COMPLEX SYSTEM MODELING AND SIMULATION
ISSN 2096-9929 02/06 pp 102−117
Volume 3, Number 2, June 2023
DOI: 10.23919/CSMS.2023.0003

Optimization of Air Defense System Deployment Against Reconnaissance Drone Swarms

Ning Li, Zhenglian Su, Haifeng Ling, Mumtaz Karatas, and Yujun Zheng*

Abstract: Due to their advantages in flexibility, scalability, survivability, and cost-effectiveness, drone swarms have been increasingly used for reconnaissance tasks and have posed great challenges to their opponents on modern battlefields. This paper studies an optimization problem for deploying air defense systems against reconnaissance drone swarms. Given a set of available air defense systems, the problem determines the location of each air defense system in a predetermined region, such that the cost for enemy drones to pass through the region would be maximized. The cost is calculated based on a counterpart drone path planning problem. To solve this adversarial problem, we first propose an exact iterative search algorithm for small-size problem instances, and then propose an evolutionary framework that uses a specific encoding-decoding scheme for large-size problem instances. We implement the evolutionary framework with six popular evolutionary algorithms. Computational experiments on a set of different test instances validate the effectiveness of our approach for defending against reconnaissance drone swarms.

Key words: drone swarms; anti-drone; air defense systems; deployment optimization; evolutionary algorithms

1 Introduction

With the rapid improvement of functionality and performance, drones, i.e., Unmanned Aerial Vehicles (UAVs), have been widely used in various applications, especially in areas that are too dangerous or inaccessible to humans[1], e.g., disaster sites and battlefields. By dynamically grouping a number of drones that interact with one another to complete common tasks cooperatively and autonomously, a drone swarm can have significantly higher flexibility, scalability, survivability, and cost-effectiveness than the simple accumulation of individual drones[2]. In recent years, drone swarms have demonstrated great superiority in modern battlefields, such as in Syria, Nagorno-Karabakh, and Ukraine.

The development of military attack and defense technologies is similar to the relationship between a spear and a shield. The aforementioned advantages of the "spears" of drone swarms have posed great challenges to the "shields" of their opponents, who need to protect against the invasion of drone swarms by reducing their mission capabilities as much as possible. However, current research on countermeasures against drone swarms, such as Global Positioning System (GPS) spoofing, electromagnetic interference, false target jamming[3, 4], and anti-aircraft weapons[5], is still limited and relatively simple. Compared to these spoofing and interference measures, anti-aircraft weapons are more active and aggressive in causing damage to drones and are, therefore, more appealing to

• Ning Li and Yujun Zheng are with the School of Information Science and Technology, Hangzhou Normal University, Hangzhou 311121, China. E-mail: 2021111011001@stu.hznu.edu.cn; yujun.zheng@computer.org.
• Zhenglian Su and Haifeng Ling are with the College of Field Engineering, Army Engineering University, Nanjing 210007, China. E-mail: suzhengliang@aeu.edu.cn; haifeng_ling@aeu.edu.cn.
• Mumtaz Karatas is with the Industrial Engineering Department, Turkish Naval Academy, National Defense University, Tuzla 34940, Turkey. E-mail: mkaratas@dho.edu.tr.
∗ To whom correspondence should be addressed.
※ This article was recommended by Executive Editor-in-Chief Ling Wang.
Manuscript received: 2022-11-25; revised: 2022-12-22; accepted: 2023-01-29

the opponents of drones. However, the flexibility and survivability of drone swarms make it difficult or impossible for a single air defense system to bring substantial damage to a swarm at a time. Typically, if one or several drones are damaged by an attack from the air defense system, the remaining drones can quickly escape from the attack area and reform the swarm that still has a majority of the original mission capability. Therefore, how to deploy available air defense systems to effectively defend against drone swarms is a challenging problem, which is important for the safety and security of the defending side.

In this paper, we study an optimization problem for deploying air defense systems against enemy reconnaissance drone swarms. Given a set of available air defense systems, the problem determines the location of each air defense system in a predetermined region where a reconnaissance drone swarm will pass through. Whenever the swarm is threatened/attacked by an air defense system, it replans the path to the target, as illustrated by the flowchart in Fig. 1. Therefore, we need to optimize the deployment of air defense systems, such that the cost for the enemy drone swarm to pass through the region will be maximized. We calculate the cost based on a counterpart drone path planning problem. To solve this adversarial problem, we first present an iterative search algorithm, which can produce exact optimal solutions to small-size problem instances; then, we propose an evolutionary framework for obtaining optimal or near-optimal solutions to large-size instances. We implement the evolutionary framework with six popular evolutionary algorithms, namely Genetic Algorithm (GA) with variable mutation rate[6], adaptive Particle Swarm Optimization (PSO) using comprehensive learning and self-adaptive parameters[7], Ecogeography-Based Optimization (EBO)[8], Quantum-inspired Tabu Search (QTS)[9], dual-strategy Differential Evolution (DE)[10], and Water Wave Optimization (WWO)[11]. We conducted computational test on a set of problem instances, the results of which validate the effectiveness of the method for defending against reconnaissance drone swarms. Among the six algorithms, EBO and WWO exhibit more competitive performance than the other ones. The main contributions of this work can be stated as follows:

● It presents a problem of deploying air defense systems against drone swarms, which, to our best knowledge, has not been addressed before.

● It proposes an exact iterative algorithm and an evolutionary framework with operators adapted for the problem, which are efficient for solving small- and large-size problem instances, respectively.

● It conducts tests to validate the effectiveness of the proposed approach for improving the defensive abilities against reconnaissance drone swarms.

In the remainder of this paper, Section 2 reviews related work, Section 3 formulates the air defense

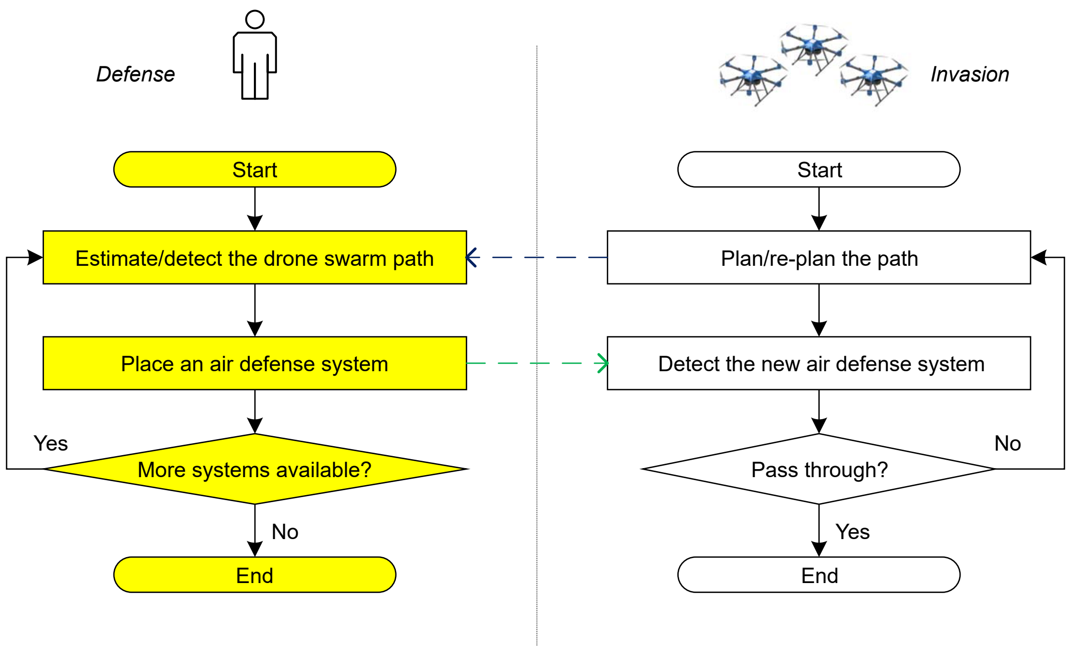

Fig. 1 Basic flow of the invading drone swarm and the opponent for defending against the swarm.

system deployment optimization problem based on a basic drone path planning problem, Section 4 presents the exact iterative search algorithm, Section 5 describes the evolutionary framework and its typical implementations, Section 6 presents the computational experiments, and finally Section 7 concludes this paper.

2 Related Work

The considered air defense system deployment problem can be classified as a facility location problem which is known to be NP-hard[12]. Exact algorithms are only applicable to small-size instances[13, 14]. Many recent studies have been conducted on heuristic and evolutionary algorithms for finding near-optimal solutions. Teran-Somohano and Smith[15] used a bi-objective evolutionary strategy algorithm to solve a semi-obnoxious and multiple capacitated facility location problem that often arises in public planning. Zheng et al.[16] presented a master-slave evolutionary algorithm for a problem of integrated civilian-military emergency supply pre-positioning. Wang et al.[17] presented a dual-population evolutionary algorithm to solve a facility location problem with two objectives on reliability and coverage under the uncertainty of facilities. Vansia and Dhodiya[18] utilized non-dominated sorting genetic algorithm and modified self-adaptive multi-population elitism Jaya algorithm for a multi-objective transportation-p-facility location problem that minimizes the overall transportation time, cost of transportation, and carbon emission. Zhang et al.[19] proposed an enhanced group theory based evolutionary algorithm to solve the uncapacitated facility location problem. Eriskin et al.[20] studied a robust multi-objective model for the location of hospitals during pandemics. Karatas et al.[21] surveyed the facility location models and solution techniques for military organizations.

The present problem aims to maximize the effectiveness of defense against enemy targets. In this sense, the problem is similar to a subclass of Weapon Target Assignment (WTA) problems[22] that aim to maximize the total expected damage or total cost of the targets. Observing that exact algorithms[23, 24] are only applicable to small-size WTA instances, most recent research efforts have been devoted to heuristic and metaheuristic algorithms, including very large neighborhood search[23, 25], ant colony optimization[26, 27], GA[28], PSO[29], eminent domain[30], etc., to efficiently solve medium- and large-size WTA instances. Moreover, our problem considers that a drone swarm can dynamically reform and replan after being attacked and hence is closer to dynamic WTA[31] which uses a sequence of decisions to tackle the dynamic change of targets and is much more complex than its static counterpart[32, 33].

Although there are many research work on general WTA problems, studies focusing on air defense system deployment are relatively limited. Han and Shi[34] proposed a simulated annealing algorithm for optimizing the deployment of air defense missile systems, where the combat effectiveness of defense systems is evaluated based on the stochastic service system theory. Wang and Guo[35] studied the disposition of air defense systems at the company level in two scenarios: one using a single system to protect multiple targets, and the other using multiple systems to protect a single target. They developed linear programming and dynamic programming methods to solve the problems. Yu et al.[36] proposed an artificial neural network approach to weapon system configuration, which first uses a backtracking network to approximate the system effectiveness, and then translates the problem to the traveling salesman problem which is solved by Hopfield network. Wang and Pan[37] considered an air defense disposition problem with uncertainties and risks measured by fuzzy entropy, and they presented a hybrid intelligent algorithm for the problem. Han et al.[38] proposed two air defense system configuration models based on integer programming, one for firing range covering and the other for firing angle covering, and they proved that the layout solution obtained with the firing angle concept is more efficient.

Currently, few studies have been conducted on air defense system deployment against drone swarms, which is much more difficult than traditional anti-aircraft air defense system deployment because drone swarms are much more flexible in reforming and replanning than traditional aircraft. To explore drone vulnerability to deceptive GPS signals, Kerns et al.[39] established necessary conditions for drone capture via GPS spoofing and explored the spoofer’s range of possible post-capture control over drones. Considering a problem of using a team of mini drones acting as a cooperative defensive system against enemy unmanned aerial systems, Castrillo et al.[5] studied sensing, mitigation, and command and control technologies to realize such a defense system. Su et al.[40] studied a problem of false target jamming against UAVs, where each false target jamming solution is evaluated based on its adversarial effects on possible UAV detection solutions. To the best of our knowledge, the problem of air defense system deployment against drone swarms has not been addressed before.

3 Problem Formulation

In this section, we formulate the air defense system deployment optimization problem, which aims to maximize the cost for enemy reconnaissance drones to pass through a predetermined region. We first describe a basic drone path planning problem from the viewpoint of the holders of drones, and then describe the adversarial defense problem from the viewpoint of the opponents of drones.

3.1 Drone path planning

First consider the basic drone path planning problem in three-dimensional space[41, 42], where a swarm of drones needs to traverse a predetermined region. According to the terrain of the region, a total number n of waypoints $\boldsymbol{\Psi} = \{\psi_1, \psi_2, \ldots, \psi_n\}$ are marked, where each waypoint ψ_i is characterized by three-dimensional coordinates (x_i, y_i, z_i) $(1 \leqslant i \leqslant n)$. The aim of the problem is to find a path $\boldsymbol{P} = \{S, P_1, P_2, \ldots, T\}$ that starts from the entry point S, passes through a subset of waypoints $\{P_1, P_2, \ldots\} \subseteq \boldsymbol{\Psi}$, and finally reaches an exit point T, such that the total cost of the path is minimized. For the convenience of expression, we denote $S = P_0$, $T = P_{n_P}$, where n_P is the number of waypoints (including T but excluding S) on the path $\boldsymbol{P}$. Here, we measure the cost of a path based on the following four criteria:

● **Length of the path:** The longer the path, the higher the cost. We calculate the length cost as the ratio of the total length to the Euclidean distance between S and T,

$$L(\boldsymbol{P}) = \frac{1}{|S,T|} \sum_{i=0}^{n_P-1} |P_i, P_{i+1}| \tag{1}$$

● **Height of the path:** The higher the path, the higher the cost. This is because flying at a low flight height can improve the reconnaissance efficiency and reduce the risk of being discovered. We calculate the height cost as the ratio of the total height to $(z_{\max} - z_{\min})$, and the difference between the maximum height and minimum height is as follows:

$$H(\boldsymbol{P}) = \frac{1}{z_{\max} - z_{\min}} \sum_{i=0}^{n_P-1} (\bar{z}(P_i, P_{i+1}) - z_{\min}) \tag{2}$$

where $\bar{z}(P_i, P_{i+1})$ denotes the average flight height of the segment $[P_i, P_{i+1}]$. For a simple straight segment, the height can be directly calculated as the mean of the heights of P_i and P_{i+1}. For a segment comprising complex parts, the height can be obtained by equally dividing the segment into a sufficient number N_e of parts and calculating the mean of the heights of the N_e parts.

● **Excessive horizontal rotation of the path:** If a horizontal rotation angle is larger than a predefined threshold $\widehat{\omega}$, the drones need to decelerate to facilitate the change of direction and then re-accelerate to the normal speed, which would reduce the flight efficiency. We calculate the excessive horizontal rotation cost as follows:

$$R(\boldsymbol{P}) = \frac{1}{n_P \widehat{\omega}} \sum_{i=0}^{n_P-2} \max(\omega(P_i, P_{i+1}, P_{i+2}) - \widehat{\omega}, 0) \tag{3}$$

where $\omega(P_i, P_{i+1}, P_{i+2})$ denotes the horizontal angle between segments (P_i, P_{i+1}) and (P_{i+1}, P_{i+2}).

● **Excessive vertical inclination angle of the path:** If an inclination angle is larger than a predefined threshold $\widehat{\theta}$, the drones need to slow down to facilitate climbing and descending and then re-accelerate to the normal speed, which would reduce the flight efficiency. We calculate the excessive inclination cost as follows:

$$S(\boldsymbol{P}) = \frac{1}{n_P \widehat{\theta}} \sum_{i=0}^{n_P-2} \max(\theta(P_i, P_{i+1}, P_{i+2}) - \widehat{\theta}, 0) \tag{4}$$

where $\theta(P_i, P_{i+1}, P_{i+2})$ denotes the vertical angle between segments (P_i, P_{i+1}) and (P_{i+1}, P_{i+2}).

The objective function of drone path planning is defined as the weighted sum of the above costs,

$$\min C(\boldsymbol{P}) = L(\boldsymbol{P}) + w_H H(\boldsymbol{P}) + w_R R(\boldsymbol{P}) + w_S S(\boldsymbol{P}) \tag{5}$$

where w_H, w_R, and w_S are the importance weights of the height cost, excessive horizontal rotation cost, and excessive inclination cost, respectively. As the weight of the length cost is explicitly one, we suggest that w_H can be set to approximately 0.5, $w_R + w_S = 1 - w_H$, and w_R can be smaller than w_S because horizontal rotation is typically simpler than vertical inclination.

3.2 Air defense system deployment against reconnaissance drone swarms

First, we assume that the holder of reconnaissance drones can always obtain the optimal path $\boldsymbol{P}^*$ that minimizes the objective of Eq. (5). Given a set of m available air defense systems, the air defense system deployment optimization problem is to determine the location of each air defense system in the predetermined region, such that the total cost for the enemy drone swarm to pass through the region under the anti-drone threat is maximized. Each air defense system can automatically detect any drone entering its defense range (within a predefined radius r_d). If two or more drones enter simultaneously, we assume that the system can shoot down at least one drone. It is also assumed that the maximum height of the drones is in the attack range of air defense systems, while the drones cannot attack the defense systems. Under the above assumptions, if there is only one enemy drone, we can simply place an air defense system at any suitable position on $\boldsymbol{P}^*$. However, for a drone swarm, whenever the first drone is intercepted by the air defense system, the remaining drones have abilities to bypass the system.

Let A_1 be the position of the first air defense system for intercepting the drones along the path $\boldsymbol{P}^*$. As illustrated in Fig. 2a, the first drone is intercepted at the intersection of $\boldsymbol{P}^*$ and the circumference of the defense

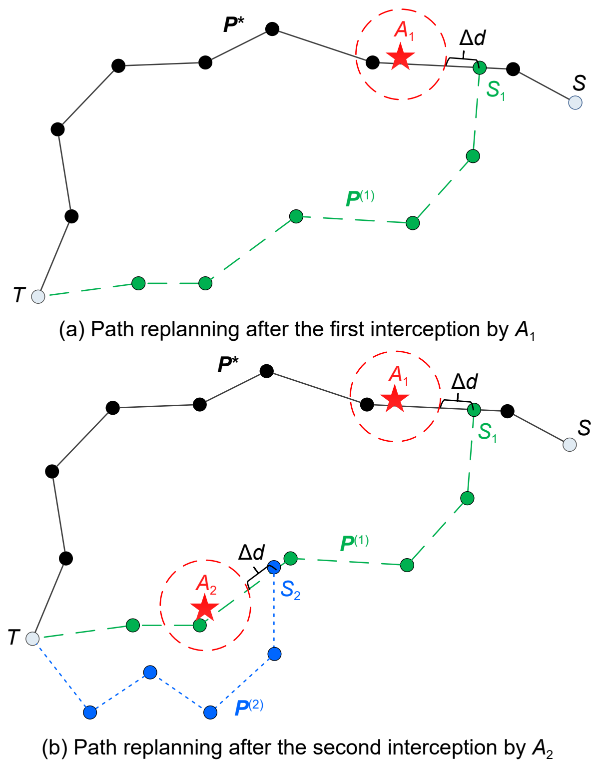

Fig. 2 Illustration of the path replanning of a drone swarm after being intercepted by air defense systems. Solid (black) line: original path; dash (green) line: first replanned path; dotted (blue) line: second replanned path; red circle: defense range.

range of A_1. Afterward, the swarm replans its path from the new starting point S_1 to the entry point T, where the distance between S_1 and the intersection is Δd (which is typically set as the average interval among the drones in the swarm). However, the new path from S_1 to T should try to avoid passing through the defense range of A_1. To reflect this, we add a new fire threat cost to the drone path as follows:

$$E_1(\boldsymbol{P}) = \frac{1}{|S,T|}\sum_{i=0}^{n_P-1} \Lambda_1(P_i, P_{i+1}) \tag{6}$$

where $\Lambda_1(P_i, P_{i+1})$ denotes the length of the part of the segment (P_i, P_{i+1}) that is in the defense range of A_1.

Consequently, the objective function of drone path replanning after being intercepted by the first air defense system should incur the fire threat cost in the following:

$$\min\ C_1(\boldsymbol{P}) = C(\boldsymbol{P}) + w_E E_1(\boldsymbol{P}) \tag{7}$$

where w_E is the importance weight of the fire threat cost. Note that when we set w_E to a sufficiently large value, the path will not be allowed to enter the defense range.

Let $\boldsymbol{P}^{(1)}$ be the new path that minimizes the updated objective of Eq. (7), and A_2 be the position of the second air defense system, which should be on path $\boldsymbol{P}^{(1)}$. Similarly, after the swarm enters the defense range of A_2, it should replan the path $\boldsymbol{P}^{(2)}$ from the new starting point S_2 to the entry point T, and the new path should try to avoid passing through the defense ranges of A_1 and A_2, as illustrated in Fig. 2b.

By analog, after placing the j-th air defense system on the path $\boldsymbol{P}^{(j-1)}$, the drone swarm replans the path from the new starting point S_j to the entry point T, and the path should consider the fire threat cost caused by the previous j air defense systems ($1 \leqslant j \leqslant m$),

$$E_j(\boldsymbol{P}) = \frac{1}{|S,T|}\sum_{i=0}^{n_P-1} \Lambda_j(P_i, P_{i+1}) \tag{8}$$

where $\Lambda_j(P_i, P_{i+1})$ denotes the length of the part of the segment (P_i, P_{i+1}) that is in the defense range of any one of $A_1, A_2, \ldots, A_j$. The objective function of the j-th path replanning is

$$\min\ C_j(\boldsymbol{P}) = C(\boldsymbol{P}) + w_E E_j(\boldsymbol{P}) \tag{9}$$

From the viewpoint of the opponent of the drone swarm, the air defense system deployment optimization problem is to determine m positions $\{A_1, A_2, \ldots, A_m\}$ in the region to place the m air defense systems, such that the total cost of the paths planned and replanned by the drone swarm is maximized,

$$\max f(A) = C(\boldsymbol{P}^*[S,S_1]) + \sum_{j=1}^{m-1} C_j(\boldsymbol{P}^{(j)}[S_j,S_{j+1}]) + C(\boldsymbol{P}^{(m)}[S_m,T]) \quad (10)$$

where $\boldsymbol{P}[S_j,S_{j+1}]$ represents the partial path from S_j to S_{j+1} in the path $\boldsymbol{P}$, $\boldsymbol{P}^{(j)} = \arg\min_{\boldsymbol{P}\in PATH(S_j,T)} C_j(\boldsymbol{P})$, and $PATH(S_j,T)$ is the set of all paths from S_j to T.

In practice, some areas are not suitable for deploying air defense systems due to terrain constraints. We use $\boldsymbol{\Omega}$ to denote the union of all positions that are suitable for placing air defense systems, $\boldsymbol{\Omega}_1$ to denote the remaining suitable positions after placing the first air defense system (by removing the defense range of A_1 from $\boldsymbol{\Omega}$), $\boldsymbol{\Omega}_2$ to denote the remaining suitable positions after placing the first two air defense systems (by removing the defense range of A_2 from $\boldsymbol{\Omega}_1$), and so on. Consequently, the feasible region for placing air defense systems is $\boldsymbol{\Omega}$ at first and updated to $\boldsymbol{\Omega}_j$ after placing the first j systems ($1 \leqslant j \leqslant m-1$).

4 Exact Iterative Search Algorithm

To solve the above air defense system deployment problem, we first propose an iterative search algorithm for finding the exact optimal solution to the problem. Suppose that the previous $(m-1)$ air defense systems have been placed, the position A_m of the m-th air defense system should be determined according to the second last replanned path $\boldsymbol{P}^{(m-1)}$ of the drone swarm, aiming at maximizing the cost of $\boldsymbol{P}^{(m-1)}$ and the last replanned path $\boldsymbol{P}^{(m)}$,

$$\max f(A_m) = C_{m-1}(\boldsymbol{P}^{(m-1)}[S_{m-1},S_m]) + C_m(\boldsymbol{P}^{(m)}[S_m,T]) \quad (11)$$

The position A_m belongs to set $\boldsymbol{\Omega}_{m-1}$, and its defense range should intersect with the path $\boldsymbol{P}^{(m-1)}$. Therefore, the procedure FindA $(m,\boldsymbol{P}^{(m-1)},\boldsymbol{\Omega}_{m-1})$ for determining the position A_m can be described by Algorithm 1 that iteratively tests all possible positions, where $\Phi(A)$ denotes the defense range of the air defense system placed at the position A.

Next, suppose that the previous $(m-2)$ air defense systems have been placed. The position A_{m-1} of the $(m-1)$-th air defense system should be determined to maximize the value of the following function:

$$\max f(A_{m-1}) = C_{m-2}(\boldsymbol{P}^{(m-2)}[S_{m-2},S_{m-1}]) + C_{m-1}(\boldsymbol{P}^{(m-1)}[S_{m-1},S_m]) + C_m(\boldsymbol{P}^{(m)}[S_m,T]) \quad (12)$$

The procedure FindA $(m-1,\boldsymbol{P}^{(m-2)},\boldsymbol{\Omega}_{m-2})$ for determining the position A_{m-1} is described in Algorithm 2, which has a framework similar to Algorithm 1 in that each possible position is tested. The difference is that, when evaluating the fitness function value for each candidate position, we not only replan the drone path $\boldsymbol{P}^{(m-1)}$ from A_{m-1}, but also call the procedure FindA $(m,\boldsymbol{P}^{(m-1)},\boldsymbol{\Omega}_{m-1})$ to find the optimal position A_m after A_{m-1} and therefore replan $\boldsymbol{P}^{(m)}$.

By analogy, we can derive the procedures for determining the positions $A_{m-2}, A_{m-3}, \ldots$ in a reverse manner, and finally obtain the procedure FindA $(1,\boldsymbol{P}^*,\boldsymbol{\Omega})$ for determining the position of A_1 for minimizing the original objective function of Eq. (10), as shown in Algorithm 3, which is an exact optimization algorithm for solving this air defense system deployment problem.

As illustrated in Fig. 3, FindA $(1,\boldsymbol{P}^*,\boldsymbol{\Omega})$ will iteratively call the sub-procedures FindA $(2,\boldsymbol{P}^{(1)},\boldsymbol{\Omega}_1)$, FindA $(3,\boldsymbol{P}^{(2)},\boldsymbol{\Omega}_2)$, …, FindA $(m,\boldsymbol{P}^{(m-1)},\boldsymbol{\Omega}_{m-1})$, and finally obtain all m positions $\{A_1,A_2,\ldots,A_m\}$ for placing the air defense systems. Let N be the cardinality of $\boldsymbol{\Omega}$. The worst time complexity of the exact optimization algorithm is

$$O(C(N,m)O(A^\star)) = O(N(N-1)(N-2)\cdots(N-m+1)O(A^\star)),$$

Algorithm 1 FindA $(m, \boldsymbol{P}^{(m-1)}, \boldsymbol{\Omega}_{m-1})$

Let V = null, $V = 0$;
for each $A \in \boldsymbol{\Omega}_{m-1}$ **do**
  if $\Phi(A) \cap \boldsymbol{P}^{(m-1)} \neq \varnothing$ **then**
    Call $A^\star$ algorithm to replan the drone path $\boldsymbol{P}^{(m)}$;
    Evaluate $f(A)$ according to Eq. (11);
    if $f(A) > V$ **then**
      $A_m \leftarrow A$;
      $V \leftarrow f(A)$;
return A_m

Algorithm 2 FindA $(m-1, \boldsymbol{P}^{(m-2)}, \boldsymbol{\Omega}_{m-2})$

Let A_{m-1} = null, $V = 0$;
for each $A \in \boldsymbol{\Omega}_{m-2}$ **do**
  if $\Phi(A) \cap \boldsymbol{P}^{(m-2)} \neq \varnothing$ **then**
    Call $A^\star$ algorithm to replan the drone path $\boldsymbol{P}^{(m-1)}$;
    Call FindA $(m, \boldsymbol{P}^{(m-1)}, \boldsymbol{\Omega}_{m-1})$ to determine the subsequent A_m and $\boldsymbol{P}^{(m)}$;
    Evaluate $f(A)$ according to Eq. (12);
    if $f(A) > V$ **then**
      $A_{m-1} \leftarrow A$;
      $V \leftarrow f(A)$;
return A_{m-1}

Algorithm 3 FindA (1, $\boldsymbol{P}^*$, $\boldsymbol{\Omega}$)

```
Let A_1 = null, V = 0;
for each A ∈ Ω do
    if Φ(A) ∩ P* ≠ ∅ then
        Call A* algorithm to replan drone path P^(1);
        Call FindA (2, P^(1), Ω_1) to determine the subsequent
            {A_2, A_3, ..., A_m} and {P^(2), P^(3), ..., P^(m)};
        Evaluate f(A) according to Eq. (10);
        if f(A) > V then
            A_1 ← A;
            V ← f(A);
return A_1
```

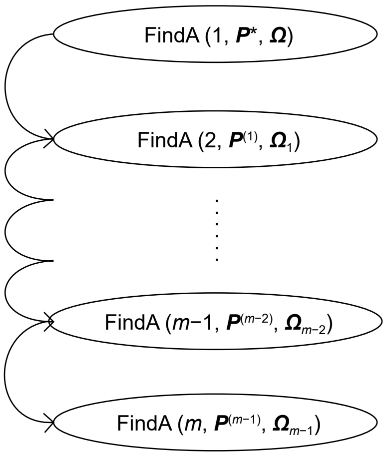

Fig. 3 Recursive invocation of the sub-procedure FindA( ).

where $O(A^\star)$ denotes the time complexity of the $A^\star$ algorithm (typically does not exceed $O(N^2)$). Nevertheless, in practice, the number of points that intersect the designated paths with the circumference of defense ranges is usually much smaller than n. Moreover, with the progress of drone path replanning, the new starting point continually becomes closer to the exit point T, and therefore the length of new replanned paths continuously becomes shorter. Consequently, the average time complexity of the algorithm is not so high.

5 Evolutionary Algorithm

As the running time of the above exact optimization algorithm increases exponentially with N and m, it is only suitable for situations where the region is relatively small and the number of air defense systems is also not large. Otherwise, the running time would be unacceptable. To solve large-size instances of the problem, we propose an evolutionary framework that evolves a population of candidate solutions to search for optimal or near-optimal solutions. Note that although this problem determining m positions among N candidates can be regarded as a multi-knapsack problem, existing evolutionary algorithms for the multi-knapsack problem are not efficient because they often test many candidate positions that are unable or inefficient to intercept drones.

To efficiently solve the air defense system deployment problem, we design a specific encoding and decoding scheme. Each solution to the problem is encoded as an m-dimensional real-valued vector $\boldsymbol{x}=\{x_1, x_2, \ldots, x_m\}$, where the value of each component x_i is in the range of [0, 1]. The solution decoding procedure consists of the following steps:

Step 1: Find out the set of all candidate positions whose defense range intersects with the path $\boldsymbol{P}^*$, and sort these positions in order of their intersections with the path from front to back (i.e., the closer the intersection to the starting point S, the higher the rank;

Step 2: Let N_1 be the number of candidate positions and k be the integer closest to $N_1 x_1$; take the k-th candidate position to place the first air defense system;

Step 3: Let $j=1$;

Step 4: Replan the drone path $\boldsymbol{P}^{(j)}$ based on A_j;

Step 5: Find out the set of all candidate positions whose defense range intersects with the path $\boldsymbol{P}^{(j)}$, and sort these positions in order of their intersections with the path from front to back;

Step 6: Let N_{j+1} be the number of candidate positions and k be the integer closest to $N_{j+1}x_{j+1}$; take the k-th candidate position to place the j-th air defense system;

Step 7: Set $j=j+1$; if $j>m$, then stop; otherwise, go to Step 4.

The above scheme encodes the position of each air defense system as the rank of its intersection with the path of the drone swarm to relate the position to the anti-drone task. The use of real-valued encoding makes it easy to adapt existing evolutionary algorithms for continuous optimization to our problem.

In the following subsections, we use GA[6], PSO[7], EBO[8], QTS[9], DE[10], and WWO[11] to implement the framework.

5.1 GA with variable mutation rate

GA performs a stochastic search based on the principle of genetic crossover and mutation. On the basis of the above encoding/decoding scheme, we use the breeder crossover operation[43] that generates offsprings $\boldsymbol{x}^c$ and $\boldsymbol{x}^b$ from two parents $\boldsymbol{x}^a$ and $\boldsymbol{x}^b$ as follows (where r is a random value uniformly distributed in [0, 1], $1 \leqslant j \leqslant m$):

$$x_j^c = rx_j^a + (1-r)x_j^b,$$
$$x_j^d = (1-r)x_j^a + rx_j^b \quad (13)$$

Another key difference of our algorithm from classical GA is that it uses a variable mutation rate $r_m(\boldsymbol{x})$ calculated as inversely proportional to the fitness of solution $\boldsymbol{x}$ as follows (such that low-fitness solutions are changed greatly while high-fitness solutions are more stable):

$$r_m(\boldsymbol{x}) = \frac{f(\boldsymbol{x})}{f_{\max}} r_m^{\max} \quad (14)$$

where $f_{\max}$ denotes the maximum objective function value in the population, and $r_m^{\max}$ is a parameter controlling the maximum mutation rate. The mutation is conducted by randomly setting each dimension in the value range.

Algorithm 4 presents the pseudo-code of the GA, where r_c is the crossover rate, and $rand\,(0,1)$ produces a random value uniformly distributed in [0, 1]. The time complexity of the fitness evaluation is $O(mN^3)$. The average mutation rate is expected to be 0.5. Let G be the maximum number of generations and N_P be the population size; the average time complexity of the GA is $O(1.5GN_PmN^3)$.

5.2 Adaptive comprehensive learning PSO

PSO[44] associates each solution (particle) $\boldsymbol{x}_i$ with a velocity vector $\boldsymbol{v}_i$ of the same dimensionality, which is used to move the solution at each iteration,

$$x_{i,j} = x_{i,j} + v_{i,j} \quad (15)$$

In the classical PSO, each velocity vector is adjusted by learning from both the personal historical best $pbest_i$ of the current solution and the global best $gbest$ of the whole population. Comprehensive learning PSO[45] improves the classical PSO by utilizing all other particles' historical best to update the velocity of each particle, such that the information of different particles is exchanged more thoroughly,

$$v_{i,j} = wv_{i,j} + c \times rand\,(0,1)(pbest_{i',j} - x_{i,j}) \quad (16)$$

The exemplar solution $pbest_{i'}$ is randomly selected based on a learning probability at each dimension j, and $pbest_{i',j}$ is the corresponding component of $pbest_{i'}$. w and c are two control parameters. We also adaptively adjust the values of the two parameters according to the evolutionary states, i.e., the improvement of each individual and the whole population over iterations[7].

Algorithm 5 presents the pseudo-code of the adaptive comprehensive learning PSO algorithm for the problem. The average time complexity of the algorithm is $O(GN_PmN^3)$.

5.3 EBO

EBO is an extended version of Biogeography-Based Optimization (BBO)[46], which associates each solution $\boldsymbol{x}$ with an immigration rate $r_\mu(\boldsymbol{x})$ and an emigration rate $r_\nu(\boldsymbol{x})$ as follows:

$$r_\mu(\boldsymbol{x}) = \frac{f_{\max} - f(\boldsymbol{x}) + \epsilon}{f_{\max} - f_{\min} + \epsilon},$$
$$r_\nu(\boldsymbol{x}) = \frac{f(\boldsymbol{x}) - f_{\min} + \epsilon}{f_{\max} - f_{\min} + \epsilon} \quad (17)$$

Algorithm 4 GA for the air defense system deployment optimization problem

Randomly initialize a population of N_P solutions;
while the stop condition is not met **do**
    Create an empty offspring population;
    while the offspring population size is smaller than N_P **do**
        Use roulette-wheel selection to randomly select two solutions $\boldsymbol{x}^a$ and $\boldsymbol{x}^b$ from the current population;
        if $rand\,(0,1) < r_c$ **then**
            Use crossover to produce two offsprings $\boldsymbol{x}^c$ and $\boldsymbol{x}^d$ according to Eq. (13);
        Calculate $r_m(\boldsymbol{x}^a)$ and $r_m(\boldsymbol{x}^b)$ according to Eq. (14);
        if $rand\,(0,1) < r_m(\boldsymbol{x}^a)$ **then**
            Mutate $\boldsymbol{x}^a$;
        if $rand\,(0,1) < r_m(\boldsymbol{x}^b)$ **then**
            Mutate $\boldsymbol{x}^b$;
        Add $\boldsymbol{x}^a$ and $\boldsymbol{x}^b$ to the offspring population;
    Set the current population to the offspring population;
return the best known solution found

Algorithm 5 PSO for the air defense system deployment optimization problem.

Randomly initialize a population of N_P solutions;
while the stop condition is not met **do**
    for all $\boldsymbol{x}_i$ in the population **do**
        for $j = 1$ to m **do**
            Select an exemplar $pbest_{i'}$;
            Update $v_{i,j}$ according to Eq. (16);
        for $j = 1$ to m **do**
            Update $x_{i,j}$ according to Eq. (15);
    Update the control parameters;
return the best known solution found

where $f_{\min}$ is the minimum objective function values in the population, and ϵ is a small positive number to avoid division by zero.

Compared to the basic BBO, EBO employs a local neighborhood structure[47] and differentiates local migration and global migration. Local migration sets the current dimension of the immigrating solution $\boldsymbol{x}$ by blending with the corresponding dimension of the emigrating solution $\boldsymbol{x}'$, where $\boldsymbol{x}'$ is a neighbor of $\boldsymbol{x}$ selected with a probability proportional to the emigrating rate. The corresponding dimention x'_j of the emigrating solution $\boldsymbol{x}'$ is as follows:

$$x_j = x_j + rand\,(0,1)(x'_j - x_j) \tag{18}$$

Global migration selects two emigrating solutions $\boldsymbol{x}'$ and $\boldsymbol{x}''$, where $\boldsymbol{x}'$ is from the neighboring solutions of $\boldsymbol{x}$, and $\boldsymbol{x}''$ is from the non-neighboring solutions. A global migration operation is performed according to the fitness comparison of the two emigrating solutions as follows:

$$x_j = \begin{cases} x'_j + rand\,(0,1)(x''_j - x_j), & f(\boldsymbol{x}') > f(\boldsymbol{x}''); \\ x''_j + rand\,(0,1)(x'_j - x_j), & f(\boldsymbol{x}') < f(\boldsymbol{x}'') \end{cases} \tag{19}$$

Whether a migration operation is a local or global migration is determined by a parameter η that linearly decreases with iteration, such that global migration has a high probability in early stages for diversifying the search, whereas local migration is more probable in later stages for improving solution accuracy.

Algorithm 6 presents the pseudo-code of the EBO algorithm, the time complexity of which is $O(GN_PmN^3)$.

Algorithm 6 EBO for the air defense system deployment optimization problem

Randomly initialize a population of N_P solutions;
while the stop condition is not met **do**
  for all $\boldsymbol{x}_i$ in the population **do**
    for $j = 1$ to m **do**
      if $rand\,(0, 1) > r_\mu(\boldsymbol{x})$ **then**
        continue; //skip migration
      Select a neighbor $\boldsymbol{x}'$ based on $r_\mu(\boldsymbol{x}')$;
      if $rand\,(0, 1) < \eta$ **then**
        Perform local migration according to Eq. (18);
      else
        Select a non-neighbor $\boldsymbol{x}''$ based on $r_\mu(\boldsymbol{x}'')$
        Perform global migration according to Eq. (19);
    if the migrated solution is better **then**
      Use the migrated solution to replace the original one;
  Update the parameter η;
return the best known solution found

5.4 QTS

Tabu search[48] is an extension of basic local search using short-term memory to save recently obtained local optimal solutions to avoid repeated searches. QTS[9] introduces quantum-inspired bits and gates into the algorithm to suppress premature convergence and better balance exploration and exploitation. The QTS algorithm first initializes a quantum matrix Q to represent the solution, and then continually performs neighborhood search by measuring the quantum matrix several times. The neighboring solutions are further improved by entanglement and local search. The search procedure continues until the stop condition is satisfied. The quantum matrix is also iteratively updated to push new solutions toward the current best solution while keeping away the current worst.

Algorithm 7 presents the pseudo-code of the QTS algorithm, the time complexity of which is $O(t_{\max}N_bmN^3)$, where N_b is the neighborhood size.

Algorithm 7 QTS for the air defense system deployment optimization problem

Let $t = 0$, and initialize a quantum matrix Q;
while the stop condition is not met **do**
  Produce a neighborhood set by measuring of $Q(t)$;
  for all $\boldsymbol{x}$ in the neighborhood set **do**
    Bound $\boldsymbol{x}$ in the feasible domain and evaluate $f(\boldsymbol{x})$;
  Perform both the dimension entanglement and the order entanglement;
  Perform local search on the best solution $\boldsymbol{x}^*$ by bit reversal;
  Perform entanglement local search on $\boldsymbol{x}^*$ by multiple bit reversal;
  Update $\boldsymbol{x}^*$ and the worst solution $\boldsymbol{x}^\dagger$;
  if $\boldsymbol{x}^*$ is not updated within a predefined number of iterations **then**
    Apply a quantum NOT gate;
  $t \leftarrow t+1$;
  Update $Q(t)$ based on $\boldsymbol{x}^*$, $\boldsymbol{x}^\dagger$, and quantum MOVE gate;
return $\boldsymbol{x}^*$

5.5 Dual-strategy differential evolution

DE is a simple but fast evolutionary algorithm that evolves solutions according to the difference between them[49]. At each iteration, a mutation vector is produced for each solution by adding the difference between two randomly selected solutions to a third one; a trial vector is then produced by the crossover of the solution and its mutation vector; finally, the better one between the solution and the trial vector is chosen for the next generation.

Dual-strategy DE[10] uses multiple sub-populations of solution. In each sub-population, the solutions are sorted in decreasing order of fitness, and the following DE/rand/1 mutation scheme is applied to the first half of the solutions:

$$\boldsymbol{v}_i = \boldsymbol{x}_{r_1} + F(\boldsymbol{x}_{r_2} - \boldsymbol{x}_{r_3}) \tag{20}$$

where r_1, r_2, and r_3 are three random indices in the population, and F is a control parameter called the scale factor.

The following DE/lbest/1 mutation scheme is applied to the second half of the solutions:

$$\boldsymbol{v}_i = \boldsymbol{x}_{lbest} + F(\boldsymbol{x}_{r_1} - \boldsymbol{x}_{r_2}) \tag{21}$$

where $\boldsymbol{x}_{lbest}$ denotes the current best solution in the sub-population.

Algorithm 8 presents the pseudo-code of the dual-strategy DE algorithm, the time complexity of which is $O(GMN_S mN^3)$, where N_S denotes the size of sub-populations.

5.6 WWO

Inspired by the shallow water wave theory, WWO assigns each solution $\boldsymbol{x}$ with a "wavelength" $\lambda(\boldsymbol{x})$, whose value is initialized as 0.5 and updated as inversely proportional to the solution fitness as follows:

$$\lambda(\boldsymbol{x}) = \lambda(\boldsymbol{x})\alpha^{-(f(\boldsymbol{x})-f_{\min}+\epsilon)/(f_{\max}-f_{\min}+\epsilon)} \tag{22}$$

where α is a control parameter set to 1.0026.

At each iteration, WWO uses a propagation operation to produce an offspring $\boldsymbol{x}'$ for each solution $\boldsymbol{x}$ within a search range proportional to $\lambda(\boldsymbol{x})$ as follows:

Algorithm 8 Dual-strategy DE for the air defense system deployment optimization problem

```
Randomly initialize a population of solutions;
while the stop condition is not met do
    Cluster the population into M sub-populations;
    for all sub-population do
        for all x in the sub-population do
            if x is inferior then
                Perform mutation according to Eq. (20);
            else
                Perform mutation according to Eq. (21);
    for all offspring u do
        if u is better than the nearest parent then
            Replace the nearest parent with u;
    for all solution x do
        if x stays for a predefined number η of iterations then
            Reinitialize x;
return the best known solution found
```

$$x'_j = x_j + rand(-1,1)\lambda(\boldsymbol{x})L_j \tag{23}$$

where L_j denotes the search range of the j-th dimension (1 for all dimensions of our problem). In this way, high (low) fitness solutions have small (large) search ranges to balance global and local search. The better one between the original solution and its offspring will be selected into the population.

WWO also performs a breaking operation on each newly found best solution $\boldsymbol{x}^*$ by generating k_N neighboring solutions, each at a distinct randomly selected dimension j as follows:

$$x'_j = x^*_j + N(0,1)\beta L_j \tag{24}$$

where $N(0,1)$ generates a Gaussian random number with mean 0 and standard deviation 1, and β is a control parameter set to 0.002. The best neighbor, if better than $\boldsymbol{x}^*$, will replace $\boldsymbol{x}^*$ in the population.

We use a variable population size[50], which linearly decreases from $N_P^{\max}$ to $N_P^{\min}$ (two control parameters). The size reduction is performed by removing the worst solution from the population.

Algorithm 9 presents the pseudo-code of the WWO algorithm. The time complexity of the algorithm is $O(G(N_P + K_N)mN^3)$, where $N_P = (N_P^{\max} + N_P^{\min})/2$.

6 Computational Experiment

The experiments were conducted on eight test

Algorithm 9 WWO for the air defense system deployment optimization problem

```
Randomly initialize a population of N_P solutions;
Let x* be the best-known solution in the population;
while the stop condition is not met do
    Calculate the wavelengths of all solutions according to
      Eq. (22);
    for all x in the population do
        Produce an offspring x' according to Eq. (23);
        if f(x') > f(x) then
            x ← x';
            if f(x) > f(x*) then
                x* ← x;
                for k = 1 to k_N do
                    Produce a neighbor x' according to
                      Eq. (24);
                    if f(x') > f(x*) then
                        x* ← x';
    Update the population size N_P (decrease N_P by
      eliminating the worst solution);
return x*
```

instances, which were constructed based on four selected maps with different topographic features in South East China. Table 1 presents the basic features of the instances, including the regional area (2nd column), number of waypoints (3rd column), number of available air defense systems (4th column), and number of drones in the swarm (5th column). The important weights w_H, w_R, and w_S in the objective function of Eq. (5) and w_E in Eq. (9) were set to 0.5, 0.2, 0.3, and 1.0, respectively.

For each test instance, we first use the $A^\star$ algorithm to plan the path of the drone swarm and calculate the cost of the path $\boldsymbol{P}^*$ without interception of air defense systems. Next, we try to use the exact Iterative Search algorithm (IS) to solve the instance (until its running time exceeds 12 hours). Afterward, we use the six evolutionary algorithms, i.e., GA, PSO, EBO, QTS, DE, and WWO, to solve the instance, each being repeated 30 times. The control parameters of each evolutionary algorithm were tuned on the whole set of instances. The stop condition of the evolutionary algorithms was set in such a way that the running time reached 30 min.

Figures 4–11 present the experimental results on the eight test instances, respectively. In each figure, the dashed line denotes the path cost $C(\boldsymbol{P}^*)$ of the drone swarm without considering air defense systems, and each box plot gives the median, minimum, maximum, first quartile (25%, denoted by Q1), and third quartile (75%, denoted by Q3) of the objective values obtained by the corresponding algorithm over the 30 runs. Within the predefined time limit of 12 hours, the exact IS algorithm only stops on the first two instances, and therefore its results are only given in Figs. 4 and 5. The results showed that using solutions obtained by the proposed algorithms to deploy air defense systems can increase the path cost of drone swarms to 500%−1100% of the original cost without air defense systems, and the ratio increases with the instance size. This finding demonstrates that the proposed method can effectively deploy air defense systems to defend against reconnaissance drone swarms especially in

Table 1 Basic features of the test instances.

Instance No.	Area (km^2)	N	m	Number of drones
1	37.3	29	5	8
2	37.3	29	6	10
3	71.0	75	7	12
4	71.0	75	8	16
5	167.5	208	9	16
6	167.5	208	12	20
7	892.2	1333	16	36
8	892.2	1333	20	42

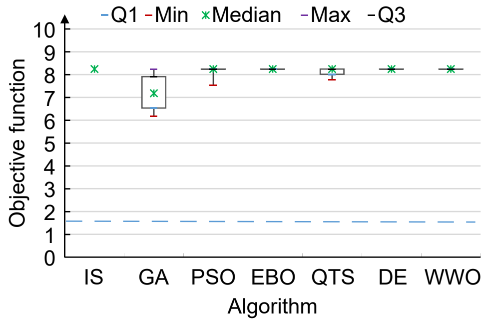

Fig. 4 Computational results on Instance 1.

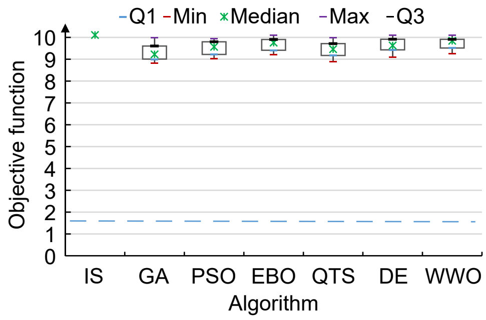

Fig. 5 Computational results on Instance 2.

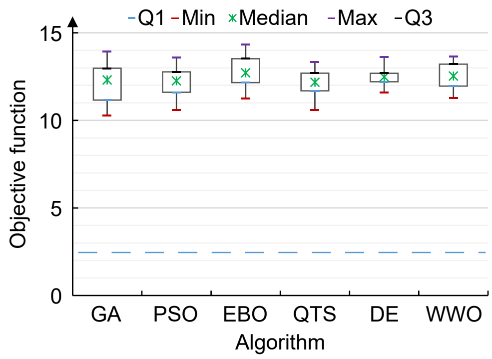

Fig. 6 Computational results on Instance 3.

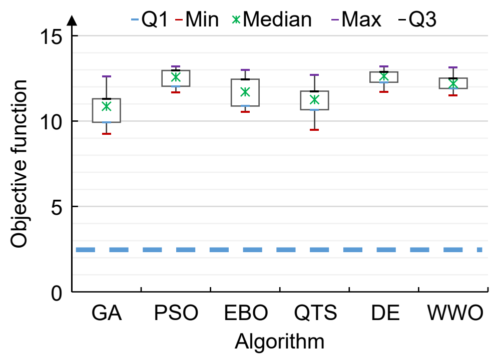

Fig. 7 Computational results on Instance 4.

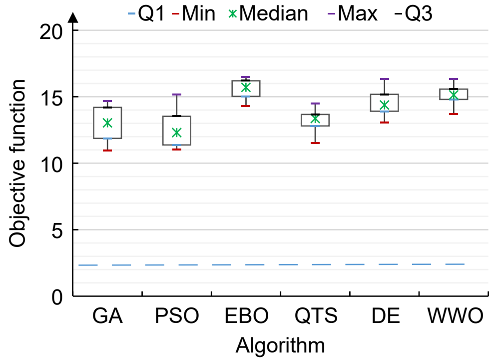

Fig. 8 Computational results on Instance 5.

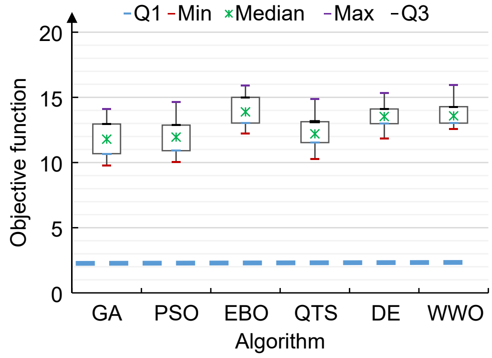

Fig. 9 Computational results on Instance 6.

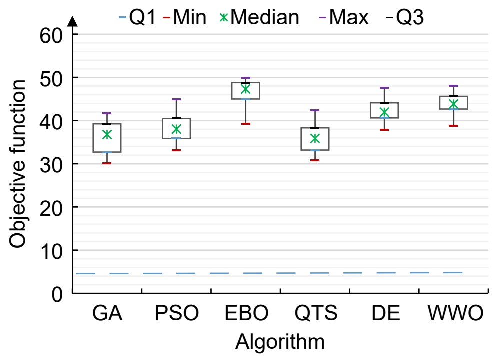

Fig. 10 Computational results on Instance 7.

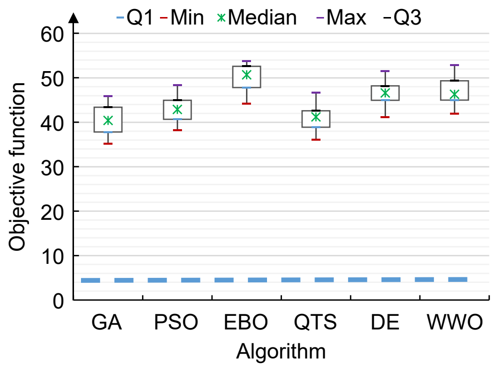

Fig. 11 Computational results on Instance 8.

large areas.

On smallest-size Instance 1, five evolutionary algorithms (except GA) obtain the same median value, which is the exact optimal objective value obtained by IS (but the computation time of the evolutionary algorithms is much shorter than that of IS). EBO, DE, and WWO always obtain the optimal solution, whereas the minimum values of PSO and QTS are smaller than the optimal value, i.e., the two algorithms occasionally fail to obtain the optimal solution. The median value of GA is smaller than the optimal value, although its maximum value is the optimal, i.e., GA occasionally obtains the optimal solution, but fails to do so in most cases. On Instance 2 where the region is the same as that of Instance 1 but the numbers of waypoints, air defense systems, and drones are all larger than those of Instance 1, WWO obtains the best median value, EBO, DE, and WWO occasionally obtain the optimal solution as IS, but GA, PSO, and QTS never do so.

On the remaining larger-size instances, IS cannot obtain the optimal solution within 12 hours, indicating that the exact algorithm can only solve small-size instances. DE obtains the best median value on Instance 4, and EBO obtains the best median values on the other five instances.

We conducted a nonparametric Wilcoxon rank sum test on the results of the six evolutionary algorithms to evaluate their differences on each instance. The results (all at a confidence level of 95%) show that, on either Instance 1 or Instance 2, there is no significant difference among the results of PSO, EBO, QTS, DE, and WWO, and they are all significantly better than the result of GA. On Instances 3 and 6, the result of EBO is significantly better than those of GA, PSO, and QTS, but it is not significantly different than those of DE and WWO. On Instance 4, the result of DE is significantly better than those of GA, EBO, and QTS, but it is not significantly different than those of PSO and WWO. On Instances 5, 7, and 8, the result of EBO is significantly better than those of the other five algorithms. Tables 2–7 present the statistical test results on Instances 3–8, respectively, where “>” denotes that the result of the algorithm in the current row is significantly better than that in the current column.

Table 8 presents the rank of the median value of each algorithm on each test instance, and Table 9 presents the total rank number (row 2), the number of times of obtaining the best median value (row 3, abbreviated to “best”), and the number of times of being significantly

Table 2 Statistical test results on Instance 3 (">" indicates "significantly better").

Algorithm	GA	PSO	EBO	QTS	DE	WWO
GA	−	−	−	−	−	−
PSO	−	−	−	−	−	−
EBO	>	>	−	>	−	−
QTS	>	−	−	−	−	−
DE	>	>	−	−	−	−
WWO	>	>	−	>	−	−

Table 3 Statistical test results on Instance 4 (">" indicates "significantly better").

Algorithm	GA	PSO	EBO	QTS	DE	WWO
GA	−	−	−	−	−	−
PSO	>	−	−	−	−	−
EBO	>	−	−	>	−	−
QTS	>	−	−	−	−	−
DE	>	−	>	>	−	−
WWO	>	>	−	>	−	−

Table 4 Statistical test results on Instance 5 (">" indicates "significantly better").

Algorithm	GA	PSO	EBO	QTS	DE	WWO
GA	−	−	−	>	−	−
PSO	−	−	−	−	−	−
EBO	>	>	>	−	>	>
QTS	−	−	−	−	−	−
DE	>	−	−	>	−	−
WWO	>	>	−	>	−	−

Table 5 Statistical test results on Instance 6 (">" indicates "significantly better").

Algorithm	GA	PSO	EBO	QTS	DE	WWO
GA	−	−	−	−	−	−
PSO	>	−	−	−	−	−
EBO	>	>	−	>	−	−
QTS	−	−	−	−	−	−
DE	>	−	−	>	−	−
WWO	>	−	−	>	−	−

Table 6 Statistical test results on Instance 7 (">" indicates "significantly better").

Algorithm	GA	PSO	EBO	QTS	DE	WWO
GA	−	−	−	−	−	−
PSO	−	−	−	−	−	−
EBO	>	>	>	−	>	>
QTS	−	−	−	−	−	−
DE	>	−	−	−	−	−
WWO	>	>	−	>	−	−

Table 7 Statistical test results on Instance 8 (">" indicates "significantly better").

Algorithm	GA	PSO	EBO	QTS	DE	WWO
GA	−	−	−	−	−	−
PSO	−	−	−	−	−	−
EBO	>	>	−	>	>	>
QTS	−	−	−	−	−	−
DE	>	−	−	>	−	−
WWO	>	>	−	>	−	−

Table 8 Ranks of the six evolutionary algorithms on the test instances.

Instance No.	GA	PSO	EBO	QTS	DE	WWO
1	6	1	1	1	1	1
2	6	4	2	5	3	1
3	4	5	1	6	3	2
4	6	2	4	5	1	3
5	5	6	1	4	3	2
6	6	5	1	4	3	2
7	5	4	1	6	3	2
8	6	4	1	5	2	3

Table 9 Summary of the comparative performance of the six evolutionary algorithms on the test instances.

Metric	GA	PSO	EBO	QTS	DE	WWO
Total rank	44	31	12	36	19	16
Best	0	1	6	1	2	2
Significantly better	1	4	25	3	12	19

better than other algorithms (row 4, abbreviated to"significantly better") over all eight instances. EBO exhibits the best overall performance, WWO performs the second best, and GA performs the worst because it is easily trapped in local optima, given that its crossover operator is relatively weak in global search. QTS only performs better than GA, as its neighborhood search mechanism is not also inefficient in searching a large-size solution space. PSO's particle motion mechanism makes it converge fast to some good solutions on small-size instances, but often leads to premature convergence on large-size instances. The differential crossover operator and multi-population strategy make DE more suitable for searching large-size solution spaces than small-size ones. The integration of global and local migrations in EBO and the wavelength-based propagation in WWO can achieve a quite good balance between global search and local search, which is the main reason why the two algorithms exhibit good performance on the test instances. Comparatively, the migration mechanism endows EBO with a stronger global search ability to solve large-size instances.

The computational results also validate that increasing the number of air defense systems can effectively increase the cost of enemy drones. For example, the ratio of the path cost after the deployment of the air defense systems (in terms of the maximum cost achieved by the algorithms) to the path cost before the deployment is around 500% on Instance 1 with five air defense systems, but the ratio increases to over 580% on Instance 2 with six systems. From the viewpoint of the drone holder, under the threat of air defense systems, increasing the number of drones can effectively decrease its total path cost. Facing the invasion of a large swarm of drones, it is crucial to obtain high-quality air defense system deployment solutions to defend against reconnaissance drones.

7 Conclusion

This paper presents an optimization problem of air defense system deployment against enemy reconnaissance drone swarms, which determines the locations of air defense systems to intercept drones and force them to continually change their paths, such that the total cost of replanned paths of the drone swarm is maximized. We propose an exact iterative search algorithm and an evolutionary framework implemented using six concrete algorithms to solve the problem. The computational experimental results validate the performance of air defense system deployment solutions obtained by the algorithms for defending against reconnaissance drones.

This study only considers defense against reconnaissance drones that cannot attack our defense systems. In our ongoing study, we consider that the target drone swarm can have ground attack drones[51], and the problem should take the possible damage caused by the drones to the air defense systems into consideration, aiming at maximizing the cost of drones and minimizing the loss of air defense systems simultaneously. In this study, we only implemented the evolutionary framework with six popular individual algorithms, and the ongoing study also includes implementing more hybrid algorithms for further possible performance improvement.

Although the countermeasure of air defense is more active than spoofing and interference, it is not sufficiently active because air defense systems are relatively static after deployment. Currently, we are also studying more comprehensive air defense systems that incorporate the cooperation of ground weapons, attacking drones, as well as human soldiers to fight against enemy drones[52]. Future studies will consider air-ground cooperative defense against enemy drones, which could be significantly more complex and require algorithms to be more efficient and intelligent in predicting drone behaviors and utilizing problem-solving knowledge [53].

Acknowledgment

This work was supported by the National Natural Science Foundation of China (No. 61872123) and the Natural Science Foundation of Zhejiang Province (No. LR20F030002).

References

[1] Y. Xia, R. Batta, and R. Nagi, Controlling a fleet of unmanned aerial vehicles to collect uncertain information in a threat environment, *Oper. Res.*, vol. 65, no. 3, pp. 674–692, 2017.

[2] A. Tahir, J. Böling, M. H. Haghbayan, H. T. Toivonen, and J. Plosil, Swarms of unmanned aerial vehicles—a survey, *J. Ind. Inf. Integr.*, https://doi.org/10.1016/j.jii.2019.100106.

[3] A. S. Banu and G. Padmavathi, Taxonomy of UAVs GPS spoofing and jamming attack detection methods, in *Computational Intelligence for Unmanned Aerial Vehicles Communication Networks*, M. Ouaissa, I. U. Khan, M. Ouaissa, Z. Boulouard, and S. B. Hussain Shah, eds. Cham, Switzerland: Springer, vol. 1033, 2022, pp. 167–201.

[4] S. Guo, Z. Dong, Z. Hu, and C. Hu, Simulation of dynamic electromagnetic interference environment for Unmanned Aerial Vehicle data link, *China Commun.*, vol. 10, no. 7, pp. 19–28, 2013.

[5] V. U. Castrillo, A. Manco, D. Pascarella, and G. Gigante, A review of counter-UAS technologies for cooperative defensive teams of drones, *Drones* , vol. 6, no. 3, p. 65, 2022.

[6] S. A. Stanhope and J. M. Daida, An individually variable mutation-rate strategy for genetic algorithms, in *Evolutionary Programming VI*, P. J. Angeline, R. G. Reynolds, J. R. McDonnell, and R. Eberhart, eds. Berlin, Germany: Springer, 1997, pp. 235–245.

[7] Y. J. Zheng, H. F. Ling, and Q. Guan, Adaptive parameters for a modified comprehensive learning particle swarm optimizer, *Math. Prob. Eng.*, vol. 2012, p. 207318, 2012.

[8] Y. J. Zheng, H. F. Ling, and J. Y. Xue, Ecogeography-based optimization: Enhancing biogeography-based optimization with ecogeographic barriers and differentiations, *Comput. Oper. Res.*, vol. 50, pp. 115–127, 2014.

[9] S. Y. Kuo and Y. H. Chou, Entanglement-enhanced

quantum-inspired Tabu search algorithm for function optimization, *IEEE Access*, vol. 5, pp. 13236–13252, 2017.

[10] Z. J. Wang, Z. H. Zhan, Y. Lin, W. J. Yu, H. Q. Yuan, T. L. Gu, S. Kwong, and J. Zhang, Dual-strategy differential evolution with affinity propagation clustering for multimodal optimization problems, *IEEE Trans. EComput.*, vol. 22, no. 6, pp. 894–908, 2018.

[11] Y. J. Zheng, Water wave optimization: A new nature-inspired metaheuristic, *Comput. Oper. Res.*, vol. 55, pp. 1–11, 2015.

[12] S. H. Owen and M. S. Daskin, Strategic facility location: A review, *Eur. J. Oper. Res.*, vol. 111, no. 3, pp. 423–447, 1998.

[13] C. Boonmee, M. Arimura, and T. Asada, Facility location optimization model for emergency humanitarian logistics, *Int. J. Disaster Risk Reduct.*, vol. 24, pp. 485–498, 2017.

[14] C. Ortiz-Astorquiza, I. Contreras, and G. Laporte, An exact algorithm for multilevel uncapacitated facility location, *Transp. Sci.*, vol. 53, no. 4, pp. 1085–1106, 2019.

[15] A. Teran-Somohano and A. E. Smith, Locating multiple capacitated semi-obnoxious facilities using evolutionary strategies, *Comput. Ind. Eng.*, vol. 133, pp. 303–316, 2019.

[16] Y. J. Zheng, Y. Wang, H. F. Ling, Y. Xue, and S. Y. Chen, Integrated civilian-military pre-positioning of emergency supplies: A multiobjective optimization approach, *Appl. Soft Comput.*, vol. 58, pp. 732–741, 2017.

[17] C. Wang, Z. Wang, Y. Tian, X. Zhang, and J. Xiao, A dual-population based evolutionary algorithm for multi-objective location problem under uncertainty of facilities, *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 7692–7707, 2022.

[18] D. O. Vansia and J. M. Dhodiya, Solution of multi-objective transportation-*p*-facility location problem with effect of variable carbon emission by evolutionary algorithms, *Soft Comput.*, vol. 25, no. 15, pp. 9993–10005, 2021.

[19] F. Zhang, Y. He, H. Ouyang, and W. Li, A fast and efficient discrete evolutionary algorithm for the uncapacitated facility location problem, *Expert Syst. Appl.*, vol. 213, p. 118978, 2023.

[20] L. Eriskin, M. Karatas, and Y. J. Zheng, A robust multi-objective model for healthcare resource management and location planning during pandemics, *Annal Oper. Res.*, doi: 10.1007/s10479-022-04760-x.

[21] M. Karatas, E. Yakıcı, and N. Razi, Military facility location problems: A brief survey, in *Operations Research for Military Organizations*, H. Tozan and M. Karatas, eds. Hershey, PA, USA: IGI Global, 2019, p. 27.

[22] R. A. Murphey, Target-based weapon target assignment problems, in *Nonlinear Assignment Problems. Combinatorial Optimization*, P. M. Pardalos and L S. Pitsoulis, eds. Boston, MA, USA: Springer, 2000, pp. 39–53.

[23] R. K. Ahuja, A. Kumar, K. C. Jha, and J. B. Orlin, Exact and heuristic methods for the weapon target assignment problem, *Oper. Res.*, vol. 55, no. 6, pp. 1136–1146, 2007.

[24] A. Kline, D. Ahner, and R. Hill, The weapon-target assignment problem, *Comput. Oper. Res.*, vol. 105, pp. 226–236, 2019.

[25] M. Z. Lee, Constrained weapon-target assignment: Enhanced very large scale neighborhood search algorithm, *IEEE Trans. Syst. Man Cybern. Part A*: *Syst. Humans*, vol. 40, no. 1, pp. 198–204, 2010.

[26] Z. J. Lee, C. Y. Lee, and S. F. Su, An immunity-based ant colony optimization algorithm for solving weapon-target assignment problem, *Appl. Soft Comput.*, vol. 2, no. 1, pp. 39–47, 2002.

[27] X. Hu, P. Luo, X. Zhang, and J. Wang, Improved ant colony optimization for weapon-target assignment, *Math. Prob. Eng.*, vol. 2018, p. 6481635, 2018.

[28] Z. J. Lee, S. F. Su, and C. Y. Lee, Efficiently solving general weapon-target assignment problem by genetic algorithms with greedy eugenics, *IEEE Trans. Syst. Man Cybern.*, *Part B*, vol. 33, no. 1, pp. 113–121, 2003.

[29] X. Zeng, Y. Zhu, L. Nan, K. Hu, B. Niu, and X. He, Solving weapon-target assignment problem using discrete particle swarm optimization, in *Proc. 6th World Congress on Intelligent Control and Automation*, Dalian, China, 2006, pp. 3562–3565.

[30] A. G. Kline, D. K. Ahner, and B. J. Lunday, A heuristic and metaheuristic approach to the static weapon target assignment problem, *J. Global Optim.*, vol. 78, no. 4, pp. 791–812, 2020.

[31] H. Cai, J. Liu, Y. Chen, and H. Wang, Survey of the research on dynamic weapon-target assignment problem, *J. Syst. Eng. Electron.*, vol. 17, no. 3, pp. 559–565, 2006.

[32] J. Chen, B. Xin, Z. Peng, L. Dou, and J. Zhang, Evolutionary decision-makings for the dynamic weapon-target assignment problem, *Sci. China Ser. F*: *Inf. Sci.*, vol. 52, no. 11, pp. 2006–2018, 2009.

[33] B. Xin, J. Chen, Z. Peng, L. Dou, and J. Zhang, An efficient rule-based constructive heuristic to solve dynamic weapon-target assignment problem, *IEEE Trans. Syst. Man Cybern.*, *Part A*: *Syst. Humans*, vol. 41, no. 3, pp. 598–606, 2011.

[34] S. C. Han and D. P. Shi, Optimization for air defense combat configuration via simulated annealing algorithm, (in Chinese), *Acta Aeronaut. Astronaut. Sin.*, vol. 20, no. 5, pp. 478–480, 1999.

[35] X. F. Wang and X. D. Guo, The optimized model research of the flak batteries' disposition in point air defence, (in Chinese), *J. Inf. Eng. Univ.*, vol. 5, no. 2, pp. 154–156, 2004.

[36] G. Yu, H. Wang, S. Sun, and Q. Wang, Optimal configuration of weapon system based on ANN, in *Proc. Chinese Control and Decision Conf.*, Yantai, China, 2008, pp. 3209–3212.

[37] Y. Wang and L. P. Pan, Study of mean-entropy models for key point air defense disposition, in *Fuzzy Information and Engineering*, B. Y. Cao, G. J. Wang, S. Z. Guo, and S. L. Chen, eds. Berlin, Germany: Springer, 2010, pp. 647–656.

[38] Y. Han, J. Wang, and X. Liao, Optimization of disposition for terminal air defense system based on set-covering model, in *Proc. 32nd Chinese Control Conf.*, Xi'an, China, 2013, pp. 8461–8464.

[39] A. J. Kerns, D. P. Shepard, J. A. Bhatti, and T. E. Humphreys, Unmanned aircraft capture and control via

GPS spoofing, *J. Field Robot.*, vol. 31, no. 4, pp. 617–636, 2014.

[40] Z. L. Su, X. L. Jiang, N. Li, H. F. Ling, and Y. J. Zheng, Optimization of false target jamming against UAV detection, *Drones*, vol. 6, no. 5, p. 114, 2022.

[41] Q. Tian, Modeling and optimization methods for multi-UAV cooperative reconnaissance mission planning problem, (in Chinese), PhD dissertation, National Defense University, Changsha, China, 2007.

[42] Y. Wang, M. X. Zhang, and Y. J. Zheng, A hyper-heuristic method for UAV search planning, in *Advances in Swarm Intelligence, Part II*, Y. Tan, H. Takagi, Y. Shi, and B. Niu, eds. Cham, Switzerland: Springer, 2017, pp. 454–464.

[43] H. Mühlenbein and D. Schlierkamp-Voosen, Predictive models for the breeder genetic algorithm I. continuous parameter optimization, *EComput.*, vol. 1, no. 1, pp. 25–49, 1993.

[44] J. Kennedy and R. Eberhart, Particle swarm optimization, in *Proc. ICNN'95 Int. Conf. Neural Networks*, Perth, Australia, 1995, pp. 1942–1948.

[45] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, *IEEE Trans. EComput.*, vol. 10, no. 3, pp. 281–295, 2006.

[46] D. Simon, Biogeography-based optimization, *IEEE Trans. EComput.*, vol. 12, no. 6, pp. 702–713, 2008.

[47] Y. J. Zheng, H. F. Ling, X. B. Wu, and J. Y. Xue, Localized biogeography-based optimization, *Soft Comput.*, vol. 18, pp. 2323–2334, 2014.

[48] F. Glover, Tabu search–part I, *ORSA J. Comput.*, vol. 1, no. 3, pp. 190–206, 1989.

[49] R. Storn and K. Price, Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces, *J. Global Optim.*, vol. 11, no. 4, pp. 341–369, 1997.

[50] Y. J. Zheng and B. Zhang, A simplified water wave optimization algorithm, in *Proc. IEEE Congress on Evolutionary Computation*, Sendai, Japan, 2015, pp. 807–813.

[51] Q. Peng, H. Wu, and R. Xue, Review of dynamic task allocation methods for UAV swarms oriented to ground targets, *Complex Syst. Model. Simul.*, vol. 1, no. 3, pp. 163–175, 2021.

[52] Y. J. Zheng, Y. C. Du, H. F. Ling, W. G. Sheng, and S. Y. Chen, Evolutionary collaborative human-UAV search for escaped criminals, *IEEE Trans. EComput.*, vol. 24, no. 2, pp. 217–231, 2020.

[53] Y. J. Zheng, X. Chen, Q. Song, J. Yang, and L. Wang, Evolutionary optimization of COVID-19 vaccine distribution with evolutionary demands, *IEEE Trans. EComput.*, vol. 27, no. 1, pp. 141–154, 2023.

Ning Li received the BEng degree in computer science from Zhejiang University of Science and Technology, China in 2021. He is currently a PhD candidate in computer science and technology at Hangzhou Normal University, Hangzhou, China. His research interests include evolutionary algorithms and intelligent scheduling.

Zhenglian Su received the PhD degree from PLA University of Science and Technology, China in 2012. He is currently a lecturer at Army Engineering University, Nanjing, China. His research interests include data and knowledge engineering and its applications in military equipment management and operations.

Haifeng Ling received the PhD degree from Nanjing University, China in 2011. She is currently a professor at Army Engineering University, Nanjing, China. Her research interests include evolutionary algorithms and theirs applications in operations research.

Mumtaz Karatas is currently an associate professor at the Industrial Engineering Department, Turkish Naval Academy, National Defense University, Turkey. His research interests include operations research applications in logistics, location planning, defense, and energy.

Yujun Zheng received the PhD degree from the Institute of Software, Chinese Academy of Sciences, China in 2010. He is now a professor at Hangzhou Normal University, Hangzhou, China. His research interests include evolutionary optimization and their applications in intelligent transportation, emergency management, and public health. He has authored over 100 papers in IEEE transactions and other famous journals. In 2014, he received the runner-up of IFORS Prize for Development due to the work on intelligent scheduling of engineering rescuing tasks in the 2013 Dingxi Earthquake. In 2018, he was selected as a finalist for the Daniel H. Wagner Prize for excellence in operations research practice. His work on emergency medical supply production was selected for the "2022 Best Paper Award" of the *Applied Soft Computing Journal*.