drones

MDPI

Article

Adjustable Fully Adaptive Cross-Entropy Algorithms for Task Assignment of Multi-UAVs

Kehao Wang [1], Xun Zhang [1], Xuyang Qiao [1], Xiaobai Li [2,*], Wei Cheng [2], Yirui Cong [3] and Kezhong Liu [4]

[1] School of Information Engineering, Wuhan University of Technology, Wuhan 430070, China
[2] Department of Early Warning Intelligence, Air Force Early Warning Academy, Wuhan 430070, China
[3] College of Intelligence Science and Technology, National University of Defense Technology, Changsha 410073, China
[4] School of Navigation, Wuhan University of Technology, Wuhan 430070, China
* Correspondence: lxb2cici@163.com

Abstract: This paper investigates the multiple unmanned aerial vehicle (multi-UAV) cooperative task assignment problem. Specifically, we assign different types of UAVs to accomplish the classification, attack, and verification tasks of targets under resource, precedence, and timing constraints. Due to complex coupling among these tasks, we decompose the considered problem into two subproblems: one with continuous and independent tasks and another with continuous and correlative tasks. To solve them, we first present an adjustable, fully adaptive cross-entropy (AFACE) algorithm based on the cross-entropy (CE) method, which serves as a stepping stone for developing other algorithms. Secondly, to overcome task precedence in the first subproblem, we propose a mutually independent AFACE (MIAFACE) algorithm, which converges faster than the CE method when obtaining the optimal scheme vectors of these continuous and independent tasks. Thirdly, to deal with task coupling in the second subproblem, we present a mutually correlative AFACE (MCAFACE) algorithm to find the optimal scheme vectors of these continuous and correlative tasks, while its computational complexity is inferior to that of the MIAFACE algorithm. Finally, numerical simulations demonstrate that the proposed MIAFACE (MCAFACE, respectively) algorithm consumes less time than the existing algorithms for the continuous and independent (correlative, respectively) task assignment problem.

Keywords: multi-UAVs; task assignment; AFACE algorithm; MIAFACE algorithm; MCAFACE algorithm

Citation: Wang, K.; Zhang, X.; Qiao, X.; Li, X.; Cheng, W.; Cong, Y.; Liu, K. Adjustable Fully Adaptive Cross-Entropy Algorithms for Task Assignment of Multi-UAVs. *Drones* **2023**, *7*, 204. https://doi.org/10.3390/drones7030204

Academic Editors: Andrey V. Savkin and Oleg Yakimenko

Received: 13 February 2023
Revised: 5 March 2023
Accepted: 13 March 2023
Published: 16 March 2023

1. Introduction

Due to its rapid deployment and nearly unlimited mobility, an unmanned aerial vehicle (UAV) has great potential in both military and civilian applications, including modern warfare, disaster search and rescue, traffic control, celestial exploration, and a variety of other fields [1–4]. UAVs for these applications have limited capabilities and require sufficient resources to perform tasks autonomously. As a result, multi-UAVs can be regarded as a promising method by which to handle complex tasks. As more attention is focused on them, two problems in multi-UAV collaboration, such as multi-UAV cooperative path planning and cooperative task assignment, are becoming more widely recognized. The main consideration of this paper is the multi-UAV cooperative task assignment problem.

In recent years, many scholars have paid attention to the multi-UAV cooperative task assignment problem, while the related research of this problem is as follows. Chen et al. [5] utilized mixed integer linear programming (MILP) to address the problem of multi-UAV cooperative task assignment and path planning for moving targets on the ground, but it had low scalability while maintaining global optimality. References [6,7] used a heuristic approach to produce near-optimal results in real time, which has been widely considered for large-scale problems and dynamic scenarios. For swarm intelligence algorithms,

e.g., particle swarm optimization (PSO) [8], ant colony optimization (ACO) [9], and genetic algorithm (GA) [10], when solving the task assignment problem, they had a fast convergence speed and could effectively obtain optimal assignment schemes, but there is a possibility of falling into local optimum. Moreover, the auction algorithm, game theory, and reinforcement learning have also been applied to the multi-UAV task assignment problem. Duan et al. [11] presented a novel hybrid "two-stage" auction algorithm that combines the structural advantages of the centralized and distributed auction algorithms, which greatly facilitates the performance of UAVs in dynamic task assignments. Chen et al. [12] studied the cooperative reconnaissance and spectrum access (CRSA) problem for task-driven heterogeneous coalition-based UAV networks, and proposed a joint bandwidth allocation and coalition formation (JBACF) algorithm to solve the task assignment and bandwidth allocation. Qie et al. [13] proposed an artificial intelligence method called simultaneous target assignment and path planning (STAPP) to solve the multi-UAV target assignment and path planning problem, and the effectiveness of the algorithm was experimentally verified. In addition, references [14–21] provide a variety of alternative algorithms for the solution of analogous problems.

Similarly, some novel works on task assignment, e.g., UAV-assisted task assignment, have been presented. Liu et al. [22] studied a UAV-assisted IoT system while presenting a nonconvex age-of-information (AoI) minimization problem, which was solved by jointly optimizing task assignment, interaction point selection (IPT), and UAV trajectories. Zhu et al. [23] considered the problem of task loss rate (TLR) fairness among IoTs and equal energy consumption (EC) fairness among UAVs, and proposed a multiagent deep deterministic policy gradient (MA-DDPG) method by which to assign UAVs to accomplish tasks and guarantee the balance between IoT TLR and UAV EC. Seid et al. [24] considered the assignment of UAVs to perform aerial base station tasks based on a multi-UAV-assisted IoT network framework, while presenting a joint optimization problem for computational offloading with energy harvesting (EH) and resource price, and the resource demands and pricing strategies between IoT devices and UAVs were continuously adjusted by the Stackelberg game. Hu et al. [25] considered the aging of cache refreshing, computation offloading, and state updates in UAV-assisted vehicle task awareness, and formulated a task-assignment energy-minimization problem that was solved by a deep deterministic policy gradient (DDPG) method. Zhou et al. [26] studied UAV-assisted mobile crowd sensing (MCS) scenarios and proposed a UAV-assisted multitasking assignment (UMA) method, while demonstrating the effectiveness of UMA. In addition, compared the UAV-assisted task assignment with the UAV task assignment, the difference is that UAVs play a secondary role in the former while serving as the primary reconnaissance and attack objects in the latter. Furthermore, the simulation scenarios in the paper are not consistent with the existing works (e.g., references [22–26]).

In the complex stochastic network, the cross-entropy (CE) method [27], a relatively new technique for dealing with combinatorial optimization problems, was initially utilized to estimate rare event probabilities. Then, references [28,29] discussed and analysed its convergence. Additionally, the cross-entropy (CE) method was proved by the authors in [30] to be particularly meaningful for handling combinatorial optimization problems. Since then, it has also been proven by many scholars to be a simple and effective tool for different fields, e.g., vehicle routing [31], buffer allocation [31], and machine learning [32]. In addition, researchers have also considered applying the cross-entropy (CE) method to the UAV task assignment [33–35]. However, the authors of these papers did not consider the specific precedence and timing constraints among these tasks.

When it comes to task-assignment schemes in the field of UAVs, some researchers usually assume that each UAV is assigned to only one target, and they rarely consider the execution sequence and the time constraints among tasks. On the other hand, multi-UAVs are sometimes needed to perform some complex combinatorial tasks, such as classifying the target, attacking it, and then verifying the target's damage level in a reasonable time on the battlefield. In addition, such deterministic approaches may not be able to find the

optimal solution in a reasonable time for large-scale task assignment problems. Under these circumstances, we present an adjustable fully adaptive cross-entropy (AFACE) algorithm based on CE method.

Therefore, the purpose of this paper is to study the AFACE algorithm for the multi-UAV cooperative task assignment problem under resource, precedence, and timing constraints. The main contributions are summed up as follows.

- We consider the multi-UAV cooperative task assignment problem in which different types of UAVs are assigned to perform classification, attack and verification tasks of targets under resource, and precedence and timing constraints. Considering complex coupling among these tasks, we decompose the considered problem into two subproblems: one with continuous and independent tasks and another with continuous and correlative tasks.
- We propose an AFACE algorithm, which changes the random sample and the quantile at each iteration and adds a parameter to adjust the maximum sample based on the CE method. Meanwhile, the algorithm serves as a stepping stone for developing other algorithms.
- To overcome task precedence and task coupling existing in these two problems, respectively, we present a mutually independent AFACE (MIAFACE) algorithm and a mutually correlative AFACE (MCAFACE) algorithm with polynomial time complexity. The former algorithm converges faster than the CE method, while the computational complexity of the latter algorithm is inferior to that of the former algorithm.
- Simulation results demonstrate that both MIAFACE and MCAFACE algorithms consume less time than other existing optimization algorithms for solving the corresponding problem.

The rest of this paper is organized as follows. In Section 2, we introduce the related works of the CE method and other algorithms for the UAV task assignment. Section 3 depicts the multi-UAV cooperative task assignment problem with its mathematical formulation. In Section 4, we decompose the considered problem into two subproblems, and propose an AFACE algorithm, a MIAFACE algorithm, and a MCAFACE algorithm, and apply the latter two algorithms to solving the corresponding problem. Section 5 conducts several simulations and comparisons to verify the feasibility and effectiveness of the proposed algorithms. This paper is concluded in Section 6.

2. Related Work

This section reviews the related works on CE method and other algorithms used for UAV task assignment.

2.1. CE Method Used for UAV Task Assignment

Due to CE's merits, the authors of [33] first proposed using the CE method for tackling the multi-UAV task assignment problem to tackle the large traveling salesman problem (TSP), the vehicle routing problem (VRP), and Markov decision process (MDP). In particular, compared to other algorithms, CE could solve optimization problems efficiently because of its ability to deal with these problems with nonlinear objective functions. Three separate multi-UAV task assignment problems were then formulated, including a nonlinear objective function with distance penalty, a nonlinear objective function with no distance penalty, and nonlinear constraints. In these problems, the authors considered the distance penalty and required that each task must be assigned to at least one vehicle. Then, the task scores were considered as nonlinear functions, and the CE method was used to determine the optimal schemes for the functions of these problems. Finally, simulation results verified that the performance of the CE method was superior to other algorithms.

The authors in [34] considered the multi-UAV task assignment problem. Then, the score function of this problem was determined with the constraint that each UAV was used for only one task. Subsequently, the CE method was used to find the optimal scheme of this problem. Finally, simulation results showed that the CE method outper-

formed the Branch and Bound algorithm in solving the above problem, especially on a large scale.

Referring to [33,34], the authors in [35] described the multitype UAV task assignment problem. In this problem, different types of UAVs, or the same type of UAV as well as resource constraints, were considered. The authors then formulated the problem and provided a score function under resource constraints. Then, the CE method was used to determine the optimal scheme of this problem by assigning multitype UAVs to complete tasks. Finally, numerical simulations of the CE method for task assignment, as well as comparisons with the exhaust search method, were conducted to verify its merits in solving the considered problem.

In [36], the authors first analyzed the CE method, then redefined its construct and applied it to UAV swarms. Subsequently, due to the robustness of this method, it could be used as an effective measure to control UAV swarms in the face of obstacles and unforeseen problems. Finally, it was validated to support UAV swarms in achieving mission objectives.

The authors of [37] considered the multi-UAV task assignment problem under resource constraint and precedence constraint. The fully adaptive cross-entropy (FACE) algorithm based on the CE method was then applied to solve the considered problem. Then, simulation results verified that the FACE algorithm was better than the CE method and PSO algorithm in terms of convergence speed.

2.2. Other Algorithms Used for UAV Task Assignment

The authors in [8] improved the PSO algorithm with an inertia weight factor and applied it to handle the multi-UAV task assignment problem, then conducted several simulations and comparisons. Then, it was verified that the improved algorithm has a faster convergence speed and global optimization capability compared with the standard PSO algorithm.

In [38], the authors presented a novel hierarchical task assignment method to solve the multi-UAV task assignment problem, and the method was decomposed into two phases, including the hierarchical decomposition phase and the task assignment phase. The former phase reduced the computational complexity by using the balance cluster method to simplify the large-scale UAV model; the latter phase maintained the diversity of the population by an improved firefly algorithm. Then, simulations showed that compared with other algorithms, the proposed hierarchical method becomes more efficient in terms of search ability and convergence speed.

The authors in [39] defined the task assignment problem for cooperative multi-UAV road network reconnaissance and formulated a multi-UAV road network reconnaissance traveling salesman problem (MRRTSP) model. Furthermore, a customized genetic algorithm for road network reconnaissance (CGA-RNR) was proposed and used to solve the considered problem. Then, simulations showed that the algorithm can quickly obtain feasible solutions and converge to the optimal solution.

3. Problem Description and Formulation

The main parameters of this paper is shown in Table 1.

Problem Description

On the battlefield, multi-UAVs are deployed to perform different tasks, for example, to classify targets before attacking them, and then to verify them to check whether these tasks have been accomplished. The problem considered in this paper is the selection of a mix of the same type of UAV or different types of UAVs from their bases to perform the classification, attack and verification tasks of targets. As shown in Figure 1, there are N_b types of UAVs with the same speed, and the related components of this problem can be defined as a 5-tuple $\{\mathcal{A}, \mathcal{B}, \mathcal{G}, \mathcal{K}, \mathcal{T}\}$. In the 5-tuple, $\mathcal{A} := \{1, 2, \ldots, N_m\}$ denotes the set of task index of targets, $\mathcal{B} := \{1, 2, \ldots, N_b\}$ represents the set of N_b bases, $\mathcal{G} := \{1, 2, \ldots, N_t\}$ denotes the set of N_t targets with known positions, $\mathcal{K} := \{K_1, K_2, \ldots, K_{N_m}\}$ represents the

set of N_{m} tasks of targets, and $\mathcal{T} := \{T_1, T_2, \ldots, T_{N_{\mathrm{m}}}\}$ denotes the set of the execution time of N_{m} tasks of targets. Note that the time required to allocate tasks is ignored.

Table 1. Simulation parameter settings.

Variables	Explanation
N_{b}	The number of bases
N_{t}	The number of targets
j	The target index
$\mathcal{K}$	The set of tasks of targets
N_{m}	The number of tasks of targets
m	The task index of targets
$\mathcal{X}$	The set of all possible UAV deployment schemes
L	The number of $\mathcal{X}$ for each task
z	The maximum number of UAVs in each scheme of $\mathcal{X}$
$\mathcal{Z}$	The set of all possible UAV deployment scheme indexes
k	A UAV deployment scheme index or a UAV formation index
$\boldsymbol{x}(m)$	A feasible UAV deployment scheme vector of task m
$x_j(m)$	A feasible UAV deployment scheme or a UAV formation of task m of target j
$g(x_j(m);k)$	A 0–1 decision variable
$\mathcal{Y}$	The set of all feasible $\boldsymbol{x}(m)$
$\Omega(x(m))$	The performance of task m
Ω	The performance vector of $\Omega(x(m))$
ρ	The total objective function
$\psi(x_j(2))$	The reward benefit of the attack task of target j
$\varphi(x_j(m))$	The cost of assigning UAV formation $x_j(m)$ to accomplish task m of target j
p_{k}^j	The probability of killing target j
p_{s}^j	The UAV survival probability of accomplishing tasks of target j
w_1, w_2 and w_3	Weight coefficients
P_{c} and V	The target identification certainty and the constant velocity of each UAV
y_j and s_j	The value and the threat level of target j
$d_j(m)$	The farthest distance from the base corresponding to UAV formation $x_j(m)$ to target j
$D_{\max}$	The maximum flying distance
T_m	The execution time of task m

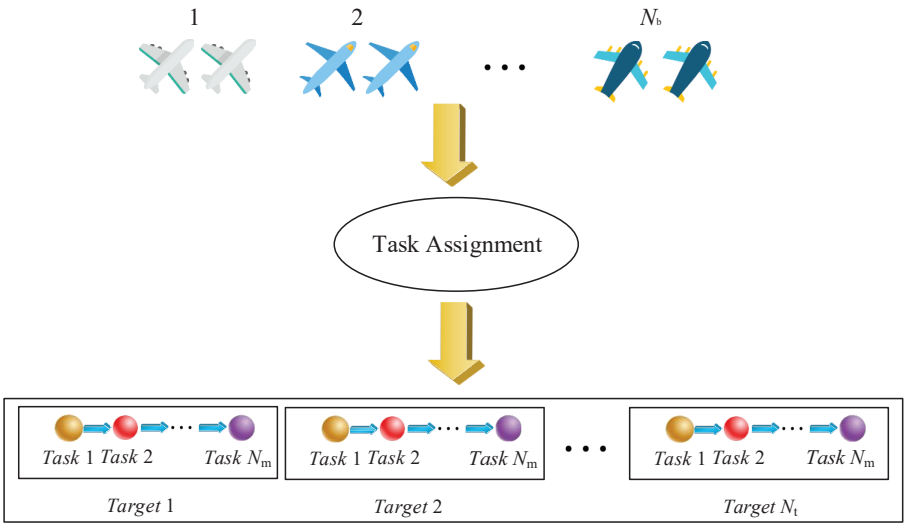


Figure 1. Task assignment diagram.

Moreover, let $\boldsymbol{x}(m) = [x_1(m), x_2(m), \ldots, x_{N_t}(m)]^T$ denote a feasible UAV deployment scheme vector, and define $\mathcal{Y}$ as the set of all feasible $\boldsymbol{x}(m)$. Let $\mathcal{Z} := \{1, 2, \ldots, L\}$ be the set of all possible UAV deployment scheme indices. Thus, $\boldsymbol{x}(m)$ satisfies

$$g(x_j(m); k) = \begin{cases} 1, if\ \phi(x_j(m)) = k, j \in \mathcal{G}, m \in \mathcal{A}, k \in \mathcal{Z} \\ 0, if\ \phi(x_j(m)) \neq k, x_j(m) \in \mathcal{X} \end{cases}, \tag{1}$$

where j denotes the target index, m designates the task index, $x_j(m)$ is a feasible UAV deployment scheme or a UAV formation of task m of target j, k represents a UAV deployment scheme index or a UAV formation index, $\phi(x_j(m))$ is an index function that serves to output the subscript corresponding to $x_j(m)$ in $\mathcal{X}$, and $g(x_j(m); k)$ is a 0–1 decision variable, i.e., the kth UAV formation is assigned to accomplish task m of target j.

Then, the total objective function ρ based on $\boldsymbol{x}(m)$ is defined as

$$\begin{aligned} \rho &= \sum_{m=1}^{N_m} \Omega(x(m)) \\ &= \sum_{j=1}^{N_t} \psi(x_j(2)) - \sum_{m=1}^{N_m} \sum_{j=1}^{N_t} \varphi(x_j(m)), \end{aligned} \tag{2}$$

where $\Omega(x(m))$ is the subobjective function of task m, $\psi(x_j(2))$ and $\varphi(x_j(m))$ denote the reward benefit of the attack task and the cost of assigning $x_j(m)$ to accomplish tasks of target j, respectively, which are

$$\psi(x_j(2)) = w_1 \times P_c \times p_k^j \times y_j \tag{3}$$

$$\varphi(x_j(m)) = w_2 \times p_s^j \times s_j + w_3 \times (V \times T_m + d_j(m)), \tag{4}$$

where P_c is the target identification certainty, y_j represents the value of target j, s_j denotes the threat level of target j, V is the constant velocity of each UAV, T_m represents the execution time of task m, $d_j(m)$ denotes the farthest distance from the bases corresponding to UAV formation $x_j(m)$ to target j, w_1, w_2 and w_3 represent weight coefficients, indicating the information about the relative importance of each subobjective, p_k^j denotes the probability of killing target j, and p_s^j is the UAV survival probability of accomplishing task m of target j.

In addition, p_k^j and p_s^j are defined as

$$p_k^j = \prod_{a \in x_j(2)} p_{aj} \tag{5}$$

$$p_s^j = 1 - \prod_{b \in x_j(m)} p_{bj}, \tag{6}$$

where a stands for a UAV in UAV formation $x_j(2)$; p_{aj} is the probability of killing target j with UAV a, b represents a UAV in UAV formation $x_j(m)$, and p_{bj} is the UAV survival probability of accomplishing task m of target j with UAV b.

According to Equations (2)–(6), ρ is rewritten as

$$\begin{aligned} \rho = \sum_{j=1}^{N_t} w_1 \times P_c \times p_k^j \times y_j - \sum_{m=1}^{N_m} \sum_{j=1}^{N_t} [w_2 \times p_s^j \times s_j \\ + w_3 \times (V \times T_m + d_j(m))]. \end{aligned} \tag{7}$$

Then, our objective is to maximize ρ, and the considered problem can be formulated as

$$\mathcal{P}: \max_{\boldsymbol{x}(m)\in\mathcal{Y}} \rho = \sum_{m=1}^{N_m} \Omega(x(m)) \tag{8}$$

$$\text{s.t. } w_1 + w_2 + w_3 = 1, 0 \le w_1, w_2, w_3 \le 1 \tag{9}$$

$$d_j(m) + V \times T_m \le D_{\max} \quad \forall j, m \tag{10}$$

$$K_1^j \prec K_2^j \prec K_3^j \quad \forall j. \tag{11}$$

Constraint (9) represents the range of w_1, w_2, and w_3. Constraint (10) is that, for target j, the sum of $d_j(m)$ and the farthest flying distance performed by the UAV formation $x_j(m)$ does not exceed the maximum flying distance $D_{\max}$. Constraint (11) means that K_1^j, K_2^j, and K_3^j are the classification, attack, and verification tasks of the target j, which are executed in a specific order, and $\prec$ denotes the preceding symbol.

According to Equation (11), the specific precedence and timing constraints are equal to

$$\begin{cases} t_{s1}^j \ge s_1, e_1 \ge t_{s1}^j + T_1 \\ t_{s2}^j \ge s_2, e_2 \ge t_{s2}^j + T_2 \\ t_{s3}^j \ge s_3, e_3 \ge t_{s3}^j + T_3 \end{cases}, \tag{12}$$

where $[s_1, e_1]$, $[s_2, e_2]$, and $[s_3, e_3]$ represent the classification, attack, and verification time windows and t_{s1}^j, t_{s2}^j and t_{s3}^j denote the start time of classification, attack, and verification tasks of the target j, respectively.

Moreover, we set a certain value γ, which ensures that the optimal scheme vector $\boldsymbol{x}^*(m)$ conforms to $\Omega(x^*(m)) \ge \gamma$. After that, the maximum ρ^* is written as

$$\rho^* = \sum_{m=1}^{N_m} \Omega(x^*(m)) \ge N_m \gamma. \tag{13}$$

Therefore, to obtain $\boldsymbol{x}^*(m)$, we present an AFACE algorithm.

4. Algorithm Analysis

In this section, an AFACE algorithm will be introduced for the considered problem, and the differences between the algorithm and cross-entropy (CE) method are that the former changes the random sample N_d^t and the quantile θ_t at each iteration t, and adds a parameter to adjust the maximum sample $N^{\max}$. For details, please refer to the analysis of the algorithm below.

4.1. Adjustable Fully Adaptive Cross-Entropy Algorithm

Referring to the principle of CE method in references [30,35] and maximizing the subobjective function $\Omega(x(m))$ of the considered problem, we have

$$\gamma^* = \Omega(x^*(m)) = \max_{x(m)\in\mathcal{Y}} \Omega(x(m)), \tag{14}$$

where γ^* is the maximum of $\Omega(x(m))$ on $\mathcal{Y}$; that is, the optimal scheme vector is $\boldsymbol{x}^*(m)$.

After that, transform this problem into a probability estimator problem, which can be explained by the probability density function (PDF) $f(\cdot; u)$ with respect to u, and the problem can be written as

$$\begin{aligned} \ell(\gamma) &= \mathbb{P}_u(\Omega(x(m)) \ge \gamma) \\ &= \sum_{\boldsymbol{x}(m)} I_{\{\Omega(x(m))\ge\gamma\}} f(\boldsymbol{x}(m); u) \\ &= \mathbb{E}_u I_{\{\Omega(x(m))\ge\gamma\}}, \end{aligned} \tag{15}$$

where γ denotes a value close to γ^*, $\mathbb{P}_u$ represents the probability measure under which the random vector $\boldsymbol{x}(m)$ has the PDF $f(\cdot;u)$, $\mathbb{E}_u$ is the corresponding expectation operator, and $I(\boldsymbol{x}(m);\gamma)$, i.e., $I_{\{\Omega(x(m))\geq\gamma\}}$, denotes the indicator function, which is

$$I(\cdot;\gamma)=\begin{cases}1, & if\ \Omega(x(m))\geq\gamma\\ 0, & if\ \Omega(x(m))<\gamma\end{cases}. \tag{16}$$

Then, at the tth iteration of AFACE algorithm, we obtain

$$\Omega_{t,1}\leq\cdots\Omega_{t,i}\leq\cdots\leq\Omega_{t,N_{\mathrm{d}}^t}, \tag{17}$$

where $\Omega_{t,i}$ $(i=1,2,\ldots,N_{\mathrm{d}}^t)$ denotes the ith sample performance, and $\Omega(x_i(m)))$ and $\Omega_{t,N_{\mathrm{d}}^t}$ are defined by $\Omega_{t,i}$ and Ω_t^* for convenience. Meanwhile, AFACE algorithm parameters N_{d}^t and θ_t satisfy

$$\begin{cases}N^{\min}\leq N_{\mathrm{d}}^t\leq N^{\max}\\ \theta_t=\beta_m/N_{\mathrm{d}}^t\end{cases}, \tag{18}$$

where N_{d}^t denotes the random sample of the tth iteration, varying between $N^{\min}$ and $N^{\max}$ ($N^{\min}=N, N^{\max}=hN, h\in\{2,3,4,5\}$) and θ_t represents the quantile of the tth iteration. The reason for presenting h is that by adjusting the size of $N^{\max}$, we can obtain the optimal $N^{\max}$ that matches the combat scenario, which can be conducted by the following simulations in Section 5.

For the AFACE algorithm, the main idea is to update N_{d}^t and θ_t based on the elite sample β_m ($\beta_m=c_mN$), where c_m and N are the elite sample influence coefficient of task m (usually $0.01\leq c_m\leq 0.1$) and the fixed random sample, respectively. Therefore, the set of elite samples ε_t ($\varepsilon_t\in\mathcal{Y}$) are comprised of such β_m samples in $\{\boldsymbol{x}_1(m),\boldsymbol{x}_2(m),\ldots,\boldsymbol{x}_{N_{\mathrm{d}}^t}(m)\}$ with the highest performances $\Omega_{t,1},\Omega_{t,2},\ldots,\Omega_{t,N_{\mathrm{d}}^t}$.

Next, referring to the formulas for solving $\widehat{\gamma}_t$ and $\widehat{v}_t$ of CE method [30], they are modified as

$$\widehat{\gamma}_t=\Omega_{(\lceil(1-\theta_t)N_{\mathrm{d}}^t\rceil)} \tag{19}$$

$$\widehat{v}_t=\arg\max_{v}\sum_{\boldsymbol{x}_i(m)\in\varepsilon_t}\ln f(\boldsymbol{x}_i(m);v), \tag{20}$$

where $\boldsymbol{x}_i(m)$ is generated from $f(\cdot;u)$, $f(\cdot;v)$ denotes another PDF with respect to v on $\mathcal{Y}$ via minimizing the Kullback–Leibler distance, $\widehat{\gamma}_t$ is equal to the worst sample performance among the elite performances, while Ω_t^* is the best sample performance among the elite performances, and $\widehat{v}_t$ converges to the probability density when Ω_t^* occurs.

Then, we devise a sampling scheme for each iteration t, ensuring high probability that

$$\begin{cases}\Omega_t^*>\Omega_{t-1}^*\\ \widehat{\gamma}_t>\widehat{\gamma}_{t-1}\end{cases}. \tag{21}$$

Moreover, we simultaneously generate two sequences to validate the correctness of AFACE algorithm. One is the levels $\widehat{\gamma}_1,\widehat{\gamma}_2,\ldots,\widehat{\gamma}_t$, and the other is the parameters $\widehat{v}_1,\widehat{v}_2,\ldots,\widehat{v}_t$. After that, the initialization process is set to $\widehat{v}_0=u$, and the quantile $(1-\theta_t)$ is calculated at the tth iteration according to Equation (18), followed by the next two steps of Algorithm 1.

In addition, the main steps of AFACE algorithm applied to solving the subobjective function $\Omega(x(m))$ of the considered problem are given by Algorithm 2.

Algorithm 1 Adaptive updating of $\widehat{\gamma}_t$ and $\widehat{v}_t$.

Adaptive updating of $\widehat{\gamma}_t$:
1: Given a fixed $\widehat{v}_{t-1}$ at the tth iteration;
2: Let γ_t be a $(1-\theta_t)$-quantile of $\Omega(x(m))$ under $\widehat{v}_{t-1}$, then γ_t satisfies $\mathbb{P}_{\widehat{v}_{t-1}}(\Omega(x(m)) \leq \gamma_t) \geq 1-\theta_t$, where $\boldsymbol{x}(m) \sim f(\cdot;\widehat{v}_{t-1})$;
3: Obtain a simple estimator $\widehat{\gamma}_t$ of γ_t by drawing N_{d}^t random samples $\boldsymbol{x}_1(m), \boldsymbol{x}_2(m), \ldots, \boldsymbol{x}_{N_{\mathrm{d}}^t}(m)$ from $f(\cdot;\widehat{v}_{t-1})$;
4: Calculate and order all performances of $\Omega(x(m))$ from smallest to biggest: $\Omega_{t,1} \leq \cdots \leq \Omega_{t,N_{\mathrm{d}}^t}$;
5: Compute $\widehat{\gamma}_t$ according to Equation (19);
Adaptive updating of $\widehat{v}_t$:
6: Given a fixed $\widehat{\gamma}_t$ and $\widehat{v}_{t-1}$ at the tth iteration, then derive $\widehat{v}_t$ according to Equation (20).

Algorithm 2 AFACE algorithm.

Input: $\widehat{v}_0$, h, N.
Output: Ω_t^*.
1: Set $t=1$, $N^{\min}=N$ and $N^{\max}=hN$;
2: **while** at the tth iteration ($t \geq 1$) **do**
3: **if** $t=1$ **then**
4: Generate N_{d}^t ($N_{\mathrm{d}}^t = N^{\min}$) random samples $\boldsymbol{x}_1(m), \boldsymbol{x}_2(m), \ldots, \boldsymbol{x}_{N_{\mathrm{d}}^t}(m)$ from $f(\cdot;\widehat{v}_0)$;
5: Calculate $\widehat{\gamma}_t$ and $\widehat{v}_t$ according to Equations (19) and (20);
6: **else**
7: Draw N_{d}^t ($N^{\min} \leq N_{\mathrm{d}}^t \leq N^{\max}$) random samples $\boldsymbol{x}_1(m), \boldsymbol{x}_2(m), \ldots, \boldsymbol{x}_{N_{\mathrm{d}}^t}(m)$ from $f(\cdot;\widehat{v}_{t-1})$;
8: **end if**
9: Update $\widehat{\gamma}_t$ and $\widehat{v}_t$ according to Algorithm 1, then calculate Ω_t^*;
10: **if** Equation (21) occurs **then**
11: Set $t=t+1$ and go to step 2;
12: **else**
13: Check whether or not $\Omega_t^* = \cdots = \Omega_{t-d}^*$ for some $t \geq d$, e.g., $d=5$;
14: **if** $\Omega_t^* = \cdots = \Omega_{t-d}^*$ **then**
15: Stop, obtain Ω_t^* and **return** Ω_t^*;
16: **else**
17: Set $t=t+1$, take random integer N_{d}^t in $[N^{\min}, N^{\max}]$ and go to step 2;
18: **end if**
19: **end if**
20: **end while**

4.2. Adjustable Fully Adaptive Cross-Entropy Algorithm for Solving Problem

Considering complex coupling among the three tasks, we decompose the considered problem $\mathcal{P}$ into two subproblems: the problem $\mathcal{P}1$ with continuous and independent tasks and the problem $\mathcal{P}2$ with continuous and correlative tasks.

Before discussing the algorithm for solving problem $\mathcal{P}$, we have to determine the number of the available schemes for each task. Please refer to Theorem 1 for the specific derivation process.

Theorem 1. *Assume that $z \geq 1$ and $N_{\mathrm{b}} = 3$, the number of the available schemes for each task of targets is L. Then, according to the mathematical formulas of permutation and combination, we can obtain*

$$L = 3z + \frac{z(z-1)(z+7)}{6}, z \geq 1. \tag{22}$$

Proof. Please see Appendix A. □

4.2.1. Mutually Independent AFACE Algorithm for Solving Problem $P1$

In problem $\mathcal{P}1$, assume that there are N_{m} continuous and mutually independent tasks for each target. Time continuity among these tasks then needs to be considered. Assume that there are L available schemes for each task, i.e., the scheme chosen by the previous

task has no effect on the choice of the scheme for the next task, indicating that the available schemes among these tasks are independent. Thus, the problem $\mathcal{P}1$ is rewritten as

$$\begin{aligned} \mathcal{P}1: \max_{\boldsymbol{x}(m)\in\mathcal{Y}} \ & \rho = \sum_{m=1}^{N_{\mathrm{m}}} \Omega(x(m)) \\ \text{s.t.} \ & (9)-(12) \\ & l_m^1 = L \quad \forall m \end{aligned}, \tag{23}$$

where l_m^1 is the available schemes when performing the mth task.

Considering time sequence and independence of the available schemes among these tasks, we present a MIAFACE algorithm, which is a combination of N_{m} AFACE algorithms. For MIAFACE algorithm, we first introduce the probability matrix vector $\boldsymbol{P} = [\boldsymbol{P}(1), \boldsymbol{P}(2), \ldots, \boldsymbol{P}(N_{\mathrm{m}})]^T$ and the performance vector $\Omega = [\Omega(x(1)), \Omega(x(2)), \ldots, \Omega(x(N_{\mathrm{m}}))]^T$, where $\boldsymbol{P}(m)$ and $\Omega(x(m))$ are the probability matrix and the performance of task m, respectively. Then, $\boldsymbol{P}(m)$ is defined as

$$\boldsymbol{P}(m) = \begin{pmatrix} p(1|1,m) & p(2|1,m) & \cdots & p(l_m^1|1,m) \\ p(1|2,m) & p(2|2,m) & \cdots & p(l_m^1|2,m) \\ \vdots & \vdots & \ddots & \vdots \\ p(1|N_{\mathrm{t}},m) & p(2|N_{\mathrm{t}},m) & \cdots & p(l_m^1|N_{\mathrm{t}},m) \end{pmatrix}_{N_{\mathrm{t}}\times l_m^1},$$

where $p(k|j,m)$ represents the probability of assigning the kth UAV formation to accomplish task m of target j and $\boldsymbol{P}(m)$ is subjected to $\sum_{k=1}^{l_m^1} p(k|j,m) = 1$.

Then, for the mth task, we initialize $\boldsymbol{P}_0(m) = (p_0(k|j,m))_{N_{\mathrm{t}}\times l_m^1}$ with a uniform distribution. Let n_{jm}^1 be the number of the feasible schemes of target j, and define $p_0(k|j,m) := \frac{1}{n_{jm}^1}$ as the element of $\boldsymbol{P}_0(m)$. After that, we set $\widehat{\boldsymbol{v}}_0 = \boldsymbol{P}_0(m)$.

At the tth iteration, we assume that the samples $\boldsymbol{x}_1(m), \boldsymbol{x}_2(m), \ldots, \boldsymbol{x}_{N_{\mathrm{d1}}^t}(m)$ are drawn from $f(\boldsymbol{x}(m); \widehat{\boldsymbol{v}}_{t-1}(m))$. In addition, we calculate the performances $\Omega_{t,i}$ $(i = 1, 2, \ldots, N_{\mathrm{d1}}^t)$, and order them from smallest to largest: $\Omega_{t,1} \le \Omega_{t,2} \le \cdots \le \Omega_{t,N_{\mathrm{d1}}^t}$. It is noted that β_m^1 is calculated by $\beta_m^1 = c_m^1 N$, and $\widehat{\gamma}_t$ is updated by Equation (20). After that, we compare $\Omega_{t,i}$ with $\widehat{\gamma}_t$, and obtain all eligible performances greater than $\widehat{\gamma}_t$ and merge them into a set $\mathcal{S}_1 := \{\Omega_{(t,\lceil(1-\theta_t)N_{\mathrm{d1}}^t\rceil)}, \Omega_{(t,\lceil(1-\theta_t)N_{\mathrm{d1}}^t\rceil+1)}, \ldots, \Omega_{t,N_{\mathrm{d1}}^t}\}$, where β_m^1 is the number of the element of $\mathcal{S}_1$, and Ω_t^* is the maximum element of $\mathcal{S}_1$. Then, $p_t(k|j,m)$ is calculated, and the specific derivation process can be seen in Theorem 2. Thus, $\boldsymbol{P}_t(m)$ is the probability matrix composed of $p_t(k|j,m)$, and $\widehat{\boldsymbol{v}}_t$ is equal to $\boldsymbol{P}_t(m)$.

Theorem 2. *Assume that there are N_{m} continuous and mutually independent tasks for each target. After that, N_{m} tasks correspond to N_{m} AFACE algorithms, which has an elite sample of $\beta_m^1 = c_m^1 N$. In the MIAFACE algorithm, $\boldsymbol{c}_1$ is a combined vector of c_m^1, e.g., $\boldsymbol{c}_1 = [c_1^1, c_2^1, \ldots, c_{N_{\mathrm{m}}}^1]^T$. Thus, when performing the mth task, we can then obtain the updating formula of $\boldsymbol{P}(m)$ as follows:*

$$\begin{cases} p(k|j,m) = \dfrac{\sum_{n=1}^{c_m^1 N} g(x_j^n(m);k)}{c_m^1 N} \\ k \in \{1,\ldots,L\}, n \in \{1,\ldots,c_m^1 N\}, c_m^1 \in \boldsymbol{c}_1 \end{cases}. \tag{24}$$

Proof. Please see Appendix B. □

Through the iterative updating of $\boldsymbol{P}(m)$, the optimal probability matrix vector $\boldsymbol{P}^*$ and the maximum performance vector Ω^* are obtained. Then, the main steps of the MIAFACE algorithm applied to solving problem $\mathcal{P}1$ are described in Algorithm 3, and the convergence of the MIAFACE algorithm is similar to that of the CE method in [40].

Algorithm 3 MIAFACE algorithm.

Input: N_{m}, l_m^1, N, h.
Output: $\boldsymbol{P}^*$, Ω^*.

Set $N^{\min} = N$ and $N^{\max} = hN$;
for $m = 1; m < N_{\mathrm{m}}; m++$ **do**
  Initialize $\boldsymbol{P}_0(m)$ with a uniform distribution and define $\widehat{\boldsymbol{v}}_0 = \boldsymbol{P}_0(m)$, then set $t = 1$;
  while at the tth iteration ($t \geq 1$) **do**
    if $t = 1$ **then**
      Generate N_{d1}^t ($N_{\mathrm{d1}}^t = N^{\min}$) random samples $\boldsymbol{x}_1(m), \boldsymbol{x}_2(m), \ldots, \boldsymbol{x}_{N_{\mathrm{d1}}^t}(m)$ from $f(\cdot; \widehat{\boldsymbol{v}}_0)$;
    else
      Draw N_{d1}^t ($N^{\min} \leq N_{\mathrm{d1}}^t \leq N^{\max}$) random samples $\boldsymbol{x}_1(m), \boldsymbol{x}_2(m), \ldots, \boldsymbol{x}_{N_{\mathrm{d1}}^t}(m)$ from $f(\cdot; \widehat{\boldsymbol{v}}_{t-1})$;
    end if
    Update $\widehat{\gamma}_t$ according to Equation (19) and calculate Ω_t^*;
    Calculate $p_t(k|j,m)$ by Equation (A14) in Appendix B;
    if $\sum_{k=1}^{l_m^1} p_t(k|j,m) = 1$ and $p_t(k|j,m) \in \{0,1\}$ **then**
      Stop, obtain $\boldsymbol{P}_t^*(m)$ and Ω_t^*, then $\boldsymbol{P}^*(m) \leftarrow \boldsymbol{P}_t^*(m)$ and $\Omega(x^*(m)) \leftarrow \Omega_t^*$;
    else
      Calculate $\boldsymbol{P}_t(m)$ and update $\widehat{\boldsymbol{v}}_t$ by $\widehat{\boldsymbol{v}}_t = \boldsymbol{P}_t(m)$;
      Set $t = t + 1$, take random integer N_{d1}^t in $[N^{\min}, N^{\max}]$, then go to step 4;
    end if
  end while
end for
Return $\boldsymbol{P}^*$ and Ω^*.

4.2.2. Mutually Correlative AFACE Algorithm for Solving Problem $P2$

In problem $\mathcal{P}2$, assume that there are N_{m} continuous and mutually correlative tasks for each target. Then, time continuity among these tasks also needs to be considered. Assume that when performing the mth task, there are only $L - m + 1$ available schemes since $m - 1$ schemes have been deleted before performing the mth task. It means that the available schemes among these tasks are correlative. Thus, the problem $\mathcal{P}2$ is rewritten as

$$\begin{aligned} \mathcal{P}2: \max_{\boldsymbol{x}(m) \in \mathcal{Y}} \rho &= \sum_{m=1}^{N_{\mathrm{m}}} \Omega(x(m)) \\ \text{s.t. } & (9) - (12) \\ & l_m^2 = L - m + 1 \quad \forall m \end{aligned}, \tag{25}$$

where l_m^2 is the remaining schemes when performing the mth task.

Considering time sequence and relevance of the available schemes among these tasks, we present a MCAFACE algorithm, which is also combined by N_{m} AFACE algorithms. For the MCAFACE algorithm, we first introduce the probability matrix vector $\boldsymbol{Q} = [\boldsymbol{Q}(1), \boldsymbol{Q}(2), \ldots, \boldsymbol{Q}(N_{\mathrm{m}})]^T$ and the performance vector $\Omega = [\Omega(x(1)), \Omega(x(2)), \ldots, \Omega(x(N_{\mathrm{m}}))]^T$, where $\boldsymbol{Q}(m)$ and $\Omega(x(m))$ are the probability matrix and the performance of task m, respectively. Then, $\boldsymbol{Q}(m)$ is defined as

$$\boldsymbol{Q}(m) = \begin{pmatrix} q(1|1,m) & q(2|1,m) & \cdots & q(l_m^2|1,m) \\ q(1|2,m) & q(2|2,m) & \cdots & q(l_m^2|2,m) \\ \vdots & \vdots & \ddots & \vdots \\ q(1|N_{\mathrm{t}},m) & q(2|N_{\mathrm{t}},m) & \cdots & q(l_m^2|N_{\mathrm{t}},m) \end{pmatrix}_{N_{\mathrm{t}} \times l_m^2},$$

where $q(k|j,m)$ represents the probability of assigning the kth UAV formation to accomplish task m of target j and $\boldsymbol{Q}(m)$ is subjected to $\sum_{k=1}^{l_m^2} q(k|j,m) = 1$.

Then, for the mth task, we initialize $\boldsymbol{Q}_0(m) = (q_0(k|j,m))_{N_t \times l_m^2}$ with a uniform distribution. Let n_{jm}^2 be the number of the feasible schemes of target j and define $q_0(k|j,m) := \frac{1}{n_{jm}^2}$ as the element of $\boldsymbol{Q}_0(m)$. After that, we set $\widehat{v}_0 = \boldsymbol{Q}_0(m)$.

At the tth iteration, we assume that the samples $\boldsymbol{x}_1(m), \boldsymbol{x}_2(m), \ldots, \boldsymbol{x}_{N_{d2}^t}(m)$ are drawn from $f(\boldsymbol{x}(m); \widehat{v}_{t-1}(m))$. In addition, we calculate the performances $\Omega_{t,i}$ $(i = 1, 2, \ldots, N_{d2}^t)$, and order them from smallest to largest: $\Omega_{t,1} \leq \Omega_{t,2} \leq \cdots \leq \Omega_{t,N_{d2}^t}$. It is noted that β_m^2 is calculated by $\beta_m^2 = c_m^2 N$, and $\widehat{\gamma}_t$ is updated by (20). After that, we compare $\Omega_{t,i}$ with $\widehat{\gamma}_t$, and obtain all eligible performances greater than $\widehat{\gamma}_t$ and merge them into a set $\mathcal{S}_2 := \{\Omega_{(t,\lceil(1-\theta_t)N_{d2}^t\rceil)}, \Omega_{(t,\lceil(1-\theta_t)N_{d2}^t\rceil+1)}, \ldots, \Omega_{t,N_{d2}^t}\}$, where β_m^2 is the number of the element of $\mathcal{S}_2$ and Ω_t^* is the maximum element of $\mathcal{S}_2$. Then, $q_t(k|j,m)$ is calculated and the specific derivation process can be found in Theorem 3. Thus, $\boldsymbol{Q}_t(m)$ is the probability matrix composed of $q_t(k|j,m)$, and $\widehat{v}_t$ is equivalent to $\boldsymbol{Q}_t(m)$.

Theorem 3. *Assume that there are N_m continuous and mutually correlative tasks for each target. After that, the selected scheme is required to be deleted after each task is accomplished. The other settings are the same as Theorem 2. Thus, when performing the mth task, we can obtain the updating formula of $\boldsymbol{Q}(m)$, as follows:*

$$\begin{cases} q(k|j,m) = \frac{\sum_{n=1}^{c_m^2 N} g(x_j^n(m);k)}{c_m^2 N} \\ k \in \{1, \ldots, L-m+1\}, n \in \{1, \ldots, c_m^2 N\}, c_m^2 \in \boldsymbol{c}_2 \end{cases}. \tag{26}$$

Proof. Please see Appendix C. □

Through the iterative updating of $\boldsymbol{Q}(m)$, the optimal probability matrix vector $\boldsymbol{Q}^*$ and the maximum performance vector Ω^* are obtained. Then, the main steps of the MCAFACE algorithm for dealing with problem $\mathcal{P}2$ are explained in Algorithm 4, and the convergence of the MCAFACE algorithm is also close to that of CE method in [40].

Algorithm 4 MCAFACE algorithm.

Input: N_{m}, l_m^2, N, h.
Output: $\boldsymbol{Q}^*$, Ω^*.

Set $N^{\min} = N$ and $N^{\max} = hN$;
for $m = 1; m < N_{\mathrm{m}}; m++$ **do**
  Initialize $\boldsymbol{Q}_0(m)$ with a uniform distribution and define $\widehat{\boldsymbol{v}}_0 = \boldsymbol{Q}_0(m)$, then set $t = 1$;
  while at the t-th iteration ($t \geq 1$) **do**
    if $t = 1$ **then**
      Generate N_{d2}^t ($N_{\mathrm{d2}}^t = N^{\min}$) random samples $\boldsymbol{x}_1(m), \boldsymbol{x}_2(m), \ldots, \boldsymbol{x}_{N_{\mathrm{d2}}^t}(m)$ from $f(\cdot; \widehat{\boldsymbol{v}}_0)$;
    else
      Draw N_{d2}^t ($N^{\min} \leq N_{\mathrm{d2}}^t \leq N^{\max}$) random samples $\boldsymbol{x}_1(m), \boldsymbol{x}_2(m), \ldots, \boldsymbol{x}_{N_{\mathrm{d2}}^t}(m)$ from $f(\cdot; \widehat{\boldsymbol{v}}_{t-1})$;
    end if
    Update $\widehat{\gamma}_t$ according to Equation (19) and calculate Ω_t^*;
    Calculate $q_t(k|j,m)$ by Equation (A15) in Appendix C;
    if $\sum_{k=1}^{l_m^2} q_t(k|j,m) = 1$ and $q_t(k|j,m) \in \{0,1\}$ **then**
      Stop, obtain $\boldsymbol{Q}_t^*(m)$ and Ω_t^*, then $\boldsymbol{Q}^*(m) \leftarrow \boldsymbol{Q}_t^*(m)$ and $\Omega(x^*(m)) \leftarrow \Omega_t^*$;
    else
      Calculate $\boldsymbol{Q}_t(m)$ and update $\widehat{\boldsymbol{v}}_t$ by $\widehat{\boldsymbol{v}}_t = \boldsymbol{Q}_t(m)$;
      Set $t = t + 1$, take random integer N_{d2}^t in $[N^{\min}, N^{\max}]$, then go to step 4;
    end if
  end while
end for
Return $\boldsymbol{Q}^*$ and Ω^*.

4.3. Complexity Analysis of the MIAFACE Algorithm and the MCAFACE Algorithm

Let N_{m} represent the number of tasks, n_{d} denote the random sample to perform each task, n_{f} represent the iteration number of AFACE algorithm to perform each task, N_{e} denote the elite sample, N_{t} represent the number of targets, and L denote the number of all possible UAV deployment schemes. The computational complexity of AFACE algorithm is divided into four parts: initialization C_1, sample C_2, sort C_3, and update C_4. Meanwhile, these parts can be defined as

$$C_1 = N_{\mathrm{t}} \times L \tag{27}$$
$$C_2 = n_{\mathrm{f}} \times n_{\mathrm{d}} \tag{28}$$
$$C_3 = n_{\mathrm{f}} \times n_{\mathrm{d}} \log n_{\mathrm{d}} \tag{29}$$
$$C_4 = n_{\mathrm{f}} \times (n_{\mathrm{d}} - N_{\mathrm{e}}). \tag{30}$$

Specifically, the computational complexity of AFACE algorithm can be written as

$$\begin{aligned} C_{\mathrm{f}} &= C_1 + C_2 + C_3 + C_4 \\ &= N_{\mathrm{t}} \times L + n_{\mathrm{f}} \times (n_{\mathrm{d}} + n_{\mathrm{d}} \log n_{\mathrm{d}} + n_{\mathrm{d}} - N_{\mathrm{e}}) \end{aligned}. \tag{31}$$

Obviously, n_{f} increases with the increment of $(N_{\mathrm{t}} \times L)$, i.e., $n_{\mathrm{f}} \propto (N_{\mathrm{t}} \times L)$. Then, Equation (31) is rewritten as

$$\begin{aligned} C_{\mathrm{f}} &= N_{\mathrm{t}} \times L \times (1 + n_{\mathrm{d}} + n_{\mathrm{d}} \log n_{\mathrm{d}} + n_{\mathrm{d}} - N_{\mathrm{e}}) \\ &= N_{\mathrm{t}} \times L \times (n_{\mathrm{d}} \log n_{\mathrm{d}} + 2n_{\mathrm{d}} - N_{\mathrm{e}} + 1) \end{aligned}, \tag{32}$$

where $n_{\mathrm{d}} \log n_{\mathrm{d}}$ is greater than the other terms in the bracket on the right side of the equation. Thus, the time complexity of AFACE algorithm can be computed as $\mathcal{O}(N_{\mathrm{t}} \times L \times n_{\mathrm{d}} \log n_{\mathrm{d}})$.

When the proposed algorithms are applied to accomplishing N_m tasks of targets in problems $\mathcal{P}1$ and $\mathcal{P}2$, respectively, according to Algorithm 3 and Equation (32), the computational complexity of MIAFACE algorithm is

$$C_{mi} = N_m \times C_f. \tag{33}$$

Thus, its time complexity is written as $\mathcal{O}(N_m \times N_t \times L \times n_d \log n_d)$. However, based on Algorithm 4 and Equation (32), the computational complexity of the MCAFACE algorithm is

$$\begin{aligned} C_{mc} =& N_t \times (L + L - 1 + \cdots + L - N_m + 1) \times (n_d \log n_d + 2n_d - N_e + 1) \\ =& N_t \times (N_m \times L - (1 + 2 + \cdots + N_m - 1)) \times (n_d \log n_d + 2n_d - N_e + 1) \\ =& N_t \times N_m \times (L - \frac{(N_m - 1)}{2}) \times (n_d \log n_d + 2n_d - N_e + 1). \end{aligned} \tag{34}$$

Since $N_m \geq 3$, its time complexity is approximately equal to $\mathcal{O}(N_m \times N_t \times (L-1) \times n_d \log n_d)$.

5. Simulation and Analysis

In order to verify the effectiveness of the proposed algorithms, we compared these proposed algorithms with the CE method and other intelligent algorithms by applying them to the multi-UAV cooperative task assignment problem. The simulations were implemented in Pycharm Community's 2019.1.1 x64 version of the programming environment on an Intel Core PC with 8 GB memory. The total cumulative reward that the UAV formations earn by successfully completing three tasks from all targets are used to measure the system performance.

On the basis of the above algorithms, various simulations were performed by assigning three types of UAVs located in the corresponding bases to accomplish three tasks of 20 targets in a 200 m × 200 m combat scenario. The position of each base and these targets are shown in Figure 2. Bases B1, B2, and B3 are located in (0,0), (0,200), and (200,0), respectively. The information of three types of UAVs and 20 target are given in Tables 2 and 3, respectively, where a and b represent two types of resources, for example, the number of resources a and b needed for different types of UAVs or to accomplish different tasks, and also they have no units.

Table 2. Information of three types of UAVs.

UAV	Base	$U_{resource}$ (Units)		p_k	p_s
		a	b		
Type A	B1	1	2	0.9	0.7
Type B	B2	2	2	0.8	0.8
Type C	B3	3	3	0.7	0.9

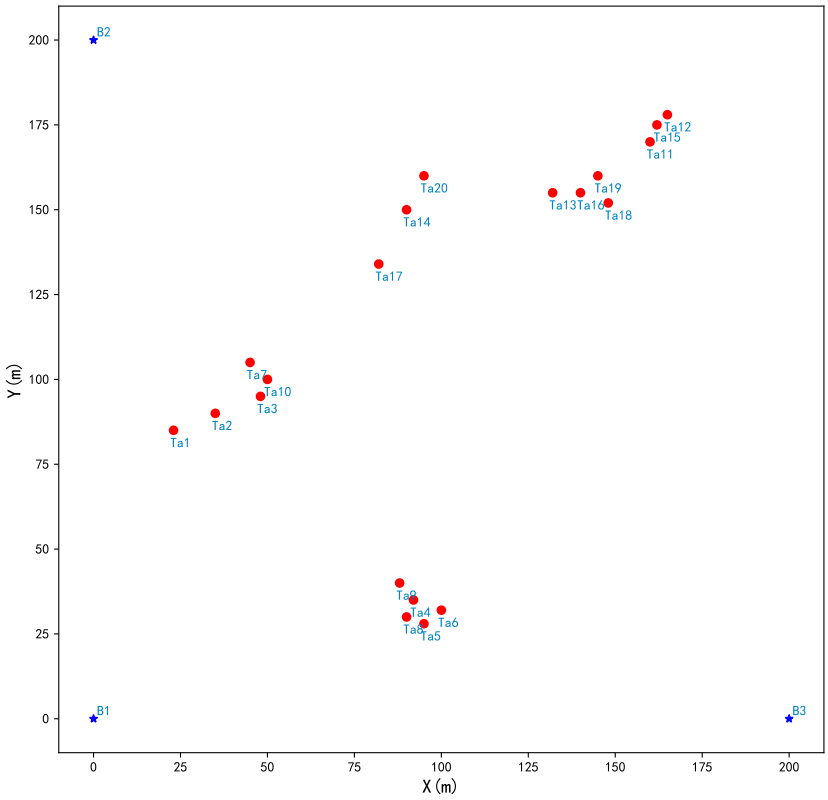


Figure 2. Initial bases and targets state.

Table 3. Information of 20 targets.

Target	Position	$T_{resource}$			y	s
		K_1 ([a,b])	K_2 ([a,b])	K_3 ([a,b])		
Target 1	(23,85)	[2,3]	[2,3]	[2,3]	30	2
Target 2	(35,90)	[3,3]	[3,3]	[3,3]	70	6
Target 3	(48,95)	[2,3]	[2,3]	[2,3]	50	4
Target 4	(92,35)	[3,2]	[3,2]	[3,2]	100	10
Target 5	(95,28)	[2,2]	[2,2]	[2,2]	120	8
Target 6	(100,32)	[3,2]	[3,2]	[3,2]	40	5
Target 7	(45,105)	[3,3]	[3,3]	[3,3]	65	3
Target 8	(90,30)	[2,2]	[2,2]	[2,2]	78	7
Target 9	(88,40)	[2,2]	[2,2]	[2,2]	35	9
Target 10	(50,100)	[3,2]	[3,2]	[3,2]	63	5
Target 11	(160,170)	[2,3]	[5,3]	[4,3]	30	2
Target 12	(165,178)	[3,3]	[5,3]	[5,3]	70	6
Target 13	(132,155)	[3,3]	[5,3]	[6,3]	50	4
Target 14	(90,150)	[3,3]	[3,5]	[4,4]	100	10
Target 15	(162,175)	[2,2]	[4,5]	[4,4]	120	8
Target 16	(140,155)	[2,3]	[6,3]	[5,3]	40	5
Target 17	(82,134)	[3,3]	[4,3]	[4,3]	65	3
Target 18	(148,152)	[2,3]	[4,2]	[5,2]	78	7
Target 19	(145,160)	[3,2]	[3,4]	[4,3]	35	9
Target 20	(95,160)	[2,2]	[3,4]	[4,4]	63	5

Referring to Theorem 1, we note that when z exceeds 3, these simulations are complicated. Thus, z is set to be 3, i.e., no more than 3 UAVs are needed to accomplish three tasks of targets in a specific order, and then the total number of each type of UAV is unrestricted. Then, each target in the following cases has 19 possible schemes, i.e., A, B, C, AA, AB, AC, BB, BC, CC, AAA, AAB, AAC, ABB, ACC, BBB, BBC, BCC, CCC, and

ABC, respectively, and these schemes correspond to numbers from 1 to 19. After that, we can use a matching approach to quickly find the feasible schemes. The resources needed to accomplish three tasks of targets are randomly generated and satisfy the maximum cooperative number of UAVs.

In the following simulations, the notations used in the tables and the figures are displayed as

- $U_{resource}$ represents the initial resources consumed by three types of UAVs;
- $T_{resource}$ represents the resources consumed by three tasks; and
- Time is CPU time in seconds for each case, and the time of each case is the average consumption time of running 100 times of each algorithm.

The parameters of the CE method, MIAFACE algorithm, MCAFACE algorithm, PSO algorithm, ACO algorithm, and GA algorithm are assumed to be set in Table 4, where the settings of the speed and maximum flying distance of the UAV are referred to [35] and they have no effect on the simulation results. For more detailed theory and parameter settings of CE, PSO, ACO, and GA (see [8–10,30,35,41]). For the targets in Table 3, there are two scenarios in the multi-UAV cooperative task assignment problem.

(1) In scenario 1, we consider the first 10 targets or more similar targets. When performing the three tasks of each target, we obtain the identical optimal scheme vector of each task. Therefore, the situation in which each target has different tasks but each task has the same optimal scheme is called the problem with continuous and independent tasks.

(2) In scenario 2, the last 10 targets or more similar targets are considered. When performing the three tasks of each target, we obtain the different optimal scheme vector of each task. Thus, the situation in which each target has different tasks and each task does not have the same optimal scheme is called the problem with continuous and correlative tasks.

Table 4. Simulation parameter settings.

Parameter	Value
The target identification	$P_c = 1$
Weight coefficients	$w_1 = 0.8, w_2 = 0.18, w_3 = 0.02$
The UAV's speed	$V = 40$ m/s
The maximum flying distance	$D_{max} = 1000$ m
Time window of task K_1 (s)	$[e_1, s_1] = [3, 10]$
Time window of task K_2 (s)	$[e_2, s_2] = [8, 20]$
Time window of task K_3 (s)	$[e_3, s_3] = [18, 26]$
Consumption time of task K_1	$T_1 = 5$ s
Consumption time of task K_2	$T_2 = 10$ s
Consumption time of task K_3	$T_3 = 5$ s
The number of targets	$N_t \in [5, 20]$
The fixed random samples	$N = 1000$
The quantile in CE	$\theta = 0.1$
Inertial weight in PSO	$w = 0.75$
Learning factors in PSO	$\eta_1 = \eta_2 = 0.5$
The number of ants in ACO	$N_a = 200$
Pheromone evaporation coefficient in ACO	$\varepsilon = 0.9$
Transfer probability in ACO	$P_a = 0.2$
Mating probability in GA	$P_1 = 0.8$
Mutation probability in GA	$P_2 = 0.01$

5.1. Scenario 1

In case 1, we used the first 10 targets in Table 3 to perform continuous and independent tasks of problem $\mathcal{P}1$, and the results are shown in Table 5.

According to Table 5, we note that the optimal scheme vector and the total result of CE and MIAFACE are identical, while that of MCAFACE is suboptimal to the other

two algorithms. Moreover, we can obtain some observations. (i) For CE, the number of iterations and the optimal scheme vector are both 4 and [3,3,3,2,2,2,3,2,2,3], respectively, and the results of each task are −79.50, 274.90, and −79.50, and the sum of the results of each task is 115.9. The situations of MIAFACE are similar to CE, except that the number of iterations is 3. (ii) For MCAFACE, the numbers of iterations and the optimal scheme vectors are 3, 2, 1 and [3,3,3,2,2,2,3,2,2,3], [9,9,9,7,7,7,9,7,7,9], [18,18,18,15,15,15,18,15,15,18], respectively, and the results of each task are −79.50, 179.0, and −82.14, and the sum of the results of each task is 17.36. (iii) The total times of using CE, MIAFACE and MCAFACE are 3.36, 3.29, and 2.17, respectively.

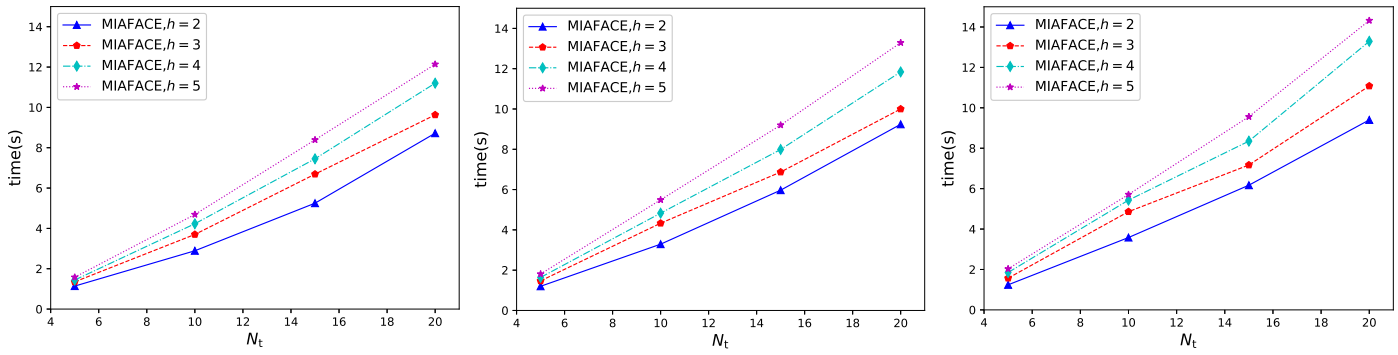
In case 2, we tested the MIAFACE algorithm and MCAFACE algorithm under h and $\boldsymbol{c}_1$, and their times change with N_t in Figures 3a–c and 4a–c, respectively.

From Figures 3 and 4, the curves of MIAFACE and MCAFACE both show an increasing trend as N_t grows, and their times increase with the increment of $\boldsymbol{c}_1$ and h. Meanwhile, the time differences between the curves gradually increase with the growth of N_t in each figure. In Figure 3a, the curve with $h = 2$ is at the lowest of the four curves, while the curve with $h = 5$ is at the highest of the four curves. The remaining two curves are in the middle, and the curve with $h = 4$ is at the top and the other one is at the bottom. Moreover, the time ranges of the four curves are both approximately in [1,12]. In Figure 3b,c, their situations are described similarly to Figure 3a, and their time ranges are in [1,14] and [1,15], respectively. From Figure 4a, the order of the four curves is similar to Figure 3a. Moreover, their time ranges are both roughly in [0.3,10]. In Figure 4b,c, their situations are analogous to Figure 4a, and their time ranges are in [0.3,10] and [0.3,12], respectively.

In case 3, the CE method, PSO algorithm, ACO algorithm, and GA algorithm are both used three times for three tasks continuously. We compared them with MIAFACE algorithm by obtaining the same optimal score under $h = 2$ and $\boldsymbol{c}_1$, and their times change with N_t in Figure 5a–c. Since MCAFACE algorithm obtains suboptimal results in scenario 1, it is not compared to other algorithms.

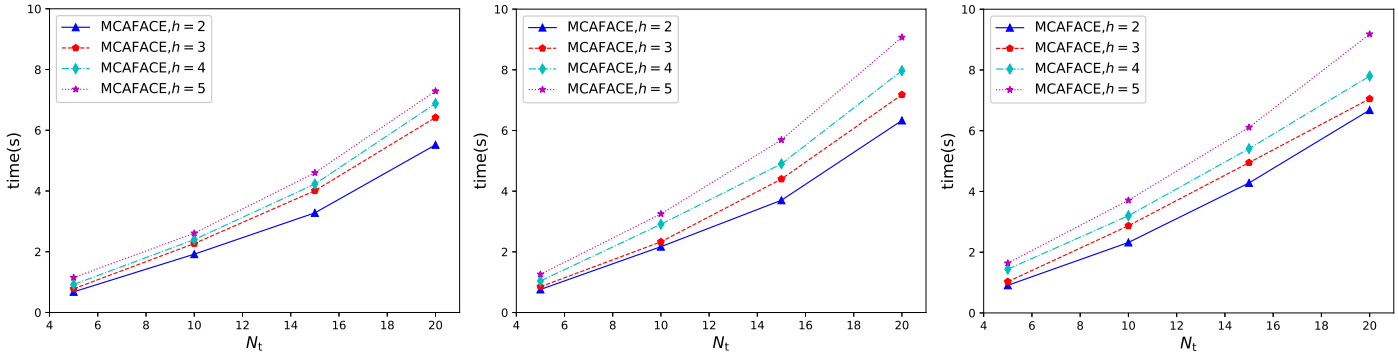
Table 5. Iterative results of three algorithms in case 1.

Algorithm	CE	MIAFACE	MCAFACE
Task	K_1 K_2 K_3	K_1 K_2 K_3	K_1 K_2 K_3
Iterations	4	3	3 2 1
Optimal scheme vector	[3,3,3,2,2,2,3,2,2,3]	[3,3,3,2,2,2,3,2,2,3]	[3,3,3,2,2,2,3,2,2,3] [9,9,9,7,7,7,9,7,7,9] [18,18,18,15,15,15,18,15,15,18]
Result of each task	−79.50 274.90 −79.50	−79.50 274.90 −79.50	−79.50 179.0 −82.14
Sum of each task's result	115.9	115.9	17.36
Total time(s)	3.36	3.29	2.17



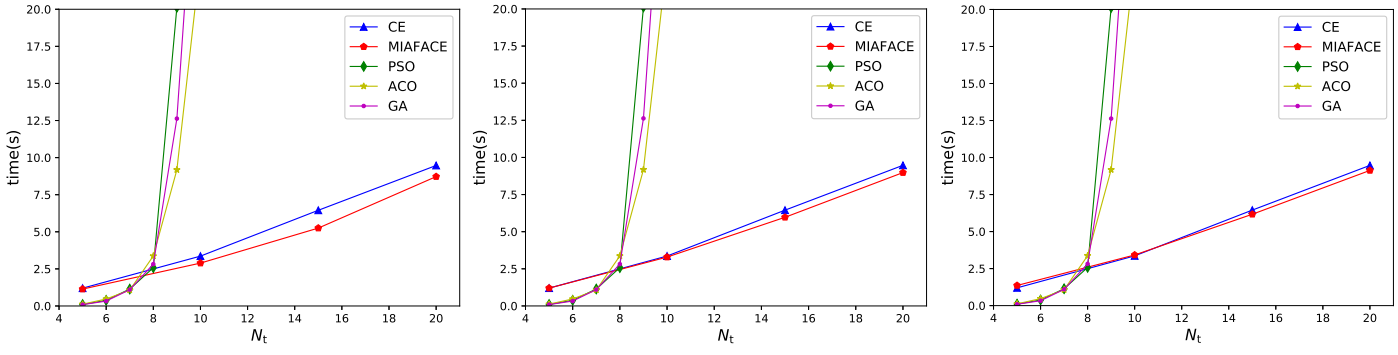
(**a**) (**b**) (**c**)

Figure 3. Time changing with N_t under MIAFACE algorithm in scenario 1. (**a**) c_1 = [0.01,0.02,0.03]. (**b**) c_1 = [0.02,0.03,0.04]. (**c**) c_1 = [0.03,0.04,0.05].



(**a**) (**b**) (**c**)

Figure 4. Time changing with N_t under MCAFACE algorithm in scenario 1. (**a**) c_1 = [0.01,0.02,0.03]. (**b**) c_1 = [0.02,0.03,0.04]. (**c**) c_1 = [0.03,0.04,0.05].



(**a**) (**b**) (**c**)

Figure 5. Time changing with N_t under $h = 2$ and different algorithms in scenario 1. (**a**) c_1 = [0.01,0.02,0.03]. (**b**) c_1 = [0.02,0.03,0.04]. (**c**) c_1 = [0.03,0.04,0.05].

From Figure 5, we note that the curves of CE and MIAFACE grow linearly, while the curves of PSO, ACO, and GA increase exponentially. In addition, their times increase gradually with the increment of c_1 and N_t. In Figure 5a, when N_t is in [5,20], the time of MIAFACE is less than that of CE, and the time difference between the two algorithms grows as N_t increases. Meanwhile, when N_t is below 8, the times of PSO, ACO, and GA are lower than that of CE and MIAFACE, but when N_t is more than 8, the situation is reversed. In addition, the time ranges of CE and MIAFACE are both approximately in [1,10], while the times of other algorithms are over 20 when N_t is larger than 10. From Figure 5b,c, their situations are similar to Figure 5a, except that the time difference between CE and

MIAFACE in Figure 5b is lower than that in Figure 5a, and the time difference in Figure 5c first decreases gradually to intersect at a point where N_t is 10, then increases slowly with the increment of N_t.

5.2. Scenario 2

In case 4, we utilized the last 10 targets in Table 3 to perform continuous and correlative tasks of problem $\mathcal{P}2$, and the results are shown in Table 6.

According to Table 6, we note that the optimal scheme vectors and the total results of CE, MIAFACE, and MCAFACE are the same. The reason for this phenomenon is that for three tasks of the same 10 targets, the optimal scheme vectors are eventually obtained and identical by using the three algorithms, which leads to the same score of the total objective function; however, the consumption time by the different algorithms varies. Moreover, some observations are available. First, for CE, the number of iterations and the sum of each task's result are 5 and 218.72, and the optimal solution vectors are [3,3,3,3,3,3,3,3,3,3], [9,9,9,7,7,7,9,7,7,9], [18,18,18,15,15,15,18,15,15,18], and the results of each task are −298.65, 819.85, and −302.48, respectively. Secondly, the situations using MIAFACE and MCAFACE are similar to that of CE, apart from the fact that the number of iterations in MIAFACE is 4 and the numbers of iterations in MCAFACE are 4, 4, and 3. Finally, the total times using CE, MIAFACE, and MCAFACE are 7.33, 7.11, and 6.9, respectively.

In case 5, we tested the MIAFACE algorithm and MCAFACE algorithm under h and c_2, and their times change with N_t in Figures 6a–c and 7a–c, respectively.

From Figures 6 and 7, the variations of the curves, the times and the time differences are both similar to Figures 3 and 4, while in Figures 6a and 7a, the time grows rapidly when N_t is over 10. The reason is that the results of these two figures are suboptimal to others. In Figure 6a, the order of the curves is the same as that of each figure in Figures 3 and 4. In addition, the time ranges of these four curves are both approximately in [1,50]. From Figure 6b,c, the situations are described similarly to that of Figure 6a and their time ranges are in [2,30] and [2,32], except that their results are the optimal results. In Figure 7, the situation of each figure is roughly similar to that of the corresponding figure in Figure 6, apart from the fact that the time range is lower than that in Figure 6.

Table 6. Iterative results of three algorithms in case 4.

Algorithm	CE	MIAFACE	MCAFACE
Task	K_1 K_2 K_3	K_1 K_2 K_3	K_1 K_2 K_3
Iterations	5	4	4 4 3
Optimal scheme vector	[3,3,3,2,2,2,3,2,2,3] [9,9,9,7,7,7,9,7,7,9] [18,18,18,15,15,15,18,15,15,18]	[3,3,3,2,2,2,3,2,2,3] [9,9,9,7,7,7,9,7,7,9] [18,18,18,15,15,15,18,15,15,18]	[3,3,3,2,2,2,3,2,2,3] [9,9,9,7,7,7,9,7,7,9] [18,18,18,15,15,15,18,15,15,18]
Result of each task	−298.65 819.85 −302.48	−298.65 819.85 −302.48	−298.65 819.85 −302.48
Sum of each task's result	218.72	218.72	218.72
Total time(s)	7.33	7.11	6.9

In case 6, we compared the CE method, MIAFACE algorithm, MCAFACE algorithm, PSO algorithm, ACO algorithm, and GA algorithm by obtaining the same optimal score under $h = 2$ and c_2, and their times change with N_t in Figure 8a–c. The CE method, PSO algorithm, ACO algorithm, and GA algorithm are also used three times for three tasks continuously.

From Figure 8, we note that for CE, MIAFACE, MCAFACE, PSO, ACO, and GA, the variations of the curves and the times are similar to the case in Figure 5. In Figure 8a, when N_t is below 11, the times of MIAFACE and MCAFACE are relatively close and less than that of CE; however, when N_t is over 11, the times of MIAFACE and MCAFACE grow quickly and more than that of CE due to obtaining the suboptimal results. Moreover, the time ranges of CE, MIAFACE, and MCAFACE are both approximately in [1,30]. Meanwhile, the times of PSO, ACO, and GA are much higher than that of CE, MIAFACE, and MCAFACE, and their time ranges are over 30 when N_t is more than 6. From Figure 8b,c, the situations of PSO, ACO, and GA are similar to Figure 8a. In Figure 8b, the time differences between CE, MIAFACE, and MCAFACE grow as N_t increases. In addition, the time of MCAFACE is lower than that of CE and MIAFACE, and the curves of CE and MIAFACE intersect at $N_t = 8$ and the time of CE is also lower than that of MIAFACE when N_t is below 8, then the situation is reversed after N_t exceeds 8. Moreover, the time ranges of CE, MIAFACE, and MCAFACE are both in [2,22]. In Figure 8c, the time differences between CE, MIAFACE, and MCAFACE decrease, and then increase as N_t grows. Furthermore, the curves of CE, MIAFACE, and MCAFACE intersect at $N_t = 9$ and the time of CE is lower than that of MIAFACE and MCAFACE when N_t is below 9, then the situation is reversed after N_t exceeds 9.

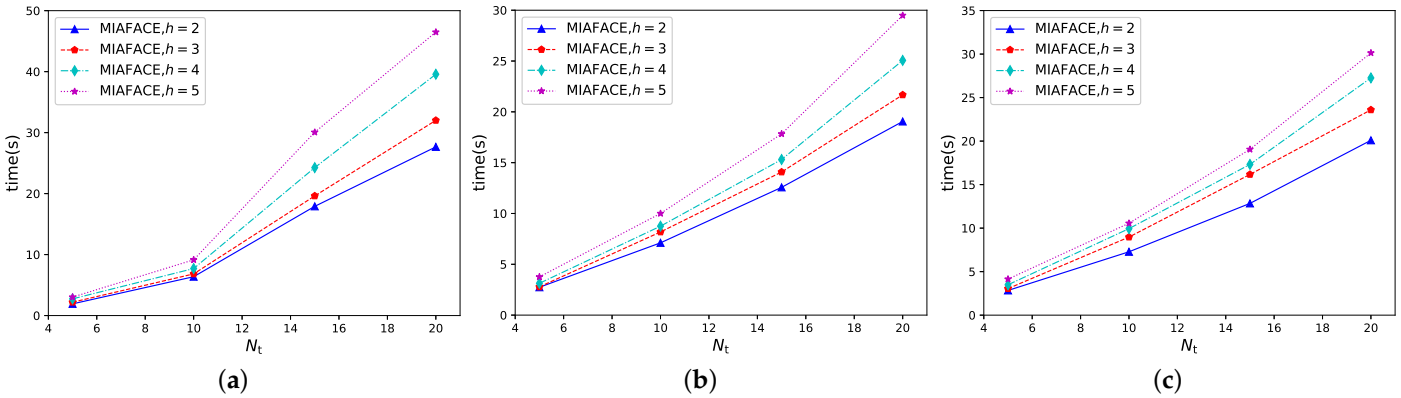


Figure 6. Time changing with N_t under MIAFACE algorithm in scenario 2. (**a**) c_2 = [0.01,0.02,0.03]. (**b**) c_2 = [0.02,0.03,0.04]. (**c**) c_2 = [0.03,0.04,0.05].

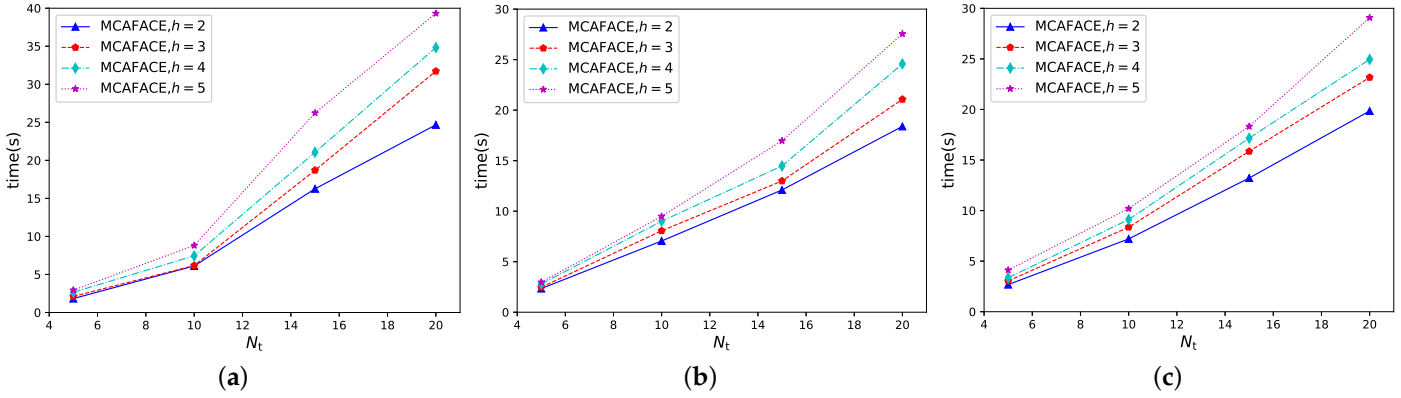


Figure 7. Time changing with N_t under MCAFACE algorithm in scenario 2. (**a**) c_2 = [0.01,0.02,0.03]. (**b**) c_2 = [0.02,0.03,0.04]. (**c**) c_2 = [0.03,0.04,0.05].

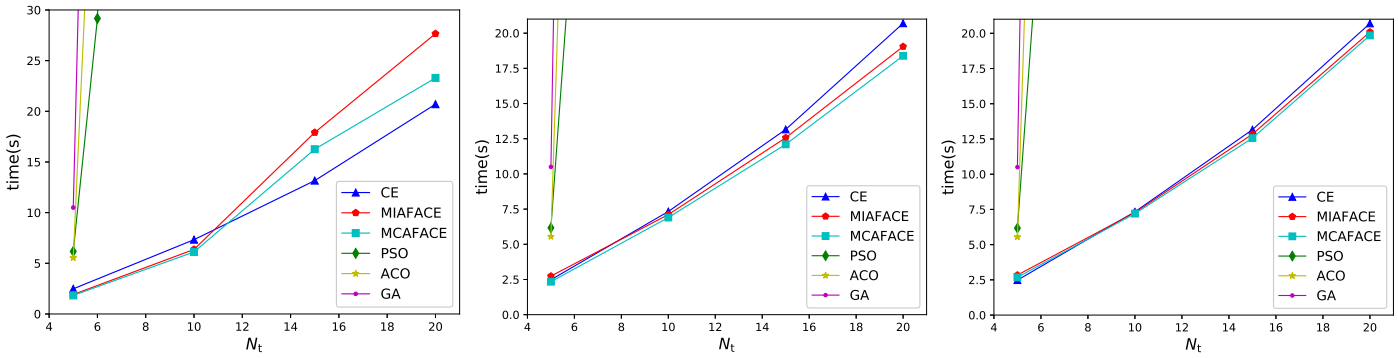


Figure 8. Time changing with N_t under $h = 2$ and different algorithms in scenario 2. (**a**) c_2 = [0.01,0.02,0.03]. (**b**) c_2 = [0.02,0.03,0.04]. (**c**) c_2 = [0.03,0.04,0.05].

5.3. Analysis

Analysing the results of case 1 and case 4, we note that the optimal scheme vectors of using MIAFACE and MCAFACE algorithms in problems $\mathcal{P}1$ and $\mathcal{P}2$, respectively, are obtained by initializing and updating the probability matrices $\boldsymbol{P}(m)$ and $\boldsymbol{Q}(m)$, which conforms to Algorithms 3 and 4 described in Section 4.2. In addition, the result of MCAFACE in case 1 is suboptimal to that of other algorithms due to deleting the corresponding optimal solution after the end of each task.

Comprehensively considering the situations of case 2 and case 5, we note that the times of CE, MIAFACE, and MCAFACE increase with the increment of N_t, h, as well as $\boldsymbol{c}$ and the time complexity of MCAFACE is lower than that of MIAFACE, and these phenomena comply with the complexity analysis of MIAFACE and MCAFACE in Section 4.3. In addition, the time of case 5 is superior to that of case 2 because there are more available solutions for each target in case 5 than in case 2 after each iteration. Meanwhile, in case 5, using MIAFACE and MCAFACE for solving this problem is easy to fall into local optimum when $\boldsymbol{c}$ is inferior to a certain vector, e.g., $\boldsymbol{c} = [0.01, 0.02, 0.03]$. The reason behind this phenomenon is that when all elements in $\boldsymbol{c}$ are small and more solutions exist after each iteration, the optimal scheme may not be selected during one of the iterations of MIAFACE and MCAFACE, leading to a suboptimal result.

Comparing the situations of case 3 and case 6, we note that the times of PSO, ACO, and GA are only related to the growth of N_t. Meanwhile, CE, MIAFACE, and MCAFACE are superior to PSO, ACO, and GA for large-scale allocation problems, e.g., more than 8 targets of case 3 and 5 targets of case 6. Moreover, CE is inferior to MIAFACE in scenario 1, e.g., Figure 5, when N_t is over 10. Moreover, e.g., Figure 8c in scenario 2, MCAFACE is superior to MIAFACE and CE when N_t is over 9.

6. Conclusions

In this paper, the multi-UAV cooperative task assignment problem was described and formulated, and three types of UAVs were considered, cooperatively accomplishing the classification, attack, and verification tasks of targets under resource, precedence, and timing constraints. After that, considering complex coupling among these three tasks, we decomposed the considered problem into two subproblems. In order to solve them, we proposed an AFACE algorithm, a MIAFACE algorithm, and a MCAFACE algorithm. Finally, simulation results verified that both MIAFACE and MCAFACE consume less time than other intelligent algorithms for solving the corresponding problem.

Nevertheless, there still exist challenges when applying the MIAFACE algorithm and MCAFACE algorithm to processing optimization problems, e.g., appropriate parameter settings, falling into local optimum when using lower elements in $\boldsymbol{c}$, etc. In future work, it will be meaningful to concentrate on promoting these two algorithms on problems where

it is vulnerable to local optimum when the number of samples is limited and on task assignment problems in complex dynamic scenarios.

Author Contributions: Conceptualization, K.W., X.Z. and X.L.; methodology, X.Z.; software, X.Z.; validation, K.W., X.Z., X.L. and W.C.; formal analysis, K.W. and X.Z.; investigation, X.Z.; resources, X.Q. and Y.C.; data curation, X.Q., Y.C. and K.L.; writing—original draft preparation, X.Z.; writing—review and editing, X.Z.; visualization, X.Z.; supervision, K.W.; project administration, Y.C. and K.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded in part by the National Natural Science Foundation of China under Grant 62172313 and 52031009, in part by the Natural Science Foundation of Hunan Province under Grant 2021JJ20054.

Data Availability Statement: Data sharing is not applied.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

If one of each type of UAVs is selected, i.e., $z = 1$, the possible schemes are written as

$$L = C_{N_b}^1 C_z^1 = 3. \tag{A1}$$

When $z = 2$, we can choose no more than three types of UAVs, and then

$$L = C_{N_b}^1 C_z^1 + C_{N_b}^2 C_z^2 = 9. \tag{A2}$$

Once $z \geq 3$, we can choose no more than three types of UAVs; thus

$$\begin{aligned} L &= \underbrace{C_{N_b}^1 C_z^1}_{1\ type} + \underbrace{C_{N_b}^2 C_z^2}_{2\ types} + \underbrace{C_{N_b}^3 C_z^3}_{3\ types} \\ &= 3z + \frac{3z(z-1)}{2} + \frac{z(z-1)(z-2)}{6}. \\ &= 3z + \frac{z(z-1)(z+7)}{6} \end{aligned} \tag{A3}$$

As a conclusion, the number of the possible schemes for each task of targets can be defined as

$$L = 3z + \frac{z(z-1)(z+7)}{6}, z \geq 1. \tag{A4}$$

Thus, we have successfully proven Theorem 1.

Appendix B

Inserting $\boldsymbol{P}(m)$ and Equation (1) into $f(\boldsymbol{x}(m); u)$, we define the problem $\mathcal{P}1$ as

$$\begin{aligned} f(\boldsymbol{x}(m); \boldsymbol{P}(m)) &= \prod_{j=1}^{N_t} p(x_j(m)|j, m) \\ &= \prod_{j=1}^{N_t} \prod_{k=1}^{l_m^1} p(k|j, m)^{g\left(x_j(m);k\right)} \end{aligned}, \tag{A5}$$

where $p(k|j, m)$ represents the coefficient in the column k and the row j of $\boldsymbol{P}(m)$, $g(x_j(m); k)$ is 1 if $\phi(x_j(m))$ equals k and 0 otherwise according to Equation (1).

After that, at the tth iteration, we assume that the samples $\boldsymbol{x}_1(m), \boldsymbol{x}_2(m), \ldots, \boldsymbol{x}_{N_{d1}^t}(m)$ are drawn from $f(\boldsymbol{x}(m); \widehat{\boldsymbol{v}}_{t-1}(m))$. In addition, we calculate the performances $\Omega_{t,i}$, and order them from smallest to largest: $\Omega_{t,1} \leq \Omega_{t,2} \leq \cdots \leq \Omega_{t,N_{d1}^t}$, and then define $\widehat{\gamma}_t(m) = \Omega_{(N_{d1}^t - \beta_m^1)}$.

Thus, Equation (A5) can be rewritten as follows:

$$\begin{aligned} \arg\max_{\boldsymbol{P}(m)} \frac{1}{N_{\mathrm{d1}}^{t}} \sum_{i=1}^{N_{\mathrm{d1}}^{t}} I_{\{\Omega(x_i(m))\geq\widehat{\gamma}_t(m)\}} \\ \times \ln f(\boldsymbol{x}_i(m);\boldsymbol{P}(m)) \end{aligned} . \tag{A6}$$

In Equation (A6), $I(\boldsymbol{x}_i(m);\widehat{\gamma}_t(m))$ is recognized, and when $N_{\mathrm{d1}}^{t} \to \infty$, the problem is equal to

$$\max_{\boldsymbol{P}(m)} \sum_{n=1}^{\beta_m^1} \ln f(\boldsymbol{x}(m);\boldsymbol{P}(m)). \tag{A7}$$

Putting Equation (A5) into Equation (A7), we have

$$\begin{aligned} &\max_{\boldsymbol{P}(m)} \sum_{n=1}^{\beta_m^1} \ln f(\boldsymbol{x}(m);\boldsymbol{P}(m)) \\ &= \max_{p(k|j,m)} \sum_{n=1}^{\beta_m^1} \ln\left(\prod_{j=1}^{\mathrm{N_t}} \prod_{k=1}^{l_m^1} p(k|j,m)^{g(x_j^n(m);k)}\right) \\ &= \max_{p(k|j,m)} \sum_{n=1}^{\beta_m^1} \sum_{j=1}^{\mathrm{N_t}} \sum_{k=1}^{l_m^1} g(x_j^n(m);k) \ln(p(k|j,m)). \end{aligned} \tag{A8}$$

Then, we assume that $r_{kj}(m) = p(k|j,m)$, $a_{kj}^n(m) = g(x_j^n(m);k)$, and Equation (A8) is modeled as

$$\begin{aligned} \mathcal{P}11: \min_{r_{kj}(m)} (&- \sum_{n=1}^{\beta_m^1} \sum_{j=1}^{\mathrm{N_t}} \sum_{k=1}^{l_m^1} a_{kj}^n(m) \ln\Big(r_{kj}(m)\Big)) \\ \text{s.t.} \quad &\sum_{k=1}^{l_m^1} r_{kj}(m) = 1 \quad \forall j,m \\ &r_{kj}(m) \geq 0 \quad \forall j,k,m \\ &l_m^1 = L \quad \forall m. \end{aligned} \tag{A9}$$

Considering $\mathcal{P}11$ as a convex problem and denoting the convex function by $f(r_{kj}(m))$, we can obtain the Lagrangian function

$$\begin{aligned} &O(r_{kj}(m),\lambda_j(m),\mu_{kj}(m)) = f\Big(r_{kj}(m)\Big) + \\ &\sum_{j=1}^{\mathrm{N_t}} \lambda_j(m)\left(\sum_{k=1}^{L} r_{kj}(m) - 1\right) + \sum_{j=1}^{\mathrm{N_t}} \sum_{k=1}^{L} \mu_{kj}(m)\Big(-r_{kj}(m)\Big) \end{aligned} , \tag{A10}$$

where $\lambda_j(m)$ and $\mu_{kj}(m)$ are the relevant restraint coefficients.

Generally, for convex optimization problem, the Karush–Kuhn–Tucker (KKT) condition is required and sufficient [42]. Thus, considering the KKT conditions of problem in Equation (A10), we have

$$\begin{cases} -\frac{a_{kj}(m)}{r_{kj}(m)} + \lambda_j(m) - \mu_{kj}(m) = 0 \\ \lambda_j(m)\left(\sum_{k=1}^{L} r_{kj}(m) - 1\right) = 0 \\ \mu_{kj}(m) r_{kj}(m) = 0 \\ \lambda_j(m) > 0 \\ \mu_{kj}(m) \geq 0 \\ r_{kj}(m) \geq 0 \end{cases} \tag{A11}$$

When solving Equation (A11), we obtain

$$\begin{cases} r_{kj}(m) = \frac{a_{kj}^n(m)}{\lambda_j(m) - \mu_{kj}(m)} \\ \lambda_j(m) = \sum_{k=1}^{L} a_{kj}^n(m) \\ \mu_{kj}(m) = 0 \end{cases}. \tag{A12}$$

Comparing $\lambda_j(m)$ and $r_{kj}(m)$, we acquire the relationship between $r_{kj}(m)$ and $a_{kj}^n(m)$, i.e.,

$$r_{kj}(m) = \frac{a_{kj}^n(m)}{\sum_{k=1}^{L} a_{kj}^n(m)}. \tag{A13}$$

Returning to our problem, the updating formula of $\boldsymbol{P}(m)$ is given by

$$\begin{aligned} p(k|j,m) &= \frac{\sum_{n=1}^{\beta_m^1} g(x_j^n(m);k)}{\beta_m^1} \\ &= \frac{\sum_{n=1}^{c_m^1 N} g(x_j^n(m);k)}{c_m^1 N} \end{aligned}, \tag{A14}$$

where $k \in \{1, \ldots, L\}$, $n \in \{1, \ldots, c_m^1 N\}$, $c_m^1 \in \boldsymbol{c}_1$.

Therefore, we have successfully proven Theorem 2.

Appendix C

Calculating the updating formulas of N_m tasks continuously and correlatively is considered.

For the mth task, if $m = 1$, its optimal solution is taken from L schemes, and if $1 < m \leq N_\mathrm{m}$, its optimal scheme is only taken from the remaining $L - m + 1$ solutions since the $m - 1$ schemes selected before performing the mth task have been deleted.

Thus, referring to the proof process of Theorem 2, the updating formula of $\boldsymbol{Q}(m)$ in problem $\mathcal{P}2$ is

$$\begin{aligned} q(k|j,m) &= \frac{\sum_{n=1}^{\beta_m^2} g(x_j^n(m);k)}{\beta_m^2} \\ &= \frac{\sum_{n=1}^{c_m^2 N} g(x_j^n(m);k)}{c_m^2 N} \end{aligned} \tag{A15}$$

where $k \in \{1, \ldots, L - m + 1\}$, $n \in \{1, \ldots, c_m^2 N\}$, $c_m^2 \in \boldsymbol{c}_2$.

Hence, we have successfully proven Theorem 3.

References

1. Singh, H.; Sharma, M. Electronic Warfare System Using Anti-Radar UAV. In Proceedings of the 2021 8th International Conference on Signal Processing and Integrated Networks, Noida, India, 26–27 August 2021; pp. 102–107.
2. Deng, Z.; Gao, Y.; Hu, A.; Zhang, Y. A Mobile Phone Uplink CPDP-DTDOA Positioning Method Using UAVs for Search and Rescue. *IEEE Sens. J.* **2022**, *22*, 18170–18179. [CrossRef]
3. Fan, B.; Jiang, L.; Chen, Y.; Zhang, Y.; Wu, Y. UAV Assisted Traffic Offloading in Air Ground Integrated Networks With Mixed User Traffic. *IEEE T. Intell. Transp.* **2022**, *23*, 12601–12611. [CrossRef]
4. D'Arcy, S.; Gonzalez, F. Design and Flight Testing of a Rocket-Launched Folding UAV for Earth and Planetary Exploration Applications. In Proceedings of the 2022 IEEE Aerospace Conference, Big Sky, MT, USA, 5–12 March 2022; pp. 1–15.

5. Chen, X.; Liu, Y.; Yin, L.; Qi, Y. Cooperative Task Assignment and Track Planning For Multi-UAV Attack Mobile Targets. *J. Intell. Robot. Syst.* **2020**, *100*, 1383–1400.
6. Sabo, C.; Kingston, D.; Cohen, K. A Formulation and Heuristic Approach to Task Allocation and Routing of UAVs under Limited Communication. *Unmanned Syst.* **2014**, *2*, 1–17. [CrossRef]
7. Wang, J.; Zhang, Y.F.; Geng, L.; Fuh, J.Y.H.; Teo, S.H. A Heuristic Mission Planning Algorithm for Heterogeneous Tasks with Heterogeneous UAVs. *Unmanned Syst.* **2015**, *3*, 205–219. [CrossRef]
8. Gou, Q.; Li, Q. Task assignment based on PSO algorithm based on Logistic function inertia weight adaptive adjustment. In Proceedings of the 2020 3rd International Conference on Unmanned Systems, Harbin, China, 1–4 September 2020; pp. 825–829.
9. Li, Y.; Zhang, S.; Chen, J.; Jiang, T.; Ye, F. Multi-UAV Cooperative Mission Assignment Algorithm Based on ACO method. In Proceedings of the 2020 International Conference on Computing, Networking and Communications, Big Island, HI, USA, 17–20 February 2020; pp. 304–308.
10. Ma, Y.; Zhang, H.; Zhang, Y.; Gao, R.; Xu, Z.; Yang, J. Coordinated Optimization Algorithm Combining GA with Cluster for Multi-UAVs to Multi-tasks Task Assignment and Path Planning. In Proceedings of the 2019 IEEE 15th International Conference on Control and Automation, Edinburgh, UK, 22–26 August 2019; pp. 1026–1031.
11. Duan, X.; Liu, H.; Tang, H.; Cai, Q.; Zhang, F.; Han, X. A Novel Hybrid Auction Algorithm for Multi-UAVs Dynamic Task Assignment. *IEEE Access* **2020**, *8*, 86207–86222. [CrossRef]
12. Chen, J.; Wu, Q.; Xu, Y.; Qi, N.; Guan, X.; Zhang, Y.; Xue, Z. Joint Task Assignment and Spectrum Allocation in Heterogeneous UAV Communication Networks: A Coalition Formation Game-Theoretic Approach. *IEEE Trans. Wirel. Commun.* **2021**, *20*, 440–452. [CrossRef]
13. Qie, H.; Shi, D.; Shen, T.; Xu, X.; Li, Y.; Wang, L. Joint Optimization of Multi-UAV Target Assignment and Path Planning Based on Multi-Agent Reinforcement Learning. *IEEE Access* **2019**, *7*, 146264–146272. [CrossRef]
14. Tang, J.; Chen, X.; Zhu, X.; Zhu, F. Dynamic Reallocation Model of Multiple Unmanned Aerial Vehicle Tasks in Emergent Adjustment Scenarios. *IEEE Trans. Aerosp. Electron. Syst.* **2022**, 1–43. [CrossRef]
15. Qie, H.; Shi, D.; Shen, T.; Xu, X.; Li, Y.; Wang, L. Distributed Cooperative Search Algorithm With Task Assignment and Receding Horizon Predictive Control for Multiple Unmanned Aerial Vehicles. *IEEE Access* **2021**, *9*, 6122–6136.
16. Fu, X.; Feng, P.; Gao, X. Swarm UAVs Task and Resource Dynamic Assignment Algorithm Based on Task Sequence Mechanism. *IEEE Access* **2019**, *7*, 41090–41100. [CrossRef]
17. Chen, Y.; Yang, D.; 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. [CrossRef]
18. Zhu, F.; Wu, F.; Chen, C.F.; Li, D.; Guo, Y.; Zhang, J.G.; Zhao, X. A coordinated assignment method for multi-UAV area search tasks. In Proceedings of the CSAA/IET International Conference on Aircraft Utility Systems, Nanchang, China, 17–20 August 2022; pp. 751–756.
19. Chen, Y.; Chen, J.; Du, C. Allocation of Multi-UAVs Timing-dependent Tasks based on Completion Time. In Proceedings of the 2022 WRC Symposium on Advanced Robotics and Automation, Beijing, China, 20 August 2022; pp. 71–76.
20. Yan, S.; Xu, J.; Song, L.; Pan, F. Heterogeneous UAV collaborative task assignment based on extended CBBA algorithm. In Proceedings of the 2022 7th International Conference on Computer and Communication Systems, Wuhan, China, 22–25 April 2022; pp. 825–829.
21. Yan, S.; Pan, F.; Zhang, D.; Xu, J. Research on Task Reassignment Method of Heterogeneous UAV in Dynamic Environment. In Proceedings of the 2022 6th International Conference on Robotics and Automation Sciences, Wuhan, China, 9–11 June 2022; pp. 57–61.
22. Liu, C.; Guo, Y.; Li, N.; Song, X. AoI-Minimal Task Assignment and Trajectory Optimization in Multi-UAV-Assisted IoT Networks. *IEEE Internet Things J.* **2022**, *9*, 21777–21791. [CrossRef]
23. Zhu, C.; Zhang, G.; Yang, K. Fairness-Aware Task Loss Rate Minimization for Multi-UAV Enabled Mobile Edge Computing. *IEEE Wirel. Commun. Lett.* **2023**, *12*, 94–98. [CrossRef]
24. Seid, A.M.; Lu, J.; Abishu, H.N.; Ayall, T.A. Blockchain-Enabled Task Offloading with Energy Harvesting in Multi-UAV-assisted IoT Networks: A Multi-agent DRL Approach. *IEEE J. Sel. Areas Commun.* **2022**, *40*, 3517–3532. [CrossRef]
25. Hu, N.; Qin, X.; Ma, N.; Liu, Y.; Yao, Y.; Zhang, P. Energy-efficient Caching and Task offloading for Timely Status Updates in UAV-assisted VANETs. In Proceedings of the 2022 IEEE/CIC International Conference on Communications in China, Sanshui, Foshan, China, 11–13 August 2022; pp. 1032–1037.
26. Gao, H.; Feng, J.; Xiao, Y.; Zhang, B.; Wang, W. A UAV-assisted Multi-task Allocation Method for Mobile Crowd Sensing. *IEEE Trans. Mob. Comput.* **2022**. [CrossRef]
27. Rubinstein, R.Y. Optimization of computer simulation models with rare events. *Eur. J. Oper. Res.* **1997**, *99*, 89–112. [CrossRef]
28. Rubinstein, R.Y. The cross-entropy method for combinatorial and continuous optimization. *Methodol. Comput. Appl. Probab.* **1999**, *1*, 127–190. [CrossRef]
29. Rubinstein, R.Y. Combinatorial optimization, cross-entropy, ants and rare events. In *Stochastic Optimization: Algorithms and Applications*; Springer: Boston, MA, USA, 2001; pp. 303–363.
30. De Boer, P.-T.; Kroese, D.P.; Mannor, S.; Rubinstein, R.Y. A tutorial on the cross-entropy method. *Ann. Oper. Res.* **2005**, *134*, 19–67. [CrossRef]

31. Chepuri, K.; Homem-de-Mello, T. Solving the vehicle routing problem with stochastic demands using the cross-entropy method. *Ann. Oper. Res.* **2005**, *134*, 153–181. [CrossRef]
32. Rubinstein, R.Y.; Kroses, D.P. The cross-entropy method: A unified approach to combinatorial optimization Monte-Carlo simulation and machine learning. *Technometrics* **2006**, *48*, 147–148.
33. Undurti, A.; How, J. A Cross-Entropy Based Approach for UAV Task Allocation with Nonlinear Reward. In Proceedings of the AIAA Guidance, Navigation, and Control Conference, Toronto, ON, Canada, 2–5 August 2010; pp. 1–16.
34. Le Thi, H.A.; Nguyen, D.M.; Dinh, T.P. Globally solving a nonlinear UAV task assignment problem by stochastic and deterministic optimization approaches. *Optim. Lett.* **2012**, *6*, 315–329. [CrossRef]
35. Huang, L.; Qu, H.;Zuo, L. Multi-Type UAVs Cooperative Task Allocation Under Resource Constraints. *IEEE Access* **2018**, *6*, 17841–17850. [CrossRef]
36. Cofta, P.; Ledziński, D.; Śmigiel, S.; Gackowska, M. Cross-Entropy as a Metric for the Robustness of Drone Swarms. *Entropy* **2020**, *22*, 597. [CrossRef] [PubMed]
37. Zhang, X.; Wang, K.; Dai, W. Multi-UAVs Task Assignment Based on Fully Adaptive Cross-Entropy Algorithm. In Proceedings of the 2021 11th International Conference on Information Science and Technology, Chengdu, China, 7–10 May 2021; pp. 286–291.
38. Wei, Y.; Wang, B.; Liu, W.; Zhang, L. Hierarchical Task Assignment of Multiple UAVs with Improved Firefly Algorithm Based on Simulated Annealing Mechanism. In Proceedings of the 2021 40th Chinese Control Conference, Shanghai, China, 26–28 July 2021; pp. 1943–1948.
39. Wang, Q.; Liu, L.; Tian, W. Cooperative Task Assignment of Multi-UAV in Road-network Reconnaissance Using Customized Genetic Algorithm. In Proceedings of the 2021 IEEE 4th Advanced Information Management, Communicates, Electronic and Automation Control Conference, Chongqing, China, 18–20 June 2021; pp. 803–809.
40. Costa, A.; Jones, O.D.; Kroese, D. Convergence properties of the cross-entropy method for discrete optimization. *Oper. Res. Lett.* **2007**, *35*, 573–580. [CrossRef]
41. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95—International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995; pp. 1942–1948.
42. Luo, Z.-Q.; Yu, W. An introduction to convex optimization for communications and signal processing. *IEEE J. Sel. Areas Commun.* **2006**, *24*, 1426–1438.