Highlights

Memetic Scheduling of Drones for Railway Catenary Deicing

Yu-Jun Zheng, Xi-Cheng Xie, Zhi-Yuan Zhang, Xin Chen

- A problem of drone scheduling for high-speed railway catenary deicing is presented.
- A memetic algorithm integrating global mutation and variable neighborhood search is proposed.
- The algorithm is validated on real-world instances.

Memetic Scheduling of Drones for Railway Catenary Deicing

Yu-Jun Zheng[a,*], Xi-Cheng Xie[a], Zhi-Yuan Zhang[a], Xin Chen[b]

[a]*School of Information Science and Technology, Hangzhou Normal University, Hangzhou 311121, China*
[b]*Information Engineering College, Hangzhou Dianzi University, Hangzhou Zhejiang 311305, China*

Abstract

Icy rainfall and snowfall in 2024 Spring Festival struck the high-speed railway catenary systems and caused serious traffic disruptions in central and eastern China. Deicing drones are an effective method in response to these freezing events due to their fast speed and high environmental tolerance. However, the large disaster-affected area and the large scale and complexity of catenary networks make deicing drone scheduling a very difficult problem. In this paper, we formulate two versions of deicing drone scheduling problem, one for single drone scheduling and the other for multiple drone scheduling, the objective of which is to minimize the total negative effect caused by the freezing events on train operations. To efficiently solve the problem, we propose a memetic optimization algorithm integrating global mutation and variable neighborhood search. Computational results on real-world problem instances demonstrate its significant performance advantages over selected popular optimization algorithms in the literature.

Keywords: drone scheduling, catenary deicing, memetic optimization, evolutionary algorithms, neighborhood search

1. Introduction

January and February 2024 were the famous Chinese Spring Festival travel rush, during which the railway transportation was subject to heavy traffic. Unfortunately, from January 31 to February 5, a wave of severe blizzards and freezing rain weather swept across 18 provincial-level areas in central and eastern China. Icy rainfall and snowfall froze on wires of the catenary systems over the railway, obstructing power supply of the high-speed trains, forcing the trains to out of action or run at lower speeds, and leaving millions of passengers stranded at railway stations.

*Corresponding author.
Email address: yujun.zheng@computer.org (Yu-Jun Zheng)

Figure 1: A fixed-wing drone equipped with a deicer.

Deicing drones (i.e., unmanned aerial vehicles, UAVs) are an effective method in response to power grid icing [1]. An illustrative example is shown in Fig. 1, where a deicer hanged on a fixed-wing drone slides along a wire to break the snow and ice on it. Compared to human workers carrying deicing equipment, deicing drones have many advantages including fast speed, high work efficiency, accessibility to human-inaccessible or dangerous sites, tolerance under harsh environments, and preventing humans from the danger of electric shock [2].

As a matter of course, the aim of deicing is to restore the catenary network operations as soon as possible and, consequently, minimize the negative effect on the train operations. However, the area by the freezing disaster is large and the deicing workload is high. Moreover, the catenary network structure and its cascading effects on the train operations are complex: when we complete the deicing work on a line of the catenary network, the corresponding railway section can be opened; however, given a train passing through the railway section, if any preceding railway section is not opened, the work does not take effect on the restoration of the train operation. Given the large number of lines (i.e., edges or segments) of the catenary network to be deiced, the problem of deicing drone scheduling is very difficult, especially when there are multiple drones to be scheduled.

To address the above issues, in this study, we formulate two versions of deicing drone scheduling problem, one for single drone scheduling and the other for multiple drone scheduling. The objective of the problem is to minimize the total negative effect caused by the freezing events on train operations, which is evaluated based on the delay or cancellation of trains under the scheduling solution. We propose a memetic optimization algorithm, which integrates global mutation search and variable neighborhood search based on history search knowledge to efficiently solve the problem. We conduct extensive computational tests on real-world problem instances, the results of which demonstrate the performance advantages of our method over a number of selected popular optimization algorithms in the literature. The main contributions of this paper can be summarized as follows:

- We formulate a practical problem of drone scheduling for railway catenary

deicing to minimize the negative effect on train operations.

- We propose a memetic algorithm integrating global mutation search and variable neighborhood search based on history search knowledge to efficiently solve the problem.
- We validate the performance of the proposed algorithm through extensive computational tests on real-world instances.

The remainder of this paper is organized as follows. Section 2 discusses related work, Section 3 presents the deicing drone scheduling problem, Section 4 proposes the memetic optimization algorithm, Section 5 presents the computational results, and Section 6 concludes.

2. Related Work

Deicing (written as de-icing in some literature) is one of the most important techniques for restoring power networks stricken by icing disasters. Current ice-melting and mechanical deicing are two most widely used techniques. Current ice melting uses the thermal effect of overcurrent/short-current on the wire to melt the ice. Although having the advantages of high efficiency, current ice melting is power consuming and, more importantly, requires to cut the power transmission during the deicing work [3]. Mechanical deicing, by contrast, uses deicing devices attached to the conductors or wires to induce sustained vibrations to shed ice, which is easier to use but more time-consuming than current ice melting [4] when deicers are carried by human or ground vehicles. The emerging drone technologies significantly improve the applicability and efficiency of mechanical deicing by using drones to carry deicers in high speed and to sites that are difficult to access by human or ground vehicles.

For current ice melting scheduling, Huneault et al. [5] proposed a dynamic programming method to determine deicing strategies by minimizing ice buildup on power lines during ice storms over the set of scenarios and over the time horizon spanning the anticipated duration of the ice storm. Hou et al. [6] presented a multi-objective deicing outage scheduling with the aim to minimize the ice thickness peak value of icing lines and the expected energy not supplied of the system; they used the non-dominated sorting genetic algorithm (NSGA-II) to find the Pareto optimal solutions, from which the final deicing schedule is chosen using a decision making method.

Nevertheless, few research works have been devoted to mechanical deicing scheduling considering the use of drones, although there are some studies on scheduling drones for power network failure detection restoration. Deng et al. [7] proposed a multi-platform drone system, in which different types of drones perform different functionalities including long-distance and short-distance imaging and communication relay in power line inspection. Considering the use of industrial Internet of drones for power line inspection, Zhou et al. [8] formulated the energy consumption minimization as a joint optimization problem involving trajectory scheduling, velocity control, relay selection, and power allocation; they

transformed the problem into a two-stage suboptimal problem and solved it by combining dynamic programming, auction theory, and matching theory. Zheng et al. [2] studied a cooperative human-drone scheduling problem, where drones are used to inspect faults, which are subsequently repaired by human technicians; the authors proposed a cooperative evolutionary algorithm that simultaneously evolves drone scheduling sub-solutions and human-team scheduling sub-solutions to obtain an optimal or near-optimal integrated solution. Atat et al. [9] proposed an optimization framework for post-disaster drone-based damage assessment of overhead power lines, which optimizes the drone flight paths to inspect the critical loads in an efficient order to minimize the total inspection time while considering drone battery recharging.

Scheduling drones to perform deicing work on icing lines, from an abstract perspective, can be regarded as a special version of vehicle routing problem (VRP) by regarding drones as vehicles and icing lines as customers. The special characteristics include that each customer (line) has a service (deicing) time, the battery capacity of drones should be explicitly considered, and the objective value should be evaluated based on the train delay and cancellation under the deicing solution rather than simply based on makespan. Dorling et al. [10] presented two multi-trip VRPs for drone delivery integrating an energy consumption model, one minimizing costs and the other minimizing the overall delivery time, which were solved by a simulated annealing heuristic for finding suboptimal solutions. Hong et al. [11] developed a method for optimizing delivery routes of drones together with a network of recharging station locations by integrating planar-space routing, range-restricted flow-refueling location, and maximal coverage location. Zheng et al. [12] proposed an evolutionary algorithm integrating comprehensive learning, variable mutation, and local search to solve a problem of collaborative human-drone search for escaped criminals, the aim of which is to minimize the expected time of capture rather than detection. Pachayappan and Sudhakar [13] presented a problem of planning optimal pickup and delivery routes of a drone, which is synchronized with a docking station for recharging during the services; they proposed a heuristic solution method for the problem. Gómez-Lagos et al. [14] studied a pickup-to-delivery drone routing problem for finding a drone scheduling such that a drone serves the customer's order from a set of available facilities, with the objective to minimize the makespan associated with the drone fleet; they designed a greedy randomized adaptive search procedure to find near-optimal solutions. Wen and Wu [15] proposed a two-echelon heterogeneous multi-drone routing problem for parcel delivery, which was solved by a three-stage iterative optimization algorithm based on the divide and conquer. Guo et al. [16] formulated a routing model of electric trackless rubber-tyred vehicles to minimize the total energy consumption under the constraint of vehicle avoidance, allowable load, and endurance power; they proposed an improved artificial bee colony algorithm integrating adaptive neighborhood search to solve the problem. Considering coverage path planning of heterogeneous drones to visit and search multiple separated regions, Chen et al. [17] presented an ant colony system based algorithm to seek approximately optimal solutions and minimize the time consumption of tasks.

Wang et al. [18] presented a problem of delivery route planning for a fleet of drones with different capacities, speeds, and maximum flight ranges; they proposed a modified, rescheduling-based genetic algorithm to search optimal or near-optimal solutions. To coordinate ground vehicles and drones to monitor urban road networks, Xu et al. [19] proposed a hybrid variable neighborhood descent search and simulated annealing algorithm for vehicle-drone arc routing. In [20], Zheng et al. studied a problem of scheduling multiple vehicles to replace and recycle batteries to keep a high operational efficiency of the shared e-bicycle system, which was efficiently solved by an evolutionary algorithm based on variable local-search-based mutation. Recently, Fan et al. [21] presented a deep reinforcement learning method, where a multi-head heterogeneous attention mechanism is designed to facilitate learning a policy for constructing the drone route in the presence of multiple charging stations. The problems studied by the above work have some characteristics similar to the problem considered in this paper, but none of them shares most key characteristics with our problem.

3. Problem Formulation

The problem is schedule one or multiple deicing drones to remove ices on wires of a railway catenary network. A catenary network hit by ices can be represented by a graph $G = \langle V, E \rangle$, where V is the set of vertices and E is the set of edges. Here, V contains not only joints connecting power grid lines, but also points dividing lines into segments such that different segments in a line have different freezing conditions. Typically, two segments can be separated according to obvious differences in altitude, temperature, humidity, wind force, etc. The set E can be divided into two subsets: $E^\dagger$ where the edges (segments) need to be deiced (otherwise the corresponding rail segments are impassable), and $E \backslash E^\dagger$ where the edges (such as segments in stations and tunnels) do not need to be deiced. For each $e \in E$, $A(e)$ denotes the amount of deicing work on the segment e ($A(e) = 0$ if $e \notin E^\dagger$). For convenience, we also use $V^\dagger$ to denote the set of vertices of the edges in $E^\dagger$. For each pair of vertices $u_i, u_j \in V$, the distance of the shortest drone path from u_i to u_j is known as $d(u_i, u_j)$. In the following two subsections, we present the two versions of the problem of deicing drone scheduling for removing the ices on the catenary network G, one using a single drone and the other using multiple drones.

3.1. Single-Drone Scheduling for Catenary Deicing

First we consider scheduling a single drone for catenary deicing. A scheduling solution can be represented by a vector $X = \{x_1, x_2, \ldots, x_n\}$, where $n = |E^\dagger|$ is the number of segments that need to be deiced, each component x_i is a tuple (x_i^u, x_i^δ), where $x_i^u \in V^\dagger$ is the starting vertex of the ith segment to be deiced, and $x_i^\delta \in \{0, 1\}$ is the direction (left or right) towards which the drone will deice from x_i^u ($1 \le i \le n$). From each component $x_i = (x_i^u, x_i^\delta)$, we can obtain the corresponding segment $e(x_i)$ to be deiced. Let u_0 be the initial location of the drone, B_0 be the full battery power of the drone, v be the velocity of the drone,

a be the amount of deicing work the drone can complete per unit time, b be the amount of battery power per unit mileage consumed by the drone in flying, and $b^\dagger$ be the amount of battery power per unit work consumed by the drone in deicing. It is assumed that the full battery power is sufficient for the drone to fly to any vertex and complete the deicing work on any segment. The time at which the deicing work on the first segment $e(x_1)$ is completed can be computed as

$$t_c(x_1) = \frac{d(u_0, x_1^u)}{v} + \frac{A\big(e(x_1)\big)}{a} \tag{1}$$

And the remaining battery power of the drone after the completion of the work on $e(x_1)$ is

$$B(x_1) = B_0 - b{\cdot}d(u_0, x_1^u) - b^\dagger{\cdot}A\big(e(x_1)\big) \tag{2}$$

After the completion of the work on each segment $e(x_i)$, the drone is located at the other endpoint of $e(x_i)$, which is denoted by $u(x_i)$ here. We also use $D(u(x_i))$ to denote the battery depot closest to $u(x_i)$, and the power required for the drone to fly from $u(x_i)$ to $D(u(x_i))$ is

$$\underline{B}(x_i) = b{\cdot}d\big(u(x_i), D(u(x_i))\big) \tag{3}$$

While the power required by the drone to directly fly to and complete the work on the next segment is

$$\widetilde{B}(x_{i+1}) = b{\cdot}d\big(u(x_i), x_{i+1}^u\big) + b^\dagger{\cdot}A\big(e(x_{i+1})\big) \tag{4}$$

At each $u(x_i)$, the drone can directly go to the next segment if the remaining battery power is not below the safe level $\widetilde{B}(x_{i+1})+\underline{B}(x_{i+1})$, otherwise it should go to $D(u(x_i))$ to replace the battery and then continue the remaining tasks. Therefore, the time at which the deicing task on each segment is completed can be iteratively computed as follows ($1<i\leq n$):

$$t_c(x_{i+1}) = \begin{cases} t_c(x_i) + \frac{d\left(u(x_i), x_{i+1}^u\right)}{v} + \frac{A(e(x_{i+1}))}{a}, & B(x_i) \geq \widetilde{B}(x_{i+1}) + \underline{B}(x_{i+1}) \\ t_c(x_i) + \frac{d\left(u(x_i), D(u(x_i))\right) + d\left(D(u(x_i)), x_{i+1}^u\right)}{v} + t^{\mathrm{R}} + \frac{A(e(x_{i+1}))}{a}, & \text{otherwise} \end{cases} \tag{5}$$

where t^{R} denotes the time duration for replacing the battery, including the time for landing and takeoff (we assume that the number of fully-charged batteries is sufficient at each depot).

And the remaining battery power $B(x_i)$ of the drone after the completion of the work on $e(x_i)$ can be iteratively computed as follows ($1<i\leq n$):

$$B(x_{i+1}) = \begin{cases} B(x_i) - b{\cdot}d\big(u(x_i), x_{i+1}^u\big) - b^\dagger{\cdot}A(e(x_{i+1})), & B(x_i) \geq \widetilde{B}(x_{i+1}) + \underline{B}(x_{i+1}) \\ B_0 - b{\cdot}d\big(D(u(x_i)), x_{i+1}^u\big) - b^\dagger{\cdot}A(e(x_{i+1})), & \text{otherwise} \end{cases} \tag{6}$$

Based on the completion time of deicing work on each segment, we can

obtain the set of segments that have been deiced at time t, denoted by $E^{\ddagger}(t)$, as follows:

$$\begin{aligned} E^{\ddagger}\big(t_c(x_1)\big) &= \big\{e(x_1)\big\} \\ E^{\ddagger}\big(t_c(x_2)\big) &= E^{\ddagger}\big(t_c(x_1)\big) \cup \big\{e(x_2)\big\} \\ &\vdots \\ E^{\ddagger}\big(t_c(x_{i+1})\big) &= E^{\ddagger}\big(t_c(x_i)\big) \cup \big\{e(x_{i+1})\big\}, \quad 1<i\leq n \end{aligned} \tag{7}$$

Then, for each block section s of the railway network affected by ices, let $\Phi(s)$ be the set of power gird line segments along the section s, that is, the section could not be opened unless all segments in $\Phi(s)$ have been deiced; the earliest open time of s under the scheduling solution X can be calculated as

$$t_o(s,X) = \min_{1\leq i\leq n} \Big\{ t_c(x_i) | \Phi(s) \subseteq E^{\ddagger}\big(t_c(x_i)\big) \Big\} \tag{8}$$

According to the train timetable, we have the set $\Gamma(s)$ of trains that will enter the railway section s, the route of each train r represented as a sequence of consecutive railway sections $\{s_1^r, s_2^r, \ldots, s_{K_r}^r\}$, and the time $t_e(r, s_k^r)$ at which the train r plans to enter the section s ($\forall r \in \Gamma(s), k = 1, 2, \ldots, K_r$), where K_r is the number of sections in the route of train r. Let $\Delta t(s)$ denote the time duration for the train to pass through section s; based on the timetable and deicing schedule, we can obtain the actual time $t_e^{\ddagger}(r, s, X)$ at which the train r can enter each section s in its route as

$$t_e^{\ddagger}(r, s_k^r, X) = \begin{cases} \max\big(t_e(r, s_k^r), t_o(s_k^r, X)\big), & k = 1 \\ \max\big(t_e(r, s_k^r), t_o(s_k^r, X), t_e^{\ddagger}(s_{k-1}^r, X) + \Delta t(s_{k-1}^r)\big), & k > 1 \end{cases} \tag{9}$$

Remark 1: Eq. (9) is a simplified equation that does not involve the contradictions among the trains. In case that multiple trains need to enter one section in a short period, we first determine the priority of each train according to the grade of the train, the time duration that the train has already delayed, and the cascading effect of the further delay on other trains; then we arrange the trains to enter the section in the order of priority, while keeping a minimum safe distance between any two consecutive trains (see [22] for more details).

We use the following function to evaluate the negative effect of the freezing events on each train r for traveling each railway section s in its route under the scheduling solution X:

$$\psi(r, s, X) = \begin{cases} 0, & t_e^{\ddagger}(r, s, X) \leq t_e(r, s) \\ w_{r,s}\big(t_e^{\ddagger}(r, s, X) - t_e(r, s)\big), & t_e(r, s) < t_e^{\ddagger}(r, s, X) < T_{\text{close}} \\ \rho_{r,s} w_{r,s}, & t_e^{\ddagger}(r, s, X) \geq T_{\text{close}} \end{cases} \tag{10}$$

where $w_{r,s}$ is the importance weight of the train r in the railway section s, which is determined according to the grade of the train, the number of the passengers

on the train, and the number and importance of the stations in the section, T_{close} is the closing time of the current operational duration (which is typically 24:00:00 of the current day, but can be prolonged by the decision-maker under special conditions), and $\rho_{r,s}$ is the penalty coefficient regarding the cancellation of the train r on the section s.

Remark 2: if a train is canceled on a section s, it is automatically canceled on the subsequent sections along its route, but it can still complete its operations on sections before s.

The aim of the problem is to minimize the total negative effect evaluated in terms of the delay or cancellation of trains caused by the freezing catenary network, which can be represented as follows:

$$\min f(X) = \sum_{s\in S}\sum_{r\in\Gamma(s)} \psi(r,s,X) \tag{11}$$

$$\text{s.t.} \quad \Big(\bigcup_{1\le i\le n}\{e(x_i)\}\Big) = E^{\dagger} \tag{12}$$

$$e(x_i)\bigcap e(x_{i'}) = \emptyset, \quad \forall 1\le i<i'\le n \tag{13}$$

$$\text{Eqs. (1)–(10)}$$

where S is the set of railway sections affected by freezing segments, and the constraints (12) and (13) indicate that each segment in $E^{\dagger}$ should be deiced once and only once.

3.2. Multi-Drone Scheduling for Catenary Deicing

Given m drones, a scheduling solution $X = \{x_{1,1},\ldots,x_{1,n_1},x_{2,1},\ldots,x_{2,n_2},\ldots,x_{m,1},\ldots,x_{m,n_m}\}$ consists of m parts, where n_j is the number of segments assigned to the jth drone, satisfying $\left(\sum_{j=1}^{m} n_j\right) = n$. For each jth part $\{x_{j,1},\ldots,x_{j,n_j}\}$, we can use the procedure similar to Eqs. (1)–(6) to iteratively calculate completion time of the deicing work on each segment $e(x_{j,i})$, where $1\le i\le n_j$. Then, by combining the results of the m drone sub-schedules, we can calculate the open time for each freezing segment and then evaluate the objective function for minimizing the effect of delay or cancellation of trains using the procedure shown in Algorithm 1.

4. Memetic Optimization to Solve the Problem

We propose a memetic optimization algorithm integrating global search and variable neighborhood search to solve the problem. In the following two subsections, we describe the algorithm versions for single drone scheduling and multiple drone scheduling, respectively.

4.1. Memetic Optimization for Single Drone Scheduling

For single drone scheduling, each initial solution is first generated as a random permutation of the set $E^{\dagger}$ of freezing segments to be deiced. Then, each

Algorithm 1: Procedure of calculating the objective function value of any scheduling solution X based on the m drone sub-schedules.

Sort all completion times $t_c(x_{j,i})$ in non-decreasing order;
foreach *completion time* $t_c(x_{j,i})$ *in the order* **do**
    Calculate $E^{\ddagger}\big(t_c(x_{j,i})\big)$ as the union of the set $E^{\ddagger}$ at the previous time and $\{e(x_{j,i})\}$;
    foreach *railway section* $s \in S$ **do**
        if $\Phi(s) \subseteq E^{\ddagger}\big(t_c(x_{j,i})\big)$ **then**
            Set $t_o(s, X) = t_c(x_{j,i})$;
            Remove s from S;
Sort all trains in the timetable in non-increasing order of their priorities;
foreach *train* r *in the timetable* **do**
    Set $t_e^{\ddagger}(r, s_1, X) = t_o(s_1, X)$ for the first segment s_1 in the route of r;
    foreach *subsequent railway section* s_k *in the route of* r **do**
        Set $t_e^{\ddagger}(r, s_k, X) = \max\big(t_e(r, s_k), t_o(s_k, X), t_e^{\ddagger}(s_{k-1}, X) + \Delta t\big(s_{k-1}\big)\big)$;
        if $t_e^{\ddagger}(r, s_k, X) = t_o(s_k, X)$ *and there is already at least one train* r' *prior to* r *enters* s_k *at time* $t_o(s_k, X)$ **then**
            set $t_e^{\ddagger}(r, s_k, X)$ to the time at which r' leaves the block section s_k;
Calculate the objective function value $f(X)$ according to Eqs. (10) and (11).

solution has a probability of p_I (a control parameter) of being tentatively improved using the following procedure:

1. For the first segment e_1 in the permutation, let x_1^u be the endpoint closer to the initial location u_0 and let x_1^δ towards the other endpoint of e_1.
2. For $i = 2$ to n do:
    (a) Calculate the remaining battery power of the drone at the other endpoint $u(x_{i-1})$;
    (b) If the remaining battery power is sufficient for the drone to complete the work on e_i, let x_i^u be the endpoint closer to $u(x_{i-1})$; otherwise, let x_i^u be the endpoint closer to the battery depot $D\big(u(x_{i-1})\big)$;
    (c) let x_i^δ towards the other endpoint of e_i.

Note that the above tentative improvement procedure does not guarantee improving the solution fitness, because making x_i^u closer to $u(x_{i-1})$ may cause $u(x_i)$ farther from the next (i+1)th segment or farther from the next depot. Nevertheless, it have a large probability of success: according to our test, approximately 80% random solutions can be improved by this procedure. Consequently, we set the initial value of p_I to 0.8. If the procedure successfully improves the solution, the updated solution replaces the initial solution; otherwise, the initial solution is kept in the population.

The global search is based on the water wave optimization metaheuristic [23] that assigns each solution a wavelength proportional to the objective function value, i.e., inversely proportional to the fitness of the solution, and mutates the solution with a degree proportional to the wavelength, such that low-fitness solutions tend to explore in large spaces while high-fitness solutions facilitate exploitation in small regions to achieve a good dynamic balance between diversification and intensification. The wavelength of each solution X is initialize to 0.5 and then updated at each generation as

$$\lambda(X) = \lambda(X) \cdot \alpha^{-(f_{\max} - f(X) + \epsilon)/(f_{\max} - f_{\min} + \epsilon)} \tag{14}$$

where $f_{\max}$ and $f_{\min}$ are the maximum and minimum objective function values among the population, respectively, α is the wavelength reduction coefficient which is suggested to be 1.0026, and ϵ is a very small number to avoid division by zero.

At each generation, each solution X is mutated as follows: for $i = 1$ to n, with a probability of $\lambda(X)$, reverse a random subsequence of X; afterwards, with a probability of p_I, use the tentative improvement procedure described above to adjust the start point and direction of each component. The value of p_I decreases with generation number g as follows (where $g_{\max}$ is the maximum number of generations of the algorithm):

$$p_I(g) = \frac{g_{\max} - 0.5g}{g_{\max}} p_I(0) \tag{15}$$

After each generation, for any newly updated solution X whose fitness is in the first half of the population, the algorithm performs a neighborhood search around the solution. Here, we consider the following three neighborhood search operators:

NS1. Continuously exchange two randomly selected components of X until the fitness is improved or the number of exchange reaches an upper limit $n^2/4$.

NS2. Continuously select a randomly component and reinsert it into another random position of X until the fitness is improved or the number of reinsertion reaches an upper limit $n^2/4$.

NS3. Continuously reverse a randomly selected subsequence of X until the fitness is improved or the number of reversion reaches an upper limit n.

The algorithm adaptively selects among three neighborhood search operators based on their past performances during the search. Initially, the selection probability of each operator is the same; at each generation, the selection probability $p_N(i)$ of each i-th operator is updated to be proportional to $N_{LP}^{(i)}$, the number of new best solutions produced by the operator in the previous LP

iterations, where LP is a parameter controlling the learning period ($i = 1, 2, 3$):

$$p_N(i) = \frac{N_{LP}^{(i)}}{N_{LP}^{(1)} + N_{LP}^{(2)} + N_{LP}^{(3)}} \tag{16}$$

If a solution X has not been improved for consecutive K_G generations (where K_G is a control parameter typically set to 6), it will be replaced by a new solution generated as follows:

1. Randomly select an exemplar solution X^* from the first half of the population.
2. For $i = 1$ to n do: with a probability of $\lambda(X)$, reverse a random subsequence of X^*.
3. With a probability of p_I, use the tentative improvement procedure to adjust the start point and direction of each component.

That is, the new solution is generated by mutating the exemplar solution X^*, but the mutation intensity is proportional to $\lambda(X)$ rather of $\lambda(X^*)$.

Algorithm 2 presents the pseudo-code of the memetic optimization algorithm for single drone scheduling.

4.2. Memetic Optimization for Multiple Drone Scheduling

In this subsection, we extend the above algorithm for multiple drone scheduling. To randomly initialize a solution, the algorithm first randomly divides the set $E^\dagger$ of freezing segments into m parts, each being assigned to one drone; then, for each jth part, the algorithm randomly generates a sub-schedule of the jth drone using the procedure of solution initialization for single drone scheduling described in the above subsection.

The global search (mutation) on a multiple drone scheduling solution X consists of two steps, one for moving components among sub-schedules and one for perform permutation within sub-schedules:

1. For $j = 1$ to $m^2/4$ do:
   (a) Randomly select two drone sub-schedules, the indices of which denoted by $j1$ and $j2$;
   (b) Randomly select a subset C of segments from the $j1$-th sub-schedule and move it into the $j2$-th sub-schedule, where the cardinality of subset is at most $n_{j1}/2$;
2. For $j = 1$ to m do:
   (a) For $i = 1$ to $n_j/2$, with a probability of $\lambda(X)$, reverse a random subsequence of the jth sub-schedule;
   (b) With a probability of p_I, use the tentative improvement procedure to the components of the jth sub-schedule.

Algorithm 2: Memetic optimization algorithm for single drone scheduling.

```
Randomly initialize a population of solutions;
while the stopping condition is not met do
    Calculate the wavelengths of the solutions;
    foreach solution X in the population do
        for i = 1 to n do
            if rand() < λ(X) then
                Reverse a random subsequence of X;
        Tune the start points and directions of the components of X;
        if the objective function is improved then
            Replace X with the mutated solution;
    foreach solution X in the first half of the population do
        if X is updated in this generation then
            Randomly select a neighborhood search operator according to the
             selection probability;
            Apply the operator to search around X;
            if the objective function of the neighboring solution is better then
                Replace X with the neighboring solution;
        else if X has not been improved for K_G generations then
            Randomly select an exemplar solution X* from the first half of
             the population;
            for i = 1 to n do
                if rand() < λ(X) then
                    Reverse a random subsequence of X*;
            Tune the start points and directions of the components of X;
            Replace X with the solution mutated from X*;
    Update N_LP^(i) and p_N(i) of the neighborhood search operators according to
     Eq. (16);
return the best-known solution.
```

Variable neighborhood search for multiple drone scheduling are extended to the following four operators :

NS1. Randomly select $m/2$ sub-schedules and for each jth sub-schedule, continuously exchange two randomly selected components of the sub-schedule until the fitness is improved or the number of exchange reaches an upper limit $n_j^2/4$.

NS2. Randomly select $m/2$ sub-schedules and for each jth sub-schedule, continuously select a randomly component and reinsert it into another random position of the sub-schedule until the fitness is improved or the number of reinsertion reaches an upper limit $n_j^2/4$.

NS3. Randomly select $m/2$ sub-schedules and for each jth sub-schedule, continuously reverse a randomly selected subsequence of the sub-schedule until the fitness is improved or the number of reversion reaches an upper limit n_j.

NS4. For $i = 1$ to $m^2/4$, randomly select two sub-schedules and for each pair of sub-schedules, continuously move a component from one sub-schedule to another until the fitness is improved or the number of movement reaches an upper limit $\min(n_{j1}, n_{j2})$.

The adaptive selection among the four operators is similar to that for single drone scheduling.

Similarly, if a solution X has not been improved for consecutive K_G generations, it will be replaced by a new solution generated by mutating exemplar solution X^* randomly select an from the first half of the population with an intensity proportional to $\lambda(X)$.

Algorithm 3 presents the framework of the memetic optimization algorithm for multiple drone scheduling.

5. Computational Results

5.1. Computational Results on Single Drone Scheduling

Single drone is mainly used for relatively small-scale catenary deicing. For testing the single deicing drone scheduling problem, we select eight problem instances from high-speed railway catenary networks in Hubei province, central China. Table 1 describes the basic information of the test instances, where $\sum_{e \in E} d(e)$ denotes the total length (in km) of the whole catenary work, $\sum_{e \in E^\dagger} d(e)$ denotes the total length of segments to be deiced (which represents the total workload of deicing), and $|\Gamma|$ denotes the number of trains in the timetable of the considered railway network.

The solution structure of single deicing drone scheduling is similar to that of the permutation flow-shop scheduling problem (FSP), except that our problem needs to determine an additional direction at each component and arrange the

Algorithm 3: Memetic optimization algorithm for multiple drone scheduling.

```
Randomly initialize a population of solutions;
while the stopping condition is not met do
    Calculate the wavelengths of the solutions;
    foreach solution X in the population do
        Perform a global mutation on X with an intensity of λ(X);
        if the objective function is improved then
            Replace X with the mutated solution;
    foreach solution X in the first half of the population do
        if X is updated in this generation then
            Randomly select a neighborhood search operator according to the
              selection probability;
            Apply the operator to search around X;
            if the objective function of the neighboring solution is better then
                Replace X with the neighboring solution;
        else if X has not been improved for K_G generations then
            Randomly select an exemplar solution X* from the first half of
              the population;
            Replace X with a new solution mutated from X* with an
              intensity of λ(X);
    Update the selection probabilities of the neighborhood search operators;
return the best-known solution.
```

Table 1: Summary of the test instances of single deicing drone scheduling.

#	$\|V\|$	$\|E\|$	$\|E^\dagger\|$	$\sum_{e\in E} d(e)$	$\sum_{e\in E^\dagger} d(e)$	$\|\Gamma\|$
1	16	35	32	37.1	28.6	39
2	18	50	42	44.9	35.6	37
3	21	77	65	72.3	66.1	45
4	27	124	103	115.8	98.5	42
5	26	158	149	142.0	122.7	49
6	36	211	188	185.4	156.5	51
7	48	406	356	340.6	314.0	54
8	60	673	512	590.8	486.6	73

drone to visit depots for battery replacement if its power is insufficient. For comparison, we adapt the following 12 efficient metaheuristic and memetic optimization algorithms for FSP to our problem by adding the additional direction part to solution representation as well as performing the tentative improvement procedure for each new solution with the same probability control mechanism as our memetic algorithm (denoted by Mem):

- An adaptive order-based GA (AGA) with multiple operators [24].
- A discrete particle swarm optimization (DPSO) algorithm [25].
- A discrete cuckoo search algorithm (DCSA) [26].
- A teaching-learning-based optimization (TLBO) algorithm [27].
- A hybrid ant colony optimization (HACO) algorithm [28].
- A biogeography-based optimization (BBO) algorithm using ecogeography-based migration [29] which outperforms other BBO variants [30].
- A memetic algorithm combining GA using semi-constructive crossover (MASC), simulated annealing, and Nawaz–Enscore–Ham (NEH) [31].
- A discrete self-adaptive differential evolution (DSADE) algorithm [32].
- A multi-local search-based variable neighborhood search (MVNS) algorithm [33].
- Three versions of the memetic algorithm based on our global mutation but only with single neighborhood search operators, NS1, NS2, and NS3, denoted by Mem-NS1, Mem-NS2, and Mem-NS3, respectively.

For a fair comparison, we set the same stopping condition that number of objective function evaluations reaches $300n$ for all algorithms. Each algorithm is independently run 51 times with random seeds on each instance. Fig. 2 – Fig. 9 present the box plots, including the mean (shown in green triangle) median, maximum, minimum, the first quartile (Q1) and the third quartile (Q3) values

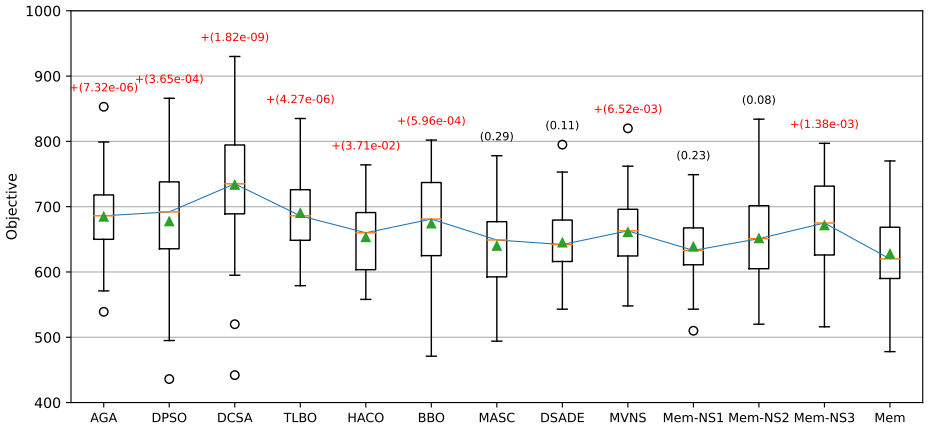

Figure 2: Box plots of the results obtained by the 13 comparative algorithms on single drone scheduling instance #1.

obtained by the comparative algorithms on the eight instances, respectively. Any objective value below the lower limit Q1$-$1.5(Q3$-$Q1) or above the upper limit Q3$+$1.5(Q3$-$Q1) is considered as an outlier. We conduct nonparametric Wilcoxon rank sum test on the result obtained by our memetic algorithm and the result obtained by each other algorithm on each instance, and present the resulting p-value above the maximum value of the corresponding box and, if the value is smaller than 0.05, mark a red '+' before the p-value to indicate there is a statistically significant difference at a confidence level of 95%.

On the first two instances #1 and #2, except that the median value of DCSA is obviously larger, the median values of the other 12 algorithms are relatively close. According to the rank sum test results, on instance #1, there are no significant differences between the result of Mem and those of MASC, DSADE, Mem-NS1 and Mem-NS2, while the result of Mem is significantly better than those of the other eight algorithms; on instance #2, the result of Mem is significantly better than those of AGA, DCSA, MVNS, and Mem-NS3, while there are no significant differences between Mem and the other seven algorithms. The remaining instances become larger and more difficult, and the performance advantages of the proposed memetic algorithm become more obvious. Considering the nine algorithm from the literature (expect the three versions of our algorithm with single neighborhood search operators), the result of Mem is not significantly different from two algorithms (HACO and DSADE) on instance #3, one algorithm (DSADE) on instance #4, and one algorithm (MVNS) on instance #5 and #6, but is significantly better than other algorithms on these instances. On the last and most two difficult instances #7 and #8, Mem performs significantly better than all these nine algorithms. On each instance, the proposed Mem algorithm uniquely obtains the best median value among all comparative algorithms.

In general, on relatively small-size instances, metaheuristic algorithms such

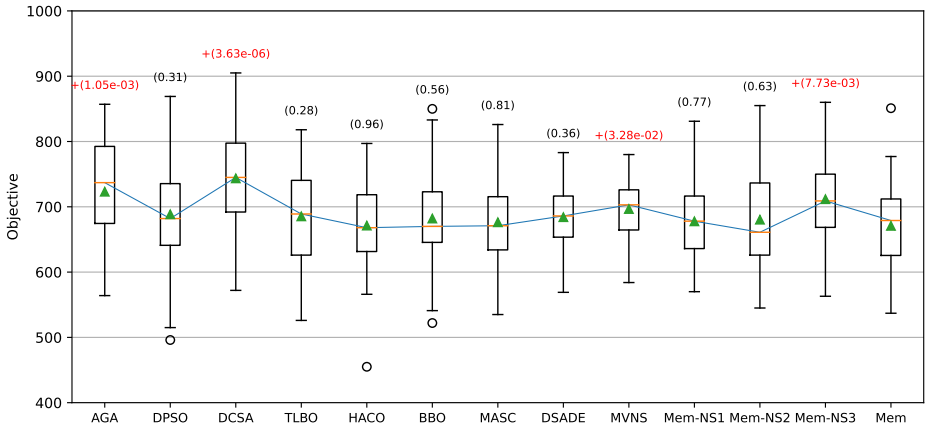

Figure 3: Box plots of the results obtained by the 13 comparative algorithms on single drone scheduling instance #2.

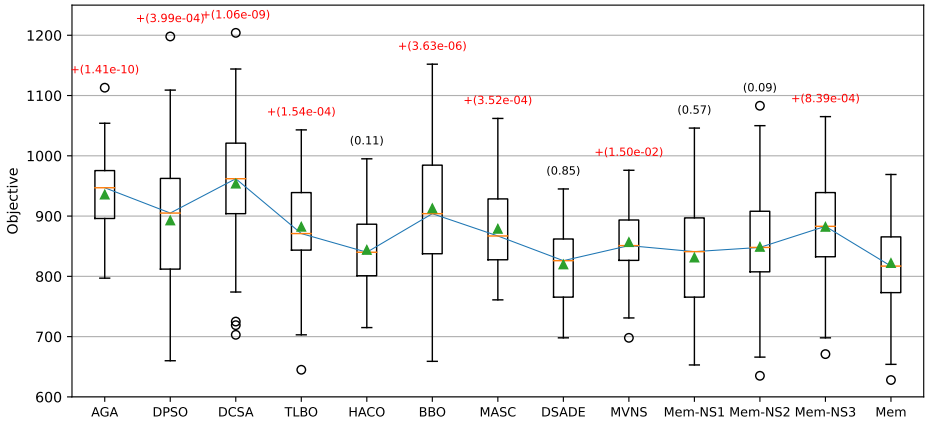

Figure 4: Box plots of the results obtained by the 13 comparative algorithms on single drone scheduling instance #3.

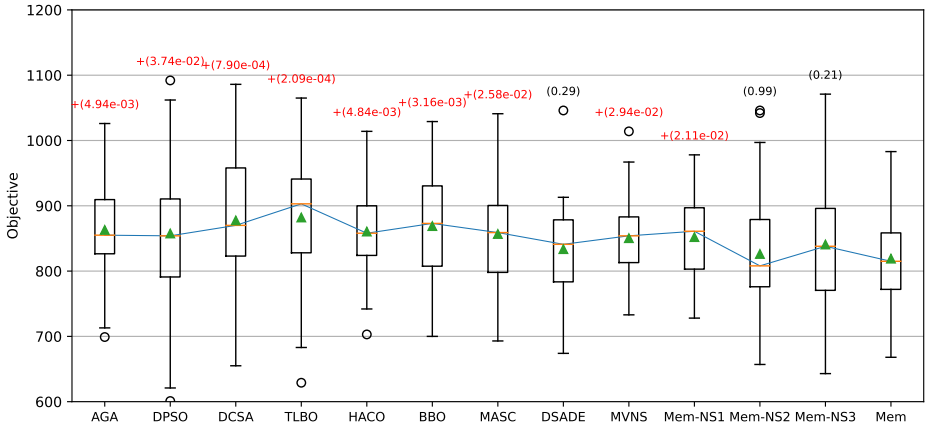

Figure 5: Box plots of the results obtained by the 13 comparative algorithms on single drone scheduling instance #4.

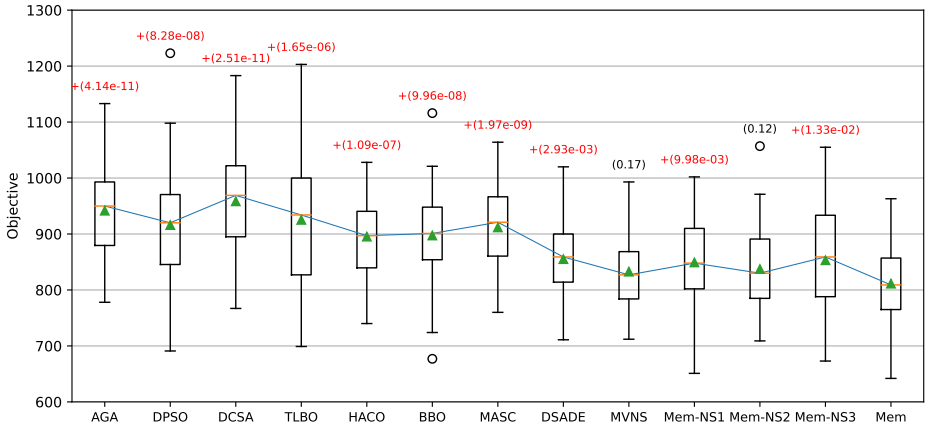

Figure 6: Box plots of the results obtained by the 13 comparative algorithms on single drone scheduling instance #5.

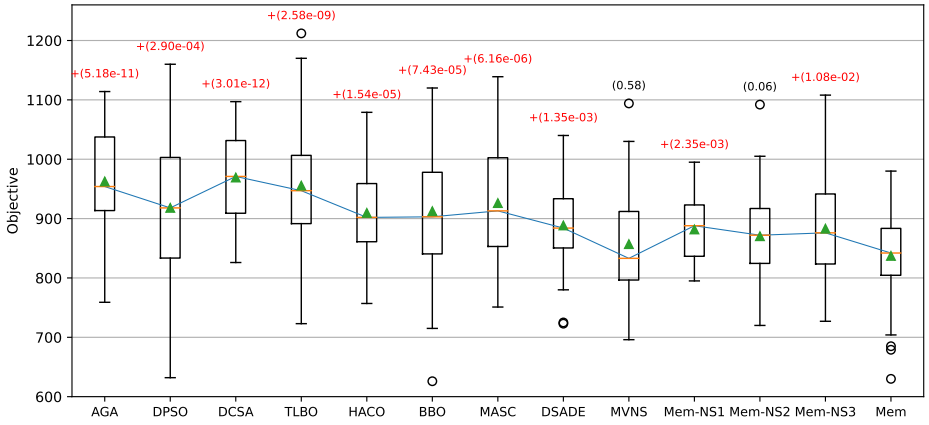

Figure 7: Box plots of the results obtained by the 13 comparative algorithms on single drone scheduling instance #6.

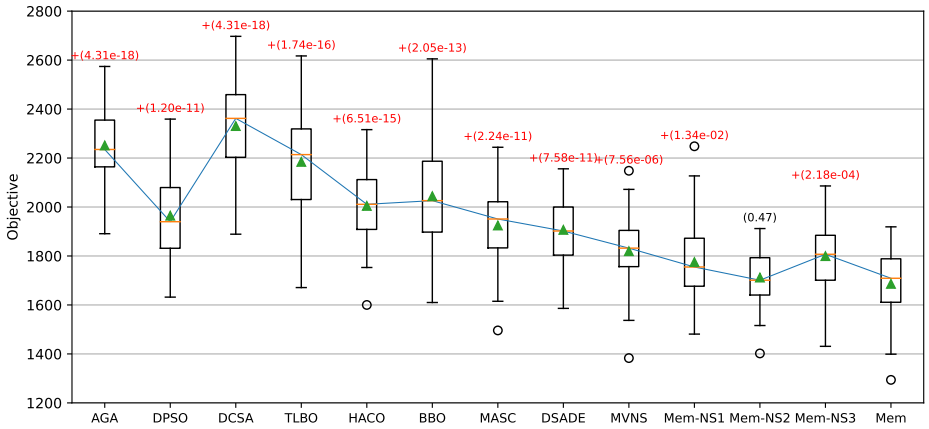

Figure 8: Box plots of the results obtained by the 13 comparative algorithms on single drone scheduling instance #7.

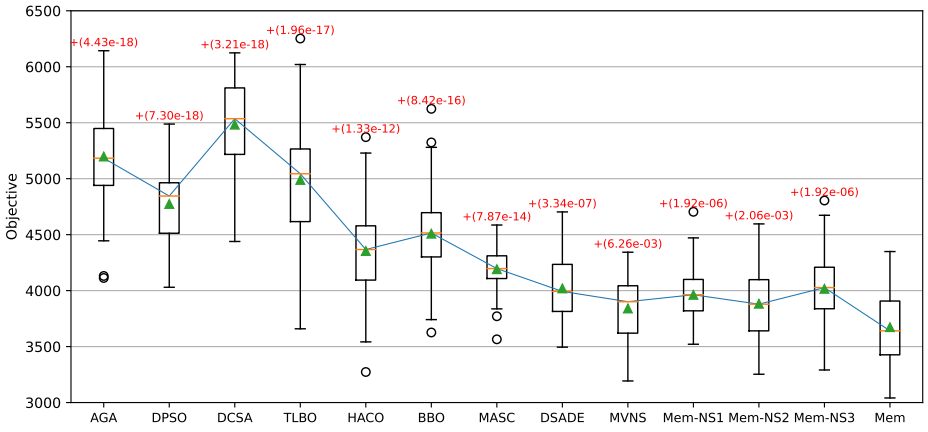

Figure 9: Box plots of the results obtained by the 13 comparative algorithms on single drone scheduling instance #8.

as DPSO, HACO, and DSADE exhibit relatively good performance because their designs aim to achieve balance between global search and local search. In particular, the self-adaptive differential mutation mechanism of DSADE shows relatively strong search ability. However, for more difficult instances, they lack strong local search abilities to improve solution accuracies; in comparison, those algorithms combining metaheuristic and neighborhood search exhibit more higher performance. In particular, MVNS and our Mem algorithm using variable neighborhood search exhibit significantly better performance than the algorithms using a fixed neighborhood search operator.

Among the three versions of our algorithm with single neighborhood search operators, the performances of Mem-NS1 and Mem-NS2 are generally better than that of Mem-NS3, which indicates that the component swap and reinsertion operators are more effective than the subsequence reversion operator for the considered problem. Nevertheless, by integrating the three neighborhood search operators in an adaptive selection schema, the proposed Mem algorithm exhibits significantly better performance than all three versions with single neighborhood search operators.

5.2. Computational Results on Multiple Drone Scheduling

We use ten test instances constructed based on real-world high-speed railway catenary networks in China for the multiple deicing drone scheduling problem, the basic information of which is presented in Table 2.

The solution structure of multiple deicing drone scheduling is similar to that of the electric VRP where vehicles need to visit stations for battery recharging or replacement, except that our problem needs to determine an additional direction at each segment (similar to a customer in VRP). For comparison, we adapt the following nine efficient metaheuristic and memetic optimization algorithms for VRP to our problem by adding the additional direction part to solution

Table 2: Summary of the test instances of multiple deicing drone scheduling.

#	m	$\|V\|$	$\|E\|$	$\|E^\dagger\|$	$\sum_{e\in E} d(e)$	$\sum_{e\in E^\dagger} d(e)$	$\|\Gamma\|$
1	2	16	35	32	37.1	28.6	39
2	3	21	77	65	72.3	66.1	45
3	4	26	158	149	142.0	122.7	49
4	4	36	211	188	185.4	156.5	51
5	4	48	406	356	340.6	314.0	54
6	6	48	406	356	340.6	314.0	54
7	6	60	673	512	590.8	486.6	73
8	8	60	673	512	590.8	486.6	73
9	9	77	1453	1216	1289.5	997.7	90
10	12	77	1453	1216	1289.5	997.7	90

representation as well as performing the tentative improvement procedure for each new solution with the same probability control mechanism as our memetic algorithm:

- An interative local search (ILS) algorithm [34].
- An adaptive large neighborhood search (ALNS) algorithm [35].
- A hybrid GA and column generation (GA-CG) algorithm [36].
- A bilevel ACO Algorithm [37].
- A diversity-enhanced memetic algorithm (DEMA) integrating new genetic operators and tabu search [38].
- Four versions of the memetic algorithm based on our global mutation but only with single neighborhood search operators, NS1, NS2, NS3, and NS4, denoted by Mem-NS1, Mem-NS2, Mem-NS3, and Mem-NS4, respectively.

For a fair comparison, we set the same stopping condition that number of objective function evaluations reaches $100mn$ for all algorithms. Similarly, each algorithm is independently run 51 times with random seeds on each instance, and nonparametric Wilcoxon rank sum test is conducted on the result obtained by Mem and the result obtained by each other algorithm. The computational results are presented in Fig. 10 – Fig. 19.

Comparing the proposed algorithm with the first five popular algorithms from the literature, according to the statistical tests, the result of Mem is not significantly different from the result of DEMA on instance #1 and the result of ACO on instance #2; in all other cases, the result of Mem is significantly better than the result of each other algorithm. On these multiple drone test instances that are more difficult, the proposed algorithm has more obvious performance advantages over the existing algorithms. Comparatively, the performance of ISL is the worst because of its global search ability is relatively weak; the performance of GA-CG is the second worst mainly due to its poor local search

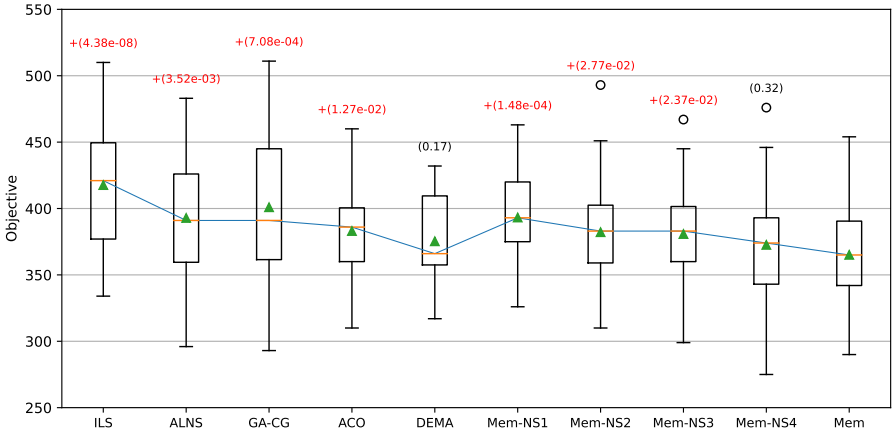

Figure 10: Box plots of the results obtained by the ten comparative algorithms on multiple drone scheduling instance #1.

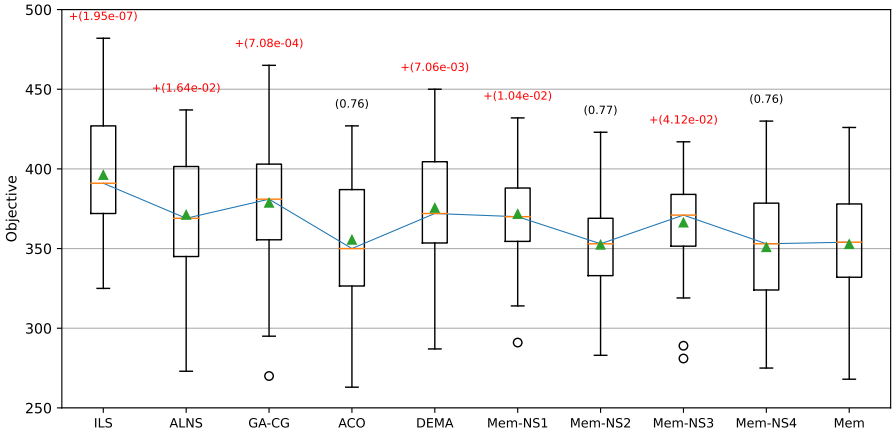

Figure 11: Box plots of the results obtained by the ten comparative algorithms on multiple drone scheduling instance #2.

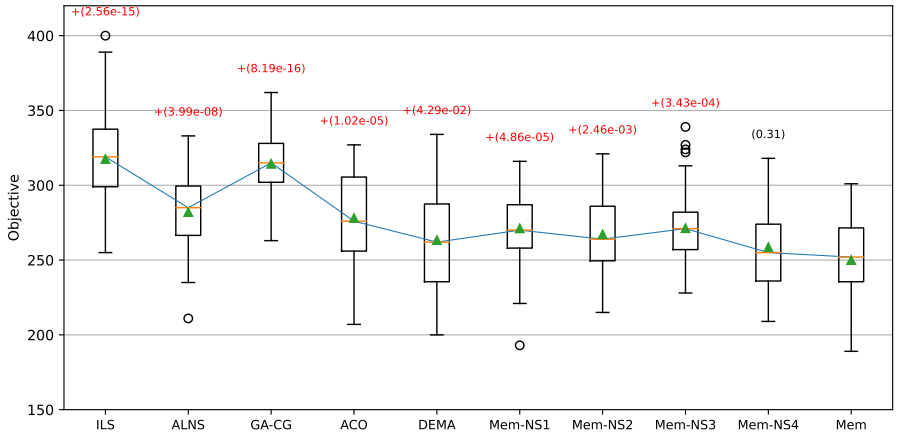

Figure 12: Box plots of the results obtained by the ten comparative algorithms on multiple drone scheduling instance #3.

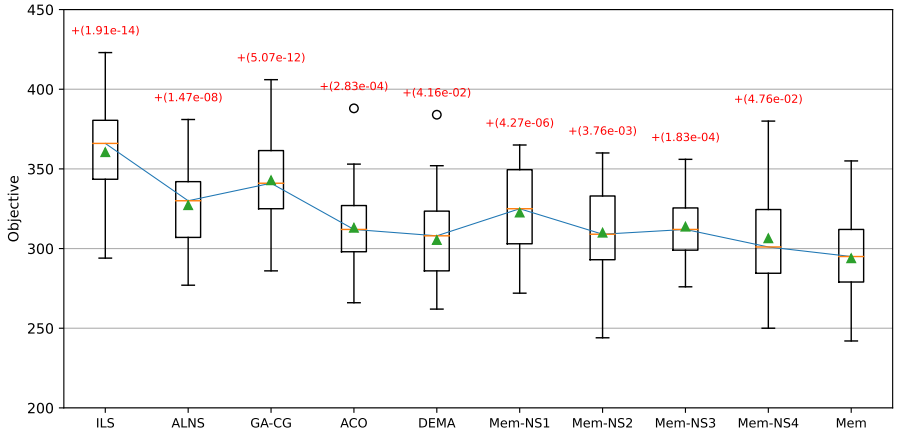

Figure 13: Box plots of the results obtained by the ten comparative algorithms on multiple drone scheduling instance #4.

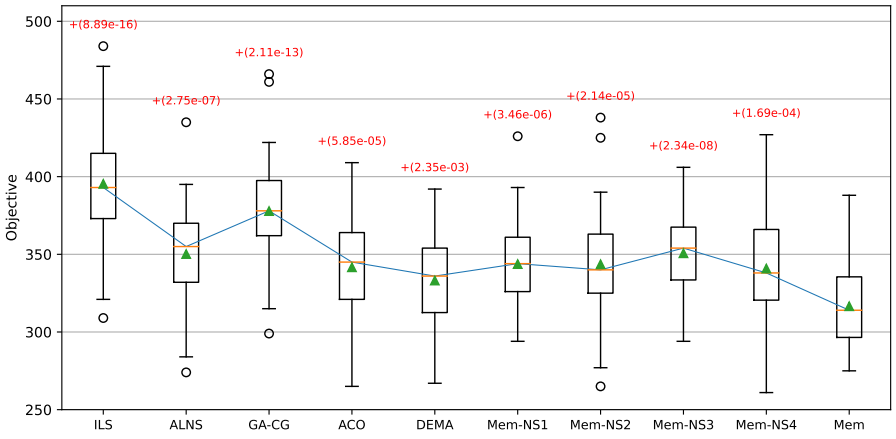

Figure 14: Box plots of the results obtained by the ten comparative algorithms on multiple drone scheduling instance #5.

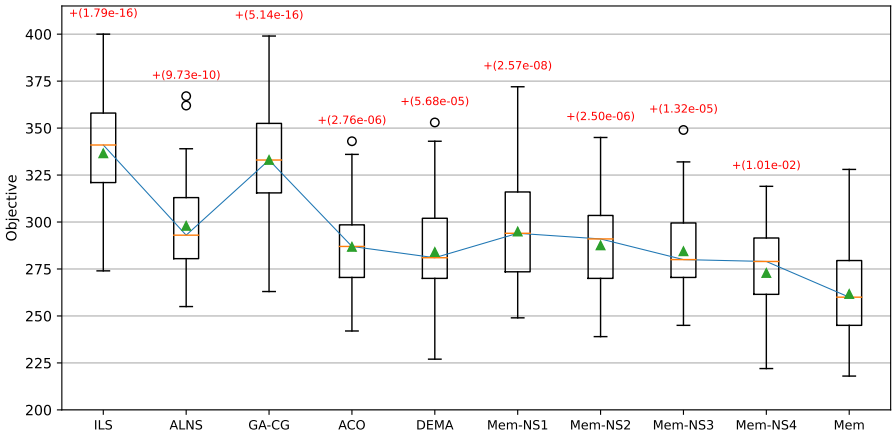

Figure 15: Box plots of the results obtained by the ten comparative algorithms on multiple drone scheduling instance #6.

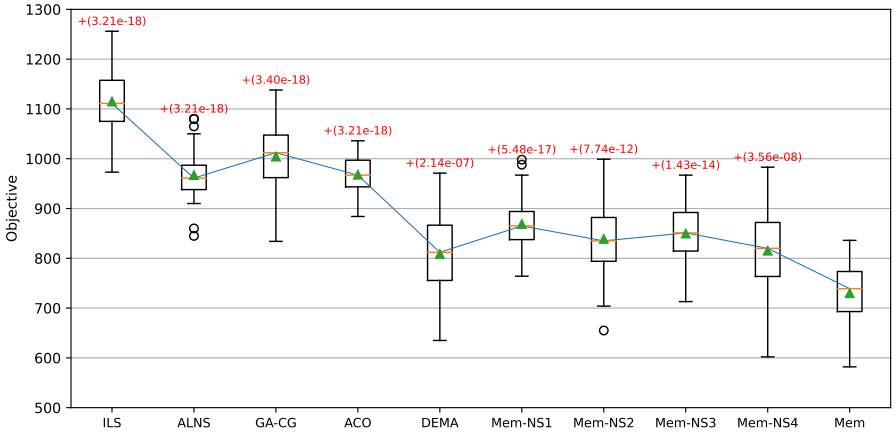

Figure 16: Box plots of the results obtained by the ten comparative algorithms on multiple drone scheduling instance #7.

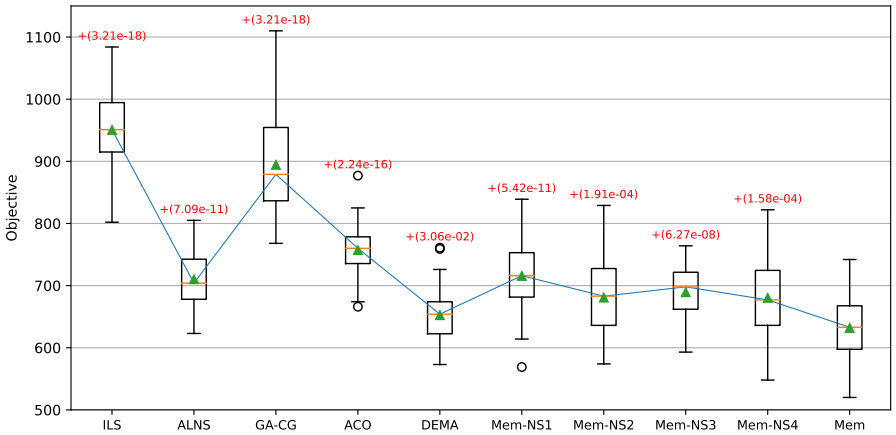

Figure 17: Box plots of the results obtained by the ten comparative algorithms on multiple drone scheduling instance #8.

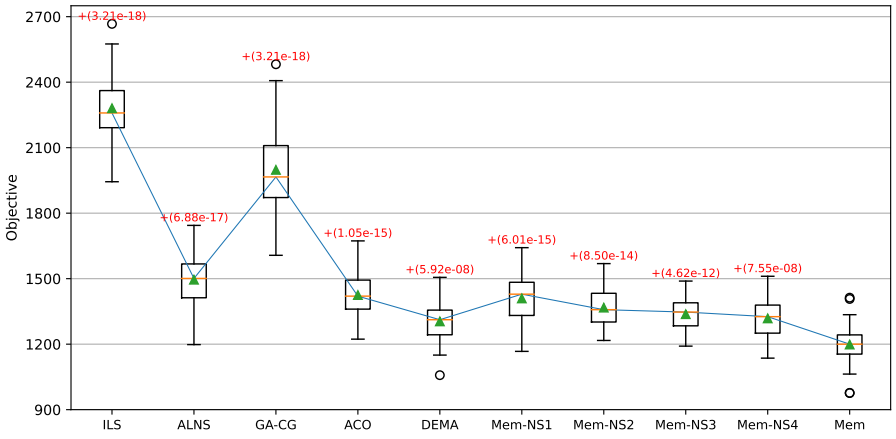

Figure 18: Box plots of the results obtained by the ten comparative algorithms on multiple drone scheduling instance #9.

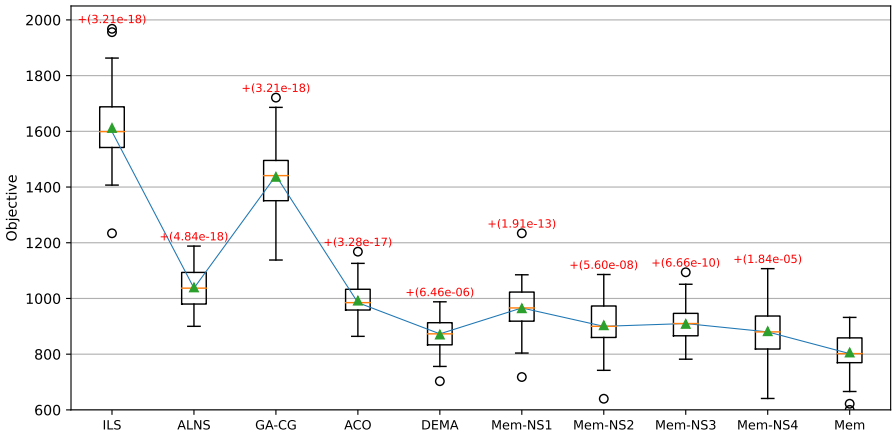

Figure 19: Box plots of the results obtained by the ten comparative algorithms on multiple drone scheduling instance #10.

ability. Similar to the case of single drone scheduling instances, the algorithms emphasizing local search exhibits better performance on more difficult instances. Particularly, another memetic algorithm DEMA performs best among these five existing algorithm, but still significantly worse than our algorithm which is elaborately designed for the considered problem.

Comparing the proposed algorithm using variable neighborhood search with the four versions of our algorithm using single neighborhood search, the result of Mem is not significantly different from the result of Mem-NS4 on instance #1, the results of Mem-NS2 and Mem-NS4 on instance #2, and the result of Mem-NS4 on instance #3; in all other cases, the result of Mem is significantly better than the result of each other version. Using neighborhood search operators that share information among different sub-schedules of different drones, Mem-NS2 and Mem-NS4 perform generally better Mem-NS1 and Mem-NS3 that exchange information within sub-schedules. By integrating the four neighborhood search operators in an adaptive selection schema, the proposed algorithm exhibits significantly better performance than all four versions with single neighborhood search.

In summary, the proposed memetic algorithm obtains the best median objective value among the ten comparative algorithms on each instance, and performs significantly better than the other algorithms in most cases. The performance advantages of the proposed algorithm become more obvious on more difficult instances, i.e., instances with larger number of segment to be deiced and larger number of drones. The computational results demonstrate that the proposed algorithm is effective and efficient for solving the considered deicing drone scheduling problem, especially for complex instances.

6. Conclusion and Discussion

Deicing drones are an effective method in response to freezing disasters striking railway catenary systems. In this study, we present a deicing drone scheduling problem for restoring catenary systems with the objective to minimize the total negative effect in terms of train delay and cancellation due to the freezing catenary. To efficiently solve the problem, we propose a memetic algorithm integrating global mutation and variable neighborhood search, the performance of which is demonstrated by computational results on real-world problem instances compared to selected popular optimization algorithms in the literature.

Our future work considers extending this study in several aspects. First, the current problem only considers stationary depots to provide battery replacement, and ongoing study is integrating stationary depots and movable depots (e.g., ground vehicles), the optimal deployment of which should be combined with the scheduling problem [39]. Second, when the icy rainfall and snowfall is continuing during the decision period, it is required to incorporate the prediction of freezing conditions and deicing workloads into the problem, for which we can use a method integrating machine learning and evolutionary optimization [40]. Third, the current objective function evaluation is based on the train delay and cancellation according to the predefined train timetable, while a more

challenging task is to enable the trains to change their routes by selecting new railway sections whose catenary lines have been or will be restored according to the deicing solution. Such extensions will significantly improve the complexity of the problem, which is to be solved by more elaborately designed methods.

References

[1] W. H. Torres, B. D. Aquino, E. I. O. Rivera, Artificial intelligence and unmanned aerial vehicle applications on electrical power systems, in: IEEE Power & Energy Society General Meeting, 2023, pp. 1–5. doi:10.1109/PESGM52003.2023.10253090.

[2] Y.-J. Zheng, Y.-C. Du, Z.-L. Su, H.-F. Ling, M.-X. Zhang, S.-Y. Chen, Evolutionary human-UAV cooperation for transmission network restoration, IEEE Trans. Ind. Informat. 17 (3) (2021) 1648–1657. doi:10.1109/TII.2020.3003903.

[3] X. Jiang, S. Fan, Z. Zhang, C. Sun, L. Shu, Simulation and experimental investigation of DC ice-melting process on an iced conductor, IEEE Trans. Power Deliv. 25 (2) (2010) 919–929. doi:10.1109/TPWRD.2009.2037632.

[4] J. Laforte, M. Allaire, J. Laflamme, State-of-the-art on power line de-icing, Atmosph. Res. 46 (1) (1998) 143–158. doi:10.1016/S0169-8095(97)00057-4.

[5] M. Huneault, C. Langheit, R. St-Arnaud, J. Benny, J. Audet, J.-C. Richard, A dynamic programming methodology to develop de-icing strategies during ice storms by channeling load currents in transmission networks, IEEE Trans. Power Deliv. 20 (2) (2005) 1604–1610. doi:10.1109/TPWRD.2004.838463.

[6] Y. Hou, X. Wang, Y. Zhang, Multi-objective transmission line de-icing outage optimal scheduling framework, IET Generat., Transm. Distrib. 10 (15) (2016) 3865–3874. doi:10.1049/iet-gtd.2016.0404.

[7] C. Deng, S. Wang, Z. Huang, Z. Tan, J. Liu, Unmanned aerial vehicles for power line inspection: A cooperative way in platforms and communications, J. Commun 9 (9) (2014) 687–692. doi:10.12720/jcm.9.9.687-692.

[8] Z. Zhou, C. Zhang, C. Xu, F. Xiong, Y. Zhang, T. Umer, Energy-efficient industrial internet of UAVs for power line inspection in smart grid, IEEE Trans. Ind. Informat. 14 (6) (2018) 2705–2714. doi:10.1109/TII.2018.2794320.

[9] R. Atat, M. F. Shaaban, M. Ismail, E. Serpedin, Efficient unmanned aerial vehicle paths design for post-disaster damage assessment of overhead transmission lines, IET Smart Grid 6 (5) (2023) 503–521. doi:10.1049/stg2.12120.

[10] K. Dorling, J. Heinrichs, G. G. Messier, S. Magierowski, Vehicle routing problems for drone delivery, IEEE Trans. Syst. Man Cybern.: Syst. 47 (1) (2017) 70–85. doi:10.1109/TSMC.2016.2582745.

[11] I. Hong, M. Kuby, A. T. Murray, A range-restricted recharging station coverage model for drone delivery service planning, Transp. Res. Part C: Emerg. Technol. 90 (2018) 198–212. doi:10.1016/j.trc.2018.02.017.

[12] Y. Zheng, Y. Du, H. Ling, W. Sheng, S. Chen, Evolutionary collaborative human-UAV search for escaped criminals, IEEE Trans. Evol. Comput. 24 (2) (2020) 217–231. doi:10.1109/TEVC.2019.2925175.

[13] M. Pachayappan, V. Sudhakar, A solution to drone routing problems using docking stations for pickup and delivery services, Transp Res Record 2675 (12) (2021) 1056–1074. doi:10.1177/03611981211032219.

[14] J. Gómez-Lagos, B. Rojas-Espinoza, A. Candia-Véjar, On a pickup to delivery drone routing problem: Models and algorithms, Comput. Ind. Eng. 172 (2022) 108632. doi:10.1016/j.cie.2022.108632.

[15] X. Wen, G. Wu, Heterogeneous multi-drone routing problem for parcel delivery, Transp. Rese. Part C: Emerg. Technol. 141 (2022) 103763. doi:10.1016/j.trc.2022.103763.

[16] Y. Guo, Y. Huang, S. Ge, Y. Zhang, E. Jiang, B. Cheng, S. Yang, Low-carbon routing based on improved artificial bee colony algorithm for electric trackless rubber-tyred vehicles, Complex Syst. Model. Simul. 3 (3) (2023) 169–190. doi:10.23919/CSMS.2023.0011.

[17] J. Chen, F. Ling, Y. Zhang, T. You, Y. Liu, X. Du, Coverage path planning of heterogeneous unmanned aerial vehicles based on ant colony system, Swarm Evol. Comput. 69 (2022) 101005. doi:10.1016/j.swevo.2021.101005.

[18] X. Wang, Z. Liu, X. Li, Optimal delivery route planning for a fleet of heterogeneous drones: A rescheduling-based genetic algorithm approach, Comput. Ind. Eng. 179 (2023) 109179. doi:10.1016/j.cie.2023.109179.

[19] B. Xu, K. Zhao, Q. Luo, G. Wu, W. Pedrycz, A GV-drone arc routing approach for urban traffic patrol by coordinating a ground vehicle and multiple drones, Swarm Evol. Comput. 77 (2023) 101246. doi:10.1016/j.swevo.2023.101246.

[20] Y.-J. Zheng, X. Chen, H.-F. Yan, M.-X. Zhang, Evolutionary algorithm for vehicle routing for shared e-bicycle battery replacement and recycling, Appl. Soft Comput. 135 (2023) 110023. doi:10.1016/j.asoc.2023.110023.

[21] M. Fan, Y. Wu, T. Liao, Z. Cao, H. Guo, G. Sartoretti, G. Wu, Deep reinforcement learning for UAV routing in the presence of multiple charging stations, IEEE Trans. Vehicular Technol. 72 (5) (2023) 5732–5746. doi:10.1109/TVT.2022.3232607.

[22] Y.-J. Zheng, Emergency train scheduling on Chinese high-speed railways, Transp. Sci. 52 (5) (2018) 1077–1091. doi:10.1287/trsc.2017.0794.

[23] Y.-J. Zheng, Water wave optimization: A new nature-inspired metaheuristic, Comput. Oper. Res. 55 (1) (2015) 1–11. doi:10.1016/j.cor.2014.10.008.

[24] W. L. Zhang, L., D.-Z. Zheng, An adaptive genetic algorithm with multiple operators for flowshop scheduling, Int. J. Adv. Manuf. Technol. 27 (2006) 580–587. doi:10.1007/s00170-004-2223-3.

[25] C.-J. Liao, C.-T. Tseng, P. Luarn, A discrete version of particle swarm optimization for flowshop scheduling problems, Comput Oper Res 34 (10) (2007) 3099–3111. doi:10.1016/j.cor.2005.11.017.

[26] P. Dasgupta, S. Das, A discrete inter-species cuckoo search for flowshop scheduling problems, Comput. Oper. Res. 60 (2015) 111–120. doi:10.1016/j.cor.2015.01.005.

[27] W. Shao, D. Pi, Z. Shao, An extended teaching-learning based optimization algorithm for solving no-wait flow shop scheduling problem, Appl. Soft Comput. 61 (2017) 193–210. doi:10.1016/j.asoc.2017.08.020.

[28] O. Engin, A. Güçlü, A new hybrid ant colony optimization algorithm for solving the no-wait flow shop scheduling problems, Appl. Soft Comput. 72 (2018) 166–176. doi:10.1016/j.asoc.2018.08.002.

[29] Y.-J. Zheng, H.-F. Ling, H.-H. Shi, H.-S. Chen, S.-Y. Chen, Emergency railway wagon scheduling by hybrid biogeography-based optimization, Comput. Oper. Res. 43 (3) (2014) 1–8. doi:10.1016/j.cor.2013.09.002.

[30] Y.-C. Du, M.-X. Zhang, C.-Y. Cai, Y.-J. Zheng, Enhanced biogeography-based optimization for flow-shop scheduling, in: J. Qiao, X. Zhao, L. Pan, X. Zuo, X. Zhang, Q. Zhang, S. Huang (Eds.), Bio-inspired Computing: Theories and Applications, Commun Comput Inf Sci, Springer, Singapore, 2018, pp. 295–306.

[31] M. Kurdi, A memetic algorithm with novel semi-constructive evolution operators for permutation flowshop scheduling problem, Appl. Soft Comput. 94 (2020) 106458. doi:10.1016/j.asoc.2020.106458.

[32] M. de Fátima Morais, M. H. D. M. Ribeiro, R. G. da Silva, V. C. Mariani, L. dos Santos Coelho, Discrete differential evolution metaheuristics for permutation flow shop scheduling problems, Comput. Ind. Eng. 166 (2022) 107956. doi:10.1016/j.cie.2022.107956.

[33] W. Shao, Z. Shao, D. Pi, Multi-local search-based general variable neighborhood search for distributed flow shop scheduling in heterogeneous multi-factories, Appl. Soft Comput. 125 (2022) 109138. doi:10.1016/j.asoc.2022.109138.

[34] G. Macrina, L. Di Puglia Pugliese, F. Guerriero, G. Laporte, The green mixed fleet vehicle routing problem with partial battery recharging and time windows, Comput. Oper. Res. 101 (2019) 183–199. doi:10.1016/j.cor.2018.07.012.

[35] M. Keskin, B. Çatay, A metaheuristic method for the electric vehicle routing problem with time windows and fast chargers, Comput. Oper. Res. 100 (2018) 172–188. doi:10.1016/j.cor.2018.06.019.

[36] C. Wang, C. Guo, X. Zuo, Solving multi-depot electric vehicle scheduling problem by column generation and genetic algorithm, Appl. Soft Comput. 112 (2021) 107774. doi:10.1016/j.asoc.2021.107774.

[37] Y.-H. Jia, Y. Mei, M. Zhang, A bilevel ant colony optimization algorithm for capacitated electric vehicle routing problem, IEEE Trans. Cybern. 52 (10) (2022) 10855–10868. doi:10.1109/TCYB.2021.3069942.

[38] J. Xiao, J. Du, Z. Cao, X. Zhang, Y. Niu, A diversity-enhanced memetic algorithm for solving electric vehicle routing problems with time windows and mixed backhauls, Applied Soft Computing 134 (2023) 110025. doi:10.1016/j.asoc.2023.110025.

[39] N. Li, Z. Su, H. Ling, M. Karatas, Y. Zheng, Optimization of air defense system deployment against reconnaissance drone swarms, Complex Syst. Model. Simul. 3 (2) (2023) 102–117. doi:10.23919/CSMS.2023.0003.

[40] X. Chen, H.-F. Yan, Y.-J. Zheng, M. Karatas, Integration of machine learning prediction and heuristic optimization for mask delivery in COVID-19, Swarm Evol. Comput. 76 (2023) 101208. doi:10.1016/j.swevo.2022.101208.