last in computational speed. [GA] exhibits inconsistent performance, finding optimal solutions in four instances but showing significant degradation on larger problems. [CG+GAg] trails in solution quality with just three wins, though it places second in computational efficiency. The "P-n40-k5" instance typifies these trends, where [CG+GAs] outperforms two competitors while maintaining reasonable computation time.

Overall, the win-times analysis demonstrates that [CG+GAs] represents the most balanced solution methodology, achieving solution quality comparable to [2E-SBD] while exhibiting improved computational efficiency.

The competitiveness of [CG+GAs] among the five solution methods is evaluated based on the "win-times" illustrated in Figure 12. The figure shows a quadrant analysis of the solution space, where each method's position reflects its relative performance in the reported gap and computational time. The size of the bubbles reflects the combined GAP and CT "win-times," with the precise values annotated near each bubble. [CG+MILP] serves as a baseline for evaluating both optimality and computational efficiency. Regarding optimality ("GAP"), the ranking is [2E-SBD] $\succ$ [CG+GAg] $\succ$ [CG+GAs] $\succ$ [GA] $\succ$ [CG+MILP], where $A \succ B$ indicates that method A outperforms B on the specified metric. For computational time, the ranking is: [CG+GAs] $\succ$ [GA] $\succ$ [CG+GAg] $\succ$ [2E-SBD] $\succ$ [CG+MILP]. [GA] excels in computational time, while [2E-SBD] is optimal in terms of solution quality, as MILP solvers achieve optimal solutions given sufficient computational resources. [CG+GAg] and [CG+GAs] occupy the quadrant where both GAP and CT metrics are competitive, demonstrating strong overall performance. Specifically, [CG+GAg] achieves 35 GAP "win-times" and 21 CT "win-times", while [CG+GAs] achieves 34 GAP "win-times" and 34 CT "win-times". Although [CG+GAg] and [CG+GAs] are not directly comparable across the two metrics, a unit loss in GAP for [CG+GAs] corresponds to a significant gain of 13 CT units compared to [CG+GAg].

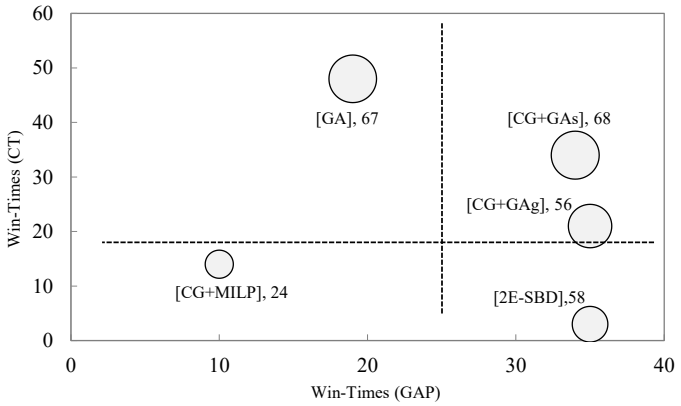


Figure 12. *The comparative "win-times" of the five solution methods.*

To further validate robustness across problem scales, we extended the evaluation to 12 benchmark instances, with each configuration run 10 times independently. Figure 13 shows that [CG+GAg] consistently outperforms [CG+GAs] across instances, with a mean gap of 5.8% versus 24.4% and a median of 2.6% versus 18.7%. This gap widened in larger instances, such as P-n76-k5 and P-n101-k4, where [CG+GAg] remained below 15% while [CG+GAs] exceeded 65%. [CG+GAg] also remained competitive with the standalone GA baseline on several instances, while consistently outperforming [CG+MILP], which

exceeded a 50% gap in multiple large-scale cases. These results, based on repeated independent trials across diverse instance sizes, confirm [CG+GAg]'s superior scalability and reliability across problem sizes.

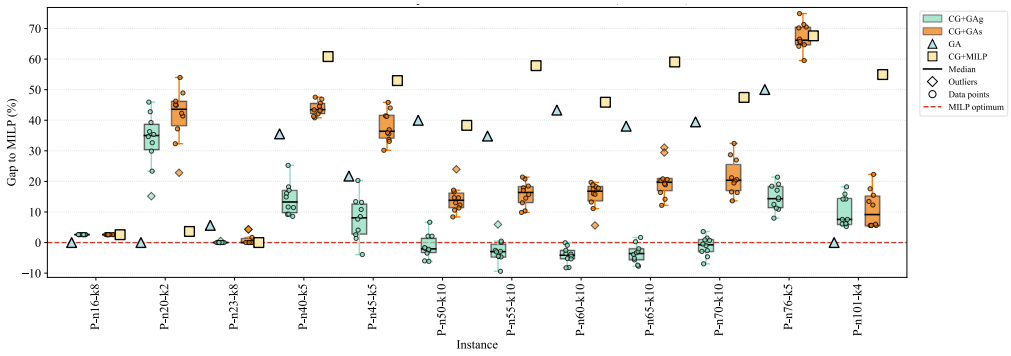
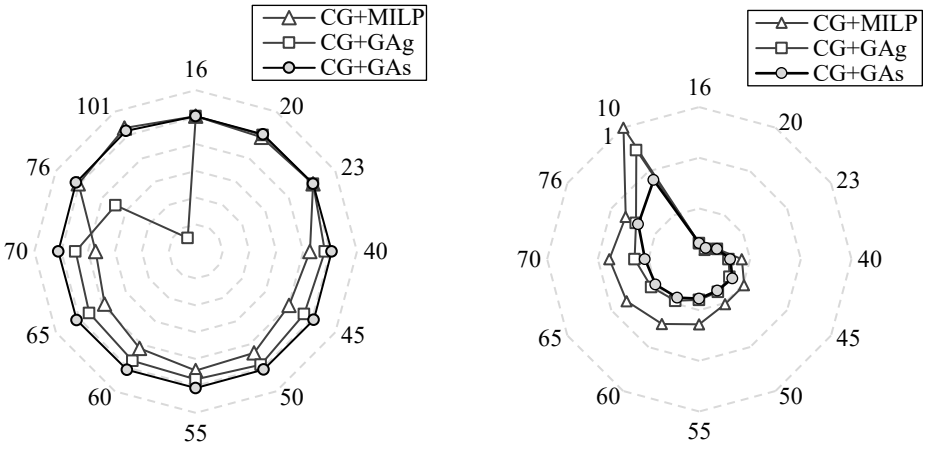


Figure 13. *Robustness of [CG+GA] solution quality across 12 benchmark instances.*

Figures 14(a) and 14(b) depict the lower and upper bounds obtained by the CG-based solution methods across 12 datasets with varying customer counts. Notably, [CG+GAs] consistently delivers the best bounds, surpassing both [CG+GAg] and [CG+MILP], the latter of which yields less satisfactory results. For datasets with fewer customers, the bounds produced by all three methods are highly similar, with some overlap. However, as the customer count increases, the disparity in the bounds becomes more pronounced, with [CG+GAs] continuing to outperform the others.



(a) *Lower bounds* **(b)** *Upper bounds*

Figure 14. *Lower and upper bounds achieved by CG-based solution methods.*

The iteration processes of the three methods for the P-n23-k8 instance are illustrated in Figure 15. All algorithms begin with identical UB and LB values and progressively

improve them. The [CG+MILP] method converges to UB and LB values of 2042.20, but the process takes 279.84 seconds, limiting its scalability for large instances. While the average and minimum computational times of [CG+MILP] are 284.73 and 240.31 seconds, respectively, it remains slower than [CG+GAs]. In contrast, [CG+GAs] achieves the same UB and LB values of 2042.20 but with a significant reduction in computation time. Specifically, [CG+GAs] shows a 79.9% reduction in average computation time over 10 runs compared to [CG+MILP] (roughly 1/5 of [CG+MILP] time). [CG+GAs] achieves these bounds with an average time of 57.3 seconds and a best-case time of 42.9 seconds. These results indicate that [CG+GAs] provides substantial efficiency improvements, outperforming [CG+MILP] in both optimality and computational time.

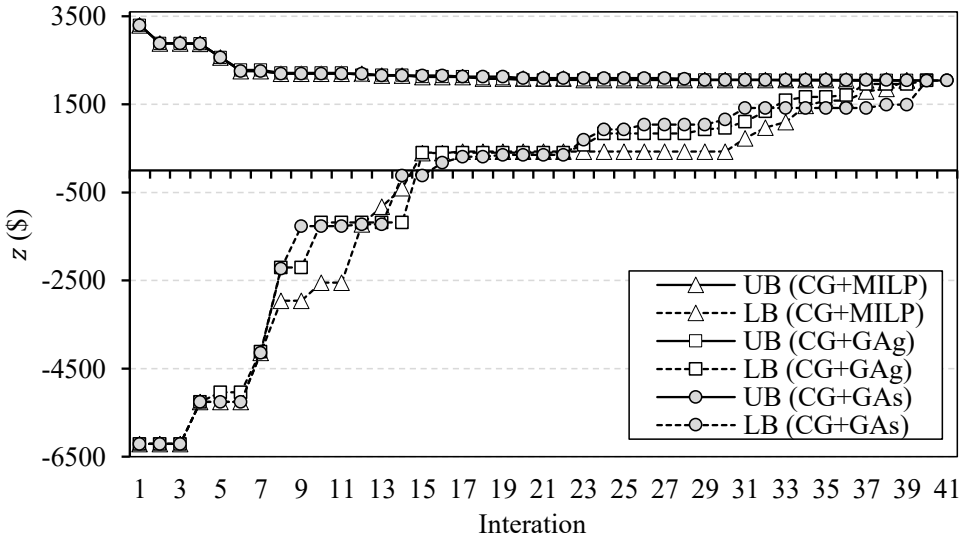
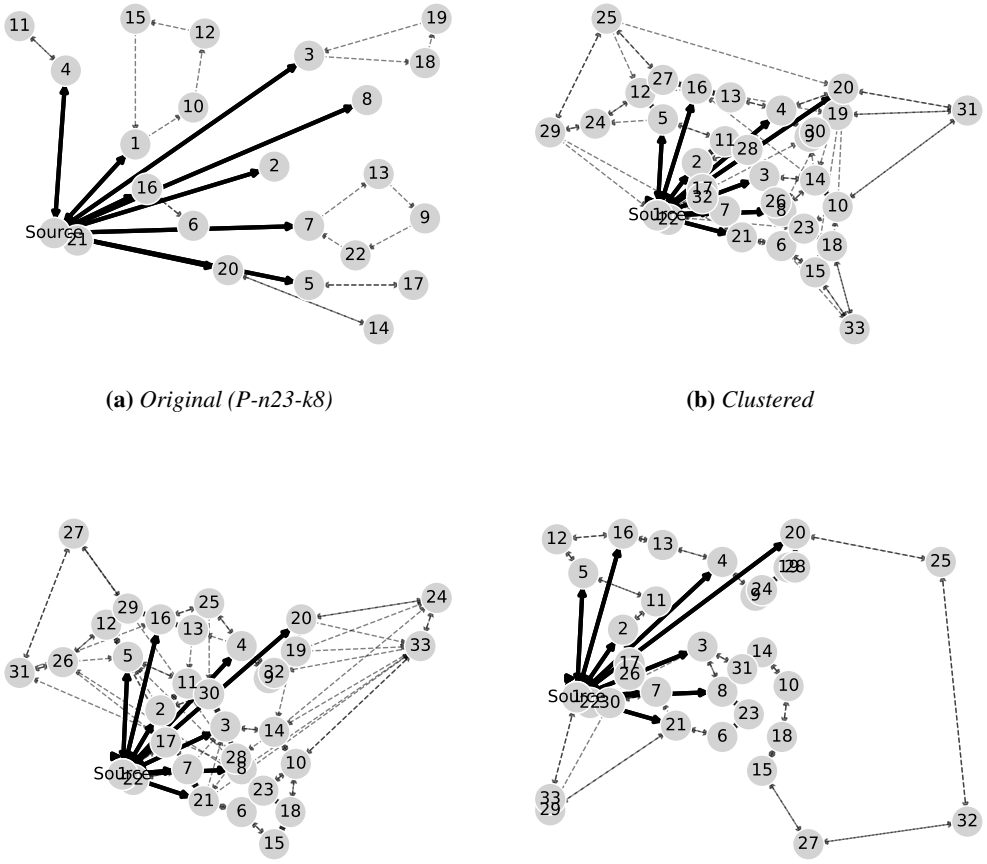


Figure 15. *The evolution of bounds of CG-based solution methods.*

5.2.5. Impacts of new customers on costs

Using the "P-n23-k8" dataset to assess the impact of adding new customers to the solution. The vehicle capacity $\bar{Q}$ is set to 40, with customer demands Q_i ranging from [5,30]. The customer nodes are located within the region [27,63] $\times$ [27,69]. First, we solved [CG+MILP] to determine 10 swap locations. Figures 16(b), 16(c), and 16(d) illustrate the results of solving [CG+MILP] with updated datasets incorporating 10 new customers (IDs 24-33) in three distributions: clustered, semi-clustered, and random, with updated locations. When new customers are added, maintaining the same total number of customers, the swap locations are re-optimized to better align with the updated customer distribution.

The results of the cost comparison between fixed and re-optimized swap locations, summarized in Table 8, reveal notable cost reductions. In the clustered dataset, the cost for



(a) *Original (P-n23-k8)*

(b) *Clustered*

(c) *Semi-clustered*

(d) *Random (uniform)*

Figure 16. *Distribution network of P-n23-k8, and after adding new customers.*

Table 8. *Comparison of the costs with fixed versus re-optimized swap locations.*

Dataset	**Fixed**		**Re-optimized**		**Cost reduction#/%**
	S^{fixed}	Z^{fixed}/RMB	S^{new}	Z^{new}/RMB	
Clustered	10	68,311.83	18	3,916.44	94.3
Semi-clustered	10	58,032.20	19	3,882.51	93.3
Random	10	67,782.33	18	3,671.37	94.6
Average	**10**	**64,708.79**	**18.33**	**3,823.44**	**94.1**

Note: S = the number of swap locations; # = $(z^{fixed} - z^{new})/z^{fixed} \cdot 100$.

fixed swap locations was 68,311.83 RMB, while the re-optimized configuration reduced the cost to 3,916.44 RMB, yielding a 94.3% reduction. In the semi-clustered dataset, the fixed swap cost was 58,032.20 RMB, compared to 3,882.51 RMB for the re-optimized

locations, resulting in a 93.3% reduction. Similarly, in the random dataset, fixed swap locations incurred a cost of 67,782.33 RMB, while the re-optimized locations reduced this to 3,671.37 RMB, achieving a 94.6% reduction. On average, the cost reduction across all datasets was 94.1%.

6. Discussions and managerial implications

Based on the computational results and sensitivity analyses, this section discusses the practical and methodological implications of the proposed 2E-SBD framework from two perspectives: operational decisions for logistics managers and algorithmic insights for optimization researchers.

6.1. Implications for logistics managers

The numerical results provide several implications for the design and operation of two-echelon swap-body distribution systems. First, the sensitivity analysis indicates that the first-echelon transportation cost coefficient (C_1) has the greatest impact on the total distribution cost, while the effects of customer demand and vehicle capacity are comparatively moderate. This finding suggests that logistics managers should pay particular attention to first-echelon transportation planning, including vehicle deployment, long-distance transportation arrangements, and cost control strategies. Improving the efficiency of first-echelon operations can yield greater cost savings than focusing solely on second-echelon routing decisions.

Second, the results highlight the importance of vehicle capacity planning in swap-body distribution systems. The sensitivity analysis demonstrates that changes in swap-body capacity influence routing structures and the selection of swap locations. Under the tested scenarios, increasing vehicle capacity can reduce the number of required routes and improve transportation efficiency when demand aggregation is beneficial. Therefore, managers should determine appropriate vehicle capacities according to demand characteristics, service requirements, and operational constraints rather than adopting larger capacities universally.

Third, the dynamic selection of swap locations provides greater operational flexibility than traditional two-echelon distribution systems that rely on fixed intermediate facilities. The computational results show that different demand distributions and customer configurations can yield distinct swap-location decisions. Therefore, logistics operators can benefit from flexible swap-location planning, allowing the distribution network to adapt to changing demand patterns without requiring permanent infrastructure expansion. However, practical implementation may still need to consider factors such as land availability, safety regulations, and operational coordination, which are beyond the scope of the current optimization model.

Fourth, the customer-integration analysis demonstrates that adding new customers affects distribution costs by altering routing structures and resource allocation. The

impact of new customer integration is not solely determined by additional demand but also depends on the spatial relationship between customers and existing swap locations. Therefore, when expanding service areas, logistics managers should evaluate customer locations and their impact on the existing distribution network, rather than focusing solely on demand growth.

6.2. Implications for algorithm development

The computational comparison among the proposed solution approaches provides insights into the design of optimization algorithms for large-scale two-echelon distribution problems. The results show that the exact [CG+MILP] approach yields reliable solutions for small-scale instances, thanks to exact pricing procedures and rigorous optimization bounds. However, its computational burden increases as the problem size grows.

In contrast, the [CG+GA] variants employ heuristic pricing strategies to improve computational efficiency for larger instances. The results demonstrate that integrating genetic algorithms into the column generation framework can generate high-quality solutions within a reasonable computational time. Among the heuristic variants, [CG+GAs] achieves a favorable balance between solution quality and computational efficiency across the tested scenarios, while [CG+GAg] offers an alternative search strategy by exploring a broader solution space via global decoding.

Nevertheless, since GA-based pricing relies on heuristic search, the solutions obtained are the best found during the search process rather than those with guaranteed optimality certification. Future algorithm development may focus on improving the robustness of heuristic pricing, adaptive parameter adjustment, and hybrid exact-heuristic strategies to further enhance scalability while maintaining solution quality.

Overall, this study demonstrates that combining mathematical decomposition with heuristic search provides an effective framework for solving complex urban distribution problems involving flexible swap-body operations. The proposed methodology offers both practical decision support for logistics planning and a potential algorithmic direction for addressing large-scale multi-echelon routing problems.

7. Conclusion

This study investigates 2E-SBD that integrates dynamic swap-location selection with multi-echelon routing decisions in urban logistics. A mixed-integer linear programming model is developed to jointly optimize swap locations, customer assignments, and distribution routes. To address the computational challenges of large-scale instances, a column generation framework combined with genetic algorithms is proposed, where exact and heuristic pricing strategies are developed for different problem scales.

The computational results demonstrate that the proposed approaches can effectively solve the 2E-SBD problem. The exact [CG+MILP] method provides reliable solutions for small-scale instances, while the CG-based heuristic approaches improve computational

efficiency for larger instances while maintaining high solution quality. The sensitivity analyses further show that first-echelon transportation costs have a dominant influence on total distribution costs, and that demand characteristics and vehicle capacity affect routing structures and swap-location decisions. These findings provide insights into the design and operation of flexible swap-body distribution networks.

This study has several limitations that suggest future research directions. First, the current model assumes deterministic customer demand and static transportation conditions. Future studies could incorporate stochastic demand, time-dependent traffic conditions, and dynamic routing decisions to better represent real-world operations. Second, practical deployment factors, such as facility constraints, operational regulations, and implementation costs for swap locations, should also be considered. Finally, more advanced hybrid algorithms and learning-based optimization methods could be explored to improve scalability for larger and more complex distribution networks.

CRediT authorship contribution statement

Yao-Na Li: Conceptualization, Methodology, Software, Visualization, Data curation, Writing-Original draft preparation. **Zhi-Hua Hu:** Methodology, Investigation, Validation, Supervision, Writing-Reviewing and Editing, and Funding acquisition. All authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgment

This research was supported by the National Natural Science Foundation of China (Grant No. 72671180) and the General Project of Humanities and Social Sciences Research of the Ministry of Education of China (Grant No. 23YJA630035).

References

Absi, N., Cattaruzza, D., Feillet, D. and Housseman, S. (2017). A relax-and-repair heuristic for the swap-body vehicle routing problem. *Annals of Operations Research*, 253(2), 957–978. doi: 10.1007/s10479-015-2098-8.

Accorsi, L. and Vigo, D. (2020). A hybrid metaheuristic for single truck and trailer routing problems. *Transportation Science*, 54(5), 1351–1371. doi: 10.1287/trsc.2019.0943.

Alves da Cruz, H.F. and Salles da Cunha, A. (2023). The profitable single truck and trailer routing problem with time windows: Formulation, valid inequalities and branch-and-cut algorithms. *Computers & Industrial Engineering*, 180, 109238. doi: 10.1016/j.cie.2023.109238.

Anderluh, A., Nolz, P.C., Hemmelmayr, V.C. and Crainic, T.G. (2021). Multi-objective optimization of a two-echelon vehicle routing problem with vehicle synchronization and 'grey zone' customers arising in urban logistics. *European Journal of Operational Research*, 289(3), 940–958. doi: 10.1016/j.ejor.2019.07.049.

Baker, B.M. and Ayechew, M.A. (2003). A genetic algorithm for the vehicle routing problem. *Computers & Operations Research*, 30(5), 787–800. doi: 10.1016/S0305-0548(02)00051-5.

Bayliss, C., Bekta, T., Tjon-Soei-Len, V. and Rohner, R. (2023). Designing a multi-modal and variable-echelon delivery system for last-mile logistics. *European Journal of Operational Research*, 307(2), 645–662. doi: 10.1016/j.ejor.2022.08.041.

Belenguer, J.M., Benavent, E., Martínez, A., Prins, C., Prodhon, C. and Villegas, J.G. (2016). A branch-and-cut algorithm for the single truck and trailer routing problem with satellite depots. *Transportation Science*, 50(2), 735–749. doi: 10.1287/trsc.2014.0571.

Chao, I-M. (2002). A tabu search method for the truck and trailer routing problem. *Computers & Operations Research*, 29(1), 33–51. doi: 10.1016/s0305-0548(00)00056-3.

Contardo, C., Hemmelmayr, V. and Crainic, T.G. (2012). Lower and upper bounds for the two-echelon capacitated location-routing problem. *Computers & Operations Research*, 39(12), 3185–3199. doi: 10.1016/j.cor.2012.04.003.

Dellaert, N., Dashty Saridarq, F., Van Woensel, T. and Crainic, T.G. (2019). Branch-and-price-based algorithms for the two-echelon vehicle routing problem with time windows. *Transportation Science*, 53(2), 463–479. doi: 10.1287/trsc.2018.0844.

Derigs, U., Pullmann, M. and Vogel, U. (2013). Truck and trailer routing-problems, heuristics and computational experience. *Computers & Operations Research*, 40(2), 536–546. doi: 10.1016/j.cor.2012.08.007.

Escobar-Vargas, D. and Crainic, T.G. (2024). Multi-attribute two-echelon location routing: Formulation and dynamic discretization discovery approach. *European Journal of Operational Research*, 314(1), 66–78. doi: 10.1016/j.ejor.2023.09.031.

García, S., Molina, D., Lozano, M. and Herrera, F. (2009). A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the cec'2005 special session on real parameter optimization. *Journal of Heuristics*, 15(6), 617–644. doi: 10.1007/s10732-008-9080-4.

Huber, S., Cordeau, J-F. and Josef Geiger, M. (2020). A matheuristic for the swap body vehicle routing problem. *OR Spectrum*, 42(1), 111–160. doi: 10.1007/s00291-019-00570-z.

Jepsen, M., Spoorendonk, S. and Ropke, S. (2013). A branch-and-cut algorithm for the symmetric two-echelon capacitated vehicle routing problem. *Transportation Science*, 47(1), 23–37. doi: 10.1287/trsc.1110.0399.

Lan, Y-L, Liu, F-G, Huang, Z., Wing, W.Y. Ng and Zhong, J. (2022). Two-echelon dispatching problem with mobile satellites in city logistics. *IEEE Transactions on Intelligent Transportation Systems*, 23(1); 84–96. doi: 10.1109/TITS.2020.3003598.

Li, H., Lv, T. and Li, Y. (2015). The tractor and semitrailer routing problem with many-to-many demand considering carbon dioxide emissions. *Transportation Research Part D: Transport and Environment*, 34; 68–82. doi: 10.1016/j.trd.2014.10.004.

Li, X., Yan, P., Yu, K., Li, P. and Liu, Y. (2024). Parcel consolidation approach and routing algorithm for last-mile delivery by unmanned aerial vehicles. *Expert Systems with Applications*, 238; 122149. doi: 10.1016/j.eswa.2023.122149.

Lin, S-W, Yu, V.F. and Chou, S-Y. (2009). Solving the truck and trailer routing problem based on a simulated annealing heuristic. *Computers & Operations Research*, 36(5); 1683–1692. doi: 10.1016/j.cor.2008.04.005.

Lin, S-W, Yu, V.F. and Chou, S-Y. (2010). A note on the truck and trailer routing problem. *Expert Systems with Applications*, 37(1); 899–903. doi: 10.1016/j.eswa.2009.06.077.

Macrina, G., Di Puglia Pugliese, L., Guerriero, F. and Laporte, G. (2020). Crowd-shipping with time windows and transshipment nodes. *Computers & Operations Research*, 113; 104806. doi: 10.1016/j.cor.2019.104806.

Miranda-Bront, J.J., Curcio, B.L., Méndez-Díaz, I., Montero, A., Pousa, F. and Zabala, P. (2017). A cluster-first route-second approach for the swap body vehicle routing problem. *Annals of Operations Research*, 253(2); 935–956. doi: 10.1007/s10479-016-2233-1.

Ramírez-Villamil, A., Montoya-Torres, J.R., Jaegler, A. and Cuevas-Torres, J.M. (2023). Reconfiguration of last-mile supply chain for parcel delivery using machine learning and routing optimization. *Computers & Industrial Engineering*, 184; 109604. doi: 10.1016/j.cie.2023.109604.

Rothenbächer, A-K, Drexl, M. and Irnich, S. (2018). Branch-and-price-and-cut for the truck-and-trailer routing problem with time windows. *Transportation Science*, 52(5); 1174–1190. doi: 10.1287/trsc.2017.0765.

Scheuerer, S. (2006). A tabu search heuristic for the truck and trailer routing problem. *Computers & Operations Research*, 33(4); 894–909. doi: 10.1016/j.cor.2004.08.002.

Sluijk, N., Florio, A.M., Kinable, J., Dellaert, N. and Van Woensel, T. (2023). Two-echelon vehicle routing problems: A literature review. *European Journal of Operational Research*, 304(3); 865–886. doi: 10.1016/j.ejor.2022.02.022.

Sutrisno, H. and Yang, C-L. (2023). A two-echelon location routing problem with mobile satellites for last-mile delivery: mathematical formulation and clustering-based heuristic method. *Annals of Operations Research*, 323(1-2); 203–228. doi: 10.1007/s10479-023-05177-w.

Tian, X-D and Hu, Z-H. (2023). A branch-and-price method for a two-echelon location routing problem with recommended satellites. *Computers & Industrial Engineering*, 184; 109593. doi: 10.1016/j.cie.2023.109593.

Todosijević, R., Hanafi, S., Urošević, D., Jarboui, B. and Gendron, B. (2017). A general variable neighborhood search for the swap-body vehicle routing problem. *Computers & Operations Research*, 78; 468–479. doi: 10.1016/j.cor.2016.01.016.

Túlio, A.M., Toffolo, J.C., Van Malderen, S., Wauters, T. and Vanden Berghe, G. (2018). Stochastic local search with learning automaton for the swap-body vehicle routing problem. *Computers & Operations Research*, 89; 68–81. doi: 10.1016/j.cor.2017.08.002.

Wang, C-L, Guo, C-C and Zuo, X-Q. (2021). Solving multi-depot electric vehicle scheduling problem by column generation and genetic algorithm. *Applied Soft Computing*, 112; 107774. doi: 10.1016/j.asoc.2021.107774.

Xue, G., Wang, Y., Guan, X. and Wang, Z. (2022). A combined ga-ts algorithm for two-echelon dynamic vehicle routing with proactive satellite stations. *Computers & Industrial Engineering*, 164; 107899. doi: 10.1016/j.cie.2021.107899.