Emission Monitoring Dispatching of Drones Under Vessel Speed Fluctuation

Poly Z. H. Sun, *Member, IEEE*, Xiaosong Luo, *Graduate Student Member, IEEE*, Tienyu Zuo, Yuguang Bao, Yan-Ning Sun, *Member, IEEE*, Rob Law, and Edmond Q. Wu, *Senior Member, IEEE*

***Abstract*— How to effectively organize drones to monitor pollutants from vessels is an important operational problem in port management. It is defined as the drone scheduling problem (DSP). The effectiveness of precise algorithms and heuristic algorithms in solving DSP has been reported in previous studies. In previous studies, the speed of the vessel was assumed to be constant. However, since the influence of sea waves and vessel power, such an assumption is difficult to satisfy in actual scenarios. The actual position of the vessel may deviate from the position information obtained through prior calculations. As the cumulative position deviation increases, it is possible to make the original feasible monitoring scheme infeasible. It is necessary to consider the emission monitoring dispatching of drones under vessel speed fluctuation in actual monitoring activities of the vessel. To deal with the problem, a dynamic dispatching strategy based on reinforcement learning (RL) is proposed. Considering the vessel speed fluctuation, the monitoring window is divided into multiple sub-time windows. The route information of the vessel in each sub-time window is updated according to the vessel speed fluctuations to reduce the accumulation of deviations between the prior position and the actual position. Then, a lightweight RL strategy is adopted to quickly (re)organize the monitoring scheme in each sub-time window. Numerical experiments illustrate the above division-conquer approach could effectively reduce the possibility of drone monitoring failure caused by vessel speed fluctuations. Also, the superiority of the RL-based dispatching strategy is illustrated by comparing it with multiple dispatching schemes.**

***Index Terms*— Drone scheduling problem, emission control area, dynamic optimization problem, reinforcement learning.**

Manuscript received 25 December 2021; revised 29 March 2022 and 23 May 2022; accepted 6 July 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 62171274 and Grant U1933125, in part by the National Natural Science Foundation of China through the Main Research Project on Machine Behavior and Human–Machine Collaborated Decision Making Methodology under Grant 72192820, and in part by the Third Research Project on Human Behavior in Human–Machine Collaboration under Grant 72192822. The Associate Editor for this article was B. F. Ciuffo. *(Corresponding authors: Rob Law; Poly Z. H. Sun.)*

Poly Z. H. Sun is with the Department of Industrial Engineering, School of Mechanical Engineering, Shanghai Jiao Tong University, and also with the Department of Automation, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: zh.sun@sjtu.edu.cn).

Xiaosong Luo, Yuguang Bao, and Yan-Ning Sun are with the Department of Industrial Engineering, School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China.

Tienyu Zuo is with the School of Automation, Nanjing University of Information Science and Technology, Nanjing 210044, China (e-mail: tian_yu_zuo@163.com).

Rob Law is with the Asia-Pacific Academy of Economics and Management, University of Macau, and also with the Department of Integrated Resort and Tourism Management, Faculty of Business Administration, University of Macau, Macau, 999078, China (e-mail: roblaw@um.edu.mo).

Edmond Q. Wu is with the Key Laboratory of System Control and Information Processing, Ministry of Education of China, the Shanghai Engineering Research Center of Intelligent Control and Management, and the Department of Automation, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: edmondqwu@163.com).

Digital Object Identifier 10.1109/TITS.2022.3189979

I. Introduction

UNDER the trend of economic globalization and supply chain globalization, there is a large demand for international cargo transportation. As the main approach, ocean shipping is considered to be the cheapest and most convenient means of long-distance transportation. It is reported that more than 80% of the global trade volume comes from container shipping [1], [2]. The power of container vessels mainly depends on the consumption of high-sulfur fuel that sulfur content is more than 3.50%. The pollution caused by it severe damage to the water and the atmosphere near the port. Some studies have shown that the air pollution generated by large and medium-sized container vessels at the speed of 70% of the maximum is equivalent to that caused by 50,000 trucks [3]. Exhaust emissions are becoming the main factor that exacerbates the deterioration of air quality [4]. Besides, the main pollutants contained in the exhaust gas emitted by vessels include carbon dioxide (CO_2), nitrogen oxides (NO_X), sulfur oxides (SO_X), and inhalable particulate matter (PM) [5], [6]. The above pollutants are aggravating ocean acidity and causing humans to suffer from many respiratory diseases [7], [8]. Severe environmental and health issues forced International Maritime Organization (IMO) to formulate rules and regulations to strictly control pollutant emissions from vessels. Among the regulations, the International Convention for the Prevention of Pollution from ships with Annex VI (MARPOL Annex VI) is one of the main international conventions for the prevention of environmental pollution by vessels [9]. It strictly stipulates that the sulfur content of fuel must be less than 3.50% (it is adjusted to 0.50% in 2020). In addition, emission control areas (ECAs) [10] have been established for the four main coasts of the world, and also more regions are gradually establishing new ECAs. For example, as a prosperous shipping country, China continues to make efforts to establish ECA and improve emission regulations of the country [11]. Vessels sailing in ECAs are subject to stricter emission supervision and restrictions. IMO stipulates that the sulfur content of fuel consumed by the vessel sailing in ECA must be no more than 0.10% [12]. With the efforts of countries all over the world, the positive effects of ECAs for mitigating environmental pollution control in coastal areas have been widely confirmed [13]–[16]. However, since various emission regulations require companies to bear additional costs for purchasing gas purification devices or using low-sulfur fuels, some vessel operators are taking the risk of violating regulations to save costs [17]–[19]. Therefore, to ensure the effective implementation of regulations, stronger pollutant emission monitoring must be conducted on vessels once entering the ECA.

Traditional monitoring methods include vessel technical file verification, hand-held analyzer onboard inspection, and sniffing telemetry. The above methods require plenty of manpower and are inefficient [20]. Thanks to the development of unmanned aerial vehicle technology [21], the use of drones equipped with the lightweight sniffer to implement vessel emission supervision is a promising supervision scheme, which has been applied in practical scenarios [22]. In the actual operation, the port manager could calculate the real-time position of the vessel in a certain time interval in the future according to the entry application information of the vessel (including carrier number, route, and speed), which can be used as the prior knowledge of the real-time position of the vessel. After that, the port manager could control drones to fly from the drone station to the target vessel, and conduct emission monitoring of this vessel. However, considering the limited size of the drone fleet and the limitations of battery capacity, it is necessary to carefully plan the flight journey of each drone. In other words, each drone needs to be assigned a monitoring sequence containing multiple target vessels so that the drone fleet could perform as many monitoring tasks as possible within a given monitoring time window. Since different vessels entering the ECA have different historical performances, the monitoring importance of each vessel is different. If the monitoring reward of each vessel is defined as the historical violation record (vessels with violation records have greater rewards), the goal of the drone fleet is to maximize the cumulative monitoring rewards. The above operations research problem is abbreviated as the drone scheduling problem (DSP). Section II will detail the model of DSP.

DSP can be considered as a variant of vehicle routing problem (VRP) or team orienteering problem (TOP) if vessels are regarded as cities/control nodes and drones are regarded as vehicles/competitors. DSP can be simply described as organizing the monitoring sequence of the drone fleet within a given time window to maximize the cumulative monitoring reward. It is an NP-hard problem. There have been some preliminary attempts to solve the DSP. Xia *et al.* [23] proposed and formally defined DSP. The characteristics of the vessel position change with time in the monitoring time window is expressed through the time-expanded network. Time-space network is a typical mathematical tool used to describe the dynamic properties of DSP referring to the real-time changes of the vessel position, *i.e.,* the sailing process. After that, DSP is modeled as mixed-integer linear programming (MILP). Then, a Lagrangian relaxation-based method is developed to improve the algorithm performance in solving DSP. However, once the scale of the vessels increases, the algorithm performance

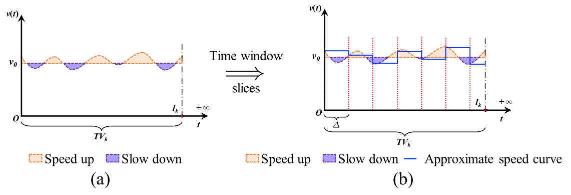

Fig. 1. An example of vessel speed fluctuation.

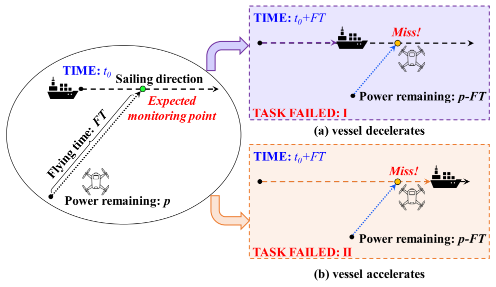

Fig. 2. Two situations that lead to task failure.

drops sharply. Luo *et al.* [24] and Sun *et al.* [25] explored the feasibility and effectiveness of the meta-heuristic method in solving DSP. The experimental results illustrate that the meta-heuristic method can obtain significantly better results than the precise algorithm in a short time, especially for large-scale problems.

In the above works, the prior knowledge of the real-time speed and position of vessels within the time window is assumed to be known and fixed in advance. In other words, the vessel is assumed to sail along a fixed channel and always maintain a constant speed. Such an assumption is helpful for the algorithm to search the solution space according to the vessel position at each time. From a macro point of view, the speed of the vessel can be regarded as stable. However, since the vessel is affected by factors such as waves and vessel power, it is still inevitable that the speed of the vessel will fluctuate. If the scheduling algorithm could not take vessel speed fluctuation into account, the feasibility of the monitoring scheme will be difficult to guarantee. As shown in Fig. 1(a), there is a gap between the ideal sailing speed and the actual sailing speed. Although the gap may be small, the accumulated errors after a while will have a devastating effect on the effectiveness of the monitoring scheme. Specifically, the accumulation of errors causes two types of damage to the monitoring scheme: (1) the vessel decelerates and the drone reaches the expected meeting point ahead of schedule; (2) the vessel accelerates and the drone misses the meeting with the vessel. For example, as shown in Fig. 2(a), according to the original speed of the vessel, the drone is expected to meet the vessel after 15 minutes and conduct the emission monitoring. Due to the vessel decelerating, the time for the vessel to reach the original meeting point is changed to 21 minutes. When the vessel reaches the meeting point, the drone had to start the next monitoring task, which causes the original monitoring task to fail. As shown in Fig. 2(b), according to the original speed of the vessel, the drone is expected to meet the vessel after 20 minutes and conduct the emission monitoring. Due to the vessel accelerating, the time for the vessel to reach the original meeting point becomes 13 minutes. When the drone reaches, the vessel had left the meeting point, which causes the original monitoring task to fail. Therefore, to obtain a practical monitoring scheme, the dynamic factor of vessel speed fluctuation should be taken seriously.

In the paper, a DSP that considers vessel speed fluctuation is defined as dynamic DSP (D-DSP). A reinforcement learning (RL)-based dispatching strategy is proposed to solve D-DSP. First, the monitoring time window is divided into multiple sub-time windows. At the beginning of each sub-time window, the vessel position and speed are updated. The monitoring task of each sub-time window can be regarded as a new DSP problem. Since the vessel speed fluctuation is a slow process, it can be considered that the speed of the vessel in a shorter sub-time window is a relatively fixed constant, and then the vessel motion could be simplified as a uniform linear motion. Through the positioning and speed measurement functions of the global positioning system (GPS) and automatic identification system (AIS), the position and speed of the vessel at the beginning of each sub-time window can be obtained [26], [27]. Therefore, the relatively accurate trajectory of the vessel can be predicted for the sub-time window based on the information of the vessel from the beginning of the sub-time window. To quickly generate the dispatching scheme for the sub-time window, then it is required a fast-time approach to rescheduling the drone. Due to the high computational cost of the meta-heuristic algorithm [28], [29], it is no longer suitable for rescheduling the drone. For the above considerations, the paper introduces Q-learning as a scheduler to replace the meta-heuristic algorithms for an efficient and fast solving performance. It should be pointed out, as shown in Fig 1(b), that although the solution of updating the vessel speed and position in time by dividing the time window cannot completely eliminate the monitoring error caused by the change of the vessel speed, time window update strategy can greatly reduce the tracking error and cumulative error of actual sailing speed and actual position. It will effectively alleviate the negative impact of dynamic factors on the dispatching system.

Besides, it is important to clarify that due to the influence of various uncertain environmental factors, only at a certain moment, the exact speed of the vessel at that moment is known. Therefore, the fluctuated vessel speed is typically not a piece of prior known information in D-DSP. If a fixed vessel speed (the speed when the vessel just entered the ECA or the historical average speed of the vessel) is set as the prior information in the entire scheduling cycle, the deviation of actual speed from the fixed prior vessel speed can lead to a decrease in the feasibility of the scheduling result. As an example in Fig. 2, it shows two situations that which the monitoring plan fails if the fixed prior speed is adopted as the input of the dispatching system. To alleviate the influence of the cumulative error of vessel speed over time, it is feasible to shorten the length of the scheduling time window, and dynamically obtain the physical position and speed of the vessel at the beginning moment of the shorter sub-time window according to the AIS. This paper adopts the scheme of dynamically obtaining vessel information through AIS. Fig. 1(b) shows an example, the dotted curve is the function of the real speed with respect to time, and the step function (approximate speed curve) is the result of calling the AIS data to identify the vessel speed at the beginning of each sub-time window.

The main contributions of the paper are as follows:

(1) D-DSP that considers vessel speed fluctuation is proposed. Compared with DSP, the model D-DSP is more in line with the actual monitoring process of the vessel. Since the vessel speed fluctuation is considered, the monitoring scheme output has stronger practical feasibility.

(2) A RL-based dispatching strategy is proposed to solve D-DSP. The developed algorithm can meet the timeliness of obtaining dispatching results within a short time. Also, it can further optimize the dispatching performance and monitoring efficiency of the drone.

(3) The proposed method enhances the feasibility of drone-based emission monitoring in actual ECA. Experiments verify that the proposed method outperforms other comparative methods.

The remaining of the paper is organized as follows. Section II gives the mathematical model of DSP and a brief review of the application of RL in the field of operations research optimization. Section III elaborates on the RL-based dispatching strategy proposed in the paper. Section IV illustrates the effectiveness of the method proposed through experiments. The theoretical and practical insights are arranged in Section V. Finally, conclusions and future research directions are given in Section VI.

II. Background

A. DSP

Suppose the total monitoring time window is $T = \{0, 1, 2, \ldots, T_{max}\}$. The set of vessels sailing in ECA during the monitoring period is V. The set of drone stations on the shore of the port is K and each drone station can provide the charging service for any drone. The set of the callable drones is D. At the initial moment, $|D|$ drones are evenly distributed among $|K|$ drone stations waiting for instructions to perform monitoring tasks on the $|V|$ vessels. Each vessel has its independent sailing time window, within which any drone can monitor it. $TV_\upsilon = \{e_\upsilon, e_\upsilon + 1, \ldots, l_\upsilon\}$ is adopted to represent the sailing time window of the υ-th vessel, where e_υ and l_υ represent the time point when the υ-th vessel enters the ECA and the time point when it arrives at the target port respectively. Monitoring rewards are determined based on the historical violation records of vessels. For the vessel that has records of violating emission regulations, a higher monitoring reward is given, and a lower monitoring reward is given. The greater the reward for a vessel, the greater the preference for monitoring the vessel. w_υ is used to represent the monitoring reward of υ-th vessel. The flying speed of drones is set to sd. The maximum battery of drones is set to Q. The time consumption for one charge and one monitoring is δ_0 and δ, respectively. Assuming that battery power consumption is proportional to time. *i.e.,* one unit of time consumes one unit of battery power.

The route of vessels in ECA is described as a time-related point set $N = \{(v, t)|v \in V, t \in TV_v\}$. During the monitoring time, the drone station also can be expressed as a time-related point set $N_0 = \{(k, t)|k \in K, t \in T\}$. u_i and t_i is used to indicate the object and time point of node i, respectively. For example, suppose the node corresponding to the v-th vessel at time t is i, then $u_i \in V$ and $t_i = t$. All elements in point set N and N_0 form a directed arc network Ω, where $e(i, j)$ represents the connecting arc between nodes i and j. G is a subset of Ω. It consists of the flight path (*i.e.,* monitoring scheme) of the drone fleet. Binary variable $x_{i,j}$ is used to indicate whether arc $e(i, j)$ exists in G. If $e(i, j)$ exists in G, then $x_{i,j} = 1$, otherwise, 0. $y_{i,t}$ indicates the number of drones parked in drone station i at the time t. $q_{i,t}$ indicates the remaining battery power of the drone when it reach node i at time t.

The objective function of DSP can be described as follows:

$$F : \arg\max_{G} \sum_{\Omega} \frac{(w_i + w_j)}{2} x_{i,j}. \tag{1}$$

When the drone monitors the vessels, it is subject to a variety of restrictions from the arc network and electricity. First of all, the out-degree and in-degree of each node should be balanced. (2) and (3) balance the in-degree and out-degree of each drone station node and vessel node, respectively.

$$\sum_{j\in N} x_{i,j} - \sum_{j\in N} x_{j,i} = y_{u_i,t_i-1} - y_{u_i,t_i}, \forall i \in N_0, \tag{2}$$

$$\sum_{j\in N\cup N_0} x_{i,j} - \sum_{j\in N\cup N_0} x_{j,i} = 0, \forall i \in N. \tag{3}$$

Secondly, the battery power changes of the drone from vessel node to vessel node or drone station node and from drone station to vessel node should be recorded, which can be expressed as (4) and (5). It should be noted that the movement of drones between drone station nodes is not allowed during the monitoring process.

$$q_{i,t_i} - (t_j - t_i)x_{i,j} \geqslant q_{j,t_j}, \forall i \in N, \forall j \in N \cup N_0, \tag{4}$$

$$Q - (t_j - t_i)x_{i,j} \geqslant q_{j,t_j}, \forall i \in N_0, \forall j \in N. \tag{5}$$

Then, the following constraints (6) and (7) stipulate that each vessel can only be monitored at most once and that only one drone is allowed to leave or return to the drone station within a safe time interval (to avoid drone collisions).

$$\sum_{i\in N\cup N_0} \sum_{j\in N, u_j=V} x_{i,j} \leqslant 1, \forall v \in V, \tag{6}$$

$$\sum_{j\in N} x_{i,j} + \sum_{j\in N} x_{j,i} \leqslant 1, \forall i \in N_0. \tag{7}$$

Finally, the value ranges of the three decision variables are expressed in (8), (9) and (10).

$$x_{i,j} \in \{0, 1\}, \forall e(i, j) \in G, \tag{8}$$

$$y_{i,t} \in \{0, 1, \ldots, |D|\}, \forall i \in K, \forall t \in T, \tag{9}$$

$$q_{i,t} \geqslant 0, \forall i \in N \cup N_0, \forall t \in T. \tag{10}$$

Based on the arc network, DSP can be defined as a MILP. The optimization object of DSP is to formulate the monitoring scheme of the drones within the total monitoring time window, and the optimization goal is to maximize the total monitoring reward of the drones.

B. D-DSP

The difference between DSP and D-DSP is whether the speed of the vessel is assumed to be constant. In DSP, the speed of the vessel is assumed to be constant, which is a simplified model of the drone-based ECA monitoring mode. However, this assumption does not correspond to the actual situation that vessels face in sailing, *i.e.,* the speed of the vessel is constantly disturbed by waves and strong winds. Further, there will be a deviation between the real-time position of the vessel and the predicted position, and the deviation is unpredictable. Therefore, in D-DSP, the fluctuation of the vessel's sailing speed due to environmental factors is considered. The entire monitoring time window is divided into multiple sub-time windows. At the beginning of each sub-time window, AIS data is called to obtain the position and speed of each vessel at the current moment. By updating the vessel position and speed at each sub-time window, the damage to the scheduling scheme caused by the dynamic factor of vessel speed fluctuation can be alleviated, that is why a stepped discontinuous function like in Fig. 1(b) is adopted to approximate the real vessel speed with respect to time.

The dynamics in D-DSP are derived from environmental factors (*e.g.,* ocean waves and strong winds) in the real offshore area. However these environmental factors act not on drones, but on vessels. MILP model involved in Section II-A depicts the process of emission monitoring of vessels by drones. The solution of the model means the control scheme (or monitoring scheme) of drones, and it can also be understood as the sequence of vessels being visited. Therefore, if the dynamic factors do not affect the operation of the drone, the mathematical expression of D-DSP is the same as that of DSP. From the perspective of dynamic scheduling theory, before and after the occurrence of dynamic events, decision-makers are faced with problems in different initial states (*i.e.,* the sailing speed of each vessel may be different in different sub-time windows), but they are all static sub-problems (*i.e.,* the scheduling problem within a certain sub-time window). Therefore, if the model of the static problem itself is not dynamic, the static model can be used to describe each sub-problem in the dynamic problem to define the entire dynamic problem.

C. RL-Based Method for Operations Research

RL has been extended to solve various operations research optimization or combinatorial optimization problems [30]. There are two common applications: (1) RL is used as a hyper-heuristic algorithm. The agent combines the optimal search methods in the meta-heuristic algorithm space to better solve the target problem; (2) Combine RL with machine learning methods, which is a new paradigm for solving combinatorial optimization problems.

Zhang *et al.* [31] proposed a deep RL method to optimize a combination of heuristic methods. Experiments show that the RL method has better performance and interpretability in dealing with uncertain combinatorial optimization problems. Qin *et al.* [32] applied the RL-based hyperheuristic searcher to solve the heterogeneous VRP. In this work, RL is used as a high-level selection strategy of the algorithm, and various heuristic methods are low-level alternative algorithms. Experimental results illustrate that the method outperforms the single meta-heuristic algorithm. Santiago Júnior *et al.* [33] studied a multi-objective selection hyper-heuristic approach combining, RL, meta-heuristic selection, and group decision-making. Choong *et al.* [34] used Q-learning to guide the hyper-heuristic model in selecting the proper components during different stages of the optimization process. Bello *et al.* [35] adopted the RL method to train the pointer network composed of recurrent neural network and long short-term memory. Pan *et al.* [36] combined RL and machine learning methods to solve the permutation flow-shop scheduling problem (PFSP). In this work, a new deep neural network (PFSPNet) is developed to encode the scheduling problem into a vector, and then the processing order could be obtained by the attention mechanism. The actor-critic method of RL is used to optimize the parameters of the PFSPNet. Zhang *et al.* [37] solved the multi-vehicle routing problem with soft time windows (MVRPSTW). A multi-agent RL method with an unsupervised auxiliary network is developed for the training process.

Besides, some works directly use RL to solve scheduling problems. In these works, the environment and actions are carefully designed and defined. Aljohani *et al.* [38] adopted a double deep Q-learning network to solve a real-time and data-driven electric vehicle routing optimization problem. Alqahtani *et al.* [39] also used DDQN to solve the energy scheduling and path planning issues of multiple electric vehicles.

Since Q-learning has a strong ability to quickly obtain dispatching results within a short time, in this paper, the RL-based scheduling strategy proposed for solving D-DSP is composed of a time window slice mechanism and Q-learning. The slice density and state-action definition of the total monitoring time window are the key parts of the proposed method, which will be introduced in detail in Section III-B.

III. Proposed Method

The general framework of the proposed method is given in Section III-A. The three important components, *i.e.,* time window slice, adjustment strategy, and the implementation of Q-learning, are elaborated in Section III-B.

A. Overview

The framework of RL-based drone dynamic dispatching is shown in Alg. 1. In the initial stage, sailing data of each vessel including the initial position, sailing speed and target port, and monitoring reward are obtained. Besides, the algorithm parameters are initialized, including the number of episodes, greedy factor, learning rate, etc. (Line 1). Lines 3 to 8 show the main process of planning a drone monitoring scheme (*i.e.,* $scheme$) within the given sub-time window. At the start time point of the given sub-time window, the sailing speed and route for each vessel are updated (Line 4). Then, Alg. 2 is employed to re-schedule the monitoring scheme constructed so far, to prevent the original plan of the drone from being failed by speed changes of the target vessel (Line 5). Next, the RL-based dispatching method is called for planning the monitoring scheme under the current sub-time window (Line 6). If the upper limit of the total monitoring time window (*i.e.,* T_{max}) is not exceeded, go to the next sub-time window for planning (Line 7); otherwise, the algorithm ends and the monitoring scheme within the total monitoring time window is returned (Line 9).

Algorithm 1 RL-Based Dynamic Dispatching Framework

```
Input: $T$: Total monitoring time window; $\Delta$: Length of the sub-time window.
Output: Monitoring scheme $scheme$ under the given time window $T$.
1: Initialize the parameters and divide $T$ into multiple sub time windows $\{\Gamma_1, \Gamma_2, \ldots, \Gamma_n\}$;
2: $time \leftarrow 0$ and $scheme \leftarrow \varnothing$;
3: while $time \le T_{max}$ do
4:     Obtain the sailing speed and route for each vessel according to AIS and generate the update vessel trajectory;
5:     Adjust the planned monitoring scheme before the current sub-time window according to Alg. 2;
6:     Call Alg. 4 and 5 to plan the monitoring scheme for the current sub-time window on the basis of adjustments, then update $scheme$;
7:     $time \leftarrow time + \Delta$;
8: end while
9: return $scheme$
```

Intuitively, the core idea of the proposed algorithm for tackling D-DSP is to dynamically update and correct the scheduling system's perception of the current state of the vessel (mainly the speed) at each decision point (*i.e.,* the beginning moment of a sub-time window). An updated sub-scheduling problem will be constructed in different sub-windows. The complete total monitoring scheme will be constructed by incremental approaches. Therefore, the validity of the monitoring scheme cannot be maintained throughout the process. When faced with a decision point, the old monitoring scheme may fail because the drone may be not able to accomplish the established old tasks in the new state. Therefore, it is necessary to dynamically update the scheme for its feasibility, according to the updated vessel speed information. An example is executed to illustrate how our algorithm handles scheme feasibility updates due to dynamic factors. Fig. 3 shows the generation process of the monitoring scheme in the entire monitoring time window. The white bar refers to the monitoring plan within the total monitoring time window. The orange bar indicates

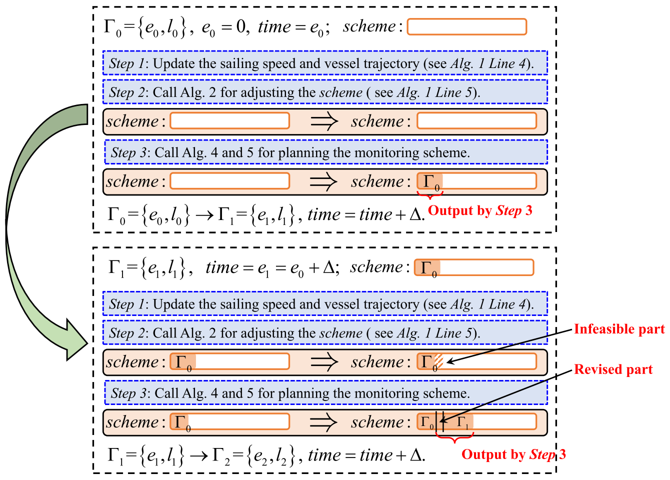
$\Gamma_0=\{e_0,l_0\}$, $e_0=0$, $time=e_0$; scheme:
Step 1: Update the sailing speed and vessel trajectory (see Alg. 1 Line 4).
Step 2: Call Alg. 2 for adjusting the scheme (see Alg. 1 Line 5).
scheme: ⇒ scheme:
Step 3: Call Alg. 4 and 5 for planning the monitoring scheme.
scheme: ⇒ scheme: Γ_0
$\Gamma_0=\{e_0,l_0\}\rightarrow\Gamma_1=\{e_1,l_1\}$, $time=time+\Delta$.
Output by Step 3
$\Gamma_1=\{e_1,l_1\}$, $time=e_1=e_0+\Delta$; scheme: Γ_0
Step 1: Update the sailing speed and vessel trajectory (see Alg. 1 Line 4).
Step 2: Call Alg. 2 for adjusting the scheme (see Alg. 1 Line 5).
Infeasible part
scheme: Γ_0 ⇒ scheme: Γ_0
Revised part
Step 3: Call Alg. 4 and 5 for planning the monitoring scheme.
scheme: Γ_0 ⇒ scheme: Γ_0 Γ_1
$\Gamma_1=\{e_1,l_1\}\rightarrow\Gamma_2=\{e_2,l_2\}$, $time=time+\Delta$.
Output by Step 3

Fig. 3. Incremental construction process of *scheme* (from Γ_0 to Γ_1).

a monitoring plan belonging to a sub-time window. Over time, multiple orange bars belonging to different sub-time windows will gradually fill up with the white bar.

Without loss of generality, it is assumed that the current moment e_1 and l_1 are the start moment and end moment of the sub-time window Γ_1 respectively, and $e_1 \neq 0$, that is, Γ_1 is not the first sub-time window. Γ_0 is the pre-order time window of Γ_1. $scheme \neq \varnothing$. At the time e_1, the speed of each vessel and the trajectory within $t \in [e_1, l_1]$ are updated (Line 4 of Alg. 1). Then, Alg. 2 is called to check if the speed of the target vessel for each drone is updated. If not updated, the journey is feasible (Guaranteed by Alg. 3). If updated, it requires to further judge whether it can monitor the target vessel (speed and position are updated) with the current state of the drone (position and remaining power at the time e_1). If cannot, it is necessary to reassign a new target vessel to the drone (Line 9 of Alg. 2). If can, it is only necessary to simply adjust the intersection point of the drone and its target vessel (Line 11 of Alg. 2). Next, Alg. 4 and 5 are called to construct the monitoring plan for each drone within $t \in [e_1, l_1]$.

It can be found that when a sub-time window transition occurs, *i.e.*, jumping from Γ_0 to Γ_1, $(t = e_1 - 1) \rightarrow (t = e_1)$, the vessel's speed is updated causing the trajectory of each vessel to be changed, which in turn causes some of the well-planned old monitoring tasks within Γ_0 to become infeasible. At this point, Alg. 2 is called to check the feasibility change of each drone's journey to reach the originally specified target vessel. The infeasible part of the *scheme* will be modified and become feasible. The infeasible part in Fig. 3 refers to the part of the monitoring plan belonging to the previous sub-time window that was damaged due to environmental changes, and the revised part in Fig. 3 refers to the result of repairing the infeasible part according to the vessel speed change of the current sub-time window. Finally, Alg. 4 and 5 will generate the monitoring scheme within Γ_1 and add it into *scheme*. The above process is repeatedly executed until the last sub-time window, *i.e.*, the monitoring plan within Γ_n is also planned.

Algorithm 2 Adjustment for Current Monitoring Scheme

Input: Start time $time$ of current sub-time window Γ; State of drones D; State of vessels V (after updating according to AIS).
Output: Updated state of drones.

```
1:  for $i = 1$; $i \leqslant |D|$;$i++$ do
2:      if Speed of $D[i].targetVessel$ is not changed then
3:          continue;
4:      else
5:          $x = D[i].Pos[time]$;
6:          $y = V[D[i].targetVessel].Pos$;
7:          $flag, EAT \leftarrow$ reachability($x$, $y$);
8:          if $flag == False$ then
9:              Trace back the status of $D[i]$ to $time$;
10:         else
11:             Adjust the monitoring scheme of $D[i]$;
12:         end if
13:     end if
14: end for
15: return Updated state of drones
```

Algorithm 3 Reachability Judgment

Input: φ: Coordinate of the drone at time $time$; υ: Target vessel; TV_υ: Sailing time window of the target vessel; $\boldsymbol{\phi} = \phi_t$: Coordinates of υ within TV_υ (*i.e.*, updated route of the target vessel and ϕ_t is the coordinate of υ at t); q_{time}: Battery remaining power of drone at $time$.
Output: Vessel υ is reachable or not ($True$ or $False$); EAT.

```
1:  for $t = (time + 1)$; $t \leqslant max(TV_\upsilon)$;$t++$ do
2:      if dist($\varphi$,$\phi_t$)$/sd > (t - time)$ then
3:          continue;
4:      else
5:          $RemainingPower \leftarrow q_{time} - \delta -$ (dist($\varphi$,$\phi_t$)$/sd$);
6:          $S_{nearest}, flightTime \leftarrow$ NearestStation($\upsilon$,$t + \delta$);
7:          if $RemainingPower - flightTime < 0$ then
8:              continue;
9:          else
10:             return $True$, $t$;
11:         end if
12:     end if
13: end for
14: return $False$, 0
```

B. Key Components

1) Time Window Slice: In the actual management of ECA, empirically speaking, the time required for the drone to complete an emission monitoring task is about 10-20 minutes (travel time consumption+monitoring execution time consumption). Shortening the length of the sub-time window as much as possible and updating vessel information in time can make the ideal speed of the vessel in the model more closely match the actual speed. However, a too-small time window may cause the drone to only perform the monitoring task for one vessel in each sub-time window or even the execution time of the task assigned to the drone spans multiple sub-

Algorithm 4 Q-Learning-Based Scheduling: Training Process

```
Input: $I$: Number of episodes; $D$: Set of drones; $V$: Set of vessels; $\Gamma$: Current sub time window; $time$: Start time of $\Gamma$; $Tabu$: Tabu list; $\epsilon$: Greedy factor; $\gamma_1$, $\gamma_2$: Discount factors; $Heap$: Min-heap.
Output: Trained Q-learning scheduling policy.
1:  Initialize $Q$ table; // Q function
2:  while $terminate\ condition$ is not met do
3:      Load $Heap$ and $Tabu$;
4:      while $Heap$ !=$\varnothing$ do
5:          $drone \leftarrow Heap.get()$; // Pop out the drone at the root of heap
6:          $state \triangleq \{\max(drone.readytime, time), Tabu\}$; // Define the state
7:          if $Q(state, :) < 0$ then
8:              $x = drone.targetVessel$;
9:              $y$=max($drone.readytime$, $time$);
10:             $action, flightTime \leftarrow$ Neareststation($x$, $y$);
11:         else
12:             $action, flightTime \leftarrow$ Greedy($Q(state, :)$, $\epsilon$);
13:         end if
14:         Calculate $reward$ according to $action$ and $flightTime$;
15:         Update $Tabu$;
16:         Update $drone.targetVessel$, $drone.readytime$;
17:         $\gamma \leftarrow \gamma_1$;
18:         if $drone.readytime \leqslant time + \Delta$ then
19:             $Heap.push(drone)$;
20:             $\gamma \leftarrow \gamma_2$;
21:         end if
22:         $drone \leftarrow Heap.top()$; // Get the drone at the root of heap (not delete);
23:         if $\max(drone.readytime.time+1) \geqslant time + \Delta$ then
24:             break;
25:         end if
26:         $state^{\#} \triangleq \{\max(drone.readytime, time + 1), Tabu\}$; // Define the next state
27:         $action^{\#}, flightTime^{\#} \leftarrow$ Greedy($Q(state, :)$, 1);
28:         $Q(state, action) = Q(state, action) + \alpha[reward + \gamma Q(state^{\#}, action^{\#}) - Q(state, action)]$;
29:     end while
30: end while
31: return Scheduler $Q$
```

time windows. It will reduce the performance of the designed algorithm for the dynamic optimization and dispatching of the drone monitoring scheme in each sub-time window. On the contrary, a too-large sub-time window will cause the error between the ideal speed and the actual speed of the vessel to accumulate. From an extreme point of view, if the time interval is not sliced, D-DSP eventually degenerates into a scheduling problem for the complete time interval (*i.e.,* native DSP).

Therefore, the time scale of the sub-time window slice is important to the performance of the algorithm. Considering the time scale of the drone to perform the monitoring task, in the paper, 30 minutes is selected as the length of each time window. This time length takes into account the effectiveness of the scheduling algorithm and the timeliness requirements generated by the dispatching plan within a single sub-time window.

2) Adjustment for Current Monitoring Scheme: The main purpose of the adjustment for the monitoring scheme is to check the current dispatch plan of drones. It is required to update the flight direction of those drones that can perform the original mission and the time of meeting with the target vessels. Besides, these original missions that cannot be performed should be identified for further processing. The detailed steps of the adjustment are shown in Alg. 2.

First, the speed of each target vessel of each drone is checked whether changed. If not changed, no update is required (Lines 2-3). Otherwise, it is necessary to update the status of the drone. Then, according to the position of i-th drone at the time $time$ and the updated route of the j-th vessel (the original target vessel of i-th drone), it is judged whether the i-th drone is capable to continue to conduct the monitoring task (Lines 5-7). If yes, adjust the monitoring scheme of the i-th drone to adapt to the updated vessel status (Line 11). Otherwise, cancel the original monitoring scheme and trace the state of the i-th drone back to the state at the beginning of the current sub-time window (Line 9).

The sub-function reachability$(\cdot,\cdot)$ in Alg. 2 is responsible for judging the reachability between the drone and the target vessel. It has two return values: $flag$ indicates whether it is reachable ($True$ is reachable, $False$ is unreachable); EAT (earliest arrival time) is the earliest time that the drone could meet the target vessel. The reachability judgment depends on whether the remaining power of the drone supports it to complete the current task and then whether there is enough remaining power of the drone to support its return to the nearest drone station after the task is completed. If one of the two conditions is not met, it is judged to be unreachable. The detailed process of reachability judgment is shown in Alg. 3. In Alg. 3, NearestStation(i,t) outputs the drone station that nearest to the i at time t and the flight time required to fly to the nearest drone station.

3) Q-Learning for Re-Scheduling: Due to the strict requirement on the solving time, the trade-off between running time and algorithm performance is an important problem for solving D-DSP. Although the meta-heuristic algorithm has been illustrated to be able to solve DSP effectively, the time-consuming of it is too long, which makes it unable to meet the requirements of scheduling time. Compared with the meta-heuristic algorithm, RL has the characteristics of high convergence efficiency, and could stably obtain higher quality results in a shorter running time. In the paper, therefore, Q-learning is designed as the scheduling solver. Q-learning is a kind of algorithm that uses the main body of the algorithm (agent) to interact with the environment (environment) autonomously and then take different actions (action, a) according to the policy in different states (state, s). Then, the Q-function is continuously updated according to the obtained reward (reward, r), in the hope that the function eventually converges to the real reward

TABLE I
DEFINITION OF STATE-ACTION IN THE PROPOSED APPROACH

Element	Detailed definition
State, s	Start time of monitoring task and access status of vessels
Action, a	No. of the target vessel
Reward, r	Ratio of the monitoring weight of the target vessel to the corresponding flight distance. If the current endurance of the drone cannot reach the target vessel, it is set to -100.

prediction function [40], [41]. In DSP, RL has to face a fact when formulating the scheduling scheme for multiple drones, task execution status of a drone before a certain moment will affect the environment of other drones after that moment during their task execution. *i.e.,* The time-related monitoring schemes of the drones affect each other. Therefore, in the proposed method, time information is used as one of the elements of the environmental state in the RL. The complete state-action definition is shown in Table I.

In addition, in the scheduling process, it is necessary to determine which drone to make a monitoring scheme according to the time-space sequence of the $readytime$ for each drone that could conduct the next task. Specifically, the drone with the earliest $readytime$ is selected as the agent. Then, the policy will develop a monitoring scheme for the agent. To achieve the drone with the earliest $readytime$, min-heap technology is introduced. Min-heap can organize an orderly sequence of drones to be planned by using the $readytime$ of each drone as the sorting index and the sorting method is descending). The $readytime$ of the drone at the root node of the min-heap is always small than that of all other drones. The detailed implementation processes are as follows. At the beginning of each sub-time window, all drones whose $readytime$ is within the time window will be put into the min-heap. Then, the drones to be planned are continuously popped up according to the sequence of $readytime$.

The training process and scheduling process of the Q-learning-based drone scheduling are illustrated as Algs. 4 and 5, respectively.

The training process of the Q-learning-based drone scheduler is illustrated as Alg. 4. In the initial stage, the Q table is initialized, and the initial value is set as the $reward$ in Table I (Line 1). Then, the Q table is trained until the termination condition (maximum number of episodes) is met (Lines 2-30).

In each episode, min-heap $Heap$ and Tabu table $Tabu$ (Line 3) are first loaded. Then, pop out the drone with the shortest $readytime$ from $Heap$ (*i.e.,* read the information of the drone and delete the drone from $Heap$) continuously and use it as the scheduling object until $Heap$ is empty (Lines 4-29). For each drone with the shortest $readytime$ popped out from $Heap$ in turn, update $state$ according to the $readytime$ and current $time$ of the obtained new drone (Lines 5-6). Also, the expected rewards of all actions in a given state can be obtained from the Q table. If they are all negative, it means that the drone must return to the nearest drone station (Lines 7-10). Otherwise, select the next vessel to be monitored based on the ϵ-greedy strategy, where ϵ-greedy is a probabilistic greedy function (Line 12). After selecting

Algorithm 5 Q-Learning-Based Scheduling: Scheduling Process

```
Input: $Q$: Scheduler trained by Q-learning; $D$: Set of drones; $V$: Set of vessels; $\Gamma$: Current sub time window; $time$: Start time of $\Gamma$; $Tabu$: Tabu list; $\epsilon$: Greedy factor; $Heap$: Min-heap.
Output: Monitoring scheme of $\Gamma$.
1: Load $Heap$ and $Tabu$;
2: $scheme \leftarrow \varnothing$
3: while $Heap$ !=$\varnothing$ do
4:     $drone \leftarrow Heap.get()$;
5:     $state \triangleq \{\max(drone.readytime, time), Tabu\}$;
6:     if $Q(state,:) < 0$ then
7:         $x = drone.targetVessel$;
8:         $y$=max($drone.readytime, time$);
9:         $action, flightTime \leftarrow$ Neareststation($x$, $y$);
10:    else
11:        $action, flightTime \leftarrow$ Greedy($Q(state,:)$, $\epsilon$);
12:    end if
13:    $scheme \leftarrow \{scheme, action\}$;
14:    Calculate $reward$ according to $action$ and $flightTime$;
15:    Update $Tabu$;
16:    Update $drone.targetVessel$, $drone.readytime$;
17:    if $drone.readytime \leqslant time + \Delta$ then
18:        $Heap.push(drone)$;
19:    end if
20: end while
21: return $scheme$
```

the target vessel, calculate the current reward obtained and update $Tabu$ and the internal parameters of the current drone object (Lines 14-16). If the $readytime$ of the updated drone object has not exceeded the upper limit of the current sub-time window, it indicates that the drone may continue to conduct the monitoring task within this sub-time window. Therefore, push the drone back into $Heap$ (Line 19). Next, read the state information of the drone that is currently at the root node (Line 22). It should be noted that for this drone, only its information is read, but it is not popped out from $Heap$. According to the $readytime$ and current $time$ of the read new drone, pure greedy strategy, *i.e.,* taking the greatest expected reward, is adopted to select the next action for the current drone (Line 27). Finally, the Q table is updated (Line 28). Besides, the discount factors in the different situations are differentiated. If the estimated completion time of the action selected by the policy is within the current sub-time window, a large discount value (*i.e.,* γ_2) is obtained, otherwise, a small discount value (*i.e.,* γ_1) is obtained. The mechanism is used to can encourage the policy to select monitoring targets in the current sub-time window, thereby increasing the possibility of the drone fleet monitoring more vessels within the allowed time window.

Alg. 5 shows the process of using the trained policy to plan the monitoring task.

IV. Numerical Experiment

In this section, the effectiveness of the proposed RL-based dynamic dispatching strategy is verified through comparative experiments. All algorithms are implanted in Python 3.8, and the experiments are performed on an Intel i9-10900K(3.7GHz) Desktop PC with 32 GB RAM.

A. Experiment Design and Data Description

1) Comparison Methods: In D-DSP, whether the scheduling algorithm can output effective scheduling results in a relatively short time is the most important indicator to evaluate the performance of the algorithm, that is, fast response-ability of the algorithm to dynamic scenes. The available scheduling algorithms mainly include (1) Rule-based greedy algorithms, which can output the scheduling results in a very short time, but the quality of the scheduling result is extremely dependent on the rule; (2) Evolutionary algorithms are represented by the ant colony optimization (ACO) algorithm, as this type of algorithm has been shown efficient in solving DSP [25] However, as a kind of random algorithm, the quality of the scheduling result depends to a certain extent on the allowable running time of the algorithm; (3) RL algorithms, which are more balanced in terms of running speed and solution quality.

In the paper, two greedy algorithms and ACO-based methods are selected as comparison methods to illustrate the effectiveness of the RL-based dispatching strategy proposed in the paper. The two greedy algorithms are denoted as Greedy1 and Greedy2. The dispatch rule of Greedy1 is: among vessels that have not been monitored, select the vessel with the largest ratio of monitoring reward to flying distance. The dispatch rule of Greedy2 is: among vessels that have not been monitored, select the vessel with the largest monitoring reward. If there are several vessels with the same monitoring reward, the vessel with the shortest flight distance is selected. The details of the ACO-based approach for comparison can be seen in [25].

The parameter setting of the RL-based approach can be seen in Table II. The learning rate is represented by lr, which can control the iterative speed. The setting of this parameter follows the recommended value. The greedy factor is represented by e_{greed}. The parameter controls the exploration behavior of the algorithm. The larger the factor, the weaker the exploration ability. To enhance the exploration ability of the algorithm, it is increased appropriately based on the recommended value of the algorithm. The maximum number of episodes of the algorithm is represented by I. To ensure the speed of the algorithm, a relatively small value is taken. The discount factors γ_1 and γ_2 represent the impact of future rewards on current rewards, where γ_2 is set according to the recommended value and the value of γ_1 is reduced to give the algorithm a certain negative incentive.

2) Data Description: In the paper, the ECA of the Pearl River Delta in China is used as an experimental simulation scenario. The area mainly includes the three ports of Shenzhen Yantian Port, Chiwan Port, and Hong Kong Port. According to data released by the Shenzhen Maritime Safety Administration, the number of vessels entering and leaving the port in Aug. 2021 is 15610. After conversion, the average number of vessels entering the port per hour is about 10.5. The total monitoring time interval is set to 5 hours, and 80 vessels are set as the upper limit of the monitoring demand according to the demand of the actual scene, *i.e.,* there are up to 80 vessels sailing within ECA within the total monitoring time window. The dataset of number of vessels is set as $\{20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80\}$. Considering that in actual monitoring scenarios, due to cost issues, the number of drones that can be called is small. Therefore, it is assumed that the upper limit of the number of drone stations is 2, and the number of drones in each drone station is 2. To ensure the rigor and reliability of the experiment, two sets of vessel sailing data are generated for algorithm performance testing and comparison (*i.e.,* Dataset A and B). Besides, at the starting point of each sub-time window, the normal distribution is used to adjust vessel speed to simulate vessel speed fluctuation. According to the investigation of the port, it is learned that the fuel consumption rate of a vessel is proportional to the third power of the speed, which causes it to balance the efficiency and cost of sailing by keeping the vessel speed as low as possible and keeping a uniform speed. Therefore, vessels usually do not actively make speed changes. If the speed changes, it is mainly caused by the influence of wind and waves. According to experience, this factor generally does not change the sailing speed by more than 10%. This paper uses 10% as the upper limit of the vessel speed fluctuation rate.

The summary of various parameters in the experiment is shown in Table III. Fig. 4 shows the ECA scenario used for numerical experiments.

TABLE II
Parameters Setting of the Proposed Approach

Parameters	Description	Value
lr	Learning rate	0.90
e_{greed}	Greedy factor	0.40
I	Maximum number of episodes	1000
γ_1	Discount factor 1	0.40
γ_2	Discount factor 2	0.90

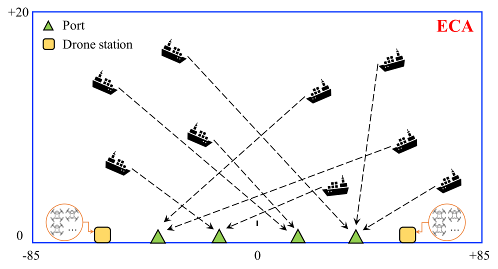

Fig. 4. Schematic diagram of the ECA.

B. Experiment I: Impact of Vessel Speed Fluctuation

Before verifying and analyzing the effectiveness of the algorithm, the impact of vessel speed fluctuation on the dispatching system should be illustrated to show the actual value of

TABLE III
EXPERIMENTAL PARAMETERS

Related to	Description	Parameter setting
Basic parameters of monitoring task	Width of the monitoring area	[-85,85] nautical mile
	Length of the monitoring area	[0,20] nautical mile
	Port location	[40,0], [20,0], [0,0]
	Drone station location	[0,0],[30,0]
	Time interval of dispatching	300 minutes
	Sliced time window length	30 minutes
Parameters of vessel	Vessel speed	5-10 knots
	Upper limit of speed fluctuation	10%
	Vessel monitoring weight	5-15
	Number of vessels	{20,25,30,35,40,45,50,55,60,65,70,75,80}
Parameters of drone	Drone flight speed	30 knots
	Number of drones in drone station	2
	Monitoring time consumption	5 minutes
	Battery replacement time consumption	5 minutes
	Maximum battery endurance	120 minutes
	Number of drone stations	{1,2}

TABLE IV
THE EXPERIMENTAL RESULTS OF EXPERIMENT I

Instance	Metrics	Upper limit of speed fluctuation rate						
		1%	2%	5%	10%	15%	20%	25%
V70K2	Number of drones arriving early	2	22	72	152	183	196	189
	Number of drones arriving late	1	30	79	119	134	180	193
	Average monitoring completion rate	99.7%	95.3%	85.2%	72.4%	67.0%	60.9%	60.3%
V80K2	Number of drones arriving early	5	41	186	253	320	346	371
	Number of drones arriving late	7	43	153	254	266	298	336
	Average monitoring completion rate	99.0%	93.2%	72.1%	58.5%	51.3%	46.9%	41.1%

our work. In this experiment, two datasets (V70K2, V80K2) are used as test cases. VxKy represents a case where there are x vessels that need to be monitored and there are y drone stations. Besides, in the initial situation, there are 2 drones arranged at each drone station. Under the two problem scales, different upper limit values of the vessel speed variation range are set for testing. To reduce the impact of the randomness of data generation, 20 sets of vessel data are generated under each setting. The number of drones arriving early, the number of drones arriving late, and the average monitoring completion rate are used as the evaluation indicators to demonstrate the impact of vessel speed fluctuation on the dispatching system. The first two evaluation indicators refer to the failure of the task shown in Fig. 2(a) and Fig. 2(b). The values of the two indicators in the table are the sum of the number of occurrences of the above two situations all 20 sets of vessel data. The average monitoring completion rate represents the ratio of the vessel monitoring rewards actually completed in the 20 sets of vessel speed change data to the total monitoring rewards theoretically obtained according to the original monitoring scheme. The higher the ratio, the degree of completion of the original monitoring scheme. The experimental results are shown in Table IV.

Through the experimental results, it can be found that when the vessel speed change rate fluctuates within 2%, the average monitoring completion rate of the original monitoring scheme could reach more than 90%, and it still has a good monitoring performance. When the upper limit of vessel speed fluctuation is further increased, the monitoring completion rate drops significantly. It is speculated that the reason for the phenomenon is that as the speed fluctuation range increases, the change of the vessel trajectory becomes more serious, causing the deviation between the actual vessel trajectory and the original trajectory to accumulate. Eventually, the original monitoring scheme is invalidated.

Experiments show that when the vessel speed changes significantly, the negative impact of the vessel speed fluctuation on the dispatching system is very obvious. Therefore, the development of a dynamic dispatch algorithm that could respond to the dynamic factors of vessel speed fluctuation promptly is very important to improve the robustness of the dispatching system.

C. Experiment II: Experiment on Non-Intensive Scenarios

In this experiment, the effectiveness of the proposed method is illustrated by comparing the performance with three comparison methods (see Section IV-A) on two datasets. Among them, the upper limit of the time consumption of the monitoring scheme in a single sub-time window of the ACO-based and RL-based method planning (*i.e.,* the length of time that the algorithm is allowed to run) is set to $V * K^2/100$ to simulate the pressure of dynamic problems on the timeliness of the solution output by the dispatching system. If the algorithm fails to complete all iterations within the upper limit, the algorithm is interrupted and the optimal monitoring scheme currently found is output. For example, in the v80k2 scenario, the upper limit of the running time in the sub-time window is $80 * 2^2/100 = 3.2$ seconds. Since the running time of the above-mentioned algorithms is limited to a small value, the

TABLE V
EXPERIMENTAL RESULTS OF EXPERIMENT II (DATASET A)

Instance		Greedy1	Greedy2	ACO-based		RL-based	
$\|V\|$	$\|K\|$	Mean±Std.	Mean±Std.	Mean±Std.	Time (s)	Mean±Std.	Time (s)
20	1	103.00 ± 0.00(≈)	84.00 ± 0.00(-)	97.00 ± 0.00(-)	0.20	105.00 ± 3.85(+)	0.20
20	2	136.00 ± 0.00(≈)	123.00 ± 0.00(-)	142.50 ± 1.50(+)	0.80	142.30 ± 2.10(≈)	0.80
25	1	123.00 ± 0.00(-)	105.00 ± 0.00(-)	120.50 ± 10.71(-)	0.25	130.00 ± 0.00(+)	0.25
25	2	177.00 ± 0.00(≈)	144.00 ± 0.00(-)	182.20 ± 10.93(+)	1.00	180.15 ± 3.48(≈)	1.00
30	1	145.00 ± 0.00(-)	142.00 ± 0.00(-)	153.65 ± 4.20(+)	0.30	151.80 ± 4.91(≈)	0.30
30	2	238.00 ± 0.00(≈)	243.00 ± 0.00(+)	241.20 ± 5.57(≈)	1.20	242.45 ± 9.19(≈)	1.20
35	1	163.00 ± 0.00(≈)	143.00 ± 0.00(-)	134.70 ± 19.58(-)	0.35	165.80 ± 2.60(+)	0.35
35	2	289.00 ± 0.00(≈)	254.00 ± 0.00(-)	283.05 ± 11.06(-)	1.40	302.05 ± 9.12(+)	1.40
40	1	168.00 ± 0.00(-)	138.00 ± 0.00(-)	171.30 ± 5.67(-)	0.40	180.80 ± 1.86(+)	0.40
40	2	297.00 ± 0.00(≈)	230.00 ± 0.00(-)	288.20 ± 6.52(-)	1.60	307.20 ± 5.43(+)	1.60
45	1	213.00 ± 0.00(≈)	173.00 ± 0.00(-)	206.95 ± 11.20(≈)	0.45	215.35 ± 5.21(+)	0.45
45	2	370.00 ± 0.00(-)	329.00 ± 0.00(-)	383.00 ± 14.52(≈)	1.80	390.35 ± 15.18(+)	1.80
50	1	240.00 ± 0.00(-)	223.00 ± 0.00(-)	233.15 ± 9.14(-)	0.50	254.60 ± 7.57(+)	0.50
50	2	420.00 ± 0.00(≈)	350.00 ± 0.00(-)	426.15 ± 18.42(≈)	2.00	431.55 ± 9.17(+)	2.00
55	1	209.00 ± 0.00(≈)	184.00 ± 0.00(-)	214.50 ± 17.37(≈)	0.55	219.75 ± 6.50(+)	0.55
55	2	440.00 ± 0.00(≈)	362.00 ± 0.00(-)	433.35 ± 15.19(≈)	2.20	446.40 ± 15.58(+)	2.20
60	1	255.00 ± 0.00(-)	200.00 ± 0.00(-)	253.00 ± 16.56(-)	0.60	284.00 ± 16.97(+)	0.60
60	2	437.00 ± 0.00(≈)	351.00 ± 0.00(-)	454.55 ± 24.24(+)	2.40	453.35 ± 13.21(≈)	2.40
65	1	271.00 ± 0.00(≈)	261.00 ± 0.00(-)	273.30 ± 22.44(≈)	0.65	280.15 ± 10.13(+)	0.65
65	2	455.00 ± 0.00(≈)	416.00 ± 0.00(-)	476.75 ± 23.46(+)	2.60	465.10 ± 24.75(≈)	2.60
70	1	279.00 ± 0.00(≈)	202.00 ± 0.00(-)	255.50 ± 18.90(-)	0.70	291.75 ± 5.93(+)	0.70
70	2	495.00 ± 0.00(≈)	419.00 ± 0.00(-)	479.55 ± 23.51(-)	2.80	508.70 ± 11.66(+)	2.80
75	1	300.00 ± 0.00(-)	221.00 ± 0.00(-)	319.10 ± 13.99(-)	0.75	346.40 ± 18.60(+)	0.75
75	2	547.00 ± 0.00(≈)	412.00 ± 0.00(-)	530.20 ± 25.26(-)	3.00	564.30 ± 16.69(+)	3.00
80	1	207.00 ± 0.00(-)	172.00 ± 0.00(-)	222.85 ± 19.45(-)	0.80	241.05 ± 5.37(+)	0.80
80	2	441.00 ± 0.00(-)	355.00 ± 0.00(-)	473.85 ± 20.36(+)	3.20	462.80 ± 25.82(≈)	3.20
+/≈/-		0/17/9	1/0/25	6/7/13		19/7/0	

TABLE VI
EXPERIMENTAL RESULTS OF EXPERIMENT II (DATASET B)

Instance		Greedy1	Greedy2	ACO-based		RL-based	
$\|V\|$	$\|K\|$	Mean±Std.	Mean±Std.	Mean±Std.	Time (s)	Mean±Std.	Time (s)
20	1	134.00 ± 0.00(≈)	133.00 ± 0.00(≈)	134.20 ± 6.98(≈)	0.20	140.00 ± 4.90(+)	0.20
20	2	195.00 ± 0.00(+)	177.00 ± 0.00(-)	195.00 ± 0.00(+)	0.80	195.00 ± 0.00(+)	0.80
25	1	91.00 ± 0.00(≈)	81.00 ± 0.00(-)	92.20 ± 3.60(+)	0.25	91.00 ± 0.00(≈)	0.25
25	2	179.00 ± 0.00(≈)	174.00 ± 0.00(-)	178.00 ± 0.00(≈)	1.00	186.00 ± 0.00(+)	1.00
30	1	184.00 ± 0.00(≈)	162.00 ± 0.00(-)	177.45 ± 7.28(≈)	0.30	186.60 ± 2.63(+)	0.30
30	2	228.00 ± 0.00(-)	215.00 ± 0.00(-)	249.10 ± 7.17(≈)	1.20	253.50 ± 14.61(+)	1.20
35	1	132.00 ± 0.00(≈)	130.00 ± 0.00(-)	98.00 ± 0.00(-)	0.35	137.70 ± 1.31(+)	0.35
35	2	230.00 ± 0.00(≈)	208.00 ± 0.00(-)	237.30 ± 15.44(+)	1.40	230.85 ± 6.75(≈)	1.40
40	1	221.00 ± 0.00(≈)	173.00 ± 0.00(-)	205.90 ± 12.01(-)	0.40	224.40 ± 4.10(+)	0.40
40	2	340.00 ± 0.00(≈)	295.00 ± 0.00(-)	351.35 ± 11.86(≈)	1.60	353.85 ± 13.08(+)	1.60
45	1	180.00 ± 0.00(-)	178.00 ± 0.00(-)	169.05 ± 15.88(-)	0.45	190.35 ± 1.53(+)	0.45
45	2	323.00 ± 0.00(≈)	280.00 ± 0.00(-)	322.95 ± 14.06(≈)	1.80	324.50 ± 5.34(+)	1.80
50	1	195.00 ± 0.00(≈)	152.00 ± 0.00(-)	173.40 ± 40.59(-)	0.50	196.15 ± 8.63(+)	0.50
50	2	381.00 ± 0.00(-)	307.00 ± 0.00(-)	406.30 ± 13.65(≈)	2.00	407.25 ± 11.94(+)	2.00
55	1	215.00 ± 0.00(-)	168.00 ± 0.00(-)	206.15 ± 17.51(-)	0.55	231.35 ± 12.63(+)	0.55
55	2	398.00 ± 0.00(-)	321.00 ± 0.00(-)	442.75 ± 36.42(≈)	2.20	449.85 ± 17.49(+)	2.20
60	1	230.00 ± 0.00(≈)	201.00 ± 0.00(-)	215.85 ± 14.96(-)	0.60	238.30 ± 7.68(+)	0.60
60	2	444.00 ± 0.00(≈)	342.00 ± 0.00(-)	448.95 ± 28.40(+)	2.40	448.50 ± 15.11(≈)	2.40
65	1	218.00 ± 0.00(-)	188.00 ± 0.00(-)	219.15 ± 19.89(≈)	0.65	230.55 ± 5.44(+)	0.65
65	2	465.00 ± 0.00(≈)	372.00 ± 0.00(-)	467.40 ± 13.79(≈)	2.60	477.60 ± 18.27(+)	2.60
70	1	245.00 ± 0.00(-)	180.00 ± 0.00(-)	239.10 ± 25.09(-)	0.70	279.20 ± 11.34(+)	0.70
70	2	418.00 ± 0.00(-)	330.00 ± 0.00(-)	438.95 ± 24.38(≈)	2.80	441.00 ± 12.28(+)	2.80
75	1	209.00 ± 0.00(-)	187.00 ± 0.00(-)	205.75 ± 13.59(-)	0.75	235.85 ± 16.66(+)	0.75
75	2	474.00 ± 0.00(≈)	361.00 ± 0.00(-)	450.15 ± 15.15(-)	3.00	490.25 ± 25.14(+)	3.00
80	1	307.00 ± 0.00(-)	225.00 ± 0.00(-)	311.70 ± 13.57(≈)	0.80	325.60 ± 10.07(+)	0.80
80	2	531.00 ± 0.00(≈)	420.00 ± 0.00(-)	538.25 ± 29.82(≈)	3.20	549.40 ± 20.13(+)	3.20
+/≈/-		1/5/10	0/1/25	4/13/9		23/3/0	

running time of different algorithms is not used as an index for comparative evaluation. In addition, to avoid the randomness of the algorithm, ACO-based and RL-based methods are run 20 times in each case. The other two greedy algorithms are only run one time because they are deterministic algorithms. The experimental results on datasets A and B are reported in Tables V and VI, respectively. "+" indicates the optimal algorithm in a case; "≈" indicates that the error between the result of the corresponding algorithm and the result of the optimal algorithm is within 5%; "-" indicates the error between the result of the corresponding algorithm and the result of the optimal algorithm is beyond 5%.

The experimental results show that the performance of the RL-based method on two datasets, a total of 52 cases, has achieved obvious advantages over the comparison method. Among them, the RL-based method is better than the comparison method in 42 cases, and the error from the optimal result is within 5% in 10 cases.

TABLE VII
EXPERIMENTAL RESULTS OF EXPERIMENT III

Instance		Greedy1	Greedy2	ACO-based		RL-based	
$\|V\|$	$\|K\|$	Mean±Std.	Mean±Std.	Mean±Std.	Time (s)	Mean±Std.	Time (s)
90	1	347.00 ± 0.00(≈)	242.00 ± 0.00(-)	311.70 ± 15.59(-)	0.90	355.25 ± 10.98(+)	0.90
90	2	607.00 ± 0.00(≈)	408.00 ± 0.00(-)	567.05 ± 24.90(-)	3.60	624.25 ± 13.56(+)	3.60
100	1	302.00 ± 0.00(≈)	203.00 ± 0.00(-)	250.75 ± 41.76(-)	1.00	306.25 ± 10.56(+)	1.00
100	2	581.00 ± 0.00(≈)	363.00 ± 0.00(-)	554.50 ± 40.07(-)	4.00	606.50 ± 32.46(+)	4.00
110	1	351.00 ± 0.00(≈)	241.00 ± 0.00(-)	331.90 ± 17.22(-)	1.10	356.80 ± 11.10(+)	1.10
110	2	675.00 ± 0.00(+)	476.00 ± 0.00(-)	601.70 ± 28.08(-)	4.40	654.65 ± 23.00(≈)	4.40
120	1	334.00 ± 0.00(≈)	212.00 ± 0.00(-)	304.05 ± 18.70(-)	1.20	349.55 ± 10.15(+)	1.20
120	2	715.00 ± 0.00(+)	433.00 ± 0.00(-)	623.05 ± 24.57(-)	4.80	701.00 ± 22.31(≈)	4.80
130	1	406.00 ± 0.00(+)	261.00 ± 0.00(-)	340.80 ± 25.08(-)	1.30	398.85 ± 13.91(≈)	1.30
130	2	723.00 ± 0.00(+)	415.00 ± 0.00(-)	625.75 ± 36.79(-)	5.20	698.75 ± 19.04(≈)	5.20
140	1	412.00 ± 0.00(+)	236.00 ± 0.00(-)	329.00 ± 25.65(-)	1.40	410.90 ± 12.43(≈)	1.40
140	2	731.00 ± 0.00(≈)	511.00 ± 0.00(-)	670.20 ± 46.38(-)	5.60	732.40 ± 16.16(+)	5.60
150	1	381.00 ± 0.00(+)	246.00 ± 0.00(-)	343.30 ± 18.89(-)	1.50	380.45 ± 10.37(≈)	1.50
150	2	714.00 ± 0.00(≈)	515.00 ± 0.00(-)	688.85 ± 25.48(-)	6.00	737.30 ± 20.78(+)	6.00
160	1	413.00 ± 0.00(≈)	323.00 ± 0.00(-)	382.25 ± 64.48(-)	1.60	429.95 ± 14.12(+)	1.60
160	2	830.00 ± 0.00(≈)	633.00 ± 0.00(-)	744.00 ± 35.56(-)	6.40	833.15 ± 19.00(+)	6.40
170	1	501.00 ± 0.00(≈)	340.00 ± 0.00(-)	460.20 ± 38.29(-)	1.70	514.25 ± 21.40(+)	1.70
170	2	883.00 ± 0.00(+)	533.00 ± 0.00(-)	765.65 ± 37.01(-)	6.80	830.25 ± 34.47(-)	6.80
180	1	441.00 ± 0.00(+)	278.00 ± 0.00(-)	392.25 ± 21.29(-)	1.80	441.00 ± 15.12(+)	1.80
180	2	811.00 ± 0.00(≈)	477.00 ± 0.00(-)	717.45 ± 44.20(-)	7.20	812.35 ± 17.84(+)	7.20
+/≈/-		8/12/0	0/0/20	0/0/20		13/6/1	

Compared with Greedy1, the RL-based method obtains significantly better results than the algorithm in 98% of the cases; and compared to Greedy2, the RL-based method also obtains significantly better results than the algorithm in 98% of the cases. It can be said that the dispatching strategy proposed in the paper has obvious advantages over the greedy algorithm in solving D-DSP. The reason is that Greedy algorithms can only take the benefits of one step in the future, which is short-sighted. The algorithm proposed in the paper is more exploratory and could further evolve subsequent events and make better choices. Although the greedy algorithm has obvious real-time advantages, the speed of the RL-based method to obtain the monitoring scheme for each sub-time window can also fully meet the needs of dynamic scheduling of drones in actual scenarios, and therefore has an obvious practical value.

Compared with the ACO-based method, the RL-based method can obtain significantly better results than the ACO-based method in 80% of the cases under the same calculation time. Also, in the remaining cases, it obtains the scheduling results close to the ACO-based method. In summary, the RL-based method has better solution quality than the ACO-based method when solving D-DSP. The main reason for the phenomenon is that the strict execution time requirement of the algorithm required a limited number of iterations that the algorithm could run, which leads to the poorer stability of the ACO-based method in solving due to the randomness. From the experimental comparison results, the RL-based method proposed in the paper can achieve better results in most cases than the comparison method.

D. Experiment III: Discussion on the Intensive Scenarios

In Experiment II, the number of vessels in ECA is set according to the actual throughput data of the port, which illustrates the feasibility of the method proposed in the practical application through the results of numerical experiments. It should be noted that the proposed RL-based method to dispatch drones is based on the actual throughput of the port. At the level of theoretical research, it is necessary to discuss the applicability of the method proposed when the number of vessels increases sharply. Consistent with previous experiments, each algorithm is run 20 times in each case. The experimental results are reported in Table VII.

When there are a large number of sailing vessels in ECA, it means that the locations of vessels in this sea area are more densely distributed. The experimental results show that the RL-based method is still the optimal strategy. But Greedy1 is also a promising strategy, which is different from the conclusion of Experiment II. The reason for the phenomenon is that the RL-based method is trying to find the optimal solution in different sub-time windows, so it may not hesitate to go near and far to obtain a larger monitoring reward, causing sometimes the planned optimal monitoring solution has to span multiple sub-time windows. However, when the monitoring task of the drone crosses from the previous sub-time window into the next sub-time window, its flight direction may be adjusted due to the fluctuation of the target vessel speed, and it may even need to travel to abandon the original target vessel in the middle and reselect the target vessel with the current location as the starting point. This behavior will waste the energy consumption of the drone and make the algorithm easy to fall into a locally optimal solution. The experimental results show that in dense scenarios, a planning method tends to be greedy algorithms or the advanced rule-based method is recommended. It can be said that the greedy strategy is suitable for solving D-DSP in this scenario.

Besides, it can be seen from the table that the ACO-based method performs particularly poorly, which illustrates that evolutionary algorithms maybe not be suitable for the dynamic scheduling of complex problems. It also verifies the rationality of choosing RL as the scheduler for solving dynamic problems.

E. Discussion

We believe that the above three experiments have shown three important conclusions in designing problem-solving strategies for D-DSP.

Firstly, the vessel speed fluctuation has an important impact on the effectiveness of the dispatching system. The given results in Experiment I show that even if the speed fluctuation is only 5-10%, its impact on the dispatching system is also serious. Therefore, taking vessel speed fluctuation factors into account in ECA emission monitoring issues is important to practice.

Secondly, the number of vessels in ECA has an important influence on the formulation of dispatching strategies. Since the number of vessels is not fixed in ECA, the evaluation of the density of vessels is the first thing that must be considered before solving the D-DSP problem. For the case of the low density of the vessel, the RL-based method has the advantages of lightweight calculation and good stability, which is a good solution strategy. For the case of the high density of the vessel, as the dynamic changes of vessels become more complicated and the scale of the scheduling problem becomes larger, the greedy strategy may be a good planning strategy.

Finally, the design of dispatching strategies is particularly important. Both experiments II and III clearly illustrate that the planning performance of Greedy1 is significantly better than that of Greedy2. It shows although they are all greedy strategies, a good definition of greedy rule can greatly improve the solution performance. Similarly, the proposed RL-based method proposed for solving characteristics is also significantly better than the ACO-based method. Therefore, designing a more targeted strategy for ECA monitoring business characteristics is the key to improving the performance of the dispatching system.

F. Theoretical and Practical Implications

The development of RL-based solution methods has contributed to the solution of DSP under vessel speed fluctuation at the theoretical level. More importantly, the development of the RL-based method has also contributed to the solution of DSP at the practical level. This is because RL-based methods can generate an effective scheduling scheme in a short time, which satisfies the strict requirements for the agility of the dispatching system in the dynamic scheduling task, and ensures that an effective scheduling scheme can be output within the allowable computing time to cope with the continuous dynamic disturbances and changes in the environment.

V. Conclusion

The paper defines a new emission monitoring problem, namely D-DSP. In the problem, the speed fluctuation of the vessel caused by natural factors such as sea waves and wind speed are considered. To solve the problem, a RL-based approach is proposed. First, the total monitoring window is divided into multiple sub-time windows; then, at the beginning of each sub-time window, the adjustment strategy updates the prior position and speed of each vessel for further dispatching. Next, RL-based strategy is adopted as a scheduler to formulate monitoring schemes in each sub-time window. Numerical experiment results illustrate that the proposed strategy could quickly respond to vessel speed changes and reorganize a high-quality scheduling plan in a timely manner.

Experimental results have also found that the number of vessels in ECA has an important influence on the formulation of dispatching strategies. Potential future work should consider the density of vessels in ECA. More optimization algorithms based on neural networks can also be considered [42]. We believe one potential study for dynamic dispatch is how to evaluate vessel density in ECA. Based on the vessel density, we expect the proposed RL-based strategy could be combined with the greedy strategy to obtain a better dynamic scheduling performance.

References

[1] D. Sheng, Q. Meng, and Z.-C. Li, "Optimal vessel speed and fleet size for industrial shipping services under the emission control area regulation," *Transp. Res. C, Emerg. Technol.*, vol. 105, pp. 37–53, Aug. 2019.

[2] *Review of Maritime Transport*. Accessed: Nov. 9, 2021. [Online]. Available: https://unctad.org/system/files/official-document/rmt2017_en.pdf

[3] *Prevention and Control of Shipping and Port Air Emissions in China*. Accessed: Dec. 1, 2021. [Online]. Available: https://www.nrdc.org/sites/default/files/china-controlling-port-air-emissions-report.pdf

[4] Z. Lv, D. Chen, and Q. Wang, "Diversified technologies in internet of vehicles under intelligent edge computing," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 4, pp. 2048–2059, Apr. 2021.

[5] X. Zhang *et al.*, "Changes in the SO_2 level and $PM_{2.5}$ components in Shanghai driven by implementing the ship emission control policy," *Environ. Sci. Technol.*, vol. 53, no. 19, pp. 11580–11587, 2019.

[6] S. Zheng, Y.-E. Ge, X. Fu, Y. Nie, and C. Xie, "Modeling collusion-proof port emission regulation of cargo-handling activities under incomplete information," *Transp. Res. B, Methodol.*, vol. 104, pp. 543–567, Oct. 2017.

[7] Z. Wan, Q. Zhang, Z. Xu, J. Chen, and Q. Wang, "Impact of emission control areas on atmospheric pollutant emissions from major ocean-going ships entering the Shanghai port, China," *Mar. Pollut. Bull.*, vol. 142, pp. 525–532, May 2019.

[8] L. Huang, Y. Wen, X. Geng, C. Zhou, and C. Xiao, "Integrating multi-source maritime information to estimate ship exhaust emissions under wind, wave and current conditions," *Transp. Res. D, Transport. Environ.*, vol. 59, pp. 148–159, Mar. 2018.

[9] K. Fagerholt, N. T. Gausel, J. G. Rakke, and H. N. Psaraftis, "Maritime routing and speed optimization with emission control areas," *Transp. Res. C, Emerg. Technol.*, vol. 52, pp. 57–73, Mar. 2015.

[10] Q. Zhang, Z. Zheng, Z. Wan, and S. Zheng, "Does emission control area policy reduce sulfur dioxides concentration in shanghai?" *Transp. Res. D, Transp. Environ.*, vol. 81, Apr. 2020, Art. no. 102289.

[11] J. Moldanová *et al.*, "Physical and chemical characterisation of PM emissions from two ships operating in European emission control areas," *Atmos. Meas. Techn.*, vol. 6, no. 12, pp. 3577–3596, Dec. 2013.

[12] T. Chu Van, J. Ramirez, T. Rainey, Z. Ristovski, and R. J. Brown, "Global impacts of recent IMO regulations on marine fuel oil refining processes and ship emissions," *Transp. Res. D, Transp. Environ.*, vol. 70, pp. 123–134, May 2019.

[13] C. Schembari *et al.*, "Impact of a European directive on ship emissions on air quality in Mediterranean harbours," *Atmos. Environ.*, vol. 61, pp. 661–669, Dec. 2012.

[14] K. Cullinane and R. Bergqvist, "Emission control areas and their impact on maritime transport," *Transp. Res. D, Transp. Environ.*, vol. 28, pp. 1–5, May 2014.

[15] T. T. Tran and N. Mölders, "Potential impacts of an emission control area on air quality in Alaska coastal regions," *Atmos. Environ.*, vol. 50, pp. 192–202, Apr. 2012.

[16] A. T. Anastasopolos *et al.*, "Air quality in Canadian port cities after regulation of low-sulphur marine fuel in the North American emissions control area," *Sci. Total Environ.*, vol. 791, Oct. 2021, Art. no. 147949.

[17] C. Wang *et al.*, "Reducing sulfur emissions from ships," *Environ. Sci. Technol.*, vol. 41, no. 24, pp. 8233–8239, 2008.

[18] Q. He, X. Zhang, and K. Nip, "Speed optimization over a path with heterogeneous arc costs," *Transp. Res. B, Methodol.*, vol. 104, pp. 198–214, Oct. 2017.

[19] L. Li, S. Gao, W. Yang, and X. Xiong, "Ship's response strategy to emission control areas: From the perspective of sailing pattern optimization and evasion strategy selection," *Transp. Res. E, Logistics Transp. Rev.*, vol. 133, Jan. 2020, Art. no. 101835.

[20] Q. Meng, Y. Du, and Y. Wang, "Shipping log data based container ship fuel efficiency modeling," *Transp. Res. B, Methodol.*, vol. 83, pp. 207–229, Jan. 2016.

[21] J.-J. Wang, C.-X. Jiang, Z. Han, Y. Ren, R. G. Maunder, and L. Hanzo, "Taking drones to the next level: Cooperative distributed unmanned-aerial-vehicular networks for small and mini drones," *IEEE Veh. Technol. Mag.*, vol. 12, no. 3, pp. 73–82, Sep. 2017.

[22] X. Peng *et al.*, "Remote detection sulfur content in fuel oil used by ships in emission control areas: A case study of the Yantian model in Shenzhen," *Ocean Eng.*, vol. 237, Oct. 2021, Art. no. 109652.

[23] J. Xia, K. Wang, and S. Wang, "Drone scheduling to monitor vessels in emission control areas," *Transp. Res. B, Methodol.*, vol. 119, pp. 174–196, Jan. 2019.

[24] X. Luo, Z.-H. Sun, and S. Qiu, "Ant colony system based drone scheduling for ship emission monitoring," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jun. 2021, pp. 241–247.

[25] Z.-H. Sun, X. Luo, E. Q. Wu, T.-Y. Zuo, Z.-R. Tang, and Z. Zhuang, "Monitoring scheduling of drones for emission control areas: An ant colony-based approach," *IEEE Trans. Intell. Transp. Syst.*, early access, Aug. 30, 2021, doi: 10.1109/TITS.2021.3106305.

[26] C. Wang, Z. Xie, L. Shao, Z. Zhang, and M. Zhou, "Estimating travel speed of a road section through sparse crowdsensing data," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 9, pp. 3486–3495, Sep. 2019.

[27] K. Gao, Y. Zhang, Y. Zhang, R. Su, and P. N. Suganthan, "Meta-heuristics for bi-objective urban traffic light scheduling problems," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 7, pp. 2618–2629, Jul. 2019.

[28] W. Deng, J. Xu, and H. Zhao, "An improved ant colony optimization algorithm based on hybrid strategies for scheduling problem," *IEEE Access*, vol. 7, pp. 20281–20292, 2019.

[29] Y. Fu, M. Zhou, X. Guo, and L. Qi, "Scheduling dual-objective stochastic hybrid flow shop with deteriorating jobs via bi-population evolutionary algorithm," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 50, no. 12, pp. 5037–5048, Dec. 2020.

[30] N. Mazyavkina, S. Sviridov, S. Ivanov, and E. Burnaev, "Reinforcement learning for combinatorial optimization: A survey," *Comput. Oper. Res.*, vol. 134, Oct. 2021, Art. no. 105400.

[31] Y. Zhang, R. Bai, R. Qu, C. Tu, and J. Jin, "A deep reinforcement learning based hyper-heuristic for combinatorial optimisation with uncertainties," *Eur. J. Oper. Res.*, vol. 300, no. 2, pp. 418–427, Jul. 2022.

[32] W. Qin, Z. Zhuang, Z. Huang, and H. Huang, "A novel reinforcement learning-based hyper-heuristic for heterogeneous vehicle routing problem," *Comput. Ind. Eng.*, vol. 156, Jun. 2021, Art. no. 107252.

[33] V. A. D. S. Júnior, E. Özcan, and V. R. D. Carvalho, "Hyper-heuristics based on reinforcement learning, balanced heuristic selection and group decision acceptance," *Appl. Soft Comput.*, vol. 97, Dec. 2020, Art. no. 106760.

[34] S. S. Choong, L.-P. Wong, and C. P. Lim, "Automatic design of hyper-heuristic based on reinforcement learning," *Inf. Sci.*, vols. 436–437, pp. 89–107, Apr. 2018.

[35] I. Bello, H. Pham, Q. V. Le, M. Norouzi, and S. Bengio, "Neural combinatorial optimization with reinforcement learning," 2016, *arXiv:1611.09940*.

[36] Z. Pan, L. Wang, J. Wang, and J. Lu, "Deep reinforcement learning based optimization algorithm for permutation flow-shop scheduling," *IEEE Trans. Emerg. Topics Comput. Intell.*, early access, Nov. 1, 2021, doi: 10.1109/TETCI.2021.3098354.

[37] K. Zhang, F. He, Z. Zhang, X. Lin, and M. Li, "Multi-vehicle routing problems with soft time windows: A multi-agent reinforcement learning approach," *Transp. Res. C, Emerg. Technol.*, vol. 121, Dec. 2020, Art. no. 102861.

[38] T. M. Aljohani, A. Ebrahim, and O. Mohammed, "Real-time metadata-driven routing optimization for electric vehicle energy consumption minimization using deep reinforcement learning and Markov chain model," *Electr. Power Syst. Res.*, vol. 192, Mar. 2021, Art. no. 106962.

[39] M. Alqahtani and M. Hu, "Dynamic energy scheduling and routing of multiple electric vehicles using deep reinforcement learning," *Energy*, vol. 244, Apr. 2022, Art. no. 122626.

[40] S. Koh *et al.*, "Real-time deep reinforcement learning based vehicle navigation," *Appl. Soft Comput.*, vol. 96, Nov. 2020, Art. no. 106694.

[41] Q. Wang and C. Tang, "Deep reinforcement learning for transportation network combinatorial optimization: A survey," *Knowl.-Based Syst.*, vol. 233, Dec. 2021, Art. no. 107526.

[42] G. Zhu *et al.*, "Relationship extraction method for urban rail transit operation emergencies records," *IEEE Trans. Intell. Veh.*, early access, 2022.

Poly Z. H. Sun (Member, IEEE) is currently pursuing the Ph.D. degree in industrial intelligent system with the Department of Industrial Engineering, School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai, China.

He is also a Research Assistant with the Department of Automation, Shanghai Jiao Tong University. He has authored/coauthored over 20 research papers in top-tier refereed international journals and conferences, such as IEEE TRANSACTIONS ON ENGINEERING MANAGEMENT, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS, IEEE TRANSACTIONS ON INTELLIGENT VEHICLES, and IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING. His current research interests include intelligent transportation systems, evolutionary optimization, neuroergonomics, brain and cognitive science, complex networks, non-parametric machine learning, operations research, and their applications in industrial or medical problems. He is a member of IEEE CIS, IEEE SMC, and the Association for Computing Machinery. He has been severing as a reviewer or the session chair for several top-tier international journals and conferences in his research field.

Xiaosong Luo (Graduate Student Member, IEEE) received the B.S. degree in mechanical engineering from Shanghai Jiao Tong University, Shanghai, China, where he is currently pursuing the master's degree in industrial engineering with the Department of Industrial Engineering, School of Mechanical Engineering. His current research interests include evolutionary computation and production plan and scheduling.

Tienyu Zuo is currently pursuing the M.S. degree with the School of Automation, Nanjing University of Information Science and Technology, Nanjing, China.

He is a Research Assistant with the Department of Automation, Shanghai Jiao Tong University, Shanghai, China. His works have been published in IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS and *Applied Soft Computing*. His current research interests include statistical machine learning, transfer learning, reinforcement learning, evolutionary algorithms, swarm intelligence, and their applications in real-world optimization problems and climate projection.

Yuguang Bao is currently pursuing the Ph.D. degree with the Department of Industrial Engineering, School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai, China. His current research interests include industrial artificial intelligence, demand engineering, system engineering, and product service system configuration.

Yan-Ning Sun (Member, IEEE) was born in Shandong, China, in 1994. He received the B.S. degree in mechanical engineering and automation from Northeastern University at Qinhuangdao, China, in 2016, and the M.S. degree in mechanical design and theory from Northeastern University, Shenyang, China, in 2019. He is currently pursuing the Ph.D. degree in mechanical engineering with Shanghai Jiao Tong University, Shanghai, China. His current research interests include the fundamental study of artificial intelligence, complex network and complex systems, data-driven decision-making methods in complex industrial processes, and development of industrial big data platform for intelligent manufacturing.

Rob Law is currently the University of Macau Development Foundation (UMDF) Chair Professor of smart tourism. He is also an Honorary Professor of several other reputable universities. Prior to joining the University of Macau in July 2021, he has worked in industry organizations and academic institutes. He is an Active Rresearcher. He has edited four books and published more than 1,000 research papers, including hundreds of articles in first-tier academic journals. His publications have received more than 53,500 citations, with H-index/i 10-index: 105/478 (http://scholar.google.com.hk/). He has received more than 90 research related awards and accolades, as well as millions of USD external and internal research grants. In addition, he serves different roles for more than 200 research journals. He is the chair/committee member of more than 180 international conferences.

Edmond Q. Wu (Senior Member, IEEE) received the Ph.D. degree in controlling theory and engineering from Southeast University, Nanjing, China, in 2009.

He is currently a Professor with the Key Laboratory of System Control and Information Processing, Ministry of Education, Shanghai Jiao Tong University, China. His research interests include deep learning, fatigue recognition, and human–machine interaction. He is an Associate Editor of IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS and IEEE TRANSACTIONS ON INTELLIGENT VEHICLES. He is also a Guest Editor of IEEE TRANSACTIONS ON COGNITIVE AND DEVELOPMENTAL SYSTEMS.