A modified genetic algorithm for task assignment of heterogeneous unmanned aerial vehicle system

Song Han*, Chenchen Fan, Xinbin Li, Xi Luo and Zhixin Liu

Abstract
This study deals with the task assignment problem of heterogeneous unmanned aerial vehicle (UAV) system with the limited resources and task priority constraints. The optimization model which comprehensively considers the resource consumption, task completion effect, and workload balance is formulated. Then, a concept of fuzzy elite degree is proposed to optimize and balance the transmission of good genes and the variation strength of population during the operations of algorithm. Based on the concept, we propose the fuzzy elite strategy genetic algorithm (FESGA) to efficiently solve the complex task assignment problem. In the proposed algorithm, two unlock methods are presented to solve the deadlock problem in the random optimization process; a sudden threat countermeasure (STC) mechanism is presented to help the algorithm quickly respond to the change of task environment caused by sudden threats. The simulation results demonstrate the superiority of the proposed algorithm. Meanwhile, the effectiveness and feasibility of the algorithm in workload balance and task priority constraints are verified.

Keywords
Heterogeneous UAV system, task assignment, workload balance, genetic algorithm

Date received: 1 February 2021; accepted: 19 February 2021

Introduction
Unmanned aerial vehicle (UAV) has made outstanding contributions in military, agriculture, and commercial applications with its unique low cost and strong maneuverability over the previous decades.1 However, it is difficult for a single UAV to satisfy the increasingly complex task requirements in many cases due to its limited airborne resources. Therefore, UAV applications are gradually developing toward clustering. Liu et al., and Zhou et al.,2,3 tackled the multi-UAV collaborative data collection problem Motlagh et al.4 studied the application of multi-UAV system in the Internet of Things (IOT) and realized the effective transmission of airborne information. Most of the above studies focus on the homogeneous UAV system. However, the diversity of tasks sets higher demands on the UAV system. The heterogeneity will be the main development direction of UAV system in the future. In the large-scale task scenario, the optimization effect of task assignment directly affects the work effectiveness of UAV system.5,6 The task assignment problem of heterogeneous UAV system has become a hot issue in the field of multiple UAVs control.7

The multiple heterogeneous UAVs task assignment problem is a complex combinatorial optimization problem. In Deng et al.,8 the total task execution time was taken as the objective function of the UAV task assignment problem. In Motlagh et al.,4 the authors jointly optimized the energy consumption and operational time of UAV system. Most of current studies focus on reducing the resource consumption of UAV system. The task completion effect of UAVs is ignored, which may lead to unnecessary resource consumption. Therefore, it is necessary to evaluate the task completion quality rationally, and jointly optimize the resource consumption and task completion effect of UAV system. In addition, many researchers have studied the workload balance in the UAV task assignment optimization problem.9,10 The algorithms considering the workload balance can effectively prevent the situation that some vehicles run out of their resources, thereby decreasing the task completion time of UAV

Institute of Electrical Engineering, Yanshan University, Qinhuangdao, China

Corresponding author:
Xinbin Li, Institute of Electrical Engineering, Yanshan University, Qinhuangdao 066004, China.
Email: lixb@ysu.edu.cn

Creative Commons CC BY: This article is distributed under the terms of the Creative Commons Attribution 4.0 License (https://creativecommons.org/licenses/by/4.0/) which permits any use, reproduction and distribution of the work without further permission provided the original work is attributed as specified on the SAGE and Open Access pages (https://us.sagepub.com/en-us/nam/open-access-at-sage).
However, the workload balance problem is difficult to solve due to the coordinated control of multiple UAVs. In Zhu and Yang, the workload balance coefficient related to the average traveled distance of UAV system was introduced in objective function to alleviate the imbalance of task assignment effectively. However, the coefficient overlooks the great influence of the real-time resource information of UAV system to the task assignment optimization process. In Chen et al., a guiding mechanism based on particle swarm optimization (PSO) algorithm was proposed to achieve a fairer task assignment scheme. However, the proposed coefficient only focuses on the real-time change of traveled distance, the assignment scheme may not satisfy the equalization requirement of other types of resources. For instance, a UAV cannot continue working if a resource carried by the UAV is consumed excessively, then the stability of UAV system will be adversely affected. Therefore, it is of great practical significance to propose a workload balance method that comprehensively considers multiple real-time changing resources to ensure the efficient operation of UAV system.

The task assignment of UAV system is a complex nondeterministic polynomial hard (NP-hard) problem with multiple constraints. It is difficult to solve by traditional convex optimization theory. In current studies, the deterministic algorithms such as mixed integer linear program (MILP), branch and bound (BAB) algorithm, and tree search algorithm (TSA) have been widely adopted to solve the UAV system task assignment problem. In the process of solving large-scale complex task assignment problem, matrix operations involved in MILP and BAB are numerous and complex, search scope of TSA is large. Thus, they are only suitable for solving small and medium-sized task assignment problems with fewer constraints. As experience-based algorithms, heuristic algorithms can avoid the high computational complexity of the NP-hard problems. Therefore, they have been more extensively studied than deterministic algorithms in the field of task assignment. In Chen et al., the authors developed a modified two-part wolf pack search (MTWPS) algorithm to solve the UAV system task assignment problem with task priority constraint effectively. In Xi et al., a guiding mechanism based on particle swarm optimization (PSO) algorithm was presented to rapidly provide feasible task assignment scheme for homogeneous strike UAV cooperative operation. In Kalayci and Gupta, the modified artificial bee colony (ABC) algorithm was proposed to effectively solve the decentralized task assignment problem. However, in the optimization process of algorithms, a solution can only be updated with the guidance of other solutions. Searching ability of the algorithms is insufficient due to the single updating way. Thus, they may fail to provide satisfactory solutions in more complex task scenarios. Compared with the above heuristic algorithms, genetic algorithm (GA) can update solutions in more diversified ways through mutation. GA shows stronger exploration ability and more advantages in solving the complex large-scale task assignment problem. In Deng et al., a GA-based mirror representation method was proposed to cope with the limited resource constraint in the multi-UAV task assignment. In Schwarzrock et al., the task assignment problem was transformed into a directed graph to simplify the problem complexity, and two different encoding methods were proposed for homogeneous UAVs and heterogeneous UAVs, respectively. In Jia et al., the authors proposed a concept of penalty items to enable GA to solve the multi-UAV task assignment problem with task priority constraint effectively. More diversified constraints must be considered due to the complexity of task scenarios. The introduction of multiple complex constraints directly affects the solving efficiency of traditional GA. In Wu et al., the authors proposed a distributed genetic algorithm (DGA) which treats each UAV as an independent processor to enhance the population diversity. In Bai et al., multiple subpopulations were created to macroscopically increase the intensity of genetic operations, thereby enhancing search ability of the algorithm. In Zhou et al., good individuals were achieved by adopting the greedy strategy to synthetically adjust the resource allocation of UAV system. These individuals can accelerate the population evolution and convergence speed. The above GAs focus on the convergence speed or search ability. They lack the comprehensive consideration of the two performances. However, it is difficult for GAs to improve these performances simultaneously. The increasing complexity of task scenarios requires higher performance of algorithms. Thus, it is significant to propose an improved GA with fast convergence speed and strong searching ability to solve the complex large-scale UAV system task assignment problem.

Given that GA is an opportunistic optimization algorithm, some infeasible solutions may be generated in the optimization process. Wherein, deadlock problem is the main reason of generating infeasible solutions. Especially in the large-scale task scenarios, deadlock solution appears with more possibility. Lemaire et al. proposed a swapping method to resolve deadlocks by exchanging the order of any two tasks which violate the task priority constraint. However, the method may fail when multiple UAVs are involved in deadlocks. In Deng et al., an unlock method based on graph theory was presented to detect and solve the deadlock problem. Nevertheless, too many matrix manipulations and loop operations are involved in the method. The tedious and large calculation greatly reduce the unlock efficiency. Therefore, it is meaningful to design a low complexity and high reliability unlocking method to adapt to the ever-expanding task background.

Based on the given analysis, some main challenges in the field of task assignment of heterogeneous UAV system are given as follows: (1) The solving difficulty of task assignment problem increases with the multiplicity.
of constraints. (2) The current heuristic algorithms show insufficient precision and slow convergence speed in solving the large-scale complex task assignment problem. (3) The existing unlocking methods are inefficient for the deadlock problem caused by the heterogeneity of UAVs and task priority constraint. In this study, we aim to propose an improved GA to solve the task assignment problem of heterogeneous UAV system with multiple constraints. First, we construct a combinatorial optimization model about the resource consumption, task completion effect, workload balance, task priority, and limited resource constraints. Different from the existing studies, constraints contained in the proposed model are more comprehensive, which are conducive to improving the effectiveness of solutions. The joint optimization of multiple constraints may reduce the solution efficiency of traditional GA. To address the issue, a concept of fuzzy elite degree considering the preservation of good genes and the change intensity of individuals is presented. Based on the concept, we propose the fuzzy elite strategy genetic algorithm (FESGA) with high computational efficiency. Specifically, a novel initialization method and a parent selection method are presented to enhance the convergence speed of algorithm. An avoiding inbreeding (AI) mechanism and a hierarchical mutation (HM) rule are proposed to solve the local extremum problem. Then, a no zero mechanism and two unlock methods are presented to deal with the infeasible solutions which violate the limited resources constraint, and the deadlock infeasible solutions, respectively. Moreover, we presented the sudden threat countermeasure (STC) mechanism to enhance the ability of algorithm to converge quickly after detecting the sudden threats.

The main contributions of this study are further concluded as follows:

1) Aiming at the optimization of resource consumption, task completion effect, and workload balance of UAV system, we propose the resource consumption function, detection revenue function, and execution capability, respectively. Further, an optimization model with multiple constraints is formulated by jointly considering the task priority and limited resource constraints.

2) We propose a concept of fuzzy elite degree to evaluate the participation of individuals in the optimization process. In the proposed FESGA, the proposed concept can rationally optimize the retention and change intensity of genes. Therefore, the convergence performance and search ability of FESGA can be improved synthetically.

3) Multiple low computational complexity and effective unlock methods are presented to solve the deadlock problem in the genetic operations. The proposed diversified unlocking ways can help the algorithm avoid being affected by local extremes.

4) The STC mechanism is presented to improve the ability of algorithm to deal with sudden threats in the changing task scenario quickly. Thus, robustness of algorithm in the various complex environments is enhanced.

The rest of this paper is organized as follows. In Section 2, the task scenario and corresponding optimization model are formulated. In Section 3, details of FESGA are introduced. Section 4 describes the methods to deal with the deadlock problem and sudden threats, respectively. Numerical simulations and analyses are provided in Section 5. Finally, Section 6 summarizes this paper.

System model and problem formulation

In this section, we describe a scenario where a heterogeneous UAV system performs multiple tasks on a set of targets. Then, the complex task assignment optimization model with multiple constraints is formulated.

System model

In this study, we consider a task scenario which consists of several targets and a group of heterogeneous UAVs. Three tasks of reconnaissance, attack, and verification should be performed successively on each target. Denote the target set as \( T = \{ t_1, t_2, \ldots, t_N \} \), the UAV set as \( U = \{ u_1, u_2, \ldots, u_M \} \). Where \( N \) and \( M \) denote the number of targets and UAVs, respectively. The set of each target is \( I = \{ task_R, task_A, task_V \} \). Where \( task_R \), \( task_A \), and \( task_V \) represent reconnaissance task, attack task, and verification task, respectively. Define \( task_i > task_k \) to indicate that \( task_i \) has higher priority than \( task_k \), the priority order of each target is expressed as: \( task_R > task_A > task_V \). The UAV system is classified into two categories depending on the type of tasks that UAVs can perform, that is, \( U = \{ U_{Su}, U_{Si} \} \). Thereinto, \( U_{Su} \) includes \( M_{Su} \) surveillance UAVs that can execute the \( task_R \) and \( task_V \). \( U_{Si} \) includes \( M_{Si} \) strike UAVs that only can execute the \( task_A \). Each UAV belongs to only one collection. All of the UAVs should return to base after completing the assigned tasks. Moreover, a state set \( S = \{ E, W, F \} \) is defined to describe three different states of the UAV. Elements in the state set denote the task execution state, waiting state, and flight state, respectively. State \( E \) represents that the UAV is performing the specific task. State \( W \) means that the UAV is being forced to wait for the reason that the higher priority task of the target has not been completed. State \( F \) indicates that the UAV is flying to the next destination after completing a task.

Considering that the resource of UAV is limited, each UAV should complete the assigned tasks in a limited time, the fuel consumption and traveled distance of each UAV are not allowed to exceed its maximum limit.

Total range of \( u_i \) can be achieved by
The total fuel consumption of \( u_i \) can be achieved as follows:

\[
d_i^{\text{total}} = \sum_{j=1}^{N+1} d_{j,i}(i) \tag{1}
\]

where \( d_{j,i}(i) \) denotes the distance of \( u_i \) from \( t_j \) to the next target \( t_j \).

Noting that the fuel consumption varies in different states of UAV, the total fuel consumption of \( u_i \) is achieved by

\[
f_i^{\text{total}} = \sum_{j=1}^{N+1} \sum_{k=1}^{3} \left( f_{ij}^{\text{flight}} x_{ijk} + f_{ij}^{\text{wait}} x_{ijk} + f_{ij}^{\text{execute}} x_{ijk} \right) \tag{2}
\]

where \( f_{ij}^{\text{flight}}, f_{ij}^{\text{wait}}, \) and \( f_{ij}^{\text{execute}} \) denote the fuel consumption of \( u_i \) when flying to execute the next task from \( t_j \), waiting to execute task \( k \) on \( t_j \), and performing task \( k \) on \( t_j \), respectively. \( x_{ijk} \) is calculated as follows:

\[
x_{ijk} = \begin{cases} 1, & \text{if task } k \text{ on } t_j \text{ is assigned to } u_i \\ 0, & \text{others} \end{cases} \tag{3}
\]

Similar to fuel consumption, the total time of \( u_i \) to complete all assigned tasks can be achieved as follows:

\[
s_i^{\text{total}} = \sum_{j=1}^{N+1} \sum_{k=1}^{3} \left( s_{ij}^{\text{flight}} x_{ijk} + s_{ij}^{\text{wait}} x_{ijk} + s_{ij}^{\text{execute}} x_{ijk} \right) \tag{4}
\]

where \( s_{ij}^{\text{flight}}, s_{ij}^{\text{wait}}, \) and \( s_{ij}^{\text{execute}} \) denote the time consumed by \( u_i \) in the state \( F \), state \( W \), and state \( E \), respectively. \( s_{ij}^{\text{execute}} \) is predetermined by the system, \( s_{ij}^{\text{flight}} \) and \( s_{ij}^{\text{wait}} \) can be achieved by the following formulas:

\[
s_{ij}^{\text{flight}} = \frac{d_{k,i}(i)}{v_i} \tag{5}
\]

\[
s_{ij}^{\text{wait}} = \psi_i^w(i,j) - \psi_k^w(i,j) \tag{6}
\]

In equation (5), \( v_i \) denotes the flight speed of \( u_i \), \( d_{k,i}(i) \) denotes the distance \( u_i \) needs to fly to the next target \( t_j \) after completing task \( k \) on \( t_j \). In equation (6), \( \psi_i^w(i,j) \) and \( \psi_k^w(i,j) \) denote the time that \( u_i \) starts to execute task \( k \) (the completion time of the task with higher priority than task \( k \)) and the time of \( u_i \) to start to get stuck into waiting after reaching \( t_j \), respectively.

In the multiple UAVs task assignment problem, we propose the concept of execution capability to better address the issue of workload unbalance. Execution capability is defined as the ability of UAV to continue executing tasks based on remaining resources, which is an important means to prevent UAVs from running out of resources. The more resources remain, the greater the execution capability is. When the remaining resources of UAV are sufficient, the ability of the UAV to continue executing tasks is relatively strong. In this condition, the execution capability decreases slowly. On the contrary, when the remaining resources of UAV are exhausted, the UAV is most likely to consume all resources and cannot continue executing tasks. To avoid this situation, the execution capability should decrease sharply in this condition. Since the resource of UAV changes over time, execution capability is a time-varying concept. In this study, traveled range is considered as one of the criteria for the evaluation of execution capability. Thus, the range-impact execution capability \( C_{\text{range}}(\psi) \) is defined. Since range is inversely proportional to remaining resources, it is inversely proportional to execution capability. Thus, \( C_{\text{range}}(\psi) \) of \( u_i \) is defined as follows:

\[
C_{\text{range}}(\psi) = \frac{\kappa_i}{\psi^i_{\text{max}}} \tag{7}
\]

where \( d_{i}(\psi) \) and \( d_{i,\text{max}} \) represent the traveled distance at time \( \psi \) and the allowed maximum range of \( u_i \), respectively. \( \kappa_i \) denotes the rate at which \( C_{\text{range}}(\psi) \) varies with \( d_{i}(\psi) \). The smaller \( \kappa_i \) is, the slower the rate changes. The value range of \( \kappa_i \) is: \( \kappa_i \in [0, 1) \).

In equation (7), with the increase of time \( \psi \), \( d_{i}(\psi) \) gradually increases, \( C_{\text{range}}(\psi) \) gradually decreases. As the value of \( d_{i}(\psi) \) approaches \( d_{i,\text{max}} \), \( C_{\text{range}}(\psi) \) drops sharply. Thus, it is reasonable to construct equation (7) based on an exponential function to express the relationship between range and execution capacity.

Naturally, \( C_{\text{range}}(\psi) \) of the UAV system is calculated as

\[
C_{\text{total}}(\psi) = \sum_{i=1}^{M} C_{\text{range}}(\psi), \forall i, d_{i}(\psi) < d_{i,\text{max}} \tag{8}
\]

The execution capability of UAV is limited due to its limited resources. The situation that traveled distance of any UAV exceeds its permitted limit, \( C_{\text{range}}(\psi) \) should be regarded as invalid. In this situation, \( C_{\text{total}}(\psi) \) is set to 0.

Note that resources of strike UAV are related to the number of carried weapons besides range. The weapon-impact execution capability \( C_{\text{weapon}}(\psi) \) is defined. \( C_{\text{weapon}}(\psi) \) of \( u_i \) is defined as follows:

\[
C_{\text{weapon}}(\psi) = \frac{\psi_{\text{max}}^{\text{w}}}{\psi_{\text{w}}^{\text{max}}}, \forall i, 1, 2, ..., M_{\text{St}} \tag{9}
\]

where \( \psi_{\text{max}}^{\text{w}} \) and \( \psi_{\text{w}}^{\text{max}} \) denote the maximum weapon number that \( u_i \) can carry and the number of weapons left in \( u_i \), respectively. \( \kappa_{\text{i}} \) denotes the rate at which \( C_{\text{weapon}}(\psi) \) varies with \( \psi_{\text{w}}(\psi) \). The larger \( \kappa_{\text{i}} \) is, the slower the rate changes. The value range of \( \kappa_{\text{i}} \) is: \( \kappa_{\text{i}} \in [0, 1) \).

In equation (9), with the reduction of \( \psi_{\text{w}}(\psi) \), \( C_{\text{weapon}}(\psi) \) decreases. When \( \psi_{\text{w}}(\psi) \) is closed to 0, \( C_{\text{weapon}}(\psi) \) greatly reduces. Therefore, the exponential function in equation (9) is suitable for describing the relationship between the execution capability and the number of remaining weapons.
Similar to equation (8), \( C_{\text{weapon}}(\cdot) \) of the UAV system can be achieved by
\[
C^\text{total}_{\text{weapon}}(\cdot) = \begin{cases} 
\sum_{i=1}^{M_0} C^\text{state}_{\text{weapon}}(\cdot), \forall i, w_i(\cdot) > 0 \\
0, \text{otherwise}
\end{cases}
\] (10)

Based on equations (7) and (9), the more residual resources UAV system has, the stronger its execution capability is. In the ideal task assignment process, UAVs with sufficient residual resources (high execution capability) should be fully utilized; UAVs with little residual resources (low execution capability) should be assigned with few tasks. In this way, the situation that some UAVs run out resources due to executing excessive tasks can be avoided, and the workload balance constraint can be satisfied. Thus, the execution capacity of UAV system is directly proportional to workload balance degree of the allocation result.

For the sake of concentrating on the main issue of this study, we make the following assumptions: (1) UAVs are not destroyed when performing tasks\(^{25}\); (2) the actual flight distances of UAVs are replaced by Euclidean distances\(^{2,24}\); (3) UAVs fly at different altitudes to prevent collisions between them\(^{19}\); (4) UAV cannot execute multiple tasks simultaneously; (5) the distance that the UAV moves in the task execution state is negligible.

**Problem formulation**

**Resource consumption function.** We adopt the total traveled distance \( d_{\text{total}} \), total fuel consumption \( \text{fuel}_{\text{total}} \), and time required to complete all tasks \( \psi_{\text{total}} \) as the main factors to describe the resource consumption of UAV system. The traveled distance, fuel consumption, and task execution time of a single UAV can be calculated by equations (1), (2), and (4), respectively. Therefore, \( d_{\text{total}} \), \( \text{fuel}_{\text{total}} \), and \( \psi_{\text{total}} \) can be achieved by following formulas:
\[
d_{\text{total}} = \sum_{i=1}^{M_0} d_i
\] (11)
\[
\text{fuel}_{\text{total}} = \sum_{i=1}^{M_0} \text{fuel}^i_{\text{total}}
\] (12)
\[
\psi_{\text{total}} = \max_i (\psi^i_{\text{total}})
\] (13)

On the premise that the three factors mentioned above are equally important, the resource consumption function can be achieved as follows:
\[
f_{\text{consume}} = \frac{d_{\text{total}}}{\sum_{i=1}^{M_0} d_i^{\max}} + \frac{\text{fuel}_{\text{total}}}{\sum_{i=1}^{M_0} \text{fuel}^i_{\text{max}}} + \frac{\psi_{\text{total}}}{\max_i (\psi^i_{\text{max}})}
\] (14)

where \( \text{fuel}^i_{\text{max}} \) and \( \psi^i_{\text{max}} \) represent the maximum fuel capacity and the endurance time of \( u_i \), respectively.

**Detection revenue function.** Some existing studies assume that UAVs only can perform tasks at the target points to ensure the effective execution of tasks.\(^ {24,29}\) The detection scope of surveillance UAVs is not considered in these task assignment model, which may lead to unnecessary resource consumption of UAV system.\(^ {30}\) In Jia et al.,\(^ {25} \) the detection scope of UAV is represented by detection radius. In the allowed detection scope, the closer UAV gets to target, the higher detection accuracy is. For the sake of quantifying the detection accuracy of UAV, the detection revenue function \( f_{\text{revenue}} \) is proposed. Given that \( f_{\text{revenue}} \) is proportional to detection accuracy, it is proportional to the distance between the UAV and target. Detection revenue function \( f_{\text{revenue}} \) can be achieved by the following formula:
\[
f_{\text{revenue}} = \frac{\sum_{i=1}^{M_0} \sum_{j=1}^{N} N \sum_{k=1}^{2} (x_{ij} \ln(2 - \frac{\text{fuel}^i_{\text{total}}}{\text{fuel}^i_{\text{max}}}))}{\sum_{i=1}^{M_0} \sum_{j=1}^{N} \sum_{k=1}^{2} x_{ij}}
\] (15)

where \( x_{ij} \) is a binary variable, which equals 1 when the task \( t_j \) on the \( u_i \) is assigned to \( u_i \). Note that task \( t_j \) can only be reconnaissance or verification task. \( r_{ij} \) denotes the distance between the surveillance UAV \( u_i \) and the position where task \( t_j \) on \( u_i \) is executed. \( r_{\text{detect}} \) is the detection radius of \( u_i \). Given that the task completion effect of UAV becomes better with the increase of detection accuracy, task completion effect is proportional to \( f_{\text{revenue}} \).

**Optimization model.** The task assignment optimization model of heterogeneous UAV system is described as follows:
\[
\max f = l_{\text{res}} f_{\text{revenue}} + \frac{C^\text{state}_{\text{weapon}}(\cdot)}{M_0} + \frac{C^\text{weapon}_{\text{weapon}}(\cdot)}{M_0}
\] (16)
\[
s.t.
\]
\[
\sum_{j=1}^{N} x_{ij} = 1
\] (17)
\[
\sum_{j=1}^{N} x_{ij} = 3
\] (18)
\[
(\psi^i_{\text{total}} + \psi^i_{\text{exec}}) \leq (\psi^i_{\text{att}} + \psi^i_{\text{exec}}) \leq (\psi^i_{\text{sta}} + \psi^i_{\text{exec}})
\] (19)
\[
\forall i \in [1, M_0], w^i_{\text{total}} \leq w^i_{\text{max}}
\] (20)
\[
\forall i \in [1, M_0], r_i \leq r_{\text{detect}}
\] (21)
\[
\forall i, \psi^i_{\text{total}} \leq \psi^i_{\text{max}}, \text{fuel}^i_{\text{total}} \leq \text{fuel}^i_{\text{max}}, d^i_{\text{total}} \leq d^i_{\text{max}}
\] (22)

In the fitness function (equation (16)), \( l_{\text{res}} \) and \( l_{\text{exec}} \) indicate the range-impact execution capability flag and weapon-impact execution capability flag, respectively. They can be achieved by
The denominator of fitness function is resource consumption function, and the molecule consists of detection revenue function and execution capability of the UAV system.

Equations (17) and (18) ensure that each task only can be completed by one UAV, and all tasks must be completely performed. Equation (19) is the task priority constraint, where $\psi_{\text{rec}}^i(j)$, $\psi_{\text{att}}^i(j)$, and $\psi_{\text{ver}}^i(j)$ represent the time when task $R_i$, task $A_i$, and task $V_i$ begin to be performed on $t_j$, respectively. $\psi_{\text{exe}}^{R_i}$, $\psi_{\text{exe}}^{A_i}$, and $\psi_{\text{exe}}^{V_i}$ represent the time required to complete task $R_i$, task $A_i$, and task $V_i$, respectively. In equation (20), $w_{\text{total}}$ and $w_{\text{max}}$ denote the consumed weapon number of UAV and the maximum weapon number it can carry. Thus, equation (20) indicates the constraint on weapon resource of strike UAVs. In equation (21), $r_i$ denotes the distance between the UAV $u_i$ and the target during the mission. Equation (21) indicates the constraint on detection scope of surveillance UAVs. Equation (22) ensures that the task execution time, fuel consumption, and range of any UAV are limited.

Fitness $f$ (equation (16)) represents the satisfaction of UAV system with current task assignment solution. $f$ should decrease with the increase of resource consumption ($f_{\text{consume}}$), and increase with the improvement of task completion effect ($f_{\text{revenue}}$) and workload balance degree ($C_{\text{range}}(\psi), C_{\text{weapon}}(\psi)$). The proposed fitness function is capable of describing these properties.

In the proposed task assignment optimization model, we jointly consider the task completion effect, workload unbalance, and other factors. Although the proposed model makes the solutions more efficient, it makes the task assignment optimization problem more complicated. Especially in the large-scale task assignment, the proposed optimization problem is more difficult to be solved.

**Fuzzy elite strategy genetic algorithm**

GA, a heuristic algorithm with fast random search ability, can provide good approximated solutions for the complex task assignment problem. However, the traditional GA converges slowly and easily converges to the local extreme when the search space is large. Therefore, the FESGA is proposed. A simple flow chart of the proposed FESGA is illustrated in Figure 1.

**Encoding**

Encoding of chromosome is the most critical part of GA. However, traditional encoding method cannot be directly applied to the proposed problem due to the heterogeneity of UAVs and the particularity of tasks. Therefore, we propose a matrix encoding method to accommodate the task assignment background. Each chromosome is encoded by a matrix of five lines and $3N$ columns.

1) **The encoding method of UAV chromosome:** The UAV chromosome is composed of $M$ UAV sub-chromosomes, a complete UAV chromosome is achieved by arranging these UAV sub-chromosomes in the order of UAV number from small to large. An encoding example of UAV chromosome is given in Figure 2. The gene (column) sequence of each UAV sub-chromosome represents the task execution order of corresponding UAV. The first three lines of chromosome represent the target number, task number, and UAV number, respectively.
number of each target contains 1, 2, and 3, which represent taskR, taskA, and taskV, respectively. Each gene represents a task assignment for a specific UAV. Since all tasks are encoded as the first two lines of UAV chromosome, the encoding method satisfies the constraint that all tasks are completely performed (equation (18)). Moreover, UAVs can be assigned to the appropriate type of tasks by encoding the target, task, and UAV as the separate lines. Thus, the proposed encoding method is applicable to the proposed task assignment background.

In addition, the fourth line of UAV chromosome is the distance \( r \) between surveillance UAV and target center, the fifth line of UAV chromosome is the course angle \( \theta \) of the UAV. Value ranges of \( r \) and \( \theta \) are: \( r \in [0, r_{\text{detect}}] \), \( \theta \in [0, 2\pi] \). Note that the fourth line and fifth line of genes in the strike UAV sub-chromosome are 0. Encoding method of the last two lines can determine the relative position of UAVs and targets, so as to help the algorithm achieve the value of detection revenue function more conveniently.

The proposed encoding method accomplishes the efficient assignment of all tasks, and the task execution order is encoded as the gene sequences of UAV chromosome. Thus, a UAV chromosome achieved by the proposed encoding method can represent a complete task assignment scheme.

2) The encoding method of target chromosome: The target chromosome is composed of \( N \) target sub-chromosomes, a complete target chromosome is achieved by arranging these target sub-chromosomes in the order of target number from small to large. Each target sub-chromosome is encoded in a matrix of five lines and three columns. An encoding example of \( t_1 \) sub-chromosome is shown in Figure 3. In Figure 3, the meaning of each line in target chromosome is same as that in UAV chromosome. However, the order of genes in target sub-chromosome is fixed, it cannot represent the task execution order of UAVs. Therefore, different from UAV chromosome, a target chromosome cannot represent a complete task assignment scheme.

Figure 2. Encoding of the UAV chromosome.

Figure 3. Encoding of the target chromosome.

Initialization

Most of studies generate the initial population by the way of random generation. The way constructs a large search space when solving complex problems, which may lead to a slow convergence speed of the algorithm. Therefore, we propose the novel initialization method for target chromosome and UAV chromosome.

1) Target chromosome initialization: Depending on the proposed encoding method, the first two lines of target chromosome can be easily achieved. We assume that matrix formed by the first two lines of target chromosome is \( X_t \), \( X_t = \{x_{1t}, x_{2t}, \ldots, x_{3N}\} \), elements in \( X_t \) represent the column of the matrix. Then, a column \( x_i \) is randomly selected from \( X_t \), and a complete gene \( g_i \) can be achieved based on \( x_i \). For the sake of describing the generation process of \( g_i \), more simply and clearly, the \( j \)-th line of \( g_i \) and \( x_i \) are expressed as \( g_i(j) \) and \( x_i(j) \), respectively. If \( x_i(2) \) is 1 or 3, a surveillance UAV should be selected, and its number is used as \( g_i(3) \). The probability of each surveillance UAV being selected is as follows:

![Figure 2. Encoding of the UAV chromosome.](image1)

![Figure 3. Encoding of the target chromosome.](image2)
Afterwards, $g_i(4)$ and $g_i(5)$ are randomly generated within their range of values. If $x_i(2)$ is 2, a strike UAV should be selected. The probability of each strike UAV being selected is as follows:

$$P_{st}^j = \frac{C_{\text{range}}^j(\psi) + C_{\text{weapon}}^j(\psi)}{\sum_{j=1}^{M_{stu}} (C_{\text{range}}^j(\psi) + C_{\text{weapon}}^j(\psi))}, j = 1, 2, ..., M_{stu}$$ (25)

Afterwards, $g_i(4)$ and $g_i(5)$ are set to 0.

The gene generation process should be repeated until all elements of $X_t$ are selected to achieve a complete target chromosome. Note that elements cannot be selected repeatedly.

The proposed target chromosome initialization method should be repeated $n$ times to make the initial population size satisfy the predefined population size $n$. In the process of the proposed target chromosome initialization, the achieved chromosomes can satisfy the workload balance constraint by generating genes based on the execution capability. These workload balanced chromosomes can constitute a good initial population.

Since excellent initial population can speed up the evolution process, the proposed method can accelerate the convergence speed of algorithm. Especially in the complex large-scale task assignment problem, advantages of the method are more prominent.

2) UAV chromosome initialization: UAV chromosome is generated during the generation of target chromosome. Once a gene $g_i$ on a target chromosome is generated, which UAV sub-chromosome that $g_i$ belongs to should be determined according to $g_i(3)$. Suppose that the UAV sub-chromosome is $u_j$ sub-chromosome, $g_i$ should be placed behind genes which have been generated on $u_j$ sub-chromosome. For example, the randomly achieved generation sequence of genes in the target chromosome is: $g_4, g_5, g_1, g_3, ...$, the generation process of UAV chromosome is demonstrated in Figure 4. $N$ UAV sub-chromosomes are achieved when all genes of the target chromosome are generated. Then, a complete UAV chromosome can be easily achieved.

In the proposed UAV initialization method, genes on the UAV chromosome correspond one to one with genes on the target chromosome. Therefore, the UAV initialization method ensures that each UAV chromosome has a unique corresponding target chromosome. In addition, the gene generation order of target
chromosome is converted into the gene sequence of UAV sub-chromosome. Given that the gene sequence represents the task execution order of UAV, the generated UAV chromosomes can represent a complete assignment scheme and satisfy the encoding requirements.

Fuzzy elite degree
Elitism has a great promotion effect on GA. Convergence speed of the algorithm can be improved by selecting some elite individuals based on the fitness function value.\(^{33}\) However, it may reduce the population diversity and finally cause that the algorithm falls into local extremum. Therefore, we propose the concept of fuzzy elite degree \(E\) to eliminate this drawback. Different from elite individuals defined in traditional GA, all individuals in the proposed FESGA are elite to a certain extent. \(E\) is adopted to measure the elite degree of an individual. The fitness function value \(f\) and the times the individual appears in current iteration \(A\) are the major factors of \(E\). Considering that the requirements of elite individuals are different in different genetic operations, we present selection fuzzy elite degree \(E_s\), crossover fuzzy elite degree \(E_c\), and mutation fuzzy elite degree \(E_m\), respectively.

Selection fuzzy elite degree \(E_s\). In the selection process of proposed FESGA, some individuals with high elite degree are selected as elite individuals. Individuals with high \(f\) are selected as elite individuals with high probability, so as to retain good genes of the good individuals. Meanwhile, individuals with few \(A\) also have the opportunity to be selected, so as to ensure the population diversity. Therefore, \(E_s\) is directly proportional to \(f\) and inversely proportional to \(A\). \(E_s\) of the \(i\)-th individual is shown below:

\[
E_s^i = \alpha_s \left( \frac{\log_2 \left( \frac{f_i}{\max(f)} + 1 \right)}{\max(\log_2 \left( \frac{f_i}{\max(f)} + 1 \right))} \right) + \beta_s \left( \frac{\max \left( 1 + e^{\frac{-A_i}{\max(A)}} \right)}{1 + e^{\frac{-A_i}{\max(A)}}} \right), i = 1, 2, \ldots, n
\]

(27)

where \(f_i\) and \(A_i\) are the fitness value and occurrence value of the \(i\)-th individual. \(\alpha_s\) and \(\beta_s\) are constants from 0 to 1. They indicate the influence extent of \(f\) and \(A\), respectively. \(\alpha_s + \beta_s = 1\). Considering that \(f\) is more important than \(A\), \(\alpha_s \in [0.5, 1]\), \(\beta_s \in [0, 0.5]\). \(l_i\) is adopted to adjust the range of part \(A\) in \(E_s\). For the sake of ensuring that the value range of part \(A\) in \(E_s\) is closer to \([0, 1]\), \(l_i \in [0.4, 0.7]\). And the larger \(l_i\) is, the slower \(E_s^i\) decreases with the increase of \(A_i\).

Crossover fuzzy elite degree \(E_c\). In the crossover process of proposed FESGA, individuals with higher elite degree are selected as parents with higher possibility. Parents with high \(f\) can improve the superiority of offsprings.\(^{34}\) Meanwhile, the operation that parents with large \(A\) are replaced by the newly generated offsprings can increase the population diversity. Therefore, \(E_c\) is directly proportional to \(f\) and \(A\). \(E_c\) of the \(i\)-th individual can be achieved in equation (28) by modifying the part of occurrence number in equation (27).

\[
E_c^i = \alpha_c \left( \frac{\log_2 \left( \frac{f_i}{\max(f)} + 1 \right)}{\max(\log_2 \left( \frac{f_i}{\max(f)} + 1 \right))} \right) + \beta_c \left( \frac{l_i e^{\frac{-l_i}{\max(A)}}}{\max(l_i e^{\frac{-l_i}{\max(A)}})} \right), i = 1, 2, \ldots, n
\]

(28)

where \(\alpha_c \in [0.5, 1]\), \(\beta_c \in [0, 0.5]\), and \(\alpha_c + \beta_c = 1\). \(l_{c1}\) is used to regulate the rate at which \(E_c^i\) varies with \(A_i\). \(l_{c1} \in (0, 1]\), and the larger \(l_{c1}\) is, the slower the rate changes. \(l_{c2}\) and \(l_{c3}\) is adopted to ensure that the range of part \(A\) in \(E_c\) is \([0, 1]\), their values depend on the value of \(l_{c1}\).

Mutation fuzzy elite degree \(E_m\). In the mutation process of proposed FESGA, individuals with lower elite degree are mutated with higher possibility and have higher mutation intensity. Similar to the \(E_s\), \(E_m\) is directly proportional to \(f\) and inversely proportional to \(A\). However, ideally, more solutions with high \(E_m\) are expected to be screened out in the mutation process to destroy good genes of the good individuals as little as possible. Therefore, take \(f\) as the example, with the increase of \(f\), \(E_m\) increases slowly in the early stage and increases greatly when \(f\) increases to nearly its maximum value. Thus, we construct new type exponential function in equation (29) to define \(E_m\) of the \(i\)-th individual.

\[
E_m^i = \alpha_m \left( \frac{\max \left( 1 + e^{\frac{-f_i}{\max(f)}} \right)}{1 + e^{\frac{-f_i}{\max(f)}}} \right) + \beta_m \left( \frac{1 - \left( e^{\frac{-f_i}{\max(f)}} \right)^{l_{m1}}}{\max \left( 1 - \left( e^{\frac{-f_i}{\max(f)}} \right)^{l_{m1}} \right)} \right), i = 1, 2, \ldots, n
\]

(29)

where \(\alpha_m \in [0.5, 1]\), \(\beta_m \in [0, 0.5]\), and \(\alpha_m + \beta_m = 1\). \(l_{m1}\) is used to adjust the range of part \(f\) in \(E_m\). For the sake of ensuring that the value range of part \(f\) in \(E_m\) is closer to \([0, 1]\), \(l_{m1} \in [0.4, 0.7]\). And the larger \(l_{m1}\) is, the slower \(E_m^i\) changes with the change of \(f_i\). \(l_{m2}\) is the rate at which
Algorithm 1. Selection process of FESGA

Input: $n$ individuals (chromosomes)
Output: $n_{elite}$ elite individuals; $n - n_{elite}$ ordinary individuals
1. Calculate $E_i$ of all individuals by equation (27)
2. Achieve the probability of an individual being elite individual
   \[ P_{elite} = \frac{E_i}{\sum_{i=1}^{n} E_i}, i = 1, 2, ..., n \]
3. Rank $n$ individuals in the order of $P_{elite}$ from large to small
4. Achieve an individual set $\Gamma = \{G_1, G_2, ..., G_n\}$
5. Select $\{G_1, G_2, ..., G_{n_{elite}}\}$ from $\Gamma$ as elite individuals
6. Calculate the probability of an individual being ordinary individual
   \[ P_{ord} = \frac{1}{\sum_{i=1}^{n} \frac{P_{elite}}{C_0}} \]
7. $b = 1$
8. While $b \leq n - n_{elite}$
9. Adopt the roulette wheel to select an ordinary individual
10. $b = b + 1$
11. End

$E_i$ varies with $A_i$. $l_{m2} \in (0, 1]$, the smaller $l_{m2}$ is, the slower the rate changes.

Fuzzy elite degree is the core concept of the proposed FESGA. In the proposed fuzzy elite degree, the part of fitness function value can help the algorithm quickly converge by largely retaining the good genes of individuals which are closed to the optimal solution; the part of the occurrence number can prevent the algorithm from converging to local extreme by changing the genes of the individuals with more occurrences.

Selection process of FESGA

In the selection process of traditional GA, individuals which do not participate in subsequent genetic operations are defined as elite individuals. Fitness function value is the only criteria for selecting elite individuals without considering other information of the optimization process.\textsuperscript{35} For the sake of comprehensively utilizing the effective information of optimization process and improving the performance of algorithm, we present a new selection method. Individuals involved in subsequent genetic operations are called as ordinary individuals in this study. The selection process of FESGA is shown in Algorithm 1.

As shown in Algorithm 1, the selection process of FESGA consists of two parts: elite individual selection process (the 1st to 5th statements), and ordinary individual selection process (the 6th to 11st statements). In the elite individual selection process, individuals with higher $E_i$ are selected with higher probability. Since $f$ and $A$ are jointly considered in $E_i$, the abilities of the algorithm to converge fast and achieve the global optimal solution are enhanced.

Crossover process of FESGA

The essence of crossover is that the new individuals are produced through the gene recombination of parents.

In Song et al.,\textsuperscript{34} a direction-based crossover operator (DBX) was presented to enhance the convergence performance and population diversity. However, DBX only can be applied to the numerically continuous system. It is not suitable for discrete system, such as the task assignment problem. Therefore, we present a parent selection method and a one-point crossover method.

Parent selection method is divided into two parts: selection of the first parent and the second parent.

Individuals with high $E_i$ are selected as the first parent $G_{fp}$ with high probability. The probability that individuals are selected as $G_{fp}$ is shown below:

\[ P_{fp}^i = \frac{E_i}{n - n_{elite}}, i = 1, 2, ..., n - n_{elite} \]

Then, $G_{fp}$ can be achieved by roulette wheel. Considering the definition of $E_i$, the way that selecting parent according to $E_i$ can accelerate the convergence speed of the algorithm and avoid the local extremum problem by improving the superiority and diversity of offspring generation, respectively.

For the sake of further improving search ability of the algorithm, we present the AI mechanism to choose the second parent. The details of AI mechanism are as follows.

First, a concept of elite distance $\Psi_{ij}$ is proposed to describe the difference degree between any two individuals. $\Psi_{ij}$ is defined as follows:

\[ \Psi_{ij} = |E_i - E_j| \]

Then, individuals which have longer elite distance from $G_{fp}$ are selected as the second parent with higher probability. Meanwhile, the probability increases slowly in the close elite distance to ensure the sufficient difference of parent generation. The selection probability of the $j$-th individual can be achieved by

\[ P_{sp}^j = \frac{1}{1 + e^{-f \frac{\Psi_{ij}}{\kappa_{sp}}}}, j = 1, 2, ..., n - n_{elite} - 1 \]

where $\kappa_{sp}$ is adopted to adjust the range of $P_{sp}^j$. For the sake of ensuring that the value range of $P_{sp}^j$ is $[0, 1]$, $\kappa_{sp} \in [0.4, 0.7]$. The larger $\kappa_{sp}$ is, the slower $P_{sp}^j$ changes with the change of $\Psi_{ij}$.

Finally, the second parent is achieved by roulette wheel.

In the selection process, more diverse offsprings can be achieved by increasing the difference between parent generation. $f$, the criterion to evaluate individuals, is adopted to measure the difference in solution between individuals in existing study.\textsuperscript{32} $A$ is the occurrence number of the individual in current iteration, it can reflect the difference between individuals during the optimization process. In the proposed AI mechanism, both the solution difference ($f$) and individual difference in the
iteration process \((A)\) are considered. Thus, difference considered in AI mechanism is more abundant. The AI mechanism is more suitable for complex large-scale task assignment problem.

After determining parents involved in the crossover, the proposed one-point crossover method is adopted to achieve offsprings. The crossover process of a pair of individuals is shown in Figure 5.

Part 1 of Figure 5 can be summarized as following steps.

Step 1: The first two lines of parent 1 are directly passed on to nonstandard offspring 1 to ensure the integrity of the tasks.

Step 2: A crossover point are randomly set in the code string of parent 1. All genes in front of the crossover point in parent 1 are directly transmitted to nonstandard offspring 1.

Step 3: Other genes of nonstandard offspring 1 are provided from parent 2. Gene \(g_{4}^{\text{parent1}}\) in Figure 5 is taken as an example. First, the position of gene \(g_{4}^{\text{parent1}}\) in parent 1 \((w_{4})\) is recorded. Then, the gene which has the same target and task with \(g_{4}^{\text{parent1}}\) \((g_{4}^{\text{parent2}})\) is found in parent 2. Finally, \(g_{4}^{\text{parent2}}\) is placed at the position \(w_{4}\) in nonstandard offspring 1.

In part 2 of Figure 5, nonstandard offspring 2 is achieved, and the generation process is similar to that of nonstandard offspring 1. Note that nonstandard offspring 2 is composed of genes in front of the crossover point in parent 2 and genes after the crossover point in parent 1.

In part 3 of Figure 5, the achieved nonstandard offsprings are standardized. Genes whose third line (UAV number) are same are put together to achieve the standard form of chromosomes.

In the one-point crossover method, the first two lines of parents are fully inherited to offsprings, which can guarantee that all tasks are assigned and these tasks are not repeatedly executed. Since the achieved offsprings can still represent a complete allocation scheme, the proposed one-point crossover method is suitable for multiple heterogeneous UAVs task assignment background.

The parent selection method and one-point crossover method should be repeated until \(nP_{c}\) pairs of offsprings are achieved, where \(P_{c}\) is the crossover probability.

**Mutation process of FESGA**

Mutation is helpful for the algorithm to avoid falling into local extremum.\(^{36}\) In the completely random traditional mutation method, the mutation intensity of good individuals and bad individuals are same. It may cause that good individuals get worse, and bad individuals do not get better.
not have mutation intensity to overstep the local extreme. Therefore, we propose the HM rule in the mutation process. Meanwhile, we present three mutation methods.

**Mutation methods**

1. Change the execution sequence of tasks: randomly exchange the position of genes on any UAV sub-chromosome.
2. Assign a different UAV to a randomly selected task: randomly select a gene, and update the third line of the gene by randomly selecting a different matching UAV based on the type of task on the gene.
3. Change the location of a randomly selected surveillance UAV: randomly select a gene $g_m$ whose third line represents the surveillance UAV. Then, change the values of $g_m(4)$ and $g_m(5)$. Note that the value range of $g_m(4)$ and $g_m(5)$ are: $g_m(4) \in [0, r_{\text{det.err}}]$, $g_m(5) \in [0, 2\pi]$.

A variety of different mutation methods can achieve different task allocation strategy from multiple aspects, so as to better avoid the algorithm being trapped in local extremum.

**The implementation process of HM rule**

Step 1: $E_m$ of all individuals are calculated by equation (29).

Step 2: Individuals which are involved in the mutation operation are selected by roulette wheel. We stipulate that individuals with higher $E_m$ participate in the mutation operation with smaller possibility to prevent the destruction of good genes in good individuals. Therefore, the probability that individuals are involved in mutation can be calculated by

$$P_{mu}^i = \frac{\frac{1}{E_m^i}}{\sum_{j=1}^{n_{\text{elite}}} \frac{1}{E_m^j}}, \quad i = 1, 2, \ldots, n - n_{\text{elite}}$$

Step 3: The elite class of the selected individual is determined based on $E_m$. The classification criteria is exhibited in Table 1.

Step 4: In the proposed HM rule, the lower the elite class of an individual is, the greater the mutation intensity of the individual. The different mutation rules are formulated for individuals of different elite classes. The specific content is shown in Table 1.

The HM rule should be repeated until $nP_m$ individuals are generated, where $P_m$ is the mutation probability.

The proposed HM rule enhances the ability of individuals with smaller $f$ to overstep the local extreme by giving these individuals larger mutation intensity. Thus, search ability of the algorithm is improved. In addition, the HM rule can generate a more diverse population by mutating individuals with high $A$. Thus, the ability of the algorithm to find the global optimum is further improved.

Note that some infeasible solutions that do not satisfy the limited resource constraint may be generated in the genetic operations, $f$ of these infeasible solutions are 0. The existence of infeasible solutions can destroy the integrity of the population. Therefore, we propose the no zero mechanism to deal with the infeasible solutions. The pseudo-code of the no zero mechanism is shown in Algorithm 2.

Algorithm 2. The process of no zero mechanism

Input: an infeasible solution (chromosome) $G_{\text{inf}} = \{g_1, g_2, \ldots, g_N\}$; $f_{\text{inf}}(f) \leq f_{\text{feasible}}$.

Output: a feasible solution $G_{\text{feasible}} = \{g_1, g_2, \ldots, g_N\}$.

1. While $f_{\text{inf}} = 0$
2. Randomly select a gene $g_i$ from $G_{\text{inf}}$
3. Achieve the task type $\eta_i$ by $g(i)$
4. Achieve a set $U_i$ consists of UAVs which can execute type $\eta_i$ of task
5. While $f_{\text{inf}} = 0$
6. Randomly select a UAV $u_i$ from $U_i$
7. Update $g(i)$ to the number of $u_i$
8. Achieved a nonstandard chromosome $G_{\text{non}}$
9. Convert $G_{\text{non}}$ to the standard from to achieve $G_{\text{feasible}}$
10. If $f_{\text{inf}} = \inf$ of $G_{\text{feasible}}$
11. If all UAVs of $U_i$ have been selected
12. Break
13. End
14. End
15. End

Methods for deadlock and sudden threat

**Deadlock**

In the task assignment problem with task priority constraint, the UAV system may fall into deadlock. The possibility that UAV system falls into deadlock

---

**Table 1.** The classification criteria of individuals.

| $E_m$ of individuals | Elite class | Mutation rule |
|----------------------|-------------|---------------|
| 0.85–1.00            | High-class  | Randomly adopt one of the mutation methods |
| 0.50–0.85            | Middle-class| Adopt any two of the mutation methods |
| 0.00–0.50            | Low-class   | Successively adopt all the three mutation methods |

---
Algorithm 3. The implement process of STC mechanism

Input: population \( P_i^{\text{in}} = \{G_i^1, G_i^2, ..., G_i^j\}; \) time \( j_{\text{th}} \) when detecting sudden threat
Output: new population \( P_i^{\text{new}} = \{G_i^1, G_i^2, ..., G_i^j\} \)

1. Achieve the state, resource consumption, and location of each UAV at \( j_{th} \)
2. \( j = 1 \)
3. While \( j \leq n \)
4. Achieve a new chromosome \( G_i^{\text{new}} \) by deleting genes which contain the information of completed tasks from \( G_i^j \)
5. Achieve genes \( \{k_{\text{per}}^1, k_{\text{per}}^2, ..., k_{\text{per}}^4\} \) containing information about tasks being performed
6. Achieve a matrix \( X_i^{\text{new}} \) by deleting the last three lines of \( G_i^{\text{new}} \) except for \( \{g_1^1, g_2^1, ..., g_k^1\} \)
7. \( k_{\text{threat}}(1) = N + 1, k_{\text{threat}}(2) = 2 \)
8. Select a strike UAV \( u_{\text{st}} \) according to \( P_{\text{threat},i} \), \( i = 1, 2, ..., M_{\text{St}} \)
9. \( k_{\text{threat}}(3) = \) the number of \( u_{\text{st}}; k_{\text{threat}}(4) = 0; k_{\text{threat}}(5) = 0 \)
10. Add \( g_2^1 \) after the last column of \( X_i^{\text{new}} \)
11. \( k_{\text{threat}}(1) = N + 1, k_{\text{threat}}(2) = 3 \)
12. Select a surveillance UAV \( u_{\text{su}} \) according to \( P_{\text{threat},i} \), \( i = 1, 2, ..., M_{\text{Su}} \)
13. \( k_{\text{threat}}(3) = \) the number of \( u_{\text{su}}; g_{\text{threat}}(4) \) and \( k_{\text{threat}}(5) \)
14. Add \( g_2^2 \) after \( k_{\text{threat}} \)
15. Achieve a target chromosome \( G_i^j \)
16. Achieve \( G_i^{\text{new}} \) based on \( G_i^j \) according to the initialization method in Section 3.2
17. \( j = j + 1 \)
18. End
19. Achieve \( P_i^{\text{new}} \)

Increases with the increase of task scale. The deadlock status mainly includes self-lock and interlock.32

**Self-lock problem.** Self-lock means that lower-priority task on a target is executed earlier than the higher-priority task on the target.19 The self-lock problem is easy to be solved by swapping the execution order of the tasks.

**Interlock problem.** In most situations, the deadlock that UAV system is stuck in is interlock. In the interlocked status, multiple UAVs fall into endless waiting as they wait for each other to execute tasks.39 A pair of corresponding UAV chromosome and target chromosome contain the information of task execution order of UAVs and targets, respectively. The state of each UAV during the mission can be determined by decoding this pair of chromosomes. When two or more UAVs are in the permanent waiting state, the UAV system gets into interlock. Since the interlock problem involves the coordinated control of multiple UAVs, it is more complex and difficult to be solved than self-lock problem. We present two unlock methods for the interlock problem.

To describe the unlock process more clearly, we define the deadlock chromosome. Each deadlocked task assignment scheme is corresponding to a unique UAV chromosome. A new chromosome can be achieved by deleting the genes carrying information about the completed tasks from the UAV chromosome. The new chromosome is the deadlock chromosome. Similar to UAV chromosome, the deadlock chromosome can be divided into several deadlock sub-chromosomes according to the UAV number. The first gene in each deadlock sub-chromosome is defined as the deadlock gene.

**Unlock Method 1:** Firstly, a deadlock gene \( g_1^{\text{deadlock}} \) is randomly selected. Suppose \( g_1^{\text{deadlock}}(1) \) indicates the target \( t_i \). Then, gene \( g_h \) which contains the information of the task with the highest priority of \( t_i \) is found from the deadlock chromosome. Meanwhile, the deadlock sub-chromosome where \( g_h \) resides is determined. Finally, deadlock gene of the determined deadlock sub-chromosome switches places with \( g_h \).

**Unlock Method 2:** Firstly, a surveillance UAV \( u_t \) that has not completed its tasks is randomly selected. A reconnaissance task \( t_k \) is randomly chosen from unexecuted tasks. Suppose the gene carrying the information of \( t_k \) is \( g_c \). Then, \( g_c \) is moved to the position in front of the deadlock gene of \( u_t \) deadlock sub-chromosome, and \( g_c(3) \) is set to the number of \( u_t \).

Each unlock method is conducted with one-half probability. Note that each method is adopted to remove current deadlock status. After each unlocking, the new solution should be determined whether is deadlocked. If the solution is deadlocked, the unlock methods are adopted again. A deadlock-free solution can be achieved by repeating the above process.

Compared with the unlock methods that only exchanging the task execution order,19,28 our proposed unlock methods can also resolve the deadlock problem by adjusting the matching relationship between UAVs and targets. Therefore, the proposed unlock methods are more diversified, they can increase the search ability of the algorithm. Moreover, compared with digraph-based unlock method,32 fewer loops are involved in the proposed unlock methods. Thus, the proposed methods have less computation and faster unlock speed.
Especially in the complex large-scale task scenario, the proposed unlock methods have more obvious advantages.

### Task assignment with sudden threat

During the execution of scheduled tasks, UAV system may discover some unknown threats. We call these threats sudden threats in this study. All sudden threats should be immediately destroyed after being detected. There are two tasks on each sudden threat: the attack task \( t^A \_threat \) and verification task \( t^V \_threat \). Note that \( t^A \_threat = t^V \_threat \). The emergence of sudden threat changes the original task environment. For the sake of making the algorithm adapt to new environment quickly, we propose the STC mechanism.

In the proposed STC mechanism, the probability of \( u_i \) to execute \( t^A \_threat \) or \( t^V \_threat \) when detecting the sudden threat (at time \( \varphi_{ih} \)) can be achieved as follows:

\[
P^i_{\text{threat}}(\varphi_{ih}) = I_{ob}(l_1 \gamma_1 \frac{e^{-\varphi_{ih}/\tau_{\text{max}}}}{e^{-\varphi_{ih}/\tau_{\text{min}}}} + l_2 \gamma_2 \frac{e^{-\varphi_{ih}/\tau_{\text{max}}}}{e^{-\varphi_{ih}/\tau_{\text{min}}}} \\
+ l_3 \gamma_3 \frac{e^{-\varphi_{ih}/\tau_{\text{max}}}}{e^{-\varphi_{ih}/\tau_{\text{min}}}} + l_4 \gamma_4 \frac{e^{-\varphi_{ih}/\tau_{\text{max}}}}{e^{-\varphi_{ih}/\tau_{\text{min}}}} + l_5 \gamma_5 \frac{e^{-\varphi_{ih}/\tau_{\text{max}}}}{e^{-\varphi_{ih}/\tau_{\text{min}}}})
\]

(34)

where \( \varphi_{ih} \) represents the remaining waiting time of \( u_i \) in the waiting state, \( \varphi_c \) represents the remaining time of \( u_i \) in the task execution state to complete the task, \( d_i(\varphi_{ih}) \) is the traveled range of \( u_i \) at \( \varphi_{ih} \), \( w_i(\varphi_{ih}) \) is the remain weapon number on \( u_i \) at \( \varphi_{ih} \), \( d_i \) represents the Euclidean distance between the location of \( u_i \) and sudden threat, \( \gamma_1, \gamma_2, \gamma_3, \gamma_4, \) and \( \gamma_5 \) represent the proportion of \( \varphi_{ih}, \varphi_c, d_i(\varphi_{ih}), w_i(\varphi_{ih}) \), and \( d_i \) in equation (34), respectively. \( I_{ob} \) is a binary variable, which equals 1 when \( u_i \) has executed tasks before \( \varphi_{ih} \). Moreover, \( l_1, l_2, l_3, \) and \( l_4 \) are binary variables. \( l_1, l_2, \) and \( l_3 \) are set as 1 when \( u_i \) is in the waiting state, task execution state, and flight state, respectively. \( l_4 \) equals 1 only when \( u_i \) is strike UAV. \( \kappa_{ob1}, \kappa_{ob2}, \kappa_{ob3}, \kappa_{ob4}, \) and \( \kappa_{ob5} \) denote the rate at which \( P^i_{\text{threat}}(\varphi_{ih}) \) varies with \( \varphi_{ih}, \varphi_c, d_i(\varphi_{ih}), w_i(\varphi_{ih}), \) and \( d_i \) respectively. Their value ranges are: \((0, 1]\). For \( \kappa_{ob1}, \kappa_{ob2}, \kappa_{ob3}, \) and \( \kappa_{ob5} \), the smaller the values of these parameters are, the slower the rate changes. But for \( \kappa_{ob4} \), the larger the value is, the slower the rate changes.

### Simulation results

In this section, we evaluate the search ability and convergence speed of algorithms via extensive numerical simulations. Specifically, we consider a small-scale task assignment scenario (three targets and three UAVs) and a large-scale task assignment scenario (seven targets and four UAVs) in the simulations. The superiority of proposed algorithms is validated through the simulation in comparison with the variants of GA (modified genetic algorithm (GA), DGA, co-evolutionary multi-population genetic algorithm (CMGA), modified genetic algorithm combined with greedy strategy (MGGS)) and other optimization algorithms (particle swarm optimization algorithm based on guidance mechanism (GMPSO), artificial bee colony (ABC) algorithm, and enumeration method). The effectiveness of proposed algorithm is confirmed by analyzing the typical solutions. All numerical simulations were conducted under the MATLAB environment (i.e. MATLAB 2014, Intel Core i5-8400 CUP@2.80GHz).

In the simulations, parameters of UAV system and targets are shown in Tables 2 and 3, respectively. The resource consumption of UAVs under different states are shown in Table 4. Moreover, parameters of the proposed FESGA are summarized in Table 5.

To indicate the computational efficiency of the proposed FESGA intuitively, we analyze the complexity of the proposed algorithm as follows. The key factors affecting the running speed of proposed FESGA are the calculation of fuzzy elite degree, genetic operations of crossover and mutation, and unlocking of deadlock solutions. The complexity of these operations depends on the population size \( n \) and maximum iteration number \( \mu_{\text{max}} \). In the worst case, all individuals need to unlock. Therefore, the complexity of the proposed FESGA can be bound by \( O(n\mu_{\text{max}}) \). Further, the calculation time comparisons are given to show the complexity. The specific comparison results are presented in Table 6 below.

To verify the universal validity of FESGA, we perform 50 Monte Carlo comparison experiments under two different task-scale scenarios. Given that strong search ability can help the algorithm overstep the local extremum and find good solution, the average of the optimal fitness value achieved by algorithms in the Monte Carlo comparison experiments \( \text{avg}(f_{\text{best}}) \) and the error between the obtained result and the optimal

---

**Table 2. Parameters of UAV system.**

| \( u_i \) | Type | Initial location (m) | Detection range (m) | Weapon resources | Maximum range (m) | Endurance time (min) | Ideal flight speed (m/min) | Maximum fuel capacity (L) |
|---|---|---|---|---|---|---|---|---|
| \( u_1 \) | \( U_{ib} \) | [0, 0] | 30 | 0 | 4400 | 23 | 200 | 20 |
| \( u_2 \) | \( U_{ib} \) | [0, 0] | 30 | 0 | 4400 | 23 | 200 | 25 |
| \( u_3 \) | \( U_{ib} \) | [0, 0] | 0 | 11 | 2600 | 23 | 150 | 8 |
| \( u_4 \) | \( U_{ib} \) | [0, 0] | 0 | 11 | 2600 | 23 | 150 | 13 |
solution $err = |\text{ave}(f_{\text{best}}) - f_{\text{enu}}|/f_{\text{enu}}$ are taken as the criteria to judge the search ability of algorithms. Where $f_{\text{enu}}$ is the optimal solution obtained by the enumeration method. In addition, the number of iterations required to converge to a stable level $m_{\text{stable}}$ and the calculation time $G$ are adopted to measure the rapidity of algorithms. The comparison results are illustrated in Table 6.

As shown in Table 6, in the small-scale task scenario, $err$ of the presented FESGA is 0. It means that FESGA can converge to the global optimal quickly. In the large-scale task scenario, only $err$ of the enumeration method equals 0. As a brute force method, the enumeration method can find the global optimal solution at all costs, but it is not suitable for practical applications due to the large amount of calculation time and cost. In addition to the enumeration method, FESGA has the smallest $err$ compared with other heuristic algorithms. Therefore, the proposed FESGA can find high-quality suboptimal solutions in limited running time. It shows strong searching ability. In addition, only CMGA and GA are slightly better than the proposed FESGA as for the calculation time ($G$). But the small reduction in running time comes at the expense of algorithm accuracy. Meanwhile, GMPSO converges to the stable level in the fastest way by comparing $m_{\text{stable}}$ of all algorithms. Nevertheless, $err$ of GMPSO is too large to provide satisfactory task assignment schemes. The presented FESGA has the smallest $m_{\text{stable}}$ except GMPSO, it can provide good enough solutions while exhibiting the characteristic of fast convergence.

Generally, the convergence speed of algorithms is judged by $m_{\text{stable}}$. As shown in Figures 6 and 7, in any task scales, the four algorithms with the fastest convergence speed are arranged in order: GMPSO, FESGA, ABC, and MGGS. The differences in convergence performance of other algorithms are not obvious. Although the convergence speed of GMPSO is faster

### Table 3. Parameters of targets.

| Parameters | $t_1$ | $t_2$ | $t_3$ | $t_4$ | $t_5$ | $t_6$ | $t_7$ |
|------------|-------|-------|-------|-------|-------|-------|-------|
| Location (m) | [525, 335] | [347, 387] | [243, 615] | [163, 450] | [578, 513] | [114, 193] | [347, 150] |
| Required tasks | $R, A, V$ | $R, A, V$ | $R, A, V$ | $R, A, V$ | $R, A, V$ | $R, A, V$ | $R, A, V$ |
| Required weapon number | 2 | 2 | 3 | 2 | 3 | 2 | 2 |

### Table 4. Parameters involved in task scenario.

| $u_i$ | Fuel consumption per unit of time (L/min) | Time to perform tasks (min) |
|-------|---------------------------------|------------------------------|
| UAV state | $E$ | $W$ | $F$ | $R_T$ | $A_T$ | $V_T$ |
| $u_1$ | 0.80 | 0.30 | 1.10 | 0.30 | / | 0.30 |
| $u_2$ | 1.00 | 0.40 | 1.20 | 0.30 | / | 0.30 |
| $u_3$ | 0.90 | 0.30 | 1.10 | / | 0.15 | / |
| $u_4$ | 1.00 | 0.35 | 1.15 | / | 0.15 | / |

### Table 5. Parameters of the proposed FESGA.

| Parameters | Description | Setting | Parameters | Description | Setting |
|------------|-------------|--------|------------|-------------|--------|
| $\mu_{\text{max}}$ | Maximum iteration number in small scale | 700 | $\mu_{\text{max}}$ | Maximum iteration number in large scale | 3000 | $\mu_{\text{max}}$ |
| $n$ | Population size | 50 | $n$ | Population size | 50 | $n$ |
| $n_{\text{elite}}$ | Elite individual number | 3 | $n_{\text{elite}}$ | Elite individual number | 3 | $n_{\text{elite}}$ |
| $P_c$ | Crossover probability | 0.7 | $P_c$ | Crossover probability | 0.7 | $P_c$ |
| $P_m$ | Mutation probability | 0.3 | $P_m$ | Mutation probability | 0.3 | $P_m$ |
| $\kappa_r$ | Adjusting the value of Execution capability | 0.1 | $\kappa_r$ | Adjusting the value of Execution capability | 0.1 | $\kappa_r$ |
| $\alpha_i$ | Calculating $E_i$ | 0.7 | $\alpha_i$ | Calculating $E_i$ | 0.7 | $\alpha_i$ |
| $\beta_i$ | Calculating $E_i$ | 0.3 | $\beta_i$ | Calculating $E_i$ | 0.3 | $\beta_i$ |
| $P_{ip}$ | Calculating $P_{ip}$ | 0.6 | $P_{ip}$ | Calculating $P_{ip}$ | 0.6 | $P_{ip}$ |
| $\alpha_c$ | Calculating $E_c$ | 0.8 | $\alpha_c$ | Calculating $E_c$ | 0.8 | $\alpha_c$ |
| $\beta_c$ | Calculating $E_c$ | 0.2 | $\beta_c$ | Calculating $E_c$ | 0.2 | $\beta_c$ |
than that of the proposed FESGA, the results obtained by GMPSO are inferior to those obtained by FESGA. In addition, Figures 6 and 7 show that FESGA can quickly obtain the global optimal solution in small-scale task scenario, and the result obtained in large-scale task scenario is similar to the global optimal solution, which is obtained by enumeration method. Therefore, the optimization effect of FESGA is verified.

Figures 6 and 7 indicate that the comprehensive performance of proposed FESGA is superior to other algorithms in any task scales. First, FESGA is slightly inferior to GMPSO in terms of convergence speed. This is because that GMPSO effectively uses the global information to search the current optimal solution space. However, GMPSO is easy to fall into the local extremum due to the over-dependence of current optimal solution. Compared with other algorithms, the proposed FESGA shows the fastest convergence speed. It benefits from the high-quality population obtained by the proposed initialization method. In addition, as shown in Figures 6 and 7, FESGA shows obvious advantages in searching ability than other algorithms. Unlike GMPSO and ABC, the variants of GA can better explore the solution space by the mutation operation. However, the inflexible genetic operation intensity may lead to the insufficient searching strength in the later stage of algorithms. Compared with other algorithms, the great advantage of FESGA lies in the proposed concept of fuzzy elite degree (\(E_s, E_c, E_m\)). The fuzzy elite degree balances the contradiction between fast converge speed and strong search ability by optimizing the retention of excellent genes and the change of individuals. Meanwhile, the presented HM rule set sufficient mutation intensity for poor individuals to further increase the probability of overstepping the local extremum. The superiority of the proposed FESGA in terms of convergence speed and searching ability are verified in Figures 6 and 7.

### Table 6. Performance comparison of algorithms.

| Task scenario        | Algorithm | \(\text{ave}(f_{\text{best}})\) | err (%) | \(\Gamma\) (s) | \(H_{\text{stable}}\) |
|----------------------|-----------|-----------------|--------|-------------|----------------|
| Small-scale task scenario | FESGA    | 3.5545          | 0      | 128.08      | 286             |
|                      | MGGS      | 3.5279          | 0.75   | 148.83      | 568             |
|                      | CMGA      | 3.4792          | 2.12   | 98.82       | 598             |
|                      | DGA       | 3.2475          | 8.64   | 139.78      | 594             |
|                      | GA        | 3.5241          | 6.06   | 133.62      | 601             |
|                      | GMPSO     | 3.3392          | 6.06   | 133.62      | 27              |
|                      | ABC       | 3.5024          | 1.47   | 135.49      | 313             |
|                      | Enumeration | 3.5545      | 0      | 1863.24     | /               |
| Large-scale task scenario | FESGA    | 1.7049          | 3.17   | 1653.03     | 1833            |
|                      | MGGS      | 1.5896          | 9.92   | 1805.52     | 2019            |
|                      | CMGA      | 1.4791          | 16.00  | 1082.69     | 2517            |
|                      | DGA       | 1.4660          | 16.74  | 2966.43     | 2721            |
|                      | GA        | 1.4298          | 18.79  | 1253.02     | 2197            |
|                      | GMPSO     | 1.4124          | 19.78  | 2148.53     | 49              |
|                      | E         | 1.5019          | 14.70  | 1821.52     | 2920            |
|                      | Enumeration | 1.7607      | 0      | 4,4268.96   | /               |

**Figure 6.** Comparison of the optimization process in the small-scale task scenario.
In one case of the large-scale task scenario, a solution achieved by FESGA is taken as an example to verify the effectiveness of FESGA. The approximate trajectory of each UAV under the achieved solution is demonstrated in Figure 8. Then, we analyze the feasibility of the solution from two aspects of task priority constraint and workload balance constraint.

**The satisfied task priority constraint**

The information of task execution order is extracted from Figure 8. Then, the task sequence of UAV system is given in Figure 9. The timeline in Figure 9 starts with the beginning of task execution and ends with the completion of all tasks. The dotted orange line corresponds to the time when all tasks on each target are completed. For each UAV, the flight time required to reach the next target, task execution time and synchronous waiting time are represented by blue, red, and green, respectively. The number of target that UAV reaches and the task to be executed are shown at the end of the blue rectangle. In Figure 9, task execution order of all targets are: taskR, taskA, and taskV. That is, tasks on all targets are performed in order of priority from high to low. Thus, the effectiveness of FESGA in task priority constraint is verified. The proposed algorithm can reasonably plan the task order based on the task priority.

**The satisfied workload balance constraint**

Define an equalization coefficient $\zeta$ to evaluate the workload balance degree of the task assignment scheme. The equalization coefficient of $u_i$ is expressed as follows:

$$\zeta = \frac{\sum_{j=1}^{n} w_{ij}}{n}$$

where $w_{ij}$ is the workload of the $i$-th target assigned to the $j$-th UAV.
\[
\zeta_i = \rho_1 \frac{d_i}{d_{\text{max}}} + \rho_2 \frac{f_i}{f_{\text{max}}} + \lambda \rho_3 \frac{w_i}{w_{\text{max}}}, \quad i = 1, 2, \ldots, M
\]  

(35)

Where \(\lambda = \begin{cases} 1, & \text{if } u_i \text{ is a strike UAV} \\ 0, & \text{otherwise} \end{cases} \)  

(36)

\[\rho_1 + \rho_2 + \lambda \rho_3 = 1 \]  

(37)

In equation (35), \(\zeta_i\) reflects the proportion of consumed resources in the maximum consumable resources. If there are no significant difference found among \(\zeta\) of different UAVs, we can consider that this scheme satisfies the workload balance constraint. Nevertheless, given that the value of \(\lambda\) corresponding to different types of UAVs is different, only \(\zeta\) of the same type of UAVs can be compared. The evaluation results of the workload balance degree of UAV system is shown in Table 7.

In Table 7, \(a_i\) and \(v_i\) are the average and variance of \(\zeta\) of the same type of UAVs, respectively. Since \(v_i\) of surveillance UAV and strike UAV are both less than 1%, we can draw the conclusion that the solution satisfies the workload balance constraint. Therefore, the proposed FESGA can reasonably allocate tasks and plan the proportion of resources according to the airborne resources of UAVs.

The above experiments mainly compare the performance of algorithms in the static task scenario. To verify the wide adaptability of proposed FESGA and the superiority of proposed STC mechanism in dealing with sudden threat, we introduce sudden threat in the original task scenarios. Then, the comparison study on the optimization process of ABC, GMPSO, GA, DGA, CMGA, MGGS, FESGA, and FESGA with STC mechanism are demonstrated. The comparison results are indicated in Figures 10 and 11.

The focus of our study is the optimization process of the algorithm after detecting the sudden threat. Therefore, in the optimization process without sudden threat, we only show the last 50 iterations in Figures 10 and 11. In the simulation of small-scale and large-scale task scenarios, the sudden threat is detected at \(m = 700\) and \(m = 3000\), respectively. Changes in the resources, state and location of UAVs caused by the sudden threat will decrease the fitness function value, and the value decreases more obviously in large-scale task scenario. This is because that targets are more widely distributed in large-scale task scenario, more resources are consumed when dispatching UAVs to destroy the sudden threat. Task scenario with sudden threats has higher requirement on rapidity. Thus, we focus on the comparison of the rapidity performance of algorithms. Indicators \(\Gamma\) and \(\mu_{\text{stable}}\) can measure the rapidity of algorithms in two aspects of calculation time and convergence speed, respectively. Where \(\mu_{\text{stable}}\) can be obtained from Figures 10 and 11. The comparison results are indicated in Table 8.

As shown in Table 8, both \(\Gamma\) and \(\mu_{\text{stable}}\) of FESGA with STC mechanism are the minimum under different task scales. Therefore, the proposed STC mechanism can greatly improve the calculation time and convergence performance of FESGA in both small-scale and large-scale task scenarios. This is because the proposed

| \(u_i\) | Sailed distance (m) | Consumed fuel (L) | Consumed weapon number | \(\zeta\) (%) | \(a_i\) (%) | \(v_i\) (%) |
|-------|----------------------|-------------------|------------------------|---------------|------------|-----------|
| Surveillance UAV | \(u_1\) | 1266.7 | 7.89 | / | 34.11 | 34.77 | 0.0087 |
| | \(u_2\) | 1443.7 | 9.51 | / | 35.43 | 35.51 | 0.0087 |
| Strike UAV | \(u_3\) | 1441.7 | 5.58 | 8 | 65.98 | 60.90 | 0.5171 |
| | \(u_4\) | 1264.3 | 5.99 | 8 | 55.81 | 55.99 | 0.5171 |

Figure 9. Task sequence of UAV system.
STC mechanism can make full use of the feedback information of the sudden threat and UAV system, and help the algorithm rapidly adjust the allocation scheme. Thus, many excellent solutions can be achieved in a short time, convergence speed of the algorithm is accelerated.

**Conclusion**

In this study, we focus on the complex task assignment problem of heterogeneous UAV system with multiple constraints. The optimization model is formulated based on the comprehensive consideration of the UAV resource consumption, task completion effect, workload balance, task priority, and limited resources. The concept of fuzzy elite degree is proposed to optimize the retention and change intensity of genes, so as to improve the efficiency of the algorithm. Based on the concept, the FESGA is presented to assign tasks to heterogeneous UAV system. In addition, two unlock
methods are presented to deal with the deadlock solutions appearing in the optimization process. Then, the STC mechanism is proposed to help the algorithm converge quickly in the complex changing task scenario with sudden threats. Finally, the comparative results indicate that the proposed algorithm has significant search ability and convergence performance. Especially in large-scale task scenarios, advantages of the algorithm are more obvious.

Declaration of conflicting interests
The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding
The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported in part by the National Natural Science Foundation of China under Grant 62003295, Grant 61873224, Grant 41976182, and Grant 61873223, in part by the S&T Program of Hebei under Grant F2020203037, and F2019203031, in part by the China Postdoctoral Science Foundation under Grant 2020M670690, in part by the Science and Technology Research Project of Universities in Hebei under Grant QN2020301, in part by the Science Foundation for Postdoctoral of Hebei under Grant B2019003019, and in part by the PhD Foundation Project of Yanshan University under Grant BL18039.

ORCID iD
Song Han https://orcid.org/0000-0002-9072-420X

References
1. Yue W, Guan X and Wang L. A novel searching method using reinforcement learning scheme for multi-UAVs in unknown environments. *Appl Sci* 2019; 9: 4964.
2. Liu D, Wang J, Xu K, et al. Task-driven relay assignment in distributed UAV communication networks. *IEEE Trans Veh Technol* 2019; 68(11): 11003–11017.
3. Zhou Z, Feng J, Gu B, et al. When mobile crowd sensing meets UAV: energy-efficient task assignment and route planning. *IEEE Trans Commun* 2018; 66(11): 5526–5538.
4. Motlagh NH, Bagaa M and Taleb T. Energy and delay aware task assignment mechanism for UAV-based IoT platform. *IEEE Internet Things J* 2019; 6(4): 6523–6536.
5. Moon S, Oh E and Shim DH. An integral framework of task assignment and path planning for multiple unmanned aerial vehicles in dynamic environments. *J Intell Robot Syst* 2013; 70: 303–313.
6. Fu X, Feng P and Gao X. Swarm UAVs task and resource dynamic assignment algorithm based on task sequence mechanism. *IEEE Access* 2019; 7: 41090–41100.
7. Yao W, Qi N, Wan N, et al. An iterative strategy for task assignment and path planning of distributed multiple unmanned aerial vehicles. *Aerosp Sci Technol* 2019; 86: 455–464.
8. Deng Q, Yu J and Wang N. Cooperative task assignment of multiple heterogeneous unmanned aerial vehicles using a modified genetic algorithm with multi-type genes. *Chin J Aeronaut* 2013; 26: 1238–1250.
9. Zhu A and Yang SX. An improved SOM-based approach to dynamic task assignment of multi-robots. In: 2010 8th world congress on intelligent control and automation, Jinan, China, 7–9 July 2010, pp.2168–2173. New York: IEEE.
10. Verma A and Rai R. Modeling multi operator-multi-UAV (MOMU) operator attention allocation problem. In: *ASME 2013 international mechanical engineering congress and exposition*, San Diego, California, 15–21 November 2013. American Society of Mechanical Engineers Digital Collection. New York: ASME.
11. Jasper P, Sibley C and Coyne J. Using heart rate variability to assess operator mental workload in a command and control simulation of multiple unmanned aerial vehicles. In: *Proceedings of the human factors and ergonomics society annual meeting*. Los Angeles, CA, 2016, vol. 60, pp.1125–1129. Los Angeles: SAGE Publications.
12. Chen M and and Zhu D. A workload balanced algorithm for task assignment and path planning of inhomogeneous autonomous underwater vehicle system. In: *IEEE transactions on cognitive and developmental systems*, 23 August 2018, vol. 11, no. 4, pp.483–93. New York: IEEE.
13. Schumacher C, Chandler P, Pachter M, et al. UAV task assignment with timing constraints via mixed-integer linear programming. In: *AIAA 3rd “unmanned unlimited” technical conference, workshop and exhibit*, Chicago, Illinois, 20–23 September 2004, p.6410. ARC: Reston: AIAA.
14. Grettli EJ and Johansen TA. Path planning for UAVs under communication constraints using SPLAT! and MILP. *J Intell Robot Syst* 2012; 65: 265–282.
15. Hu X, Ma H, Ye Q, et al. Hierarchical method of task assignment for multiple cooperating UAV teams. *J Syst Eng Electron* 2015; 26: 1000–1009.
16. Sun BQ and Wang L. An estimation of distribution algorithm with branch-and-bound based knowledge for robotic assembly line balancing. *Complex Intell Syst*. Epub ahead of print 12 June 2020. DOI: 10.1007/s40747-020-00166-z.
17. Shima T, Rasmussen S and Gross D. Assigning micro UAVs to task tours in an urban terrain. *IEEE Trans Control Syst Technol* 2007; 15: 601–612.
18. Rasmussen SJ and Shima T. Tree search algorithm for assigning cooperating UAVs to multiple tasks. *Int J Robust Nonlinear Control* 2008; 18: 135–153.
19. Chen Y, Yang D and Yu J. Multi-UAV task assignment with parameter and time-sensitive uncertainties using modified two-part wolf pack search algorithm. *IEEE Trans Aerosp Electron Syst* 2018; 54: 2853–2872.
20. Xi C, Yongtai L, Liyuan Y, et al. Cooperative task assignment and track planning for multi-UAV attack mobile targets. *J Intell Robot Syst* 2020; 100(3): 1383–1400.
21. Kalayci CB and Gupta SM. Artificial bee colony algorithm for solving sequence-dependent disassembly line balancing problem. *Expert Syst Appl* 2013; 40(18): 7231–7241.
22. Bai X, Yan W, Ge SS, et al. An integrated multi-population genetic algorithm for multi-vehicle task assignment in a drift field. *Inf Sci* 2018; 453: 227–238.
23. Zhou Z, Li F, Zhu H, et al. An improved genetic algorithm using greedy strategy toward task scheduling
optimization in cloud environments. *Neural Comput Appl* 2020; 32(6): 1531–1541.

24. Schwarzrock J, Zacarias I, Bazzan AL, et al. Solving task allocation problem in multi unmanned aerial vehicles systems using swarm intelligence. *Eng Appl Artif Intell* 2018; 72: 10–20.

25. Jia Z, Yu J, Ai X, et al. Cooperative multiple task assignment problem with stochastic velocities and time windows for heterogeneous unmanned aerial vehicles using a genetic algorithm. *Aerosp Sci Technol* 2018; 76: 112–125.

26. Wu W, Wang X and Cui N. Fast and coupled solution for cooperative mission planning of multiple heterogeneous unmanned aerial vehicles. *Aerosp Sci Technol* 2018; 79: 131–144.

27. Lemaire T, Alami R and Lacroix S. A distributed tasks allocation scheme in multi-UAV context. In: IEEE international conference on robotics and automation, 2004. Proceedings. ICRA’04, New Orleans, LA, 26 April–1 May 2004, vol. 4, pp.3622–3627. New York: IEEE.

28. Deng Q, Yu J and Mei Y. Deadlock-free consecutive task assignment of multiple heterogeneous unmanned aerial vehicles. *J Aircr* 2014; 51: 596–605.

29. Wang J, Zhang Y, Geng L, et al. A heuristic mission planning algorithm for heterogeneous missions with heterogeneous UAVs. *Unmanned Syst* 2015; 3: 205–219.

30. Dai X, Yao Z and Zhao Y. A discrete adaptive auction-based algorithm for task assignments of multi-robot systems. *J Robot Mechatron* 2014; 26: 369–376.

31. Michalewicz Z and Fogel DB. *How to solve it: modern heuristics*. Springer Science & Business Media, 2013. Berlin: Springer Science & Business Media.

32. Guerrero JL, Berlanga A and Molina JM. Initialization procedures for multiobjective evolutionary approaches to the segmentation issue. In: Corchado E, Snášel V, Abraham A, et al. (eds) *International conference on hybrid artificial intelligence systems*. Berlin, Heidelberg: Springer, 2012, pp.452–463.

33. Pachauri A and Srivastava G. Automated test data generation for branch testing using genetic algorithm: an improved approach using branch ordering, memory and elitism. *J Syst Softw* 2013; 86: 1191–1208.

34. Song Y, Wang F and Chen X. An improved genetic algorithm for numerical function optimization. *Appl Intell* 2019; 49: 1880–1902.

35. Qiu M, Ming Z, Li J, et al. Phase-change memory optimization for green cloud with genetic algorithm. *IEEE Trans Comput* 2015; 64: 3528–3540.

36. Edison E and Shima T. Integrated task assignment and path optimization for cooperating uninhabited aerial vehicles using genetic algorithms. *Comput Oper Res* 2011; 38: 340–356.

37. Kim J, Song BD and Morrison JR. On the scheduling of systems of UAVs and fuel service stations for long-term mission fulfillment. *J Intell Robot Syst* 2013; 70: 347–359.

38. Jiang X, Zhou Q and Ye Y. Method of task assignment for UAV based on particle swarm optimization in logistics. In: *Proceedings of the 2017 international conference on intelligent systems, metaheuristics & swarm intelligence*, March 2017, pp.113–117. New York: ACM.

39. Balampanis F, Maza I and Ollero A. Coastal areas division and coverage with multiple UAVs for remote sensing. *Sensors* 2017; 17: 808.

40. Miao Y, Zhong L, Yin Y, et al. Research on dynamic task allocation for multiple unmanned aerial vehicles. *Trans Inst Meas Control* 2017; 39: 466–474.