Journal of Network Intelligence
Taiwan Ubiquitous Information
 ISSN 2414-8105 (Online)
Volume **10**, Number **4**, November **2025**

Enhanced Resource Optimization for Vehicle Routing Problems with Time Windows Using A Hybrid SSA-ALNS Algorithm

Yifan Suo

College of Computer Science and Mathematics
Fujian University of Technology
Fuzhou City, Fujian 350118, China
yfsuo80102@hotmail.com

Chia-Hung Wang[1,2]

[1]College of Computer Science and Mathematics,
Fujian University of Technology,
No. 69, Xuefu South Road, University Town,
Fuzhou City, Fujian 350118, China
[2]Fujian Provincial Key Laboratory of Big Data Mining and Applications,
Fujian University of Technology,
No. 69, Xuefu South Road, University Town,
Fuzhou City, Fujian 350118, China
jhwang728@hotmail.com

Trong-The Nguyen[1,2,*]

[1]Multimedia Communications Lab.,
VNU-HCM, University of Information Technology, Vietnam
[2]Vietnam National University, Ho Chi Minh City 700000, Vietnam
thent@uit.edu.vn

Kun Hu

College of Computer Science and Mathematics
Fujian University of Technology
Fuzhou City, Fujian 350118, China
KunHu2532@hotmail.com

Jiongbiao Cai

College of Computer Science and Mathematics
Fujian University of Technology
Fuzhou City, Fujian 350118, China
jbcai99082@hotmail.com

Qing Ye

College of Computer Science and Mathematics
Fujian University of Technology
Fuzhou City, Fujian 350118, China
qye004554@hotmail.com

[*]Corresponding author: Trong-The Nguyen
Received January 6, 2024, revised August 1, 2024, accepted May 21, 2025.

Abstract. *In light of the explosive growth of e-commerce logistics, optimizing vehicle delivery routes has become a pressing concern. This paper delves into the complexities of vehicle routing problems with time windows (VRPTW), where vehicles must efficiently transport goods from depots to customers within specified time windows, while also considering capacity limitations. To address these challenges, we introduce a hybrid intelligent optimization algorithm that combines the enhanced sparrow search algorithm (SSA) with adaptive large-scale neighborhood search (ALNS). This hybrid SSA-ALNS method aims to overcome issues related to local optima and stability encountered when using the ALNS algorithm. We achieve this by integrating three operation mechanisms of SSA into ALNS and implementing a warning mechanism to optimize the potential population, thus enhancing the search process. Furthermore, we expand the types of operators in the improved ALNS algorithm to perform more targeted operations for different identities in the search space. Our experimental analysis, conducted on 56 instances in the Solomon Benchmark, demonstrates that the proposed hybrid SSA-ALNS method outperforms traditional ALNS algorithms, showcasing an average optimization improvement of 30.04%. Additionally, our method outperforms BKS in several test instances, highlighting its efficacy in finding superior solutions.*

Keywords: Meta-heuristic Algorithm, Adaptive Large-scale Neighborhood Search, Vehicle Routing Problems, Swarm Intelligence Algorithms, Heuristic Random Search

1. **Introduction.** The widespread adoption of Internet technology and the rapid growth of e-commerce have made online shopping an integral part of people's daily lives [1]. The convenience of online shopping has led to a significant shift away from traditional brick-and-mortar stores, resulting in a surge in the logistics industry [2]. However, this growth has also brought several challenges to the forefront [3], including rising logistics costs [4], delivery efficiency, and service quality [5]. Notably, last-mile delivery has emerged as a particularly costly and inefficient aspect of logistics distribution [6]. As the final point of contact with customers, any errors during this phase can significantly impact their experience with express delivery services. Consequently, the industry has increasingly focused on addressing these challenges in recent years [7,8]. The issue of logistics terminal distribution aligns with the Vehicle Routing Problem (VRP) and warrants further study to develop effective solutions.

The VRP was originally proposed by Dantzig et al. in 1959 as a method for solving truck scheduling issues [9]. As a classic combinatorial operations optimization problem, it has consistently garnered significant attention from researchers both domestically and internationally. Over the years, numerous scholars have introduced various VRP variants and corresponding solutions [10]. The classic vehicle routing problem involves a depot node serving as a distribution center with goods to fulfill customer needs, a set of customer nodes with diverse requirements, and a fleet comprising a specific number of vehicles with identical cargo capacities. The objective is to plan the optimal driving routes to satisfy the needs of the customer nodes and other constraints, ultimately minimizing the total cost. The fundamental depiction of the vehicle routing problem is illustrated in Figure 1(a). Given the spatial relationship between the depot and the customer in Figure 1(a), the route depicted in Figure 1(b) is determined.

With the continuous deepening of related research in this field, many variant problems have emerged, including Vehicle Routing Problems with Time Windows (VRPTW) [11], Capacitated Vehicle Routing Problems (CVRP) [12], Multi-Depot Vehicle Routing Problems with Time Windows (MDVRPTW) [13], and others. These problems could find applications in specific fields such as logistics distribution, medical services [14], job shop scheduling [15], potential exploration, and inspection [16, 17]. In this study, our focus is

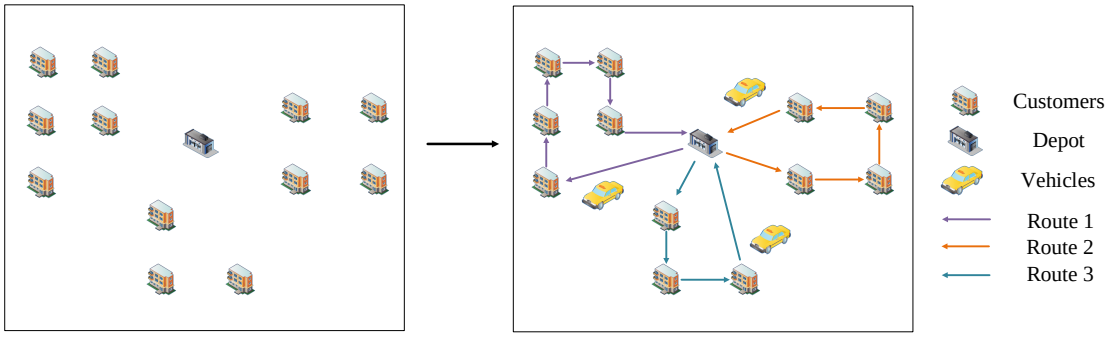


FIGURE 1. An illustrative example for the studied vehicle routing problem

primarily on VRPTW. The key distinction between VRPTW and VRP lies in the addition of a time window constraint in VRPTW. This constraint specifies that each customer has a designated time window for receiving goods, and vehicles must deliver goods to the customer within this time frame. Additionally, we take into account vehicle capacity constraints, meaning that each vehicle has a limited loading capacity. Consequently, the total cargo demand of each customer in a planned vehicle route cannot exceed the loading capacity limit of the corresponding vehicle [18].

Certainly, the time sensitivity of customer needs, particularly for items like fresh or medical goods, underscores the critical importance of studying VRPTW. Failure to meet the designated time limits can result in significant losses for delivery companies [19, 20]. Given that VRPTW is a proven NP-hard problem with substantial complexity, many researchers are exploring the use of meta-heuristic algorithms to address such NP-hard problems. These algorithms can yield relatively acceptable solutions within a short timeframe and have been widely applied in diverse fields to optimize decision objectives [21–32].

One such meta-heuristic method is the Adaptive Large Neighborhood Search (ALNS), introduced by Ropke and Pisinger in 2006 [33]. Building upon neighborhood search, ALNS incorporates a mechanism for evaluating the effectiveness of operators, enabling the algorithm to automatically select the best operator from multiple options to transform the solution, thereby increasing the probability of obtaining an improved solution. However, designing different transformation operators for various optimization objectives is crucial, as an inadequate design can hinder effective target optimization.

Another notable algorithm is the Sparrow Search Algorithm (SSA), a swarm intelligence optimization method proposed by Xue and Shen in 2020 [34,35]. Based on the foraging and predator evasion behavior of sparrows, SSA boasts robust optimization capabilities and high efficiency. When tackling global optimization problems, SSA demonstrates strong search ability and parallelism, presenting clear advantages in convergence accuracy and stability compared to other representative heuristic algorithms. Consequently, SSA has found application across a range of domains.

In this paper, we introduce the SSA-ALNS algorithm for addressing VRPTW, aiming to overcome the limitations of ALNS, such as susceptibility to local optima and low algorithm stability. Our contributions are twofold: Firstly, we incorporate the concept of SSA, integrating three operation mechanisms of SSA into ALNS and categorizing the population into two identities: producers and scroungers. We also implement an early warning mechanism to optimize the population for further enhancement. Secondly, we expand the range of operators in ALNS, including 2-optimization (2-opt), Or-optimization (Or-opt), improved 2-optimization (2-opt*), swap/shift, and others [36]. These operators are divided into two groups, allowing for more targeted operations for different population identities and broadening the search space of ALNS. Furthermore, we conduct experimental analysis on 56 instances from the Solomon Benchmark, each containing 100 customers, and compare the results with other algorithms to demonstrate the feasibility and effectiveness of our proposed algorithm.

The paper is structured as follows: Section 2 provides an overview of relevant works on VRP problem-solving and summarizes key characteristics of existing literature. In Section 3, we define the VRPTW and derive the mathematical model for our proposed solution. Section 4 presents the hybrid SSA-ALNS algorithm, detailing the combination of SSA and ALNS, operator design, and the solution process. Section 5 covers parameter tuning and a comparison of SSA-ALNS with other solving algorithms. Finally, in Section 6, we summarize the conclusions drawn from this study.

Table 1. VRP-related problems and solving algorithms

Reference	Problem	Number of depots	Time window	Capacity constraints	Algorithm
Alvarez et.al. [37]	VRPTWMD	1	√	√	BPC- ILS- LNS
Gong et.al. [11]	VRPTW	1	√	√	S-PSO-VRPTW
Altabeeb et.al. [12]	CVRP	1	×	√	IFA
Yesodha et.al. [13]	MDVRPTW	n	√	√	IFA
Chen et.al. [38]	VRPTWDR	1	√	√	ALNS
Lin et.al. [39]	TTRPTW	1	√	√	SA
Ropke et.al. [33]	PDPTW	1	√	√	ALNS
Luo et.al. [40]	VRPTW	1	√	√	HSFLA
Esam et.al. [41]	VRPTW	1	√	√	meta-HSA
zhang et.al. [42]	VRPRC	1	√	√	TSABC
Arit et.al. [43]	CVRP	1	×	√	MAS-SA-PR
Juan et.al. [44]	FCVRP	1	×	√	ISCM
This work	VRPTW	1	√	√	SSA-ALNS

2. **Related work.** In recent years, researchers have made many efforts to solve VRP, proposing various types of solution methods from different research perspectives [45–48]. VRP related issues are mainly solved using heuristic methods [49]. In Table 1, we summarize several characteristics of the relevant works in the literature. Based on set heuristic rules, these algorithms can search within an appropriate range and find the optimal solution within a certain amount of time [50–52]. Alvarez et al. [37] provide a hybrid approach to solve the vehicle routing problem with time windows and multiple deliveries (VRPTWMD). VRPTWMD is a variant of VRPTW. This method combines the branch and price reduction (BPC) algorithm, large neighborhood search (LNS), and iterative

local search (ILS). In this issue, the customer's service time depends on the number of delivery personnel assigned to the route they serve. Gong et al. [11] provide a set-based particle swarm optimization algorithm (S-PSO-VRPTW) for solve VRPTW. This algorithm is based on PSO and defines the arithmetic operators on the set, transforming the solution through set transformation. Consider the arc set of the complete graph of all nodes in the problem as the search space. Altabeeb et al. [12] introduce an improved Firefly algorithm (IFA) to solve CVRP. Two local search processes were introduced. In addition, the crossover operator and two mutation operators of the Genetic Algorithm (GA) are combined to balance local and global searches. Yesodha et al. [13] provide an improved Firefly algorithm (IFA) to solve MDVRPTW. This algorithm combines the IFA with the inter-depot method in the allocation phase to optimize the search space of the algorithm. At the same time, it expands the use of local search methods in the scheduling phase to optimize local optimal solutions. Chen et al. [38] proposed the Vehicle Routing Problem with Time Windows and Delivery Robots (VRPTWDR). They used an Adaptive Large Neighborhood Search heuristic (ALNS) algorithm to solve the above problem. This problem is related to routing a delivery vehicle equipped with multiple autonomous package delivery robots. Seven removal operators and five insertion operators were introduced into the algorithm. The experimental results have demonstrated that this algorithm has significant advantages over the CPLEX solver in solving VRPTWDR problems. Lin et al. [39] introduce an improved simulated annealing heuristic (SA) algorithm. This algorithm solves a vrp variant problem called the Truck and Trailer Routing Problem with Time Windows (TTRPTW). Six neighborhood structures were introduced in the algorithm to generate new solutions. Ropke et al. [33] provide an Adaptive Large Neighborhood Search Heuristic (ALNS) algorithm. A novel VRP variant problem has been solved, which is the Pickup and Delivery Problem with Time Windows (PDPTW). The algorithm consists of multiple competing subdomains whose usage frequency corresponds to their historical performance. To handle VRPTW, Luo et al. [40] suggest a Hybrid Shuffled Frog Leaping Algorithm (HSFLA). For the quality of the solution, an optimized clonal selection procedure was used. At the same time, in order to expand the search scope, they used improved and extended extreme optimization (EO) with alternative shift operators. Esam et al. [41] introduce a meta-Harmony Search Algorithm (meta-HSA) to solve VRPTW. The HSA optimizer and HSA solver are two important components of this algorithm. The HSA optimizer modifies the solver's configuration through adaptive means based on the current search situation. The HSA solver is a hybrid of HSA and local search algorithm (LS), which takes the configuration processed by the HSA optimizer as input to solve a given problem. Zhang et al. [42] proposed a hybrid Tabu search and the Artificial Bee Colony algorithm (TSABC). This algorithm can effectively solve the routing problem with realistic constraints (VRPRC). This problem includes time window constraints and additionally considers the spatial volume capacity of the goods. Arit et al. [43] proposed a hybrid strategy consisting of three strategies - Modified Ant System (MAS), Sweep Algorithm (SA), and Route Relinking (PR), also known as MAS-SA-PR, to solve CVRP. Generate initial solution using SA and assign customers to vehicles. Then, use MAS to generate a new generation of reasonable solutions. PR is used to build better solutions from multiple solutions. Juan et al. [44] proposed an Iterative Soft Constraints Method (ISCM). This method is used to solve a special VRP problem, which is the Fuzzy Capacity Vehicle Routing Problem (FCVRP) with fuzzy capacity requirements. Due to the lack of available or reliable data, there are some uncertain capacity vehicle routing issues. Therefore, they represent these uncertain costs or requirements as fuzzy numbers. These fuzzy numbers are combined with iterative integer programming methods to optimize the solution.

In summary, previous studies have investigated the issue of various variants of VRP, indicating the importance of VRP and VRPTW. We will focus on solving VRPTW. In this study, we introduce the SSA-ALNS algorithm for solving VRPTW. Improvements are made to address the drawbacks of ALNS being prone to locally optimal solutions and low algorithm stability.

TABLE 2. The definitions of mathematical notations in the studied VRPTW model

Notation	Definition
F	Objective function
α	The weighting factor cost, which the weight ratio of the cost of using the number of vehicles and the distance traveled
V	Node set
C	Customer set
D	Depot set
K	Vehicle set
Q	Maximum capacity of the vehicle
m	Number of customer nodes
u	Number of vehicles
$TS_{k,i}$	The starting service time of the k-th vehicle on node i
h_i	Service time of the vehicle to node i
$c_{i,j}$	The distance between node i and node j
r	Cost unit distance
v	Vehicle speed
e_i	The left time window of node i
l_i	The right time window of node i
p	Vehicle call cost
q_i	Customer i's needs
$x_{k,i,j}$	Decision variable, indicating whether the k-th vehicle passes through a directed route from customer i to customer j, with a value of {1,0}
y_k	Decision variable, indicating whether the k-th vehicle is used, with a value of {1,0}

3. **Mathematical Model.** The VRPTW can be described as having a certain number of vehicles in the depot, which departs from the depot to deliver goods to customers and return them [37]. The cost required for the entire process should be as low as possible and not violate corresponding constraints. The main constraints we consider include time window constraints, vehicle capacity constraints, etc. Time window constraint refers to the depot and customers having a corresponding service time interval [53]. Vehicles must provide services to the depot and customers within the corresponding time interval, which is a challenging time window. Vehicle capacity constraint refers to each vehicle having a

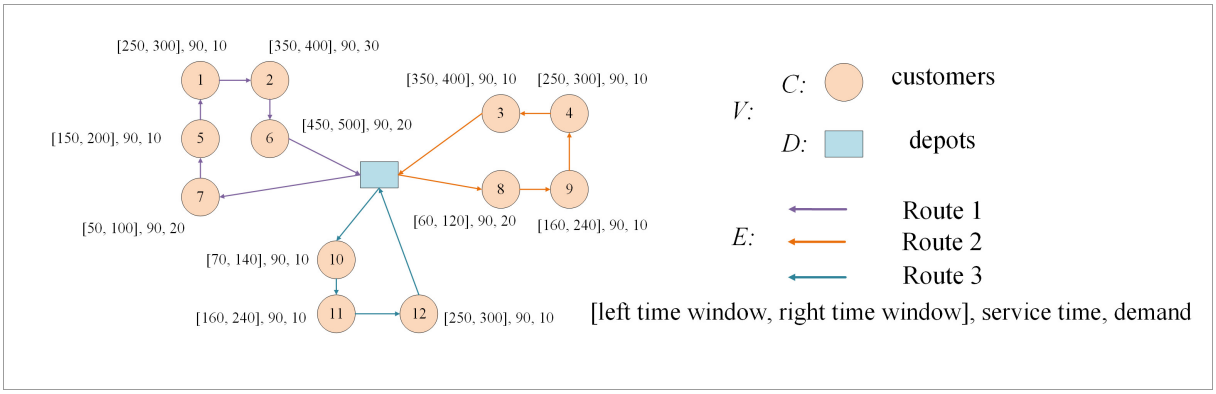


FIGURE 2. A mathematical example for the studied vehicle routing problem with time windows

maximum loading capacity, which cannot be exceeded during loading. In addition, the number of vehicles is limited [37]; The capacity of each vehicle is the exact and known; Each customer can only be served by one vehicle. Constraints such as a single depot are also considered.

We define VRPTW as a directed connected graph $G = (V, E)$, where $V = \{0, 1, ..., m\}$ represents the node-set, $V = D \bigcup C$, $D = \{0\}$ represents the depot node, and the number of depots is 1; $C = \{1, 2, ..., m\}$ represents the set of customer nodes, with the number of customers being m; $E = \{\langle i, j\rangle | i, j \in V, i \neq j\}$ represents a set of arcs, with each arc corresponding to a nonnegative cost c_ij, represents the distance between node i and node j; $K = \{1, 2, ..., u\}$ represents the set of vehicles, and u represents the number of vehicles. For each vehicle $k \in K$ is of the same type, each vehicle has the same capacity Q, usage cost p, driving speed v, and driving distance cost r. For each customer $i \in C$, there is a nonnegative demand q_i and service time h_i. The customer must be served within the time window $[e_i, l_i]$. $TS_{k,i}$ represents the starting service time of the $k-th$ vehicle for the $i-th$ customer. To display the diagram's structure more clearly, we have drawn Figure 2 based on the content of Figure 1, which includes 12 customer nodes. The time in the figure is represented in minutes. Assuming the maximum number of vehicles used is $5(u = 5)$ and the current number of vehicles is 3, it meets the limit. The symbol representation table is shown in Table 2.

Based on the above description of the VRPTW problem, the definition is as follows:

$$minF = \alpha \sum_{k \in K} \sum_{i \in V} \sum_{j \in V} c_{i,j} x_{k,i,j} r + (1 - \alpha) \sum_{k \in K} y_k p, \tag{1}$$

Subject to:

$$\sum_{i \in C} x_{k,g,i} = \sum_{i \in C} x_{k,i,j} = 1, \quad \forall g \in V, \forall j \in V, \forall k \in K, \tag{2}$$

$$x_{k,i,j} \left(TS_{k,i} + h_i + \frac{c_{i,j}}{v} - TS_{k,j} \right) \leq 0, \quad \forall k \in K, \forall i, j \in V, \tag{3}$$

$$e_i \leq TS_{k,i} \leq l_i, \forall i \in C, \quad \forall k \in K, \tag{4}$$

$$\sum_{i \in V} \sum_{j \in V} q_i x_{k,i,j} \leq Q, \quad \forall k \in K, \tag{5}$$

$$x_{k,i,j}, y_k \in \{0, 1\}, \quad \forall i, j \in V, \forall k \in K, \tag{6}$$

$$TS_{k,i} \geq 0, \quad i \in V, k \in K, \tag{7}$$

$$\sum_{i\in V}\sum_{j\in V} x_{x,i,j} \leq |LL_k| + 1, \quad k \in K, \tag{8}$$

where Equation (1) is the objective function of the optimization problem represents the total cost of minimizing the delivery plan, including the cost of distance and the cost of using vehicles. The objective function is the weighted sum of two cost components. The weighting factor α represents the importance of two costs. The larger the value of α, the more important the cost of distance. Equation (2) ensures that each customer can only be served once. The routes to reach and leave the current customer are selected, and the index i represents the current customer; g and j represent the previous and next nodes of the vehicle service, respectively. Equation (3) represents the order of accessing nodes. Assuming that vehicle k directly travels from node i to node j, the start service time$TS_{k,i}$ of arriving at node j must be equal to $TS_{k,i} + h_i + c_{i,j}/v$. Equation (4) ensures that the $i-th$ customer is serviced within the time window $[e_i, l_i]$. Equation (5) ensures that the load does not exceed the capacity Q of vehicle k. Equation (6) defines the decision variables x and y as binary. In Equation (7), it gives that the start service time $TS_{k,i}$ cannot be negative. Equation (8) are standard subtour elimination constraints, the $|LL_k|$ represents the number of nodes that the current route has passed through, excluding depots.

$$x_1 = \begin{bmatrix}
0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0
\end{bmatrix}
\quad
x_2 = \begin{bmatrix}
0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0
\end{bmatrix}$$

$$x_3 = \begin{bmatrix}
0&0&0&0&0&0&0&0&0&0&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1\\
1&0&0&0&0&0&0&0&0&0&0&0&0
\end{bmatrix}
\quad
y = \begin{bmatrix}1&1&1&0&0\end{bmatrix}$$

where $\boldsymbol{x_1}$ represents the solution matrix of $\boldsymbol{x}$ when $k = 1$, corresponding to the first route: 0-7-5-1-2-6-0. In this instance, the matrix's dimension is $\mathbf{13 \times 13}$, including 1 depot node and 12 customer nodes. The first row (column) represents the depot node, while rows (columns) 2 to 13 represent the customer node. The formula $x_{1,1,8} = 1$ indicates that the value of the first row and the eighth column of the matrix $\boldsymbol{x_1}$ is 1. Represents: 0-7 in the route. Which means choosing the route from the depot to customer 7. The matrix $\boldsymbol{y}$ represents 5 vehicles, with only 3 selected. The matrices $\boldsymbol{x_4}$ and $\boldsymbol{x_5}$ are both zero matrices.

4. **A hybrid solution algorithm based on SSA and ALNS.**

4.1. **Key Concepts of SSA-ALNS.**

4.1.1. *Basic Sparrow Search Algorithm.* SSA mainly simulates the foraging process of sparrows. The foraging process of sparrows is a model of collaborative foraging between producers and followers, and it also incorporates a warning mechanism. Find individuals with better food quality in the population as producers and other individuals as scroungers. Select a certain proportion of individuals in the population for investigation and warning, and if danger is found, give up food [34].

Algorithm 1 provides the main steps of SSA. When executing SSA, three critical factors should be considered: the producer's search rules, the follower's search rules, and the warning mechanism's setting. In the basic version of SSA, these operations are as follows.

(a) Producer

Firstly, assume that the number of individuals in the sparrow population is n. Before each iteration, the first PD individuals sorted from the population will be labeled as producers, where PD represents the number of producers. They search near their location. The location update formula is described as follows:

$$xs_{i,j}^{t+1} = \begin{cases} xs_{i,j}^{t} \cdot exp\left(\frac{-i}{\gamma \cdot iter_{max}}\right), & R_1 < ST \\ xs_{i,j}^{t} + Z \cdot L, & R_1 \geq ST \end{cases}, \tag{9}$$

where $xs_{i,j}^{t}$ represents the j dimensional position of the i individual in the t generation of the population, where $j = 1, 2, ..., D$ and D represent dimensions. Besides, γ represents the uniform random number in $(0, 1]$, and Z is a standard normal distribution random number. The R_1 is the uniform random number in $[0, 1]$, ST is the alert threshold, and the value range is $[0.5, 0.1]$. The $iter_{max}$ represents the maximum number of iterations, and L represents a unit row vector of the D dimension.

(b) Scrounger

Secondly, label the last (n – PD) individuals sorted from the population as followers. Some move towards the optimal global position, while the other part randomly searches within the search space. The search rules are described as follows:

$$xs_{i,j}^{t+1} = \begin{cases} Z \cdot exp\left(\frac{xs_{worst}^{t} - xs_{i,j}^{t}}{i^2}\right), & \text{if } i > \frac{n}{2} \\ xs_{p}^{t+1} + |xs_{i,j}^{t} - xs_{p}^{t+1}| \cdot A^{+} \cdot L, & \text{otherwise} \end{cases}, \tag{10}$$

where xs_{worst}^{t} is the worst position of sparrows in the current population, and xs_{p}^{t+1} is the optimal location for sparrows among producers. The Z is a standard normal distribution random number, and L is the unit row vector of a D dimension. In addition, the matrix $\boldsymbol{A}$ represents $1 \times D$ dimensional matrix, where each element is randomly assigned as 1 or -1, and we also have $\boldsymbol{A}^{+} = \boldsymbol{A}^{+}(\boldsymbol{A}\boldsymbol{A}^{T})^{-1}$.

(c) Warning mechanism

Finally, some sparrows will be responsible for implementing warning mechanisms, and when danger arises, they will abandon their current food. Whether the sparrow is a producer or follower, it will abandon its current food and move to a new location. After each iteration, SD individuals will be randomly selected from the population for early warning behavior, where SD represents the number of early warning individuals. The location update formula is described as follows:

$$xs_{i,j}^{t+1} = \begin{cases} xs_{b}^{t} + \beta \cdot |xs_{i,d}^{t} - xs_{b}^{t}|, & f_i \neq f_g \\ xs_{i,j}^{t} + R \cdot \left(\frac{|xs_{i,j}^{t} - xs_{w}^{t}|}{|f_i - f_w| + \varepsilon}\right), & f_i = f_g \end{cases}, \tag{11}$$

where β is the random number conforming to the standard normal distribution, and R is the uniform random number of $[-1, 1]$. The ε represents a smaller number, preventing the denominator from being 0. The xs_w^t is the global worst position of the sparrow, and xs_b^t is the global optimal position of the sparrow. Moreover, f_i represents the current fitness value of the sparrow. The f_g and f_w are defined as the current global best and worst fitness values, respectively.

Algorithm 1 Pseudo-code of SSA

Input: G, PD, SD, R_1, n
Output: xs_{best}, f_g

```
Initialize a population of n sparrows
t = 1
while t < G do
    Rank the fitness values and find the current best individual and the current worst individual
    for i = 1 to n do
        if i ≤ PD then
            Using Equation (9) update the sparrow's location
        else
            Using Equation (10) update the sparrow's location
        end if
    end for
    for i = 1 to SD do
        Using Equation (11) update the sparrow's location
    end for
    Rank the sparrow, find the current best among all sparrow's location xs_best and fitness f_g
    t = t + 1
end while
return xs_best, f_g
```

4.1.2. *fitness function.* Fitness is the criterion to judge whether a solution is better. Due to the Equation (1) cannot directly participate in algorithm operations, the fitness is constructed by expanding on the basis of Equation (1). The fitness in this paper is expressed by Equation (12). The lower the fitness, the better the quality of the solution. VRPTW has two goals. The main goal is to minimize the number of vehicles (NV). The secondary goal is to minimize the total distance (TD) under the same number of routes. We adjust the optimization direction of the algorithm by assigning different weight parameters to these two target costs. The algorithm in this paper allows for generating solutions that violate constraints. A penalty coefficient will be added to the violated constraints to prevent the algorithm from jumping out of infeasible solutions. The specific settings of these parameters will be assigned during the experimental phase. The formula of fitness value is given as follows, and the mathematical symbols used in the formulation are explained in Table 3.

$$fit(s) = TD(s) \cdot r \cdot \alpha + VN(s) \cdot p \cdot (1 - \alpha) + q(s) \cdot pun_1 + w(s) \cdot pun_2. \quad (12)$$

TABLE 3. The definitions of mathematical notations in the studied fitness function

Symbol	Definition
s	Current solution
α	Weight factor for distance cost, range (0,1)
$TD(s)$	The total distance the current solution
$VN(s)$	The number of vehicles for the current solution
p	Cost of vehicle
r	Cost of driving distance
$q(s)$	The total number of routes that violate vehicle capacity constraints for the current solution
$w(s)$	The total number of customers who violate the time window constraint for the current solution
pun_1	Penalty coefficient for violating vehicle capacity constraints
pun_2	Penalty coefficient for violating time window constraints

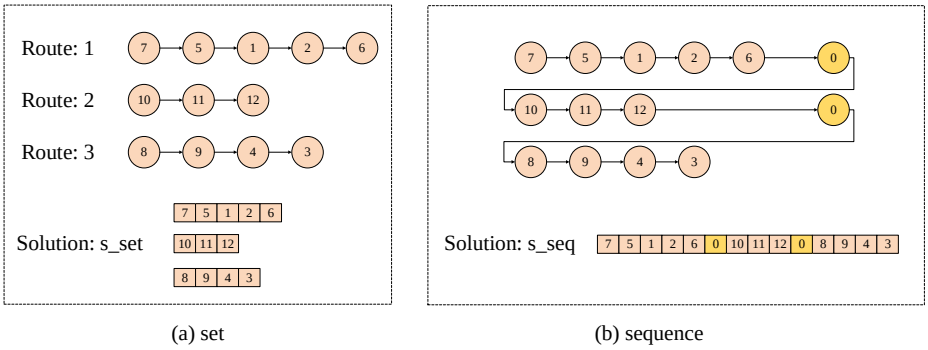


FIGURE 3. Representation of solutions for the proposed algorithm

4.1.3. *Representation of Solutions.* In this paper, we use two methods to represent the solution of the algorithm, which can be used in different operator transformation methods and mutually transformed. We use the instance in Figure 2 for an explanation. The solution in Figure 2 consists of three routes: 7-5-1-2-6, 10-11-12, and 8-9-4-3. The first method: In Figure 3 (a), the solution is represented as a set: s_set, which includes three sequences representing vehicle routes. The second method: In Figure 3(b), the solution is represented as a sequence: s_seq, which connects all subsets in Figure 3(a) and separates them with 0.

4.1.4. *Generation of initial solutions.* The initial solutions of all individuals are generated randomly. The advantage of the random approach is that it enables the algorithm to have higher universality in different situations. For each individual, generate a random solution sequence s with a length of m; s represents the solution sequence of this individual; m represents the total number of customers. Then randomly insert $u-1$ zeros into the sequence s, where u represents the maximum number of vehicles used. Finally, a random solution sequence s with a length of Ls is generated, $Ls = m + u - 1$. As shown in Figure 4, we assume that the maximum number of vehicles used in this instance is 5. Figure 4 shows the generation of initial solutions. We assume that the maximum number of vehicles used in this instance is 5.

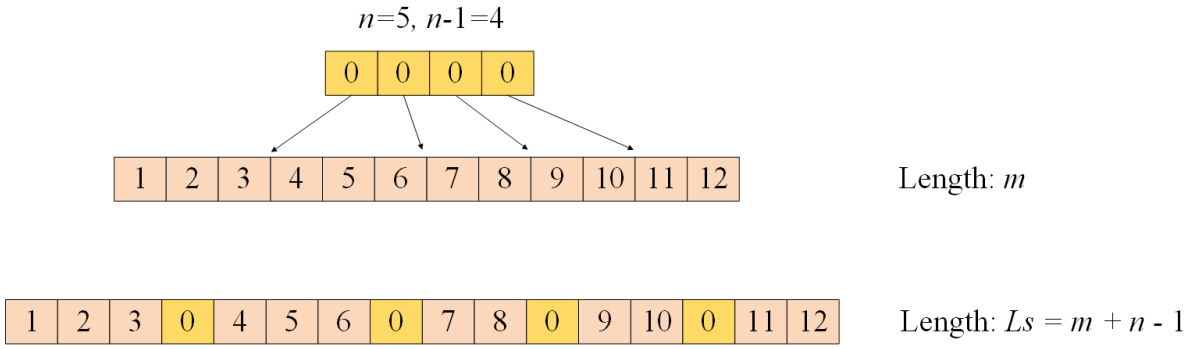


FIGURE 4. Generation of initial solutions

4.1.5. *Simulated annealing algorithm.* The simulated annealing algorithm (SA) is a random optimization algorithm based on Monte Carlo iterative solution strategy [38]. In the simulated annealing algorithm, the generating function of the new solution is not strictly set. The main idea is to use the Metropolis criterion to determine whether a new solution is accepted.

Metropolis criterion: When a new solution $s^{'}$ is generated, compare the fitness value of the current solution s and judge whether to accept the new solution $s^{'}$ through probability pro. The formula is derived as follows:

$$T = T0 \cdot fit(best_0), \tag{13}$$

$$T = T \cdot \eta, \tag{14}$$

$$prob(s \to s^{'}) = \begin{cases} 1, & fit(s^{'}) < fit(s) \\ exp\left(-\frac{fit(s^{'}) - fit(s)}{T}\right), & fit(s^{'}) \geq fit(s) \end{cases}. \tag{15}$$

When $fit(s^{'}) < fit(s)$, the new solution is better than the current solution, then the probability of acceptance is 1, that is, it must be accepted. If $fit(s^{'}) \geq fit(s)$, and the new solution is worse than the current solution, then the accepted probability is related to the formula. The $T0$ represents the initial temperature factor; $fit(best_0)$ represents the optimal fitness value in the initial population with Equation 13. In the iteration process, temperature T is updated with Equation 14; η represents the annealing rate, and the range is (0,1). Note that the temperature T decreases gradually during the iterations. The probability *prob* is also reducing. The probability of accepting the worse new solution is higher in the early iteration, and the acceptance rate of the worse new solution gradually decreases with the continuous iteration of the algorithm.

4.1.6. *Adaptive large neighborhood search algorithm.* Adaptive Large Neighborhood Search (ALNS) is a heuristic method proposed by Ropke and Pisinger [26]. Based on neighborhood search [54], it adds a measure of the effect of the operator so that the algorithm can automatically select a good operator from many operators to transform the solution to get a better solution. The weight-updating formula of the operator is as follows:

$$\omega_d = \begin{cases} \omega_d, & u_d = 0 \\ (1-\rho)\omega_d + \rho\frac{s_d}{u_d}, & u_d > 0 \end{cases}, \tag{16}$$

where d represents the neighborhood operator used, and ω_d represents operator weight. The s_d represents operator fraction, and u_d represents the number of times the operator

is used, must be greater than or equal to 0. Besides, ρ is the weight updating coefficient used to control the speed of weight change. We calculate the current operator's weight based on each round's score. The weight of an operator is related to the degree to which the operator produces a solution. The operator selection method is applied according to the operator's weight, and by using the roulette to be selected. In the following Table 4 , it shows the score setting.

TABLE 4. Solution reception and score setting

<table>
<tr><td colspan="3">The reception of the solution</td><td>score</td></tr>
<tr><td colspan="3">The new global optimal solution
is obtained after the operation</td><td>sc_{d1}</td></tr>
<tr><td rowspan="3">The optimal global solution
is not obtained
after the operation</td><td colspan="2">The new solution is better than
the current one. Accept it.</td><td>sc_{d2}</td></tr>
<tr><td rowspan="2">The new solution is worse than
the current one</td><td>Meet the criteria, accept</td><td>sc_{d3}</td></tr>
<tr><td>Do not meet the criteria, do not accept</td><td>sc_{d4}</td></tr>
</table>

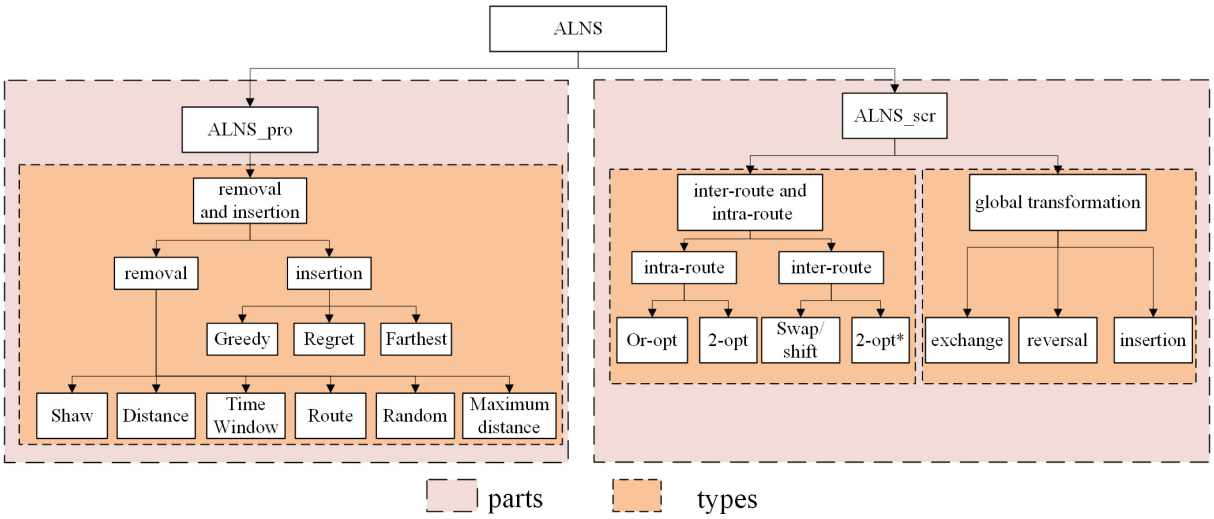


FIGURE 5. Grouping diagram of operators used in the improved neighborhood design of ALNS

4.2. **Improved neighborhood operator design in ALNS.** Traditional ALNS only includes removal operators and insertion operators. Sometimes, new solutions generated solely by these operators do not have good advantages. So, it is necessary to expand the operator to increase individual diversity. Therefore, in this paper, we have expanded ALNS by adding various transformation operators to assist traditional ALNS in transformation and combined them with SSA. The neighborhood operators in our proposed ALNS include three types of operators: The first type is the removal operators and insertion operators, the Type 2 is the global transformation operators, and the Type 3 is the inter-route and intra-route neighborhood operators.

We will divide three neighborhood generation algorithms into two parts.Each part performs different transformations on the producer and follower [55]. In Figure 5, it shows the grouping diagram of these operators. ALNS_pro is mainly aimed at producers, including removal and insertion operators. ALNS_scr primarily targets followers, including global transformation operators and inter-route and intra-route neighborhood operators.

4.2.1. *ALNS_pro.* ALNS_pro is an algorithm that performs neighborhood transformations on producers, also known as traditional ALNS. We used six removal and three insertion operators. Removal operators include Shaw removal, time window removal, distance removal, route removal, random removal and maximum distance removal. Insertion operators include greedy, regret, and farthest insertion operators.

Removal Operators.Remove several customer nodes meeting the corresponding conditions from solution sequence s. The number of customer nodes removed each time is represented by n_r, and the expression is as follows:

$$n_r = rand(\lfloor kr_{min} \cdot n + 1 \rfloor, \lfloor kr_{max} \cdot n + 1 \rfloor), \tag{17}$$

where $rand(A, B)$ indicates that an integer is randomly selected between A and B. The calculation formula $\lfloor kr_{min} \cdot n + 1 \rfloor$ means $kr_{min} \cdot n + 1$ is rounded down, where kr_{min} and kr_{max} are the coefficients between (0,1), $kr_{min} < kr_{max}$. The reasonable setting of kr_{min} and kr_{max} can make the algorithm have strong performance and control time consumption. The specific Settings are given in the experimental section. kr_{min} andkr_{max} can vary with the functions are as follows:

$$kr_{min} = kr_{min} \cdot kk, \tag{18}$$

$$kr_{max} = kr_{max} \cdot kk, \tag{19}$$

$$kk = \frac{2 - \left(\frac{t}{maxItera}\right)}{2}, \tag{20}$$

where kk represents the change parameter. The t represents the current number of iterations, and $maxItera$ indicates the maximum number of iterations.

(1) Shaw removal

This method was first proposed by Shaw [56]. The basic idea is to remove a set of nodes related to correlation. The correlation between customer i and customer j is represented by $R(i, j)$, which is expressed as follows:

$$R(i,j) = \varphi_1 \cdot d_{i,j} + \varphi_2 \cdot (|e_i - e_j| + |l_i - l_j|) + \varphi_3 \cdot |q_i - q_j| + sr_{i,j}, \tag{21}$$

where φ_1, φ_2 and φ_3 represent the weight coefficients of distance, time window and demand, respectively. The $d_(i, j)$ represents the Euclidean distance between two nodes, and the calculation formula $|e_i - e_j| + |l_i - l_j|$ represents the time interval between two nodes, where e_i indicates the left time window of the customer i, and l_i indicates the right time window for the customer i. Besides, the calculation formula $|q_i - q_j|$ represents the demand difference between the two nodes, where the q_i indicates the requirements of the customer i. Note that the smaller $R(i, j)$ is, the higher the correlation between two nodes is; otherwise, the higher the correlation is.

The correlation between removed nodes should be as low as possible. If customer i and customer j are on the same route, then $sr_{i,j} = 1$; otherwise, it gives 0. The time complexity is estimated as $O(m)$, where the m represents the number of customers.

(2) Distance removal

Distance removal is removal based on Shaw removal only considering the distance. In Equation (21), let $\varphi_2 = \varphi_3 = 0$. The time complexity is estimated as $O(m)$.

(3) Time Window removal

Time window removal only considers time window factors based on Shaw removal. In Equation (21), let $\varphi_1 = \varphi_3 = 0$. The time complexity is estimated as $O(m)$.

(4) Maximum distance removal

We calculate the sum of the distance travelled in the front segment and the distance travelled in the back segment for each customer in the current solution. Next, we remove the top n_r customers with the longest distance based on the wheel. The time complexity is estimated as $O(m)$.

$$mDis(i) = \begin{cases} d_{0,i} + d_{i,i+1}, & i-1 \text{ node is depot} \\ d_{i-1,i} + d_{i,0}, & i+1 \text{ node is depot} \\ d_{i-1,i} + d_{i,i+1}, & \text{else} \end{cases}, \tag{22}$$

where the $mDis(i)$ represents the sum of the driving distance of the front section and the driving distance of the back section of node i. The i represents the node corresponding to i in the current sequence, which may represent a customer or a depot. The $i-1$ and $i+1$ represent the previous and next nodes of the node corresponding to i in the current sequence. The $d_{i-1,i}$ represents the Euclidean distance between customer $i-1$ and customer i. The $d_{i,0}$ represents the Euclidean distance between the customer and the depot.

(5) Route removal

We remove an entire route to reduce the total number of vehicles, and preferentially select a route with no more than n_r customers. If the number of customers on all routes is greater than n_r, select a route at random among all routes. The time complexity is estimated as $O(m)$.

(6) Random removal

We randomly select n_r customers from the current solution to the removal. The time complexity is estimated as $O(1)$.

Insertion operators.After using the above removal operator, we put the customers removed into the collection L_r. The customer in L_r is inserted back into the current solution s_i using the insertion operator during the customer insertion phase.

(1) Greedy insertion

We select the location where the effect on fitness is minimal. A random node is selected to calculate the total fitness increment value of the route after the node is inserted into a certain position. Select the position corresponding to the lowest increment value. Repeat until L_r is an empty set. The time complexity is estimated as $O(m)$.

(2) Regret insertion

Based on the optimal greedy insertion operator, this operator takes one more step to ensure the plurality of operators. Evaluate the optimal and suboptimal locations where each removal node is inserted, and calculate the difference between the least total cost increment and the least total cost increment. All the removed customer nodes should be sorted from largest to smallest, and the next node to be inserted is selected to insert in an optimal greedy way. Choose the node with the large difference first. Repeat until L_r is an empty set. The time complexity is estimated as $O(m)$.

(3) Farthest insertion

When selecting the insertion position, the operator calculates the reasonable position with the maximum increment of the distance to insert each customer, and these positions must meet the constraint conditions. Select the customer with the smallest distance increment to insert into the corresponding position. The customer to be inserted and the corresponding location are recalculated each time. Repeat until L_r is an empty set. The time complexity is estimated as $O(m)$.

4.2.2. *ALNS_scr.* The procedure ALNS_scr is a neighborhood transformation algorithm for followers. We use the global transformation and inter-route/intra-route transformations operators, respectively. During the execution of the algorithm, the solution s is transformed by selecting one of the following seven transformation operators using roulette. The global transformation operator operates on the routine s_seq. The inter-route/intra-route transformation operator operates on the routine s_set. Global transformation operators include exchange, reversal and insertion. Inter-route conversion operators include 2-opt and Or-opt. Intra-route conversion operators include 2-opt* and swap/shift.

Global transformation operators.For the exchange structure, we randomly select two positions and exchange the nodes corresponding to the selected positions. The time complexity is estimated as $O(1)$. For the reversal structure, we randomly select two positions and arrange all nodes between the nodes corresponding to the selected positions in reverse order, including the selected position nodes. The time complexity is estimated as $O(1)$. We randomly select two positions for the insertion structure and insert the first node corresponding to the selected position after the second node. The time complexity is estimated as $O(1)$.

Inter-route and intra-route transformation operators.

(a) Intra-route transformation operators

(1) 2-optimization (2-opt)

2-opt is an intra-route transformation algorithm proposed by Lin [57]. We randomly select a route from the route set and randomly select two nodes $i, j, i < j$ from this route. Arrange all nodes between nodes i and j in reverse order, including i, j. Figure 6(a) shows a schematic diagram of the transformation. The time complexity is estimated as $O(1)$.

(2) Or-optimization (Or-opt)

Or-opt is an intra-route transformation algorithm proposed by Babin et al. [58]. We randomly select a route from the route set and assume the route length is nl. We randomly select two nodes $i, j, i < j$ from this route. This operator has two transformation methods, with a probability of 50% each. The time complexity is estimated as $O(m)$. The first method is shown in Figure 6(b). Generate a random number ki that is not greater than the length between two nodes i and j. Insert node i and the following ki nodes after node j. The second method is shown in Figure 6(c). Generate a random number kj not greater than $(nl - j)$. Insert node j and subsequent kj nodes after node i.

(b) Inter-route transformation operators

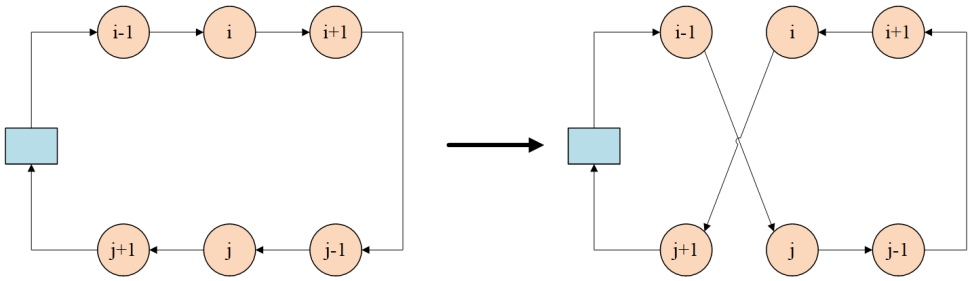
(1) Improved 2-Optimization (2-opt *)

It is an improved method for inter-route transformation proposed by Potvin and Rousseau [59]. We randomly select two routes in the current solution and randomly select one exchange position each. Swap all customer nodes after the position of the first route with all customer nodes after the position exchange of the second route, as shown in Figure 6(d). $L1$ and $L2$ represent the last customer of Route 1 and Route 2, respectively. The time complexity is estimated as $O(m)$.

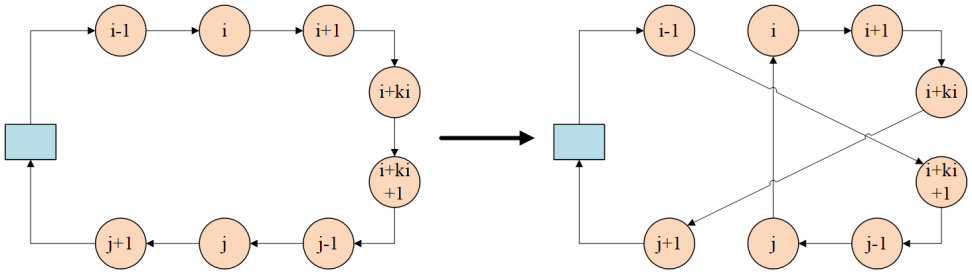
(2) swap/shift

It is an inter-route transformation algorithm proposed by Chen and Wu [60]. There are two ways to use this method. First, we randomly select two routes from the routes and exchange multiple nodes from the two routes, as shown in Figure 6(e). The a and b represent the number of exchange nodes from the two routes. Second, transfer some nodes from one route to another, as shown in Figure 6(f). The c represents the number of transfer nodes. The time complexity is estimated as $O(m)$.

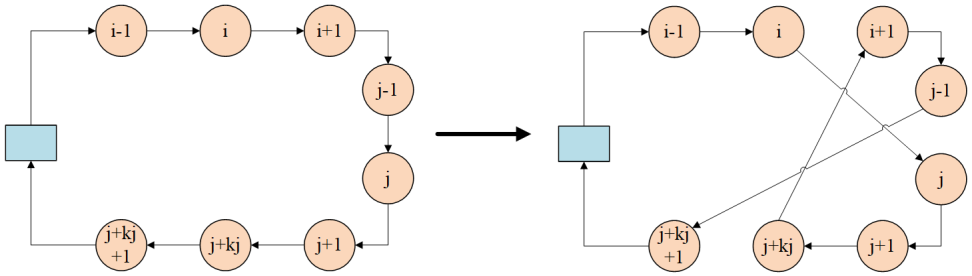
4.3. **A hybrid SSA-ALNS algorithm.**



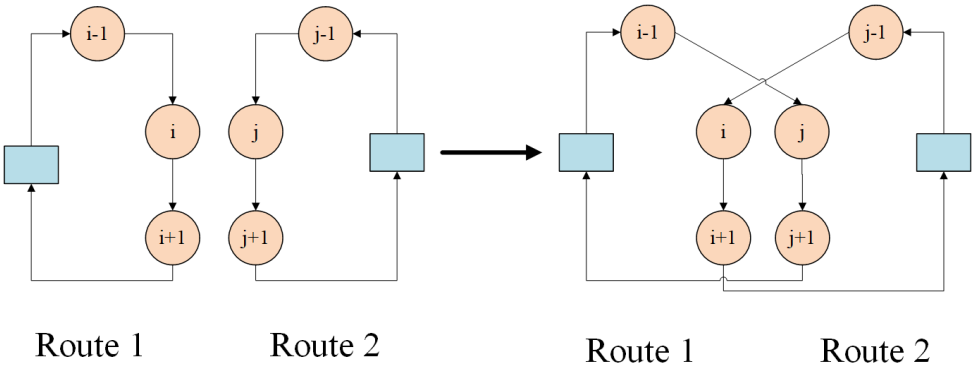
(a) 2-opt



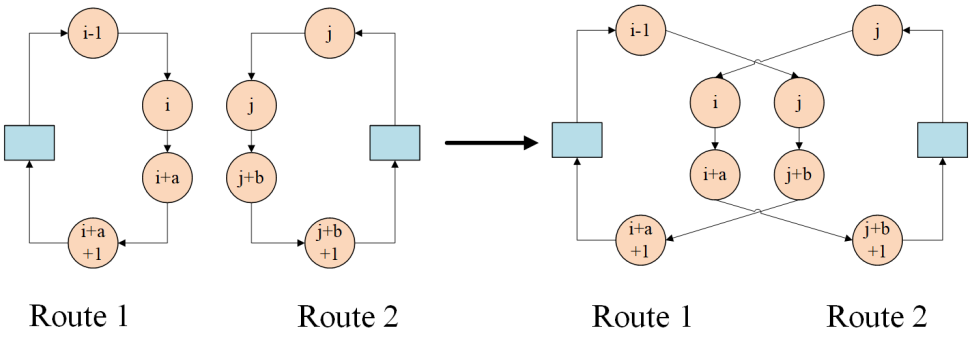
(b) Or-opt (1)



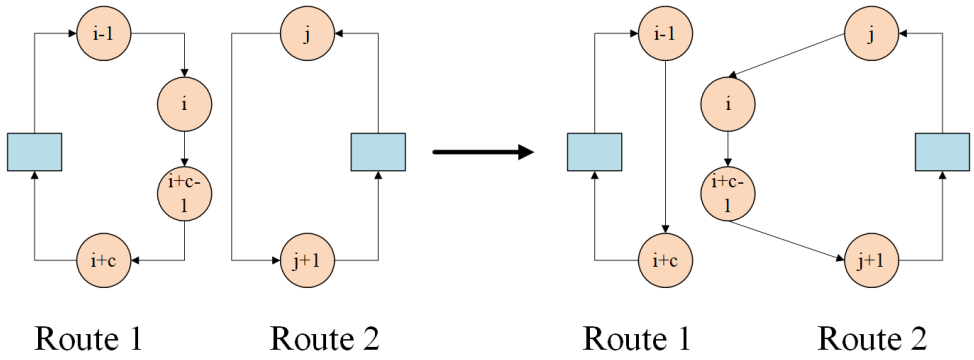
(c) Or-opt (2)



(d) 2-opt *



(e) swap/shift (1)



(f) swap/shift (2)

FIGURE 6. Inter-route and intra-route transformation operators

4.3.1. *Use of SSA.* We refer to the SSA algorithm and introduce three operational mechanisms. First, before each iteration, we will take the pro_{num} individuals with the best fitness in the current population as the producer. The procedure ALNS_pro is used for neighborhood transformation on such individuals during the iteration process. The calculation formula for pro_{num} is as follows:

$$pro_{num} = pro_{rate} \cdot pop_{num}, \tag{23}$$

$$pro_{rate} = pro_{prate} + (t/1.8)/maxItera, \tag{24}$$

where the pro_{num} represents the number of discovers; pro_{num} represents the number of populations, and pro_{rate} represents the proportion of producers. The pro_{prate} represents the initial proportion of producers. Besides, t represents the current number of iterations, and $maxItera$ represents the maximum number. The proportion of producers increases proportionally with iteration. Secondly, the remaining individuals act as scroungers, and we use ALNS_scr for neighborhood transformation during the iteration process for these individuals. Finally, after completing the transformation of all individuals, we perform an early-warning operation on the population. We select sd_{num} individuals with the worst fitness in each iteration randomly transform into others. The calculation formula of sd_{num} is derived as follows:

$$sd_{num} = sd_{rate} \cdot pop_{num}, \tag{25}$$

where sd_{num} represents the number of individuals alerted, and sd_{rate} represents the proportion of early warning recipients.

4.3.2. *Process of SSA-ALNS.* We integrate the above content to construct a hybrid algorithm, SSA-ALNS. In the Algorithm 2, we summarize the pseudocode for the proposed hybrid SSA-ALNS algorithm. In Figure 7, it illustrates the presented algorithmic framework, and the symbol interpretation is summarized in Table 5.

5. **Results and discussion.** To verify the effectiveness and feasibility of the algorithm in solving VRPTW problems, we will conduct comparative experiments between the proposed SSA-ALNS algorithm and other algorithms. The algorithms are evaluated using Solomon VRPTW instances, where the data set can be accessed at `http://web.cba.neu.edu/~msolomon/problems.htm`.

TABLE 5. The symbol interpretation for the proposed hybrid SSA-ALNS algorithm

Symbol	Definition
pop_{num}	Number of populations
$maxItera$	Maximum number of iterations
t	Current Iterations
i	Current individual
T	Simulated annealing temperature
$T0$	Initial temperature factor
fit_{best}	The global optimal fitness value
s_{best}	The optimal global solution
s_i	Current individual's solution
$s_i^{'}$	The solution of the current individual after neighborhood transformation
$fit(s_i)$	Current individual fitness
$fit(s_i^{'})$	Fitness of the current individual after neighborhood transformation
pro_{num}	Number of producers
scr_{num}	Number of scroungers
sd_{num}	Number of individuals for warning operations
pro_{rate}	The proportion of producers
pro_{prate}	The initial proportion of producers
scr_{rate}	Proportion of scroungers
sd_{rate}	The proportion of early-warning personnel

The experiment involves the dataset, including 56 Solomon instances with 100 customers, divided into six groups (R1, R2, C1, C2, RC1, and RC2). In R1 and R2, customers are generated based on a uniform distribution. In C1 and C2, customers are concentrated and distributed into several clusters. In RC1 and RC2, some customers are evenly distributed, while others form groups. R1, R2, RC1, and RC2 time windows are randomly generated. For C1 and C2, customers gather by route and then create windows. R1, C1, and RC1 are Type 1, and logistics delivery vehicles' cargo carrying capacity and time window are relatively small. R2, C2, and RC2 are Type 2, and logistics delivery vehicles have a large cargo capacity and time window. Display the information of the top 10 customers of instance C101 in Table 6. Numbers 1 to 10 represent customers, and 0 illustrates the depot. The maximum number of vehicles is 25. The capacity limit of each vehicle is 200. CUST NO.(Customer Number) means the customer number, 1-10 represents the customer, and 0 represents the depot. XCOORD. (X Coordinate) represents the horizontal axis coordinate. YCOORD. (Y Coordinate) represents the vertical axis coordinate. Demand represents the demand for goods; Ready Time represents the left

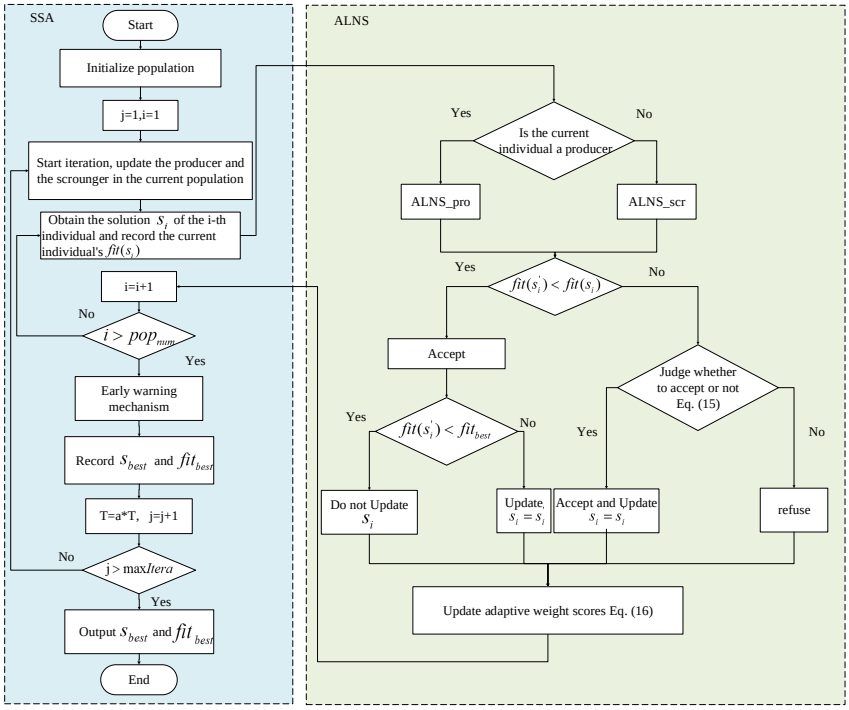


FIGURE 7. The flowchart for the proposed hybrid SSA-ALNS algorithm

time window. Due Time represents the right time window. Service Time represents the service time.

TABLE 6. Information on the top 10 customers of C101

CUST NO.	XCOORD.	YCOORD.	Demand	Ready Time	Due Time	Service Time
0.00	40.00	50.00	0.00	0.00	1236.00	0.00
1.00	45.00	68.00	10.00	912.00	967.00	90.00
2.00	45.00	70.00	30.00	825.00	870.00	90.00
3.00	42.00	66.00	10.00	65.00	146.00	90.00
4.00	42.00	68.00	10.00	727.00	782.00	90.00
5.00	42.00	65.00	10.00	15.00	67.00	90.00
6.00	42.00	96.00	20.00	621.00	702.00	90.00
7.00	40.00	66.00	20.00	170.00	225.00	90.00
8.00	40.00	68.00	20.00	255.00	324.00	90.00
9.00	38.00	70.00	10.00	534.00	605.00	90.00
10.00	38.00	66.00	10.00	357.00	410.00	90.00
...						

5.1. **Experimental Environment.** The algorithm testing experimental environment used in this paper is shown in Table 7. The calculation parameter settings of fitness

Algorithm 2 Pseudo-code of the proposed SSA-ALNS algorithm

Input: $pop_{num}, maxItera, T0, pro_{prate}, sd_{rate}$
Output: s_{best}, fit_{best}

Initialize parameters
Generate the initial population using the random method and calculate s_{best} and fit_{best}
$T = T0 * fit_{best}$
$j = 1$
while Stopping criteria is not satisfied **do**
    Update producers and scroungers in the current population
    for $i = 1$ to pop_{num} **do**
        if Current individual s_i belong to the producer **then**
            Execute ALNS_pro transformation, and calculate fitness $fit(s_i^{'})$;
        else
            Execute ALNS_scr transformation, and calculate fitness $fit(s_i^{'})$;
        end if
        $sc = 0$
        if $fit(s_i^{'}) < fit(s_i)$ **then**
            $s_i = s_i^{'}$
            if $fit(s_i^{'}) < fit_{best}$ **then**
                $s_{best} = s_i^{'}$
                $sc = sc_{d1}$
            else
                $sc = sc_{d2}$
            end if
        else
            if Accept new solution, Equation(15) **then**
                $s_i = s_i^{'}$
                $sc = sc_{d3}$
            else
                $sc = sc_{d4}$
            end if
        end if
        Update the adaptive weights based on the rating, Equation(16)
        $i = i + 1$
    end for
    Implement early warning mechanism
    Rank the sparrow, find the current best among all sparrow s_{best} and fintness fit_{best};
    $T = T \cdot \eta$
    $j = j + 1$
end while
return s_{best}, fit_{best}

in the experiment are shown in Table 8. Table 10 shows the experimental results under different simulated annealing parameter settings. It can be seen that when the experimental parameters are set to $T0 = 0.0001$ and $\eta = 0.93$, the optimal results can be obtained, with Avg reaching 0.88%.

TABLE 7. Experimental environment configuration

Name	Model
operating system	Windows 11 64bit
CPU	Intel(R) Core (TM) i7-10870H 2.20GHz 2.21GHz
Memory	16.00GB
compiler	MATLAB 2020b

TABLE 8. Experimental environment configuration

Symbol	Value
α	0.3
p	200
r	1
pun_1	100
pun_2	1000

TABLE 9. Experimental environment configuration

			C104	C109	R105	R108	R206	R208	RC103	RC104	RC204	
pop_{num}	pop_{prate}	sd_{rate}	gap	gap	gap	gap	gap	gap	gap	gap	gap	Avg.
10	0.1	0.2	2.23%	2.32%	0.66%	0.93%	1.44%	0.92%	1.46%	1.53%	1.25%	1.42%
10	0.2	0.2	2.03%	2.23%	0.59%	0.67%	1.41%	1.33%	1.37%	1.22%	1.45%	1.37%
10	0.2	0.3	1.98%	1.67%	0.47%	0.69%	1.27%	1.24%	1.51%	1.47%	1.31%	1.29%
20	0.1	0.2	2.27%	1.17%	0.50%	0.61%	1.16%	1.41%	1.60%	1.49%	1.10%	1.26%
20	0.2	0.2	1.99%	1.44%	0.42%	0.67%	1.40%	1.22%	1.37%	1.41%	1.42%	1.26%
20	0.2	0.3	1.77%	1.68%	0.44%	0.57%	1.33%	1.10%	1.27%	1.47%	1.40%	1.23%
50	0.1	0.2	1.13%	1.01%	0.30%	0.40%	0.93%	0.81%	1.32%	0.86%	1.23%	**0.89%**
50	0.2	0.2	1.10%	1.26%	0.25%	0.39%	1.05%	0.74%	1.15%	1.16%	0.94%	0.90%
50	0.2	0.3	1.36%	1.04%	0.37%	0.41%	1.16%	0.71%	1.00%	1.18%	1.15%	0.93%

5.2. **Parameter experiments.** To determine the parameter settings in the SSA-ALNS algorithm, we selected 9 difficult instances to solve, such as C104, C109, R105, R108, R206, R208, RC103, RC104, and RC204. Execute these instances 30 times under different parameter settings. We select the evaluation index *gap* as the evaluation index of the algorithm, which represents the relative error percentage between the average value of results and the currently known Best Know Solution (BKS). The formula is as follows:

$$gap = (f_{avg} - f_{BKS})/f_{BKS} \cdot 100\% \tag{26}$$

where f_{avg} represents the average of the total cost of the shortest delivery vehicle travel planning searched by the algorithm in multiple experiments; f_{BKS} represents the total cost of shortest delivery vehicle travel planning that is currently known to be solved by heuristic algorithms.

TABLE 10. Experimental results of simulated annealing parameters

		C104	C109	R105	R108	R206	R208	RC103	RC104	RC204	
$T0$	η	gap	gap	gap	gap	gap	gap	gap	gap	gap	Avg.
0.001	0.93	1.42%	0.93%	0.40%	0.76%	0.84%	1.39%	1.30%	1.15%	1.34%	1.06%
0.001	0.9996	3.68%	5.05%	1.91%	1.79%	1.31%	1.34%	2.32%	2.52%	1.43%	2.37%
0.001	0.99975	3.75%	4.91%	1.81%	1.59%	1.65%	2.02%	2.07%	2.64%	1.03%	2.38%
0.0005	0.93	1.44%	0.86%	0.45%	0.42%	1.07%	0.84%	1.30%	1.01%	1.04%	0.94%
0.0005	0.996	3.67%	3.96%	1.72%	1.74%	1.09%	0.50%	2.30%	2.63%	1.24%	2.09%
0.0005	0.99975	3.57%	4.13%	1.93%	1.87%	0.97%	0.99%	2.36%	2.96%	1.19%	2.22%
0.0001	0.93	1.36%	0.98%	0.34%	0.53%	1.04%	0.68%	1.19%	0.91%	0.92%	**0.88%**
0.0001	0.996	2.22%	1.56%	1.29%	1.17%	1.27%	0.84%	2.12%	2.62%	1.03%	1.57%
0.0001	0.99975	2.38%	1.43%	1.58%	1.34%	1.30%	1.54%	2.20%	2.29%	1.46%	1.73%

We need to tune some parameters in the algorithm, which play a crucial role in the optimization quality of the algorithm. Firstly, Table 9 shows the experimental results under the ratio settings of different population numbers and initial identity settings. Avg. represents the average gap of these nine instances. It can be seen that when the experimental parameters are set to $pop_{num} = 50$, $pro_{prate} = 0.1$ and $sd_{rate} = 0.2$, the optimal results can be obtained, with Avg reaching 0.89%.

5.3. **Experimental Comparison.** To verify the effectiveness and feasibility of SSA-ALNS in solving VRPTW problems, we will conduct comparative experiments with the proposed SSA-ALNS and other algorithms. For 56 instances with 100 customers in the Solomon Benchmark, conduct 30 experiments on each instance and calculate the average result. Table 11 shows the parameter settings for SSA-ALNS, ALNS [38], S-PSO [11], and SA [39].

We will display the adaptive operators in the SSA-ALNS score average results in Tables 12 and 13. In Figure 8, the probability ratio of operator use is shown in the pie chart. The proportion of operators removed in Figure 8(a) is relatively uniform, ranging from 10% to 20%. The proportion of path removal operators is the lowest, only 10.09%. In Figure 8(b), the greedy insertion operator has the lowest proportion of insertion operators, only 13.74%. The other two operators have similar proportions, 19.44% and 17.43%, respectively. In Figure 8(c), the two operators with the highest proportion in ALNS_scr are 2-opt and Or-opt, with a proportion exceeding 50%. It can be considered that they produce the best results. The proportion of other operators is relatively uniform, accounting for about 9% each.

Algorithm comparison section. Firstly, in Table 14, we present the optimal results obtained by the algorithm. In some instances, better solutions than BKS were found, represented by * in the table. Such as R101, R201, R208, R209, RC205, and RC207. Our algorithm found shorter driving distances than BKS in these instances when using the same number of vehicles. When facing type C instances, there is not much difference between the algorithms. Our algorithm can demonstrate certain advantages when faced with type R and type RC instances. At the same time, these two types are closer to the real situation, indicating that our algorithm has more advantages.

TABLE 11. Parameter Settings of the Algorithm

Algorithm	Parameter	Symbol	Value
SSA+ALNS	Number of populations	pop_{num}	50
	Maximum number of iterations	$maxItera$	300
	Initial temperature factor	$T0$	0.0001
	Cooling factor	η	0.93
	The lower bound of the initial removal factor	kr_{min}	0.1
	The upper bound of the initial removal factor	kr_{max}	0.15
	Weights used for Shaw removal	$(\varphi_1, \varphi_2, \varphi_3, sr_{ij})$	(6, 5, 4, 1)
	The initial proportion of producers	pro_{prate}	0.1
	The initial proportion of scroungers	scr_{rate}	0.9
	The initial proportion of early warning Personnel	sd_{rate}	0.2
	Adaptive score	$(sc_{d1}, sc_{d2}, sc_{d3}, sc_{d4})$	(1.6, 1.2, 0.8, 0.1)
	Adaptive update weights	ρ	0.4
ALNS	Maximum number of iterations	$MaxIter$	2000
	Initial temperature parameters	η	0.001
	Cooling rate	c	0.99975
	Solution Score	$(\sigma_1, \sigma_2, \sigma_3)$	(30, 10, 20)
	Reaction factor	r	0.1
	Number of iterations in each segment	$Iter$	200
	Shaw removes weights used	$(\phi_1, \phi_2, \phi_3, \phi_4)$	(8, 2, 1, 1)
	Weights used for optimal insertion	(g_1, g_2, g_3, g_4)	(0.5, 0.2, 0.2, 0.1)
	Randomize parameters	δ	16
	Noise parameters	μ	0.08
S-PSO-VRPTW	Inertia weight	ω	0.9 to 0.4
	Acceleration coefficient	c	2
	Refresh interval	rg	7
	Learning probability	Pc	0.05 to 0.5
	Population size	M	20
	Greedy initialization probability	φ	0.3
SA	Maximum number of iterations for outer loop	$MaxOutIter$	2000
	Maximum number of iterations for inner loop	$MaxInIter$	300
	Initial temperature	$T0$	100
	Cooling factor	η	0.99

Secondly, we also compare the fitness value calculated from BKS according to Table 8 with the fitness value calculated from the test algorithm, as shown in Table 15. Keep the results to two decimal places. Where min. represents the minimum fitness; Avg. represents the average fitness in 30 experiments; CPU time represents the calculation time of the algorithm in seconds; C, R and RC lines represent the average values of the corresponding results for type C, type R, and type RC instances, respectively; gapAvgAll. represents the average value of gap for all instances. When solving C and R instances,

TABLE 12. Average Score and Usage of Operators in ALNS_pro

	Removal Operators						Insertion Operators		
	Random	Route	Shaw	Time Window	Distance	Maximum distance	Farthest	Regret	Greedy
C1	1.254	0.92	1.215	1.106	1.236	1.249	1.263	1.37	0.487
C2	1.386	0.492	1.274	1.203	1.29	1.372	1.276	1.396	0.979
R1	1.127	0.848	0.937	0.804	1.037	1.128	1.022	1.17	0.179
R2	1.21	0.339	1.093	0.961	1.124	1.196	1.076	1.233	0.253
RC1	1.096	0.789	0.916	0.761	1.026	1.095	0.993	1.128	0.188
RC2	1.224	0.347	1.106	0.99	1.128	1.206	1.111	1.24	0.219
Avg.	1.216	0.622	1.09	0.971	1.14	1.207	1.124	1.256	0.384
Per.	19.44%	10.09%	17.43%	15.51%	18.23%	19.30%	40.73%	45.53%	13.74%

TABLE 13. Average Score and Usage of Operators in ALNS_scr

	Exchange	Reversal	Insertion	2-opt	Or-opt	Swap/Shift	2-opt*
C1	0.232	0.205	0.237	0.614	0.794	0.246	0.233
C2	0.227	0.202	0.244	0.441	0.825	0.235	0.231
R1	0.242	0.218	0.245	0.677	0.763	0.264	0.238
R2	0.242	0.21	0.27	0.454	0.779	0.249	0.253
RC1	0.24	0.22	0.245	0.683	0.769	0.262	0.235
RC2	0.235	0.212	0.26	0.462	0.788	0.245	0.241
Avg.	0.236	0.211	0.25	0.555	0.786	0.25	0.238
Per.	9.34%	8.34%	9.88%	22.07%	31.06%	9.90%	9.42%

our algorithm yields an average gap of only 0.63% and 10.66%, which is better. However, our algorithm achieved 11.73% when solving RC instances, slightly worse than S-PSO-VRPTW. Our proposed algorithm achieves a $gapAvgAll.$ of 7.92%, which improves the average optimization ability of SSA-ALNS by 30.04% compared to the 11.32% obtained by traditional ALNS. Compared with the 8.04% obtained from S-PSO-VRPTW, the average optimization ability of SSA-ALNS increased by 1.5%.

Regarding computational time, our algorithm significantly shortens the time compared to S-PSO-VRPTW. But it takes longer than traditional ALNS. It can be seen that in most calculation instances, SSA-ALNS can produce better results and be closer to BKS.

Result display section. Firstly, we will present 6 instances of the optimal solution obtained in Figure 9, including R101, R201, R208, R209, RC205, and RC207. In Table 16, we will present the optimal route results obtained from the proposed algorithm search. Secondly, we also selected some SSA-ALNS solutions to display the results, as shown in Figure 10. In Figure 11, their iterative process is shown. It can be seen that our proposed algorithm has a fast convergence speed and finds a better optimal value.

6. **Conclusion.** This study proposed an enhanced SSA-ALNS algorithm to tackle the VRPTW problem, addressing the limitations of ALNS related to local optima and algorithm stability. The key innovations include integrating SSA's operation mechanisms into traditional ALNS and expanding the types of operators in ALNS, such as 2-opt, Or-opt, 2-opt*, swap/shift, and more. Additionally, the algorithm is divided into ALNS_pro and ALNS_scr to conduct targeted searches for producers and scroungers in the population.

Table 14. Optimal Solution

	BKS		SSA-ALNS		ALNS		S-PSO-VRPTW		SA	
	NV	TD	NV	TD	NV	TD	NV	TD	NV	TD
C101	10	828.94	10	828.94	10	828.94	10	828.94	10	828.94
C102	10	828.94	10	828.94	10	828.94	10	829.71	10	828.94
C103	10	828.06	10	828.06	10	828.06	10	851.37	10	882.76
C104	10	824.78	10	847.05	10	853.08	10	868.52	10	886.95
C105	10	828.94	10	828.94	10	828.94	10	828.94	10	828.94
C106	10	828.94	10	828.94	10	828.94	10	828.94	10	828.94
C107	10	828.94	10	828.94	10	862.37	10	828.94	10	828.94
C108	10	828.94	10	828.94	10	828.94	10	828.94	10	863.50
C109	10	828.94	10	828.94	10	828.94	10	828.94	10	924.42
C201	3	591.56	3	591.56	3	591.56	3	591.56	3	591.56
C202	3	591.56	3	591.56	3	591.56	3	591.56	3	591.56
C203	3	591.17	3	591.17	3	603.37	3	591.17	3	591.56
C204	3	590.60	3	590.60	3	607.15	3	615.43	3	612.72
C205	3	588.88	3	588.88	3	588.88	3	588.88	3	588.88
C206	3	588.49	3	588.49	3	588.49	3	588.88	3	588.49
C207	3	588.29	3	588.29	3	588.29	3	591.35	3	588.29
C208	3	588.32	3	588.32	3	588.32	3	588.49	3	588.32
R101	19	1645.79	19	1642.88*	19	1689.09	19	1652.00	21	1707.70
R102	17	1486.12	18	1477.32	18	1519.85	17	1500.81	19	1530.78
R103	13	1292.68	14	1234.19	14	1264.98	14	1242.65	15	1299.53
R104	9	1007.24	10	1012.87	10	1032.54	10	1042.22	12	1050.69
R105	14	1377.11	14	1382.20	14	1428.61	14	1385.08	17	1507.51
R106	12	1251.98	13	1262.25	13	1299.67	12	1294.87	15	1360.33
R107	10	1104.66	11	1090.77	11	1132.33	11	1123.98	13	1133.34
R108	9	960.88	10	956.49	10	995.85	10	1011.68	11	1027.81
R109	11	1194.73	12	1164.89	12	1270.82	12	1211.63	15	1253.71
R110	10	1118.59	11	1099.70	11	1144.14	11	1190.36	14	1182.65
R111	10	1096.72	11	1083.36	11	1129.14	11	1102.99	13	1161.26
R112	9	982.14	10	965.49	10	1023.74	10	1029.12	12	1051.56
R201	4	1252.37	4	1206.02*	4	1300.15	4	1274.97	8	1289.70
R202	3	1191.70	3	1254.38	4	1122.67	3	1247.03	8	1114.41
R203	3	939.54	3	949.09	3	973.75	3	1052.71	6	952.18
R204	2	825.52	3	755.36	3	782.37	3	844.16	5	796.69
R205	3	994.42	3	1010.15	3	1090.17	3	1061.46	7	1060.45
R206	3	906.14	3	937.07	3	998.74	3	1016.35	6	1007.19
R207	2	893.33	3	818.13	3	919.22	3	946.78	5	891.63
R208	2	726.75	2	715.53*	2	779.17	2	834.72	4	775.17
R209	3	909.16	3	893.90*	3	981.78	3	1003.19	7	969.86
R210	3	939.34	3	957.61	3	1011.70	3	1040.54	7	1020.27
R211	2	892.71	2	980.05	3	876.95	3	861.32	6	847.17
RC101	14	1696.94	15	1652.47	15	1758.79	15	1641.20	18	1779.37
RC102	12	1554.75	14	1503.15	14	1527.71	13	1510.95	16	1620.90
RC103	11	1261.67	12	1327.63	12	1380.88	11	1294.74	14	1422.85
RC104	10	1135.48	10	1170.86	10	1204.01	10	1190.55	12	1223.58
RC105	13	1629.44	15	1544.62	15	1600.43	14	1603.71	17	1624.26
RC106	11	1424.73	13	1380.06	13	1463.87	12	1410.93	16	1522.40
RC107	11	1230.48	12	1252.57	12	1366.82	11	1249.80	13	1358.09
RC108	10	1139.82	11	1165.76	11	1272.01	11	1181.87	12	1202.17
RC201	4	1406.91	4	1427.95	4	1497.74	4	1423.52	10	1430.03
RC202	3	1367.09	4	1161.29	4	1239.13	4	1193.59	8	1216.99
RC203	3	1049.62	3	1058.48	4	1146.29	3	1123.42	7	1049.75
RC204	3	798.41	3	820.48	3	835.91	3	894.12	5	889.43
RC205	4	1297.19	4	1253.48*	4	1338.48	4	1321.43	9	1305.31
RC206	3	1146.32	3	1406.18	4	1184.80	3	1307.90	7	1210.87
RC207	3	1061.14	3	1013.24*	3	1114.37	3	1130.37	7	1103.89
RC208	3	828.14	3	871.90	3	933.31	3	958.24	6	875.92

TABLE 15. Comparison of fitness values

	BKS	SSA-ALNS				ALNS				S-PSO-VRPTW				SA			
	min.	min.	avg.	gap	CPU Time(s)	min.	avg.	gap	CPU Time(s)	min.	avg.	gap	CPU Time(s)	min.	avg.	gap	CPU Time(s)
C101	1648.68	**1648.68**	1665.81	1.04%	421.68	1648.68	1648.68	0.00%	148.04	1648.68	1648.68	0.00%	336	1648.68	1653.57	0.30%	166.06
C102	1648.68	**1648.68**	1653.78	0.31%	427.97	1648.68	1659.08	0.63%	180.21	1648.91	1655.25	0.40%	581.4	1648.68	1652.29	0.22%	167.78
C103	1648.42	**1648.42**	1655.6	0.44%	431.07	1648.42	1685.86	2.27%	198.6	1655.41	1665.9	1.06%	1726.8	1664.83	1697.35	2.97%	167.99
C104	1647.43	**1654.12**	1667.45	1.22%	424.94	1655.92	1692.69	2.75%	206.93	1660.56	1687.55	2.44%	2404.2	1666.09	1704.83	3.48%	162.28
C105	1648.68	**1648.68**	1652.78	0.25%	424.65	1648.68	1652.14	0.21%	211.85	1648.68	1649.13	0.03%	364.8	1648.68	1665.5	1.02%	167.33
C106	1648.68	**1648.68**	1648.68	0.00%	420.06	1648.68	1687.49	2.35%	204.78	1648.68	1648.68	0.00%	379.2	1648.68	1720.18	4.34%	170.58
C107	1648.68	**1648.68**	1664.29	0.95%	428.74	1658.71	1704.46	3.38%	168.74	1648.68	1659.59	0.66%	417	1648.68	1694.44	2.78%	168.12
C108	1648.68	**1648.68**	1653.61	0.30%	420.73	1648.68	1680.66	1.94%	143.8	1648.68	1648.72	0.00%	759.6	1659.05	1728.68	4.85%	179.53
C109	1648.68	**1648.68**	1665.16	1.00%	421.17	1648.68	1713.81	3.95%	144.63	1648.68	1648.87	0.01%	934.8	1677.32	1725.11	4.64%	171.92
C201	597.47	**597.47**	597.47	0.00%	357.51	597.47	597.47	0.00%	87.94	597.47	606.53	1.52%	377.4	597.47	611.09	2.28%	117.9
C202	597.47	**597.47**	597.47	0.00%	364.3	597.47	633.71	6.07%	89.48	597.47	604.91	1.25%	432.6	597.47	616.82	3.24%	110.63
C203	597.35	**597.35**	606.39	1.51%	277.03	601.01	691.4	15.74%	90.41	597.35	601.75	0.74%	845.4	597.47	630.07	5.48%	143.94
C204	597.18	**597.18**	619.34	3.71%	381.44	602.14	634.54	6.26%	91.58	604.63	651.43	9.08%	2285.4	603.82	656.96	10.01%	189.76
C205	596.66	**596.66**	596.66	0.00%	359.62	596.66	596.66	0.00%	88.41	596.66	600.09	0.57%	355.2	596.66	611.36	2.46%	170.1
C206	596.55	**596.55**	596.55	0.00%	340.23	596.55	596.55	0.00%	93.24	596.66	597.88	0.22%	385.8	596.55	622.64	4.37%	168.25
C207	596.49	**596.49**	596.49	0.00%	317.96	596.49	596.49	0.00%	88.64	597.41	598.96	0.41%	385.8	596.49	629.98	5.61%	114.26
C208	596.5	**596.5**	596.5	0.00%	309.39	596.5	600.35	0.65%	92.2	596.55	597.48	0.16%	382.8	596.5	622.3	4.33%	105.59
R101	3153.74	**3152.86**	3229.54	2.40%	953.56	3166.73	3217.46	2.02%	265.79	3155.6	3157.37	0.12%	439.8	3452.31	3602.9	14.24%	269.34
R102	2825.84	2963.2	2969.22	5.07%	723.73	2975.96	2966.52	4.98%	252.1	**2830.24**	2943.99	4.18%	882.6	3119.23	3249.29	14.98%	245.2
R103	2207.8	**2330.26**	2347.38	6.32%	556.61	2339.5	2332.44	5.65%	240.85	2332.79	2340.58	6.01%	1656	2489.86	2695.87	22.11%	206.41
R104	1562.17	**1703.86**	1726.36	10.51%	464.73	1709.76	1732.24	10.89%	290.4	1712.66	1799.33	15.18%	1935	1995.21	2183.75	39.79%	171.33
R105	2373.13	**2374.66**	2506.22	5.61%	575.01	2388.58	2533.5	6.76%	312	2375.52	2423.58	2.13%	1056.6	2832.25	3007.1	26.71%	226.48
R106	2055.59	2198.67	2247.22	9.32%	507.14	2209.9	2188.05	6.44%	196.36	**2068.46**	2192.11	6.64%	1366.2	2508.1	2629.39	27.91%	202.98
R107	1731.4	**1867.23**	1877.68	8.45%	449.74	1879.7	1882.56	8.73%	171.82	1877.19	1888.01	9.05%	1696.2	2160	2320.21	34.01%	183.42
R108	1548.26	**1686.95**	1696.21	9.56%	453.51	1698.76	1705.61	10.16%	157.65	1703.5	1733.89	11.99%	1832.4	1848.34	2016.21	30.22%	161.57
R109	1898.42	**2029.47**	2093.78	10.29%	768.37	2061.25	2137.36	12.59%	192.97	2043.49	2111.84	11.24%	2509.8	2476.11	2563.71	35.04%	196.16
R110	1735.58	**1869.91**	1992.44	14.80%	567.35	1883.24	2001.86	15.34%	182.14	1897.11	2037.11	17.37%	2196.6	2314.79	2412.79	39.02%	255.78
R111	1729.02	**1865.01**	1929.02	11.57%	532.01	1878.74	1885.14	9.03%	176.04	1870.9	1930.98	11.68%	1755.6	2168.38	2322.43	34.32%	276.18
R112	1554.64	**1689.65**	1731.12	11.35%	483.1	1707.12	1832.3	17.86%	170.21	1708.74	1842.92	18.54%	2464.8	1995.47	2063.46	32.73%	229.68
R201	935.71	**921.81**	970	3.66%	339.08	950.05	1003.72	7.27%	108.24	942.49	949.48	1.47%	1457.4	1506.91	1815.93	94.07%	113.79
R202	777.51	796.32	893.24	14.88%	385.56	896.8	959.88	23.46%	106.7	**794.11**	867.95	11.63%	2052	1454.32	1654.04	112.74%	112.44
R203	701.86	**704.73**	725.93	3.43%	429.84	712.13	798.47	13.76%	98.26	735.81	750.24	6.89%	2031	1125.65	1311.63	86.88%	108.02
R204	527.66	**646.61**	653.66	23.88%	443.84	654.71	671.09	27.18%	107.73	673.25	698.41	32.36%	2337.6	939.01	1056.85	100.29%	114.96
R205	718.33	**723.05**	794.64	10.62%	389.8	747.05	870.05	21.12%	102.54	738.44	760.71	5.90%	2125.2	1298.14	1555.92	116.60%	112.7
R206	691.84	**701.12**	709.08	2.49%	305.6	719.62	783.39	13.23%	116.27	724.9	739.68	6.91%	2702.4	1142.16	1321.24	90.97%	107.7
R207	548	**665.44**	676.46	23.44%	299.07	695.77	683.52	24.73%	127.1	704.03	731.06	33.41%	2436.6	967.49	1104.36	101.53%	118.28
R208	498.03	**494.66**	587.24	17.91%	292.07	513.75	643.87	29.28%	127.02	530.42	628.09	26.11%	2563.2	792.55	900.25	80.76%	104.7
R209	692.75	**688.17**	783.52	13.10%	314.33	714.53	828.03	19.53%	130.6	720.96	743.08	7.27%	2502.6	1270.96	1427.23	106.02%	109.38
R210	701.8	**707.28**	729.61	3.96%	309.46	723.51	863.69	23.07%	149.07	732.16	747.11	6.46%	2619.6	1286.08	1396.63	99.01%	106.93
R211	547.81	**574.01**	670.78	22.45%	301.34	683.09	758.13	38.39%	152.54	678.4	701.45	28.05%	2724.6	1094.15	1218.92	122.51%	114.47
RC101	2469.08	2595.74	2713.79	9.91%	596.13	2627.64	2742.79	11.09%	323.21	**2592.36**	2600.66	5.33%	765.6	3053.81	3200.52	29.62%	245.52
RC102	2146.43	2410.95	2461.34	14.67%	554.23	2418.31	2410.97	12.32%	310.15	**2273.29**	2384.04	11.07%	1098.6	2726.27	2930.34	36.52%	218.45
RC103	1918.5	2078.29	2100.83	9.50%	498.33	2094.27	2075.99	8.21%	257.51	**1928.42**	2029.78	5.80%	1252.2	2386.85	2563.77	33.63%	193.37
RC104	1740.64	**1751.26**	1885.68	8.33%	452.06	1761.2	1897.46	9.01%	185.62	1757.16	1877.8	7.88%	1617	2047.07	2227.42	27.97%	196.67
RC105	2308.83	2563.39	2643.97	14.52%	585.28	2580.13	2569.68	11.30%	248.41	**2441.11**	2560.54	10.90%	1080	2867.28	2987.98	29.42%	284.22
RC106	1967.42	2234.02	2347.9	19.34%	534.25	2259.16	2302.55	17.03%	237.89	**2103.28**	2216.85	12.68%	1424.4	2696.72	2740.25	39.28%	283.16
RC107	1909.14	2055.77	2123.98	11.25%	498.96	2090.05	2122.25	11.16%	207.44	**1914.94**	1965.39	2.95%	2259.6	2227.43	2396.23	25.51%	248.88
RC108	1741.95	**1889.73**	1957.83	12.39%	479.08	1921.6	2009.89	15.38%	189.23	1894.56	1909.28	9.61%	2167.2	2040.65	2178.72	25.07%	189.49
RC201	982.07	988.38	1024.13	4.28%	325.13	1009.32	1147.29	16.82%	113.99	**987.06**	1001.81	2.01%	1329	1829.01	2041.87	107.91%	117.28
RC202	830.13	**908.39**	935.65	12.71%	316.11	931.74	953.4	14.85%	109.65	918.08	945.96	13.95%	1726.2	1485.1	1717.71	106.92%	116.7
RC203	734.89	**737.54**	856.69	16.57%	313.37	903.89	885.58	20.51%	105.94	757.03	786.84	7.07%	2289	1294.92	1403	90.91%	113.91
RC204	659.52	**666.14**	674.38	2.25%	290.86	670.77	741.55	12.44%	108.86	688.24	709.35	7.56%	2056.8	966.83	1081.66	64.01%	116.89
RC205	949.16	**936.04**	1072.12	12.95%	322.74	961.54	1083.31	14.13%	115.55	956.43	974.66	2.69%	2393.4	1651.59	1875.79	97.63%	117.99
RC206	763.9	841.85	910.03	19.13%	372.16	915.44	937.56	22.73%	160.13	**812.37**	927.62	21.43%	2538.6	1343.26	1592.13	108.42%	118.77
RC207	738.34	**723.97**	862.94	16.88%	401.18	754.31	896.42	21.41%	155.97	759.11	875.77	18.61%	2748	1311.17	1468.2	98.85%	117.57
RC208	668.44	**681.57**	688.37	2.98%	295.32	699.99	786.94	17.73%	143.81	707.47	745.48	11.53%	3216	1102.78	1292.26	93.32%	110.29
C				0.63%	384.03			2.72%	137.03			1.09%	785.54			3.67%	155.41
R				10.66%	471.52			15.28%	171.06			12.20%	1971.47			63.59%	167.3
RC				11.73%	427.2			14.76%	185.84			9.44%	1872.6			63.44%	174.32
gapAvgAll.				7.92%	432.3			11.32%	164.95			8.04%	1583.21			45.35%	165.7

TABLE 16. Route of experimental results for optimal values obtained by SSA-ALNS

instances	Routes
R101	Route 1 0->12->76->79->3->54->24->80->0 Route 2 0->39->23->67->55->25->0 Route 3 0->63->64->49->48->0 Route 4 0->45->82->18->84->60->89->0 Route 5 0->92->42->15->87->57->97->0 Route 6 0->95->98->16->86->91->100->0 Route 7 0->2->21->73->41->56->4->0 Route 8 0->62->11->90->10->0 Route 9 0->72->75->22->74->58->0 Route 10 0->27->69->30->51->20->32->70->0 Route 11 0->40->53->26->0 Route 12 0->65->71->9->66->1->0 Route 13 0->5->83->61->85->37->93->0 Route 14 0->36->47->19->8->46->17->0 Route 15 0->28->29->78->34->35->77->0 Route 16 0->31->88->7->0 Route 17 0->14->44->38->43->13->0 Route 18 0->59->99->94->96->0 Route 19 0->33->81->50->68->0 Route 20 0->52->6->0
R201	Route 1 0->27->69->31->52->82->7->18->99->85->87->57->41->22->56->4->54->55->25->24->80->77->0 Route 2 0->28->33->65->71->29->12->76->79->81->51->9->78->34->50->3->68->26->74->13->58->0 Route 3 0->5->45->83->47->36->63->64->11->19->62->88->30->90->10->20->66->35->32->70->1->0 Route 4 0->2->72->39->75->23->67->21->73->40->53->94->6->96->97->37->43->100->91->93->0 Route 5 0->95->59->92->42->15->14->98->61->16->44->38->86->84->8->49->46->48->17->60->89->0
R208	Route 1 0->28->76->50->1->69->70->30->20->66->65->71->35->9->51->33->81->34->78->79->3->77->29->24->55->25->4->72->74->73->21->40->58->0 Route 2 0->27->52->31->88->7->82->48->19->11->62->10->32->90->63->64->49->36->47->46->8->45->17->84->83->18->60->5->99->59->97->87->2->57->42->100->91->98->37->95->94->13->0 Route 3 0->89->6->96->92->93->85->61->16->86->44->38->14->43->15->41->22->75->56->23->67->39->54->80->68->12->26->53->0
R209	Route 1 0->40->2->73->21->72->75->23->67->39->25->55->80->68->77->50->3->79->34->35->65->66->20->32->10->70->1->0 Route 2 0->27->52->69->31->88->62->63->11->19->47->7->82->83->5->84->17->45->18->6->94->13->53->58->41->22->74->56->4->54->24->26->0 Route 3 0->95->99->59->98->85->61->16->86->38->44->14->87->57->15->43->42->97->92->37->100->91->93->96->0 Route 4 0->28->12->76->29->78->71->9->81->33->51->30->90->64->49->36->46->8->48->60->89->0
RC205	Route 1 0->65->83->64->23->21->18->19->57->99->52->86->87->59->75->97->74->24->48->25->77->58->0 Route 2 0->2->45->42->39->36->44->61->90->98->88->78->73->79->7->8->6->46->5->3->4->1->43->35->37->93->96->0 Route 3 0->92->95->33->28->27->29->31->30->63->76->51->84->49->22->20->66->56->85->89->26->32->34->50->91->80->0 Route 4 0->69->82->11->15->16->47->14->12->53->9->10->13->17->60->100->70->0 Route 5 0->94->62->67->71->72->38->40->41->81->54->68->55->0
RC207	Route 1 0->82->53->98->90->61->72->93->94->81->42->44->40->39->38->37->35->36->43->41->54->96->0 Route 2 0->92->95->67->71->31->29->27->28->30->85->63->76->18->19->49->51->84->62->50->34->26->32->33->89->48->20->24->66->56->91->80->0 Route 3 0->65->99->52->83->64->22->21->23->25->77->75->58->59->87->86->57->74->97->13->9->10->17->78->60->55->0 Route 4 0->69->88->12->11->15->16->47->14->73->79->7->6->2->8->5->3->1->45->46->4->100->70->68->0

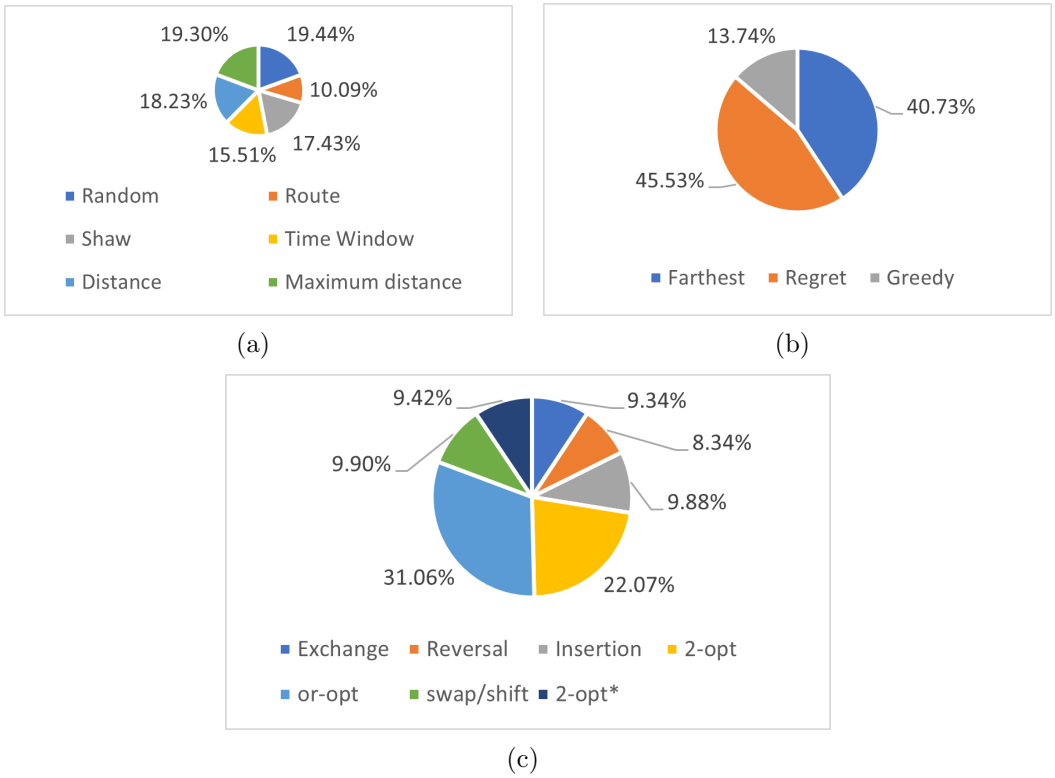
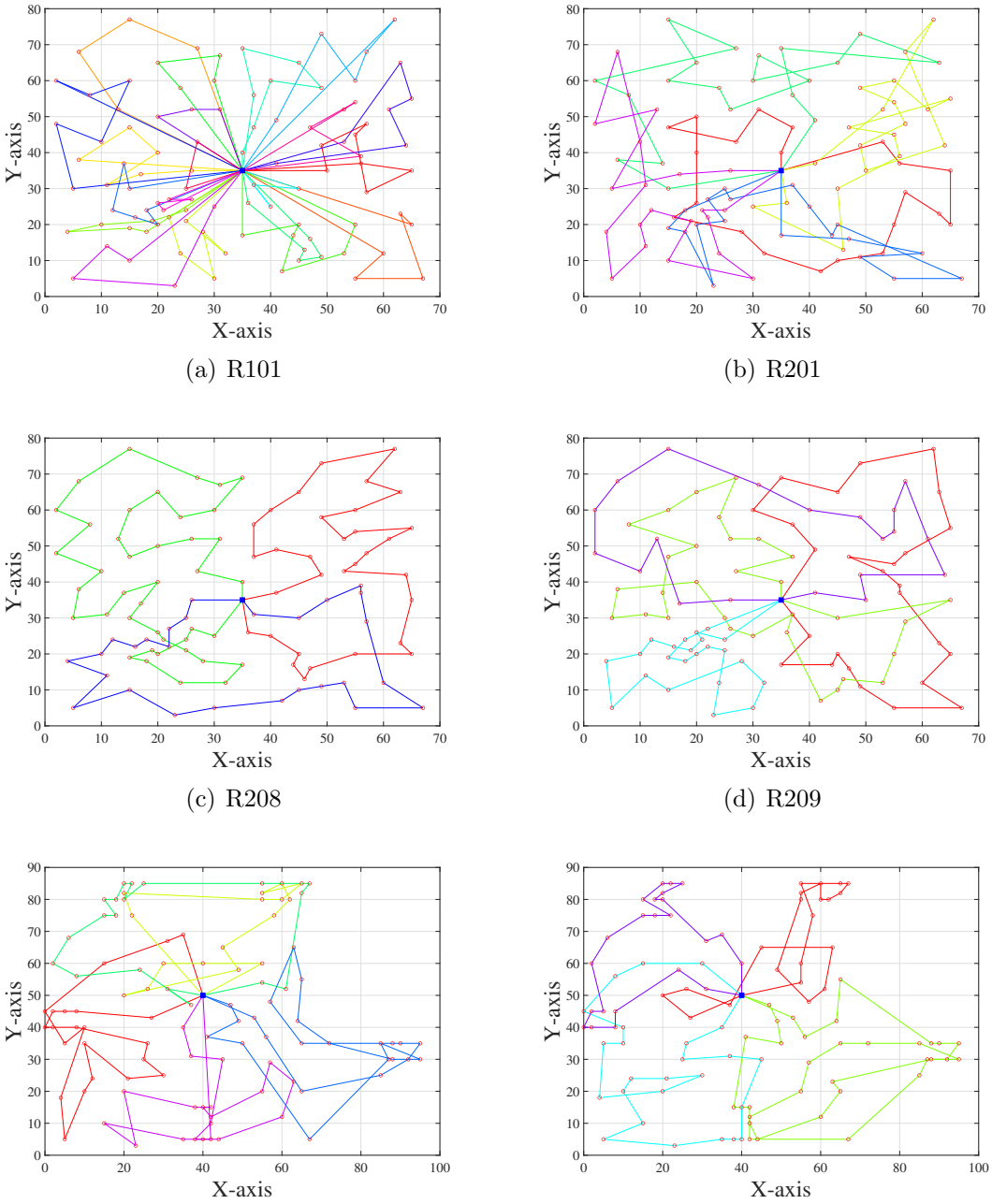


FIGURE 8. Proportion of operator usage

The contributions of this paper are presented as follows: Firstly, the proposed SSA-ALNS algorithm builds upon the traditional ALNS, enhancing population diversity and algorithm stability. Secondly, by extending multiple operators for ALNS, the algorithm's local development ability is improved, and the search space is expanded. Mathematical experiments optimized the algorithm's parameters, leading to suitable settings for 100 customers, as demonstrated in Table 11.

Conduct comparative experiments with ALNS, S-PSO, and other algorithms on 56 instances in Solomon Benchmark. The algorithm proposed in this article can find better results when facing most cases. At the same time, we will display the result graph, and it can be seen that the result paths are relatively reasonable. Notably, the removal operators showed relative balance, while the greedy insertion operator produced suboptimal results. In ALNS_scr, 2-opt and Or-opt consistently generated superior solutions. The experimental results underscore the feasibility and effectiveness of our proposed algorithm.

While our algorithm achieved remarkable results, there are avenues for future exploration. These include combining SSA-ALNS with other neighborhood search algorithms that have not been studied yet, making the algorithm's time complexity smaller so that it takes less time to run, and using SSA-ALNS to solve other VRPs like TDVRP, MDVRP, MDVRPTW, and similar ones. These extensions hold promise for further advancing the field of vehicle routing optimization.

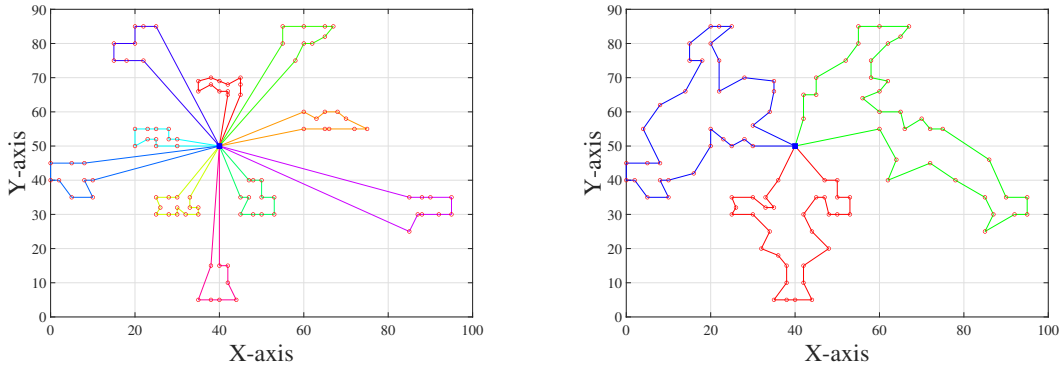


(a) R101 (b) R201

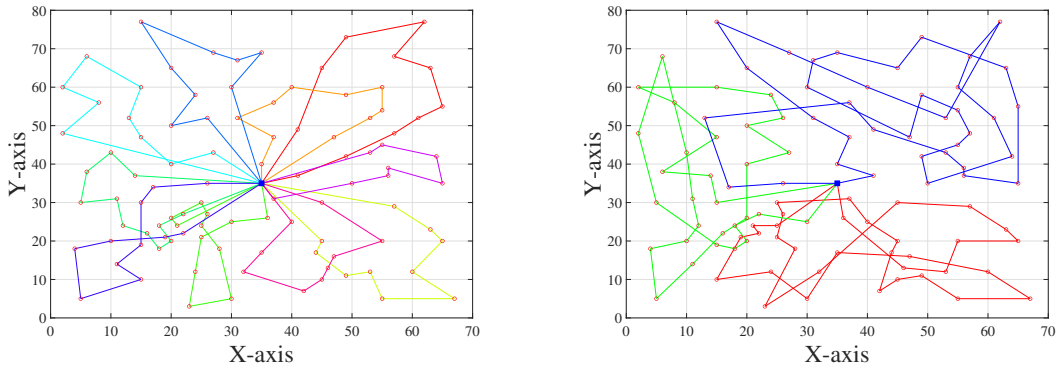
(c) R208 (d) R209

(e) RC205 (f) RC207

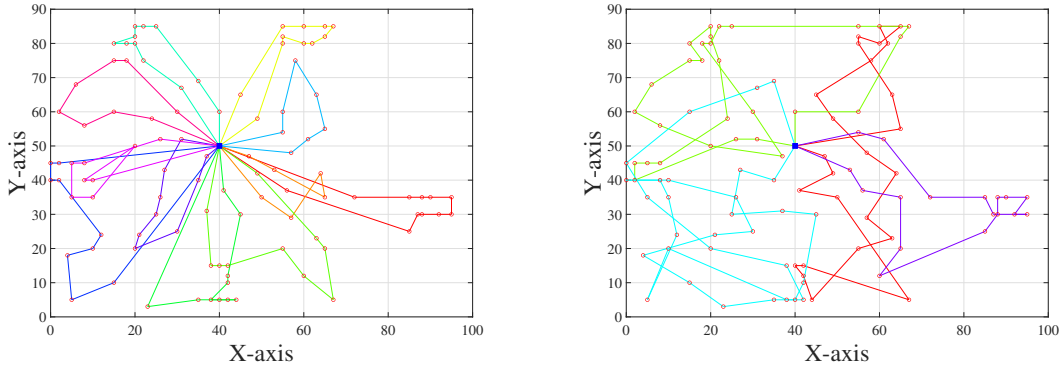
FIGURE 9. Experimental result graph of the optimal values obtained by SSA-ALNS



(a) C105 (b) C207

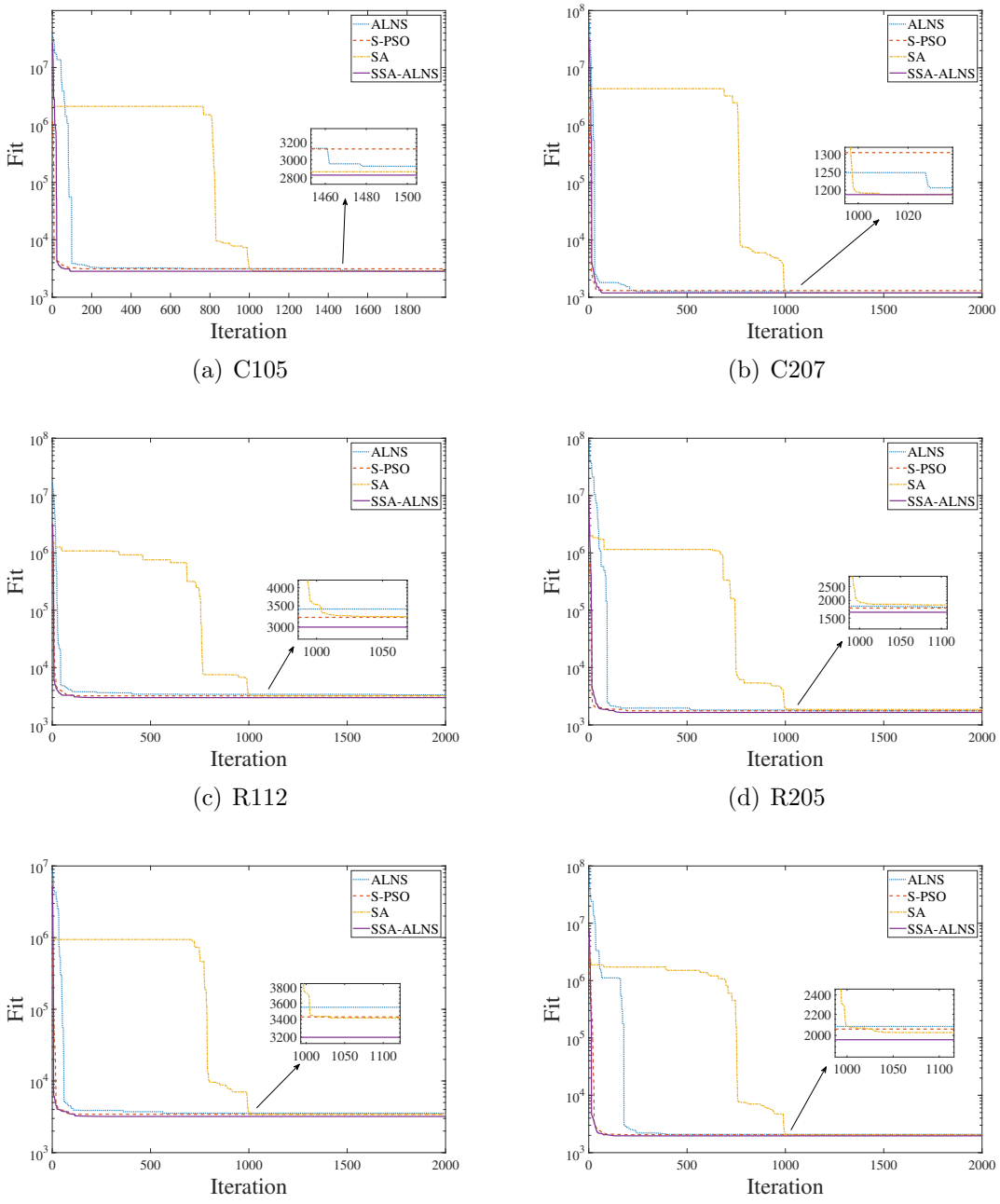


(c) R112 (d) R205



(e) RC104 (f) RC206

FIGURE 10. Experimental result graph of the optimal values obtained by SSA-ALNS



(a) C105 (b) C207

(c) R112 (d) R205

(e) RC104 (f) RC206

FIGURE 11. Experimental result graph of the optimal values obtained by SSA-ALNS

Acknowledgment. This work was supported in part by Fujian Provincial Department of Science and Technology under Grant No. 2021J011070, and Fujian University of Technology under Grant No. GY-Z18148.

REFERENCES

[1] B. Han, R. Gao, J. Li, and H. Chen, "Research and analysis of electronic commerce in network platforms with mathematical model and cloud computing," in *Journal of Physics: Conference Series*, vol. 1952, 42116, no. 4. IOP Publishing, 2021.

[2] T.-K. Dao, T.-S. Pan, T.-T. Nguyen, and J.-S. Pan, "Parallel bat algorithm for optimizing makespan in job shop scheduling problems," *Journal of Intelligent Manufacturing*, vol. 29, no. 2, pp. 451–462, 2018.

[3] T.-T. Nguyen, T.-K. Dao, M.-F. Horng, and C.-S. Shieh, "An energy-based cluster head selection algorithm to support long-lifetime in wireless sensor networks." *Journal of Network Intelligence*, vol. 1, no. 1, pp. 23–37, 2016.

[4] T.-S. Pan, T.-K. Dao, T.-T. Nguyen, and S.-C. Chu, "Hybrid particle swarm optimization with bat algorithm," in *Genetic and Evolutionary Computing: Proceeding of the Eighth International Conference on Genetic and Evolutionary Computing*. Springer, 2015, pp. 37–47.

[5] K. Huang, J. Wang, and M. Khader, "Research on the impact of environmental regulation on total factor energy effect of logistics industry from the perspective of green development," *Mathematical Problems in Engineering*, vol. 2022, pp. 1–17, 2022.

[6] J.-S. Pan, T.-K. Dao, T.-S. Pan, T.-T. Nguyen, S.-C. Chu, and J. F. Roddick, "An improvement of flower pollination algorithm for node localization optimization in wsn." *Journal of Information Hiding and Multimedia Signal Processing*, vol. 8, no. 2, pp. 486–499, 2017.

[7] B. L. Golden, S. Raghavan, and E. A. Wasil. Springer, 2008, vol. 43.

[8] T.-T. Nguyen, H.-J. Wang, T.-K. Dao, J.-S. Pan, and S.-W. Weng, "An improved slime mold algorithm and its application for optimal operation of cascade hydropower stations," *IEEE Access*, vol. 8, pp. 226 754–226 772, 2020.

[9] G. B. Dantzig and J. H. Ramser, "The truck dispatching problem," *Management Science*, vol. 6, no. 1, pp. 80–91, 1959.

[10] R. Elshaer and H. Awad, "A taxonomic review of metaheuristic algorithms for solving the vehicle routing problem and its variants," *Computers & Industrial Engineering*, vol. 140, 106242, 2020.

[11] Y.-J. Gong, J. Zhang, O. Liu, R.-Z. Huang, H. S.-H. Chung, and Y.-H. Shi, "Optimizing the vehicle routing problem with time windows: a discrete particle swarm optimization approach," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 2, pp. 254–267, 2011.

[12] A. M. Altabeeb, A. M. Mohsen, and A. Ghallab, "An improved hybrid firefly algorithm for capacitated vehicle routing problem," *Applied Soft Computing*, vol. 84, 105728, 2019.

[13] R. Yesodha and T. Amudha, "A bio-inspired approach: firefly algorithm for multi-depot vehicle routing problem with time windows," *Computer Communications*, vol. 190, pp. 48–56, 2022.

[14] C.-H. Wang, R. Tian, K. Hu, Y.-T. Chen, and T.-H. Ku, "A markov decision optimization of medical service resources for two-class patient queues in emergency departments via particle swarm optimization algorithm," *Scientific Reports*, vol. 15, p. 2942, 2025.

[15] T.-T. Nguyen, Y. Zeng, C.-H. Wang, J. Yuan, and T.-K. Dao, "A solution to the job shop scheduling problem based on an enhanced slime mould algorithm," *International Journal of Computing Science and Mathematics*, vol. 21, no. 4, pp. 289–302, 2025.

[16] C.-H. Wang, "A three-level health inspection queue based on risk screening management mechanism for post-covid global economic recovery," *IEEE Access*, vol. 8, pp. 177 604–177 614, 2020.

[17] C.-H. Wang, Y.-T. Chen, and X. Wu, "A multi-tier inspection queueing system with finite capacity for differentiated border control measures," *IEEE Access*, vol. 9, pp. 60 489–60 502, 2021.

[18] C.-M. Chen, S. Lv, J. Ning, and J. M.-T. Wu, "A genetic algorithm for the waitable time-varying multi-depot green vehicle routing problem," *Symmetry*, vol. 15, 124, no. 1, 2023.

[19] C.-H. Wang, J. Cai, Q. Ye, and Y. Suo, "A two-stage convolution network algorithm for predicting traffic speed based on multi-feature attention mechanisms," *Journal of Intelligent & Fuzzy Systems*, vol. 45, no. 3, pp. 5181–5196, 2023.

[20] J. Cai, C.-H. Wang, and K. Hu, "Lcdformer: Long-term correlations dual-graph transformer for traffic forecasting," *Expert Systems with Applications*, vol. 249, 123721, 2024.

[21] S. Chen, C.-H. Wang, Z. Dong, Q. Zhao, Q. Yang, Y. Wei, and G. Huang, "Performance evaluation of three intelligent optimization algorithms for obstacle avoidance path planning," in *Genetic and Evolutionary Computing: Proceedings of the Fourteenth International Conference on Genetic and Evolutionary Computing, October 21-23, 2021, Jilin, China 14*. Springer, 2022, pp. 60–69.

[22] Z. Dong, C.-H. Wang, Q. Zhao, Y. Wei, S. Chen, and Q. Yang, "A study on intelligent optimization algorithms for capacity allocation of production networks," in *Proceedings of 2021 Chinese Intelligent Systems Conference: Volume II*. Springer, 2022, pp. 734–743.

[23] C.-H. Wang, C.-J. Lee, and X. Wu, "A coverage-based location approach and performance evaluation for the deployment of 5G base stations," *IEEE Access*, vol. 8, pp. 123 320–123 333, 2020.

[24] C.-H. Wang, T.-T. Nguyen, J.-S. Pan, and T.-k. Dao, "An optimization approach for potential power generator outputs based on parallelized firefly algorithm," in *Advances in Intelligent Information Hiding and Multimedia Signal Processing: Proceeding of the Twelfth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Nov., 21-23, 2016, Kaohsiung, Taiwan, Volume 2*. Springer, 2017, pp. 297–306.

[25] C.-H. Wang and M.-E. Wu, "Numerical evaluation of two management schemes for sharing limited bandwidth," in *2016 Third International Conference on Computing Measurement Control and Sensor Network (CMCSN)*. IEEE, 2016, pp. 182–185.

[26] M.-H. Hung, C.-H. Wang, and Y. He, "A real-time routing algorithm for end-to-end communication networks with QoS requirements," in *2016 Third International Conference on Computing Measurement Control and Sensor Network (CMCSN)*. IEEE, 2016, pp. 186–189.

[27] C.-H. Wang, W.-H. Chung, C.-J. Lee, and M.-E. Wu, "An atomic routing game for multi-class communication networks with quality of service requirements," in *2015 24th Wireless and Optical Communication Conference (WOCC)*. IEEE, 2015, pp. 206–210.

[28] C.-H. Wang and H. P. Luh, "Analysis of bandwidth allocation on end-to-end QoS networks under budget control," *Computers & Mathematics with Applications*, vol. 62, no. 1, pp. 419–439, 2011.

[29] C.-H. Wang, S. Chen, Q. Zhao, and Y. Suo, "An efficient end-to-end obstacle avoidance path planning algorithm for intelligent vehicles based on improved whale optimization algorithm," *Mathematics*, vol. 11, 1800, no. 8, 2023.

[30] C.-H. Wang, Q. Zhao, and R. Tian, "Short-term wind power prediction based on a hybrid markov-based PSO-BP neural network," *Energies*, vol. 16, 4282, no. 11, 2023.

[31] T.-Y. Wu, H. Li, and S.-C. Chu, "Cppe: an improved phasmatodea population evolution algorithm with chaotic maps," *Mathematics*, vol. 11, 1977, no. 9, 2023.

[32] T.-Y. Wu, A. Shao, and J.-S. Pan, "Ctoa: toward a chaotic-based tumbleweed optimization algorithm," *Mathematics*, vol. 11, 2339, no. 10, 2023.

[33] D. Pisinger and S. Ropke, "A general heuristic for vehicle routing problems," *Computers & Operations Research*, vol. 34, no. 8, pp. 2403–2435, 2007.

[34] J. Xue and B. Shen, "A novel swarm intelligence optimization approach: sparrow search algorithm," *Systems Science & Control Engineering*, vol. 8, no. 1, pp. 22–34, 2020.

[35] T.-T. Nguyen, T.-G. Ngo, T.-K. Dao, and T.-T.-T. Nguyen, "Microgrid operations planning based on improving the flying sparrow search algorithm," *Symmetry*, vol. 14, 168, no. 1, 2022.

[36] T.-K. Dao, T.-T. Nguyen, V.-T. Nguyen, and T.-D. Nguyen, "A hybridized flower pollination algorithm and its application on microgrid operations planning," *Applied Sciences*, vol. 12, 6487, no. 13, 2022.

[37] A. Alvarez and P. Munari, "An exact hybrid method for the vehicle routing problem with time windows and multiple deliverymen," *Computers & Operations Research*, vol. 83, pp. 1–12, 2017.

[38] C. Chen, E. Demir, and Y. Huang, "An adaptive large neighborhood search heuristic for the vehicle routing problem with time windows and delivery robots," *European Journal of Operational Research*, vol. 294, no. 3, pp. 1164–1180, 2021.

[39] S.-W. Lin, F. Y. Vincent, and C.-C. Lu, "A simulated annealing heuristic for the truck and trailer routing problem with time windows," *Expert Systems with Applications*, vol. 38, no. 12, pp. 15 244–15 252, 2011.

[40] J. Luo, X. Li, M.-R. Chen, and H. Liu, "A novel hybrid shuffled frog leaping algorithm for vehicle routing problem with time windows," *Information Sciences*, vol. 316, pp. 266–292, 2015.

[41] E. T. Yassen, M. Ayob, M. Z. A. Nazri, and N. R. Sabar, "Meta-harmony search algorithm for the vehicle routing problem with time windows," *Information Sciences*, vol. 325, pp. 140–158, 2015.

[42] D. Zhang, S. Cai, F. Ye, Y.-W. Si, and T. T. Nguyen, "A hybrid algorithm for a vehicle routing problem with realistic constraints," *Information Sciences*, vol. 394, pp. 167–182, 2017.

[43] A. Thammano and P. Rungwachira, "Hybrid modified ant system with sweep algorithm and path relinking for the capacitated vehicle routing problem," *Heliyon*, vol. 7, no. 9, 2021.

[44] J. C. Figueroa–García, J. S. Tenjo–García, and C. Franco, "A global satisfaction degree method for fuzzy capacitated vehicle routing problems," *Heliyon*, vol. 8, no. 6, 2022.

[45] C.-H. Wang, K. Hu, and X. Wu, "Multi-robot path planning in online dynamic obstacle environments based on parallel cooperative strategy optimization algorithm," *Discover Computing*, vol. 28, p. 132, 2025.

[46] C.-H. Wang, C.-J. Lee, W.-H. Chung, and M.-E. Wu, "An atomic routing game for multi-class communication networks with quality of service requirements," in *Proceedings of the 24th Wireless and Optical Communication Conference (WOCC 2015)*. IEEE, 2015, pp. 206–210.

[47] C.-H. Wang and H. Luh, "A precomputation-based scheme for qos routing and fair bandwidth allocation," *Lecture Notes in Computer Science*, vol. 4297, pp. 595–606, 2006.

[48] ——, "Network dimensioning problems of applying achievement functions," *Lecture Notes in Operations Research – Operations Research and Its Applications*, vol. 6, pp. 35–59, 2006.

[49] K. Corona-Gutiérrez, S. Nucamendi-Guillén, and E. Lalla-Ruiz, "Vehicle routing with cumulative objectives: A state of the art and analysis," *Computers & Industrial Engineering*, vol. 169, 2022, 108054.

[50] C.-H. Wang, K. Hu, X. Wu, and Y. Ou, "Rethinking metaheuristics: Unveiling the myth of "novelty" in metaheuristic algorithms," *Mathematics*, vol. 13, no. 13, p. 2158, 2025.

[51] K. Hu and C.-H. Wang, "Parameter identification of solar photovoltaic cell model based on an enhanced swarm intelligence optimization with adaptive parameter tuning and alternating strategies," *Arabian Journal for Science and Engineering*, pp. 1–27, 2025.

[52] H. Liu and C.-H. Wang, "Seams: A surrogate-assisted evolutionary algorithm with metric-based dynamic strategy for expensive multi-objective optimization," *Expert Systems with Applications*, vol. 265, p. 126050, 2025.

[53] T.-T. Nguyen, T.-K. Dao, and T.-D. Nguyen, "A sensor network coverage planning based on adjusted single candidate optimizer," *Intelligent Automation & Soft Computing*, vol. 37, no. 3, pp. 3213–3234, 2023.

[54] P. Karakostas, A. Sifaleras, and M. C. Georgiadis, "Variable neighborhood search-based solution methods for the pollution location-inventory-routing problem," *Optimization Letters*, vol. 16, no. 1, pp. 211–235, 2022.

[55] T.-K. Dao, T.-D. Nguyen, and V.-T. Nguyen, "An improved honey badger algorithm for coverage optimization in wireless sensor network," *Journal of Internet Technology*, vol. 24, no. 2, pp. 363–377, 2023.

[56] P. Shaw, "Using constraint programming and local search methods to solve vehicle routing problems," in *International Conference on Principles and Practice of Constraint Programming*. Springer, 1998, pp. 417–431.

[57] S. Lin, "Computer solutions of the traveling salesman problem," *Bell System Technical Journal*, vol. 44, no. 10, pp. 2245–2269, 1965.

[58] G. Babin, S. Deneault, and G. Laporte, "Improvements to the or-opt heuristic for the symmetric travelling salesman problem," *Journal of the Operational Research Society*, vol. 58, pp. 402–407, 2007.

[59] J.-Y. Potvin and J.-M. Rousseau, "An exchange heuristic for routeing problems with time windows," *Journal of the Operational Research Society*, vol. 46, no. 12, pp. 1433–1446, 1995.

[60] J.-F. Chen and T.-H. Wu, "Vehicle routing problem with simultaneous deliveries and pickups," *Journal of the Operational Research Society*, vol. 57, pp. 579–587, 2006.