Hindawi
Wireless Communications and Mobile Computing
Volume 2021, Article ID 5530454, 14 pages
https://doi.org/10.1155/2021/5530454

Research Article

Solving "Limited" Task Allocation Problem for UAVs Based on Optimization Algorithms

Xiaopan Zhang[1] **and Xingjun Chen**[2]

[1]*School of Resources and Environmental Engineering, Wuhan University of Technology, Wuhan 430070, China*
[2]*Institution of Dalian Naval Academy, Dalian 116018, China*

Correspondence should be addressed to Xingjun Chen; cxj_dna@yeah.net

Received 26 February 2021; Accepted 22 June 2021; Published 8 July 2021

Academic Editor: Giovanni Pau

With the rapid development of science and technology, unmanned technology has been widely used in many fields. One of the most important applications is in the field of civil and military UAVs. In the field of military UAVs (unmanned aerial vehicles), UAVs usually have to complete a series of tasks. In this series of tasks, there are often some key tasks. Key tasks play an important role, which is highly related to the feasibility of the whole action or task; mission failure sometimes causes incalculable damage. When assigning tasks to UAVs, it is necessary to ensure the accurate implementation of key tasks, so as to ensure the orderly implementation of the overall task. This paper not only successfully solved the previous problems but also comprehensively considered the minimization of resource consumption and the maximization of task revenue in the process of UAV task allocation. On the basis of considering the key system, considering the constraints and multiobjective problems in the UAV task allocation process, the violence allocation algorithm, constraint optimization evolutionary algorithm, PSO algorithm, and greedy algorithm combined with a constraint evolutionary algorithm are improved and optimized; it has been proven that they can solve the above difficulties. At the same time, several comparison experiments have been carried out; the performance and conclusion of the above four algorithms in the "limited" UAV task allocation scheme are analyzed in the experimental part.

1. Introduction

An unmanned aerial vehicle (UAV) is an important part of an unmanned system. The traditional unmanned system also includes a ground control platform and communication system between UAV and ground. A military unmanned combat system adds a mission system to the traditional unmanned combat system. The construction of the whole combat system is mission centred, because in the unmanned combat system, there is no need to consider the personal safety, human physiology, environment, time, and other factors that will limit human activities. The unmanned combat system will focus more on the operation itself and improve the operational efficiency and capability of the system as much as possible [1]. With the shift of the combat centre from human to task, researchers were focusing more on the tasks. The effective planning and reasonable allocation of tasks have become a research hotspot. In recent years, researchers domestically and internationally have made a lot of attempts and experiments on task allocation, among which the use of BDD [2], brute-force method, genetic algorithm [3], particle swarm optimization algorithm [4], ant colony algorithm [5], and Hungarian algorithm [6] have made remarkable achievements.

Eun and Bang used the genetic algorithm to solve the task allocation and path planning problem of UAV [3]. Fei et al. used the ant colony algorithm to solve the cooperative multitask assignment problem (CMTAP) of the unmanned aerial vehicle (UAV) [5]. The Hungarian algorithm was proposed for task allocation. Fang et al. studied the communication and information collection of the underwater unmanned aerial vehicle (AUV) [7, 8]. Zhu et al. proposed an integrated biologically inspired self-organizing map (SOM) algorithm for task assignment and path planning of an autonomous underwater vehicle (AUV) system in 3-D underwater environments with objective avoidance [9]. Amorim et al.

assessed a swarm-GAP-based solution for the task allocation problem in dynamic scenarios [10].

With the rapid development of task allocation technology, researchers began to move from the overall task allocation to the specific tasks [11]. In the researches and experiments, they have paid more attention to the task itself, the combat action and process. An event-based mission planning scheme has been proposed, which fully considered the problems of cooperative operation in the process of UAV cluster operation. In the process of UAV cooperative operation, each UAV has its own task, which helps to achieve the overall goal of the task. Some of these UAVs carry a decisive role in the mission; they have a higher income and a greater role in promoting the mission and even directly determine the feasibility of the whole mission. The failure of these missions is a devastating blow to the entire operational plan. The researchers have tried to apply the binary decision diagram to the reliability modelling of task allocation; they tried to use a binary decision diagram [2] to predict the success probability of the whole UAV group to execute the mission together. If an emergency occurs, or the success probability of the overall task is too low, the task allocation scheme will be reconfigured and modified; this is to ensure the smooth progress of the whole task planning process and the implementation of the scheme.

It is not difficult to find that the previous research content is biased to two aspects. One is to optimize the allocation of the overall task and maximize the use of resources and time in the whole allocation process, but it lacks the consideration of the task itself and the measurement of the feasibility of the whole operation plan. However, in real combat, it may often consume a lot of resources and time for a specific target task, because it plays a decisive role in the whole task. On the other hand, the whole planning process is too biased towards the stability of the whole task planning system and the task process, which ensures the orderly progress of tasks as far as possible and lacks the thinking of some nonkey tasks. The revenue accumulation of these nonkey tasks may not be underestimated for the whole combat mission. Because of the problems in the front, it leads to the research ideas of this paper. This paper attempts to improve the original task method, on the premise of ensuring the maximum revenue, and takes the stability of the system into account to ensure the smooth implementation of key tasks. Finally, it can realize the smooth implementation of the whole operation plan, the inevitable implementation of key tasks, and the maximization of nonkey task benefits.

This paper focused on the smooth implementation of key tasks and the maximization of nonkey task revenue. At the same time, this paper also considers the constraints between tasks and UAVs and the optimization of multiobjective problems. Here, this paper improves and optimizes the four research algorithms, the violence allocation algorithm, constraint optimization evolutionary algorithm [12], PSO algorithm, and greedy algorithm [13] combined with the constraint evolutionary algorithm, so that the scheme can adapt to and solve the previous problems. The research scheme proposed in this paper improves the coding mode and optimization mode of the former four algorithms. On the basis of retaining the original efficiency and effect of the algorithm, it can still solve the research topic efficiently and quickly. The whole algorithm has better generality, more practical application of the combat plan, and better certainty in the process of task allocation. Finally, it is realized, and the original research results are further improved, which fills the blank in the research field of taking into account the smooth implementation of key tasks and maximizing the benefits of nonkey tasks and promotes the research and development of UAV combat task allocation.

This paper used forced coding. Before the plan is started, the key tasks must be put in. This paper will generate a list of UAVs that can perform key tasks in advance. In the process of algorithm iteration and solution, key tasks always ensure that key tasks will not be lost because of the heuristic rules used in algorithm solution. At the same time, the priority of tasks is processed in the evaluation function. The execution of key tasks must be in the highest priority and then consider the multiobjective problem in task allocation. In the selection of an algorithm, this paper chooses the brute-force method as the basic contrast object, which can solve the optimal task allocation scheme for the reference. In addition, the multiobjective constrained optimization evolutionary algorithm was selected, which has excellent performance in solving the task allocation problem. The paper tried to improve the evolutionary algorithm and combine it with the greedy algorithm and found that it can enrich the diversity of the population and got better global optimal solution. This paper also chose the particle swarm optimization algorithm; the study found that the algorithm has a good performance in solving efficiency and solving effect.

The main contributions of this paper are as follows:

(1) A "limited" UAV task allocation scheme is proposed

(2) Four specific task allocation schemes are designed and compared, and the advantages and disadvantages of each scheme are obtained

The rest of this paper is arranged as follows. Section 2 introduces the normal description of the concepts and symbol representation of task planning. Four improved and optimized "limited" task allocation schemes are shown in Section 3. Section 4 presents the experimental results and analysis. Section 5 summarizes the work in this paper and envisions future work.

2. Related Work and Background

This part will introduce the concept and symbol representation of basic traditional task allocation. The concept and other related information of task assignment are based on an event model. Also, this paper focuses on the concept of the "limited" task allocation scheme and other related information.

2.1. Traditional Task Allocation. Traditional task allocation mainly involves UAV, undifferentiated task, execution sequence, execution probability, and other key concepts [14].

(1) Problem model: assume that there are m UAVs and n tasks globally. Then, an allocation method is solved, which makes the scheme meet the multiobjective optimization objectives and the constraints of UAV task allocation

(2) UAVs: $U = \{u_1 \cdots u_M\}$, where m is the number of UAVs. The current position of UAV u_i is expressed as $ul_i = (x, y)$ (where x, y can be two-dimensional coordinates or longitude and latitude), and the configured resources are expressed as $R = \{r_1 \cdots r_n\}$, where r_i (real number, $i = 1 \cdots M$) is the number of resources carried by the UAV u_i (in order to simplify the symbolic representation, the ammunition resources, fuel resources, medical resources, etc. carried by the UAV are converted into resources according to the proportion and importance, and usually the forces are also converted into resources)

(3) Tasks (no distinction between key and nonkey tasks): $MS = \{ms_1 \cdots ms_N\}$, where N is the number of tasks, the position of task ms_i is expressed as $msl_i = (x, y)$ (where x, y can be two-dimensional coordinates or longitude and latitude), the resource consumed to execute ms_i is rc_i, the time consumed is t_i, the effective time window that can be executed is $[\text{start}T_i, \text{end}T_i]$, and the benefit obtained is gain_i

(4) Task allocation: $\prod = \{\pi_1 \cdots \pi_M\}$, where $\pi_i (i = 1 \cdots M)$ indicates the assignment of task MSI. For example, $\pi_i = ms_i \longrightarrow u_1$ indicates that task ms_i is assigned to UAV u_1 for execution. For simplicity, it is directly expressed as $\pi_i = 1$. If the task ms_j is not assigned to any UAV, it is expressed as $\pi_j = \text{Null}$

(5) Execution sequence: $Q = \{q_1 \cdots q_m\}$, where $q_i (i = 1 \cdots m)$ represents the task sequence to be executed by the UAV u_i; for example, $q_1 = ms_2 ms_4 ms_1$ means that u_1 will execute tasks ms_2, ms_4, and ms_1, respectively

2.2. Task Planning Based on Event Models. In operational task planning, the superior usually sets the overall operational objectives and then formulates a series of key operational tasks according to the battlefield situation and environment [15]. Finally, it is necessary to plan a set of operational task execution schemes that can achieve the overall operational objectives and evaluate the success probability of the scheme implementation, so as to maximize the success probability of the overall operational scheme [16, 17].

(1) Event model: under the premise of a given overall operational objective and a series of key operational tasks, how to plan the operational task execution scheme that can achieve the overall operational objective and evaluate the probability of successful implementation of the scheme is important. Assuming that the overall combat target is O (objective) and the set of key combat tasks is $\{T_1 \cdots T_K\}$, $x_i (i = 1 \cdots k)$ can be used to represent the basic event "task T_i successfully executed" and $\bar{x}_i$ to represent the basic event "task T_i not successfully executed".

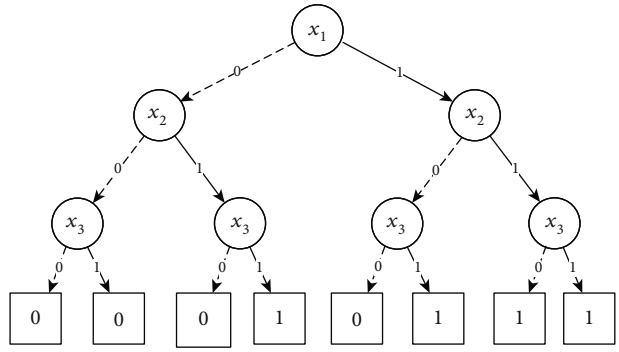

Figure 1: Decision tree.

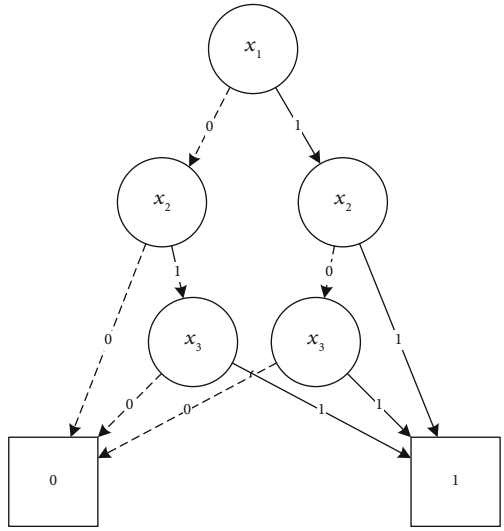

Figure 2: Decision chart.

(2) Binary decision diagram: the binary decision diagram (BDD) [18] is a kind of directed acyclic graph, which can accurately evaluate the process of the whole operation scheme and calculate the success probability of each allocation scheme. It includes from the root node to the terminal node (terminal node is 1, which means the overall combat mission is successfully completed; terminal node is 0, which means the overall combat mission failed). For example, for the Boolean expression $f(x_1, x_2, x_3) = x_1x_2 + x_2x_3 + x_1x_3$, the following binary decision graph can be constructed (Figure 1 shows the decision tree [19], and Figure 2 is transformed from Figure 1 to a decision graph)

(3) The main steps of decision-making are as follows:

(i) Based on the event model, each element of combat task allocation is represented formally, such as mission and UAV

Input: UAVs, Tasks, Key Tasks.
Output: The best task assignment sequence and its objective function value.
1: **For** each UAV u_i
2: Get all of its executable tasks;
3: Arrange the tasks in full order;
4: Get all the executable task sequences of the UAV ***execQueue***;
5: **For** each task sequence in ***execQueue***
6: **If** not all key-tasks exist
7: Delete current task sequence;
8: **For** each task sequence in ***execQueue***
9: Evaluation function of computing task sequence;
10: Sort the ***execQueue*** according to the evaluation results;
11: Select the best task sequence after sorting.

Pseudocode 1: Pseudocode of the brute-force algorithm.

(ii) According to the battlefield situation, the Boolean function f of the overall combat target about the key combat task set is extracted

(iii) The binary decision graph is constructed according to f

(iv) According to the binary decision diagram, all the schemes that can achieve the overall goal are searched

(v) According to the binary decision diagram and the occurrence probability of each event, the scheme with the highest success probability is solved

3. "Limited" Task Allocation Scheme

In this paper, the brute-force method, constrained evolutionary algorithm, PSO algorithm, constrained evolutionary algorithm, and greedy algorithm are used for comparative experiments. The four allocation schemes are composed of the following main steps: test data generation, algorithm initialization, algorithm solving, algorithm evaluation, and data output.

The traditional task planning scheme lacks the thinking of the task itself and considers all tasks the same random tasks to solve, so as to maximize the benefits as much as possible. But there is a lack of assessment of the overall mission action. It is precise in real combat that the importance of tasks is different. Some tasks have to be carried out. Decision-makers should not only consider the simple quantitative benefits but also correctly handle the absolutely key task here. They must be carried out in the process of mission planning, and only on this basis can the quantitative benefits of the overall operation be considered.

The latter allocation scheme based on the event model and binary decision graph also has two obvious disadvantages.

First, only key tasks are considered. The so-called key tasks are those that have a substantial impact on the success of the overall goal. But in the actual task planning, in addition to key tasks, there are many nonkey tasks (the proportion may be much larger than that of key tasks). Although the implementation of nonkey tasks will not affect the achievement of the overall goal, it will increase revenue or reduce losses. For example, in order to capture area A, it is necessary to carry out the key task of "annihilating enemy forces." After the key task is completed, you can choose to carry out the nonkey task of "rescuing hostages." "Hostage rescue" may not affect the final "capture of area A," but it will increase revenue.

Secondly, it only considers the task planning from the high level and does not consider the time, resources, forces, losses, benefits, task attributes, and other factors of specific tasks. It is possible to get a huge set of schemes through the binary decision graph, and the benefit with the largest success probability of this set may be small (this phenomenon usually exists in reality). It is necessary to consider the selection of schemes from many aspects for the so-called high risk and high benefit. If the probability of success is slightly lower, other benefits may increase significantly.

In view of the problems existing in the previous research scheme, this paper comprehensively consider the two research ideas. On the basis of the event-based task allocation model, this paper considers the benefits of nonkey tasks and then comprehensively measures the nonkey tasks and key tasks, abstracts a new task planning solution model, and designs a new problem model. Finally, this paper uses the improved traditional task allocation scheme for task planning.

(1) Problem model: assume that according to the binary decision graph, the top n scheme $\{S_1 \cdots S_n\}$ with the highest probability is selected, and the scheme $S_i = x_1 \cdots x_k$ represents the sequence of task execution, which are all key tasks. Suppose that there are M UAVs and N missions (including key and nonkey missions) globally. To solve a task allocation scheme, (1) include all the key tasks in S_i, (2) maximize the number of successfully assigned tasks, (3) maximize the benefits of executing the assigned tasks, (4) minimize the loss of resources, and (5) minimize the maximum time of executing tasks in UAV

(2) Execution probability:

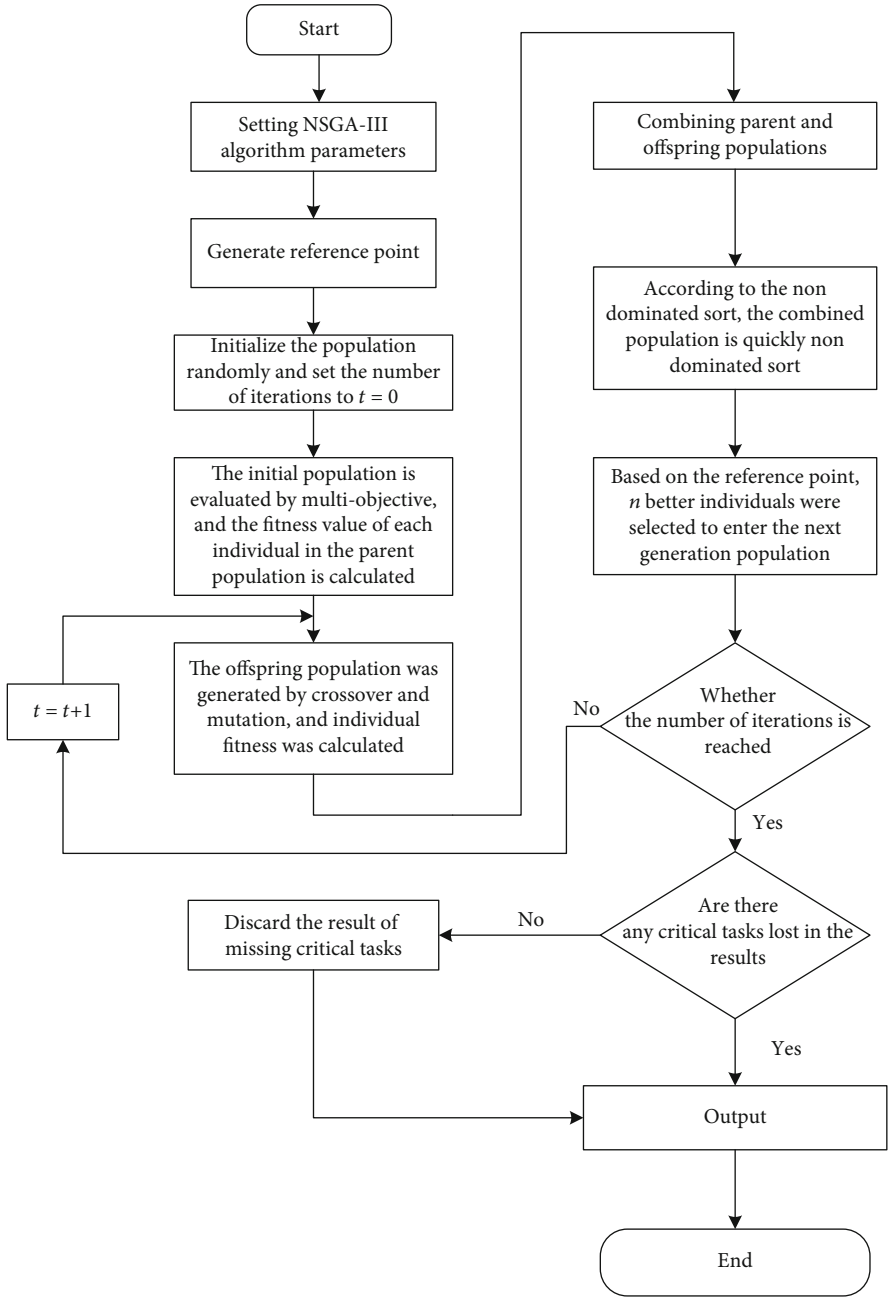

Figure 3: NSGA-III flowchart.

$$P = \begin{bmatrix} p_{11} & \cdots & p_{1N} \\ \vdots & \ddots & \vdots \\ p_{M1} & \cdots & p_{MN} \end{bmatrix}. \quad (1)$$

$p_{ij}(1 \le i \le M, 1 \le j \le N)$ represents the success rate when the UAV u_i performs the task ms_j

(3) Constraint condition:

(1) Time constraint: the UAV must first arrive at the location of the task, and then, it can execute the task. The earliest start event and the latest end time of the task are in an interval $[\text{start}T_i, \text{end}T_i]$. The arrival time of the UAV u_i meets

$$\text{arrivalTime}_{i,j} = \text{flyTime}_{i,j} + \text{curTime} \le \text{end}T_i. \quad (2)$$

$\text{arrivalTime}_{i,j}$ represents the time when the UAV u_i arrives at the location of task ms_j, and $\text{flyTime}_{i,j}$ represents the time required for the UAV u_i to fly from the current position to task ms_j. curTime means the current time for the task allocation

Input: UAVs, Tasks, Key Tasks.
Output: The best task assignment sequence and its objective function value.
1: The UAV, mission and key task are encoded as particles.
2: **DO**
3: **For** per particle
4: Calculated the fitness (using evaluation function);
5: **If** (fitness is better than historical best value of particles)
6: Use *Xi* update the best individual in history *Pi*;
7: **End**
8: Select the best particle in the current particle swarm;
9: **If** (the current best particle is better than the historical best particle)
10: Update with the best particle of current swarm *Pg*;
11: **For** per particle
12: Update the particle velocity according to formula (11);
13: Update the particle position according to formula (12);
14: **End**
15: **While** The maximum number of iterations is not reached

Pseudocode 2: Pseudocode of the PSO algorithm.

Input: Individual x
Output: Optimized individual x
1: $oldF \longleftarrow getFvalue(x)$;
2: **do**
3: **for** i =1 **to** m **do**
4: **for**$\forall v_{ij}, \in x_i$ $(j \neq k)$**do**
5: Exchange the positions of v_{ij} and v_{ik} in x_i;
6: Calculate $newF \longleftarrow getFvalue(x)$;
7: **if** $oldF - newF \leq 0$ **then**
8: Restore the positions of v_{ij} and v_{ik} in x_i;
9: **end if**
10: **end for**
11: **end for**
12: **while** (x is updated)
13: **return** x

Pseudocode 3: Greedy algorithm

(2) Resource constraints: on the premise that the UAV can arrive at the task location on time, each task needs a certain amount of resource consumption to be executed. Only when the UAV meets the resource requirements of the task can the task be executed. If the UAV can meet the resource requirements of the task, the UAV will go to the location of the task; otherwise, it will not perform the task

$$\text{carryResourcere}_i \geq \text{costResource}_j. \quad (3)$$

carryResourcere_i is the resources carried by the current UAV u_i, and costResource_j is the resources consumed by the task ms_j to be executed

(3) Functional constraints: during actual combat, UAVs may be divided into many types; each UAV has its own specific function or has outstanding ability in a certain aspect. For example, the reconnaissance UAV has good concealment and mobility, the combat UAV can carry more weapons and equipment with an accurate strike system, the transport UAV has better transport capacity to provide resource support for other UAVs or provide support for combat personnel, and the rescue UAV is equipped with various medical equipment and has certain personnel transport capacity. With this in mind, establish a functional constraint between the mission and the UAVs. ms_j can only be executed by a specific UAV or a class of UAVs. Similarly, UAV u_i can only perform a specific task or category of tasks

Table 1: Algorithm efficiency of $m = 20$, $n = 50$, and $k = 20$.

	Brute force	NSGA-III	PSO	NSGA-III-greedy
Times	1	20	20	20
Total time (s)	—	2215.29	530.93	3775.07
Avg time (s)	—	110.76	26.55	188.75

Table 2: Implementation of key tasks of $m = 20$, $n = 50$, and $k = 20$.

	Brute force	NSGA-III	PSO	NSGA-III-greedy
Achieve goal	Yes	Yes	No	Yes

For task constraints,

$$ms_j \longrightarrow \{u_k \cdots u_l\}. \quad (4)$$

This means that task ms_j can only be executed by UAV $u_k \cdots u_l$:

$$\{ms_k \cdots ms_l\} \longrightarrow u_i. \quad (5)$$

This means that the function of UAV u_i can only execute $ms_k \cdots ms_l$.

(4) Objective function:

(a) Include all the key tasks in S_i:

$$\text{goal}_1 = S_i \in Q. \tag{6}$$

$Q = \{q_1 \cdots q_n\}$ is the final execution sequence of UAV, and $S_{i=x_1 \cdots x_k}$ is one of the key tasks. That means that the final UAV task allocation scheme contains all the key tasks.

(b) Maximize the number of tasks:

$$\text{goal}_2 = \max_{\forall D} \sum_{i=1}^{n} \{\pi_i \neq \text{NULL}\}. \tag{7}$$

(c) Maximize the mission revenue:

$$\text{goal}_3 = \max_{\forall D} \sum_{i=1}^{n} \{\pi_i \neq \text{NULL}\}. \tag{8}$$

(d) Minimize resource consumption:

$$\text{goal}_4 = \max_{\forall D} \sum_{i=1}^{m} \left(\text{move\_cost}(p_i^r, p_{i1}^s) + \sum_{j=2}^{|s_i|} \text{move\_cost} \cdot \left(p_{i(j-1)}^s, p_{ij}^s\right) + \sum_{j=1}^{|s_i|} \text{task\_cost}\left(s_{ij}\right) \right). \tag{9}$$

$\text{move\_time}(p_i^r, p_{i1}^s)$ indicates the movement time of UAV from the previous task to the current task. $\text{task\_time}(s_i, j-1)$ represents the time spent on the previous task of the current task.

(e) Minimize time cost:

$$\text{goal}_5 = \min_{\forall D} \max_{i=1}^{m} \text{time\_cost}(s_i, |s_i|). \tag{10}$$

3.1. Brute-Force Method

(1) Load UAV information, mission information, and key task information

(2) Solve all the permutations and combination situations of the tasks that each UAV can perform

(3) For each combination of UAVs in the front, the executable test is carried out, leaving the length of task sequences that UAVs can execute in each task sequence

Table 3: Target optimization of goal_2 to goal_5.

	Goal_2	Goal_3	Goal_4	Goal_5
Brute force	—	—	—	—
NSGA-III	0.18	0.14	0.28	0.71
PSO	0.56	0.47	0.08	0.70
NSGA-III-greedy	0.18	0.14	0.28	0.87

(4) Judge whether the key tasks assigned to the UAV are lost in the remaining tasks. If lost, delete the execution sequence; otherwise, keep it

(5) The existing task sequences of each UAV are arranged and combined to get the total task sequence

(6) The total sequence of tasks is then evaluated using the nondominated function

(7) Select the optimal task sequence and its objective function value to output

The pseudocode of the brute-force method is shown in Pseudocode 1.

3.2. Constrained Optimization Evolutionary Algorithm. After studying and testing a variety of constrained evolutionary algorithms, this paper chose the NSGA-III algorithm as the main algorithm of the comparative experiment.

NSGA-III [20] is a novel multiobjective optimization evolutionary algorithm based on fast nondominated sorting proposed by DEB in 2014. The algorithm can well deal with multiobjective optimization problems [21], and the effectiveness of the algorithm is verified by a multiobjective function. Moreover, from the known findings of researchers, the NSGA-III has a very good effect in solving the task allocation problem of multitarget UAV [22].

The main flow of the algorithm is as follows:

(1) Load UAV information, task information, and key task information

(2) The UAV information, task information, and key task information are encoded, and the key task is compulsorily encoded into the corresponding UAV executable task sequence

(3) The number of iterations t is set to 0

(4) Evaluate the initial population and calculate the individual fitness

(5) The offspring population was generated by crossover and mutation; then, individual fitness was calculated

(6) The parent population and the offspring population are combined, and then, nondominated sorting is performed

(7) Based on the reference point, n better individuals were selected; then, put them into the next-generation population

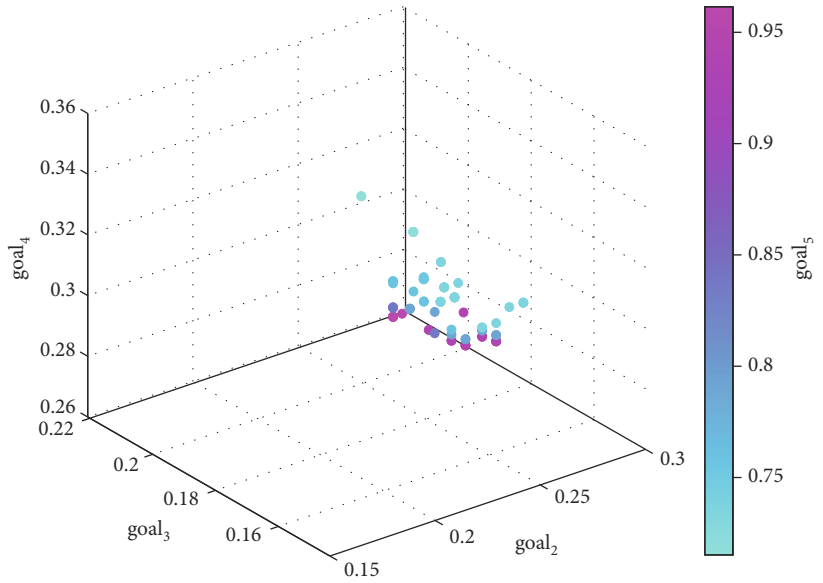

Figure 4: NSGA-III for $m = 20$, $n = 50$, and $k = 20$.

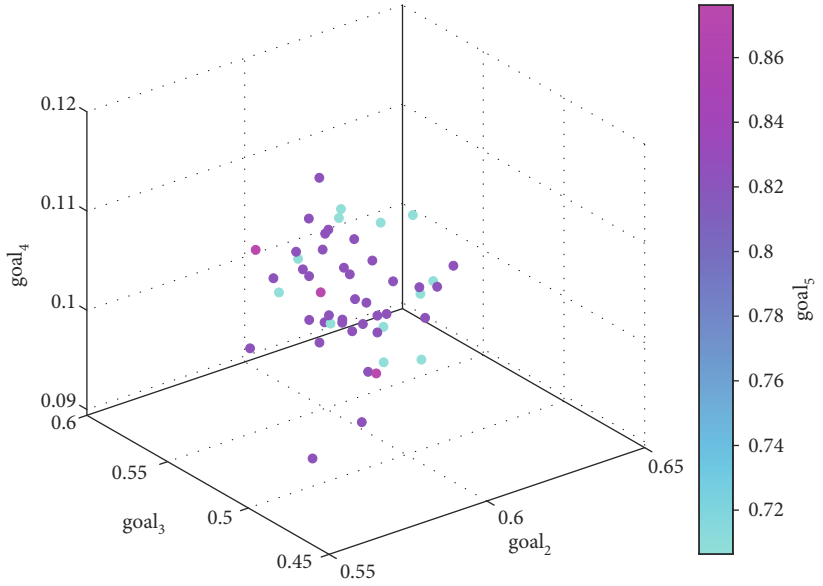

Figure 5: PSO for $m = 20$, $n = 50$, and $k = 20$.

(8) Repeat the operations of 4–7 until the number of iterations is reached

(9) After processing the results, the solution set containing all the key tasks is selected

The program flowchart is shown in Figure 3.

3.3. PSO Algorithm. Particle swarm optimization (PSO) was proposed by Dr. Eberhart and Dr. Kennedy in 1995 which is based on the study of birds' predation behavior. Its basic core is to make use of the information sharing of individuals in the group, so that the movement of the whole group will have an evolutionary process from disorder to order in the problem solving space, so as to obtain the optimal solution of the problem. Particle swarm optimization has the advantages of easy implementation, high accuracy, and fast convergence. This paper chooses the particle swarm optimization algorithm mainly because of its high precision and fast convergence, because in the real combat plan, people need to plan the task allocation scheme in the quickest time possible [4, 23].

The pseudocode of the PSO algorithm is shown in Pseudocode 2.

Speed update equation:

$$V_{id} = \omega V_{id} + C_1 \text{random}(0,1)(P_{id} - X_{id}) + C_2 \text{random}(0,1)\left(P_{gd} - X_{id}\right). \quad (11)$$

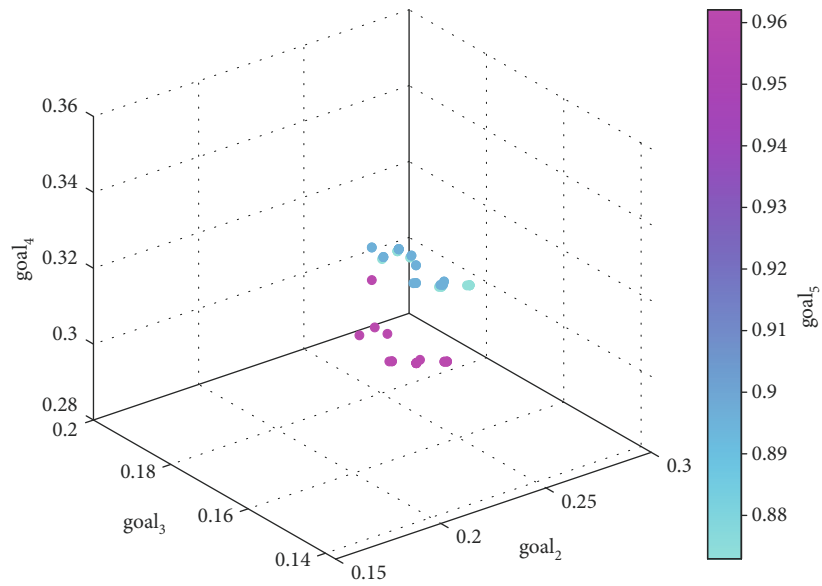

Figure 6: NSGA-III-greedy for $m = 20$, $n = 50$, and $k = 20$.

Table 4: Algorithm efficiency of $m = 50$, $n = 200$, and $k = 50$.

	Brute force	NSGA-III	PSO	NSGA-III-greedy
Times	1	20	20	20
Total time (s)	—	2821.77	1159.54	15217.41
Avg time (s)	—	141.10	57.98	760.87

Table 5: Implementation of key tasks of $m = 50$, $n = 200$, and $k = 50$.

	Brute force	NSGA-III	PSO	NSGA-III-greedy
Achieve goal	Yes	Yes	No	Yes

Position update equation:

$$X_{id} = X_{id} + V_{id}. \tag{12}$$

3.4. Constrained Optimization Evolutionary Algorithm with the Greedy Algorithm. Here, this paper still uses the NSGA-III constrained evolutionary algorithm. This paper adds the greedy algorithm, which can find the local fast optimization in the population evolution process of the NSGA-III algorithm. The purpose of this is to enrich the diversity of the population, avoid local deadlock while accelerating the speed of local optimization, optimize the global optimal solution, and obtain better UAV task allocation effect [4].

The pseudocode of the greedy strategy, where x_i is the execution sequence of agent u_i, is shown in Pseudocode 3.

4. Results and Discussion

This section will discuss the effect of the four schemes in the "limited" UAV task allocation scheme based on the event model by comparing the data of the four experiments and analyzing the experimental results.

Table 6: Target optimization of $goal_2$ to $goal_5$.

	$Goal_2$	$Goal_3$	$Goal_4$	$Goal_5$
Brute force	—	—	—	—
NSGA-III	0.46	0.28	0.31	0.94
PSO	0.74	0.75	0.11	0.86
NSGA-III-greedy	0.46	0.28	0.32	0.94

The experimental environment of this paper is as follows: Windows 10 operating system 64-bit professional edition, version 2004, processor Intel i7-4720hq, main frequency 2.60 GHz, memory 16 G, hard disk 512gssd, programming environment PyCharm 2020, and visual studio 2019.

For the fairness of experimental comparison, this paper uses the same test data for UAV, mission, and key task. First of all, this paper designs three scales of test data: 20 UAVs and 50 tasks, including 20 key tasks; 50 UAVs and 200 tasks, including 50 key tasks; and 100 UAVs and 400 tasks, including 100 key tasks. In terms of the number of key tasks, this paper's design is the same as that of UAVs. In reality, there are few key tasks. In order to test the effectiveness of the improved algorithm, this paper increased the number of key tasks to the same level as the number of UAVs and designed the initialization data of UAV and mission.

In terms of functional constraints, both key tasks and nonkey tasks may be executed by multiple UAVs; each UAV's task execution sequence can include both key tasks and nonkey tasks; there is no necessary sequence between nonkey tasks and key tasks.

4.1. Small-Scale Task Allocation Experiment. Under the small-scale UAV task allocation, the number of UAVs is assumed to be 20 and the total number of tasks is 50, including 20 key tasks. This paper will show and analyze the data from three aspects: the algorithm solution time, whether it

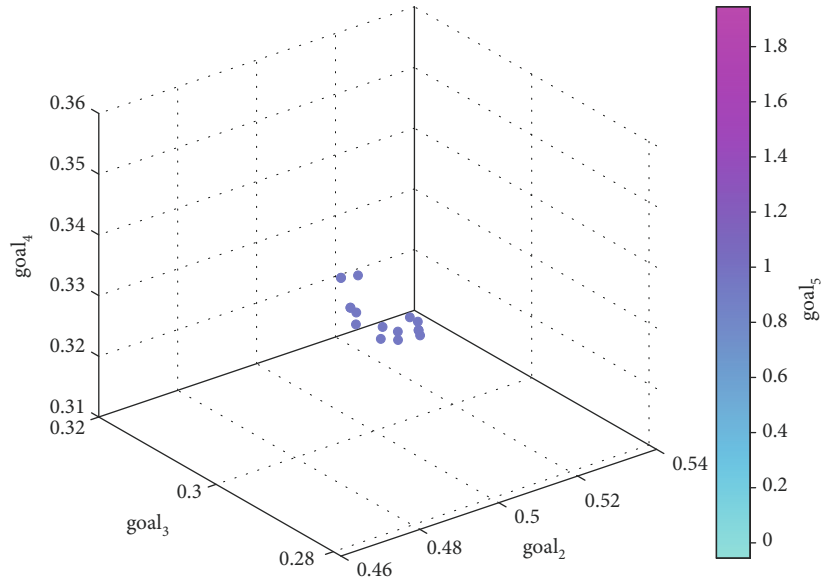

Figure 7: NSGA-III for $m = 50$, $n = 200$, and $k = 50$.

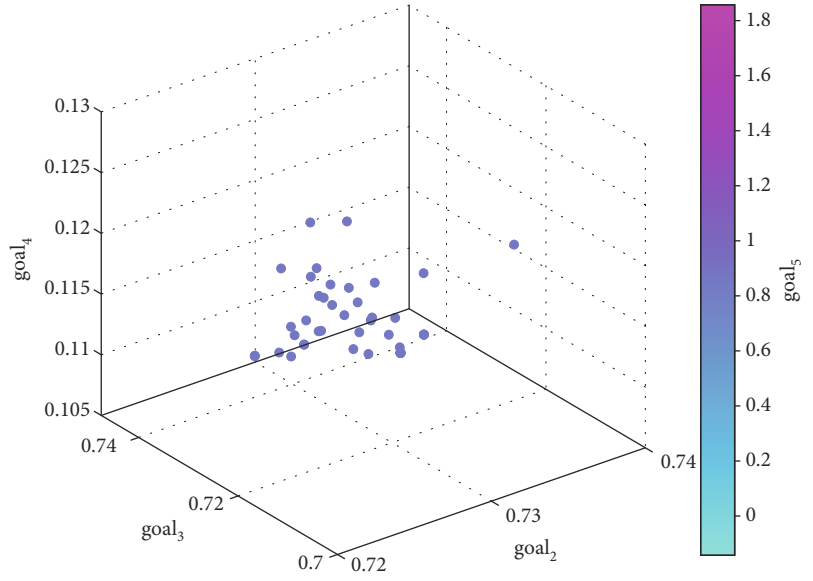

Figure 8: PSO for $m = 50$, $n = 200$, and $k = 50$.

can ensure the implementation of key tasks, and the final optimization effect of the objective function.

First, the time efficiency of the algorithm is shown in Table 1.

This paper finds that the time complexity of the brute-force method increases exponentially with the increase in the task scale. When the task scale reaches 20, 50, and 20, the computer for the experiment cannot solve the task planning scheme with the existing equipment. Obviously, the brute-force method can only be applied to the task allocation of small teams, but it is not applicable to the task allocation of large-scale cluster operations. This paper will focus on the other three options. From Table 1, the PSO algorithm has great advantages in solving efficiency.

Then, there is the ability to ensure the smooth progress of key tasks (see Table 2).

This paper finds that only in the particle swarm optimization algorithm, key tasks cannot be realized to be executed inevitably. This is because the particle swarm optimization algorithm is easy to fall into the local optimal situation in the process of optimization. Because this paper set more targets, the algorithm converges quickly after a target has better effect. In fact, it is also a trade-off between the optimization effect and optimization time of the particle swarm optimization algorithm.

Finally, the four goals of $goal_2$-$goal_5$ are compared. Table 3 shows the comparison of optimal solutions, and Figures 4–6 show the solution set distribution.

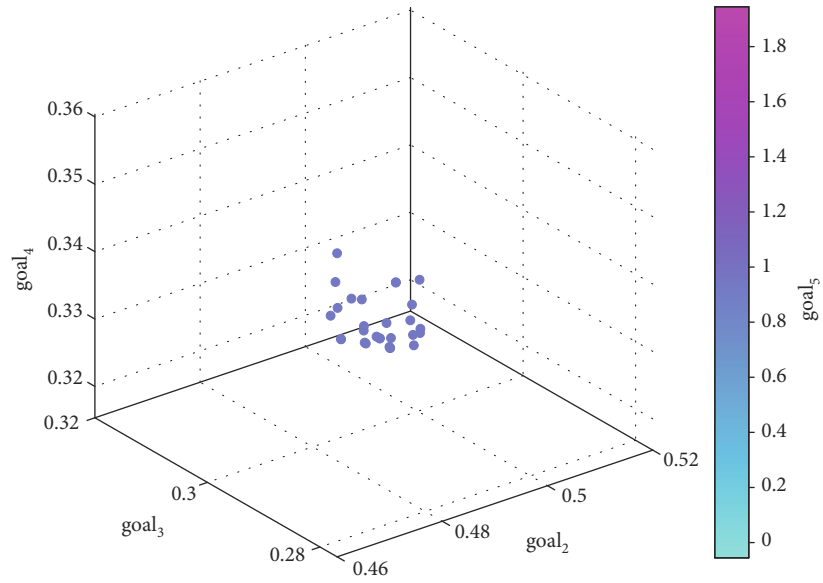

Figure 9: NSGA-III-greedy for $m = 50$, $n = 200$, and $k = 50$.

Table 7: Algorithm efficiency of $m = 200$, $n = 1000$, and $k = 200$.

	Brute force	NSGA-III	PSO	NSGA-III-greedy
Times	1	20	20	5
Total time (s)	—	4904.00	1501.91	21532.62
Avg time (s)	—	245.20	75.10	4306.52

Table 8: Implementation of key tasks of $m = 200$, $n = 1000$, and $k = 200$.

	Brute force	NSGA-III	PSO	NSGA-III-greedy
Achieve goal	Yes	Yes	No	Yes

4.2. Middle-Scale Task Allocation Experiment. Under the middle-scale UAV task allocation, the number of UAVs is assumed to be 50 and the total number of tasks is 200, including 50 key tasks. This paper will show and analyze the data from three aspects: the algorithm solution time, whether it can ensure the smooth implementation of key tasks, and the final optimization effect of the objective function.

First, the time efficiency of the algorithm is shown in Table 4.

It can be seen from Table 4 that the PSO algorithm still has the fastest solution speed, followed by NSGA-III. After adding the greedy algorithm, the speed of NSGA-III is greatly reduced with the increase in the problem size.

Then, there is the ability to ensure the smooth progress of key tasks (see Table 5).

Here, like the small-scale problem, in four cases, the inevitable execution of key tasks cannot be realized only in the particle swarm optimization algorithm.

Finally, the four goals are compared. Table 6 shows the comparison of optimal solutions, and Figures 7–9 show the solution set distribution.

Table 9: Target optimization of $goal_2$ to $goal_5$.

	$Goal_2$	$Goal_3$	$Goal_4$	$Goal_5$
Brute force	—	—	—	—
NSGA-III	0.43	0.29	0.36	0.89
PSO	0.68	0.63	0.12	0.79
NSGA-III-greedy	0.43	0.28	0.33	0.93

4.3. Large-Scale Task Allocation Experiment. Under the large-scale UAV task allocation, the number of UAVs is assumed to be 200 and the total number of tasks is 1000, including 200 key tasks. This paper will show and analyze the data from three aspects: the algorithm solution time, whether it can ensure the smooth implementation of key tasks, and the final optimization effect of the objective function.

First, the time efficiency of the algorithm is shown in Table 7.

The PSO algorithm still maintains excellent efficiency in solving large-scale problems. The efficiency of NSGA-III with the greedy algorithm is further reduced.

The next part is whether the smooth progress of key tasks can be ensured (see Table 8).

The four cases here are the same as the previous two cases; only in the particle swarm optimization algorithm, the time key task cannot necessarily be executed.

Finally, the four goals are compared. Table 9 shows the comparison of optimal solutions, and Figures 10–12 show the solution set distribution.

From the above several different scale experiments, the four methods show different characteristics and advantages. The brute-force algorithm needs to enumerate all the possibilities, which leads to slow efficiency. But the brute-force algorithm can get the best result. The NSGA-III algorithm needs a lot of crossover and mutation operations in the iterative process, and its efficiency is not the best. But the NSGA-III has an excellent balance. Algorithm efficiency and effect

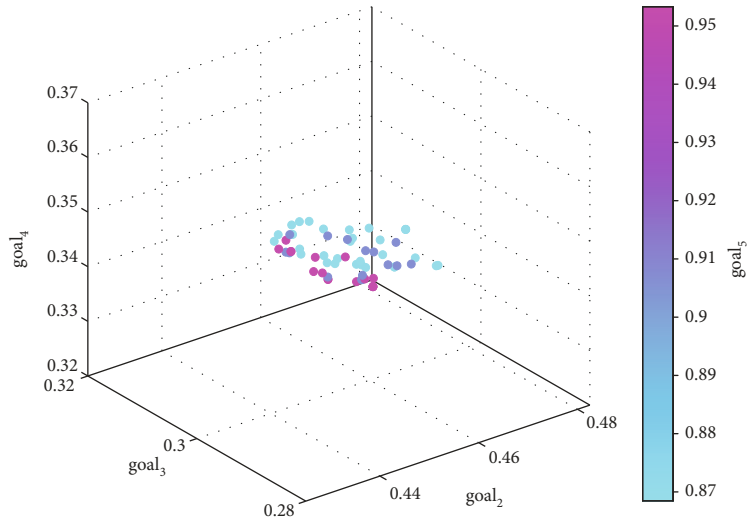

Figure 10: NSGA-III for $m = 200$, $n = 1000$, and $k = 200$.

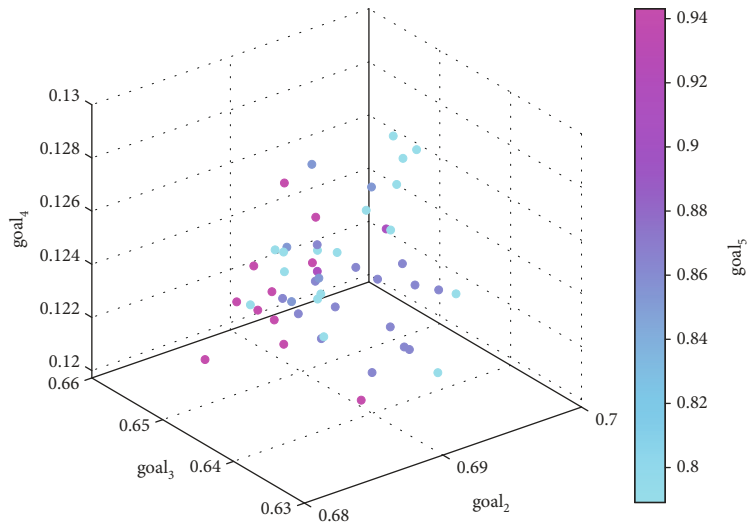

Figure 11: PSO for $m = 200$, $n = 1000$, and $k = 200$.

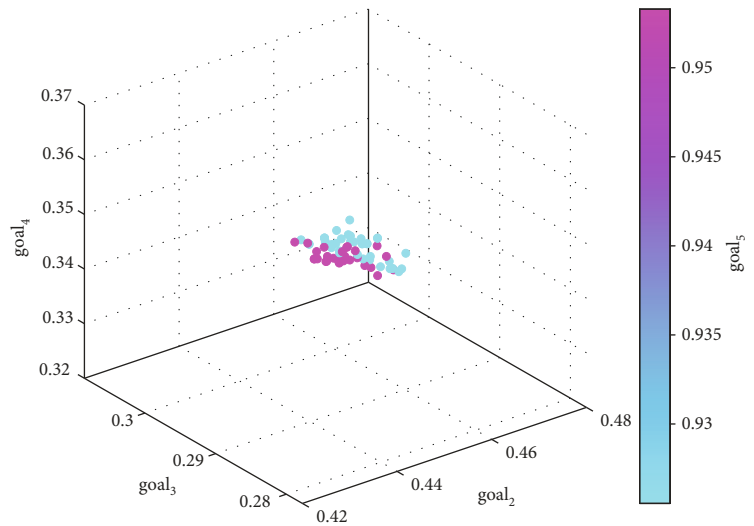

Figure 12: NSGA-III-greedy for $m = 200$, $n = 1000$, and $k = 200$.

have excellent performance. In the process of iteration, the PSO algorithm needs few operations, and the iteration efficiency is very fast. PSO has a local optimal problem, which leads to poor results. After adding the greedy algorithm, the population diversity of NSGA-III is improved. At the same time, due to the increase in computation, the efficiency of the algorithm is reduced.

5. Conclusions

In order to solve the problem of "limited" task allocation in UAV combat, four existing algorithms are improved and optimized in this paper to ensure that the nonkey tasks are optimized to maximize the revenue when the key tasks are executed. The brute-force method, because its solution time increases exponentially with the increase or decrease in task size, is only suitable for small-scale task allocation and can achieve the optimal effect; NSGA-III, an evolutionary algorithm for multiobjective optimization, has a very good effect in solving such problems and has a good balance between solution efficiency and solving effect; PSO is suitable for solving different scale task allocation problems, and it has the fastest solving efficiency, but because the algorithm itself is easy to fall into the local optima, it often leads to premature convergence of the algorithm. In the final, the effect of the constrained optimization evolutionary algorithm with the greedy algorithm is not very obvious, but with the increase in the size of the problem, the optimization effect begins to show gradually. In the actual UAV combat task allocation problems, users can choose different algorithms with different problem sizes and constraints, which can give full play to the advantages of each algorithm to adapt to different task allocation requirements.

In the future work, we intend to focus on optimizing the constrained evolutionary algorithm, because its performance is excellent in solving the constrained task allocation in the existing work. In the future, we may try to enrich the diversity of its population, improve its global search ability, and avoid the local optimal problem, so that the algorithm can achieve better performance. At the same time, using more efficient algorithms to solve the existing problems will also be our work direction.

Data Availability

All the data can be generated according to the steps described in our paper, and readers can also ask for the data by contacting cxj_dna@yeah.net.

Conflicts of Interest

There is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work is partially supported by the Natural Science Foundation of China (71701208).

References

[1] J. Bellingham, M. Tillerson, A. Richards, and J. P. How, "Multi-task allocation and path planning for cooperating UAVs," in *Cooperative Control: Models, Applications and Algorithms*, pp. 23–41, Springer, Boston, MA, 2003.

[2] W. R. Van Hage, V. Malaisé, R. Segers, L. Hollink, and G. Schreiber, "Design and use of the simple event model (SEM)," *Journal of Web Semantics*, vol. 9, no. 2, pp. 128–136, 2011.

[3] Y. Eun and H. Bang, "Cooperative task assignment/path planning of multiple unmanned aerial vehicles using genetic algorithm," *Journal of Aircraft*, vol. 46, no. 1, pp. 338–343, 2009.

[4] A. Salman, I. Ahmad, and S. Al-Madani, "Particle swarm optimization for task assignment problem," *Microprocessors and Microsystems*, vol. 26, no. 8, pp. 363–371, 2002.

[5] S. Fei, C. Yan, and S. H. Lin-Cheng, "UAV cooperative multitask assignment based on ant colony algorithm," *Acta Aeronautica et Astronautica Sinica*, vol. 29, no. 5, pp. 188-189, 2008.

[6] G. A. Mills-Tettey, A. Stentz, and M. B. Dias, *The Dynamic Hungarian Algorithm for the Assignment Problem with Changing Costs*, Robotics Institute, Pittsburgh, PA, 2007.

[7] Z. Fang, J. Wang, C. Jiang, Q. Zhang, and Y. Ren, "AoI inspired collaborative information collection for AUV assisted Internet of underwater things," *IEEE Internet of Things Journal*, p. 1, 2021.

[8] Z. Fang, J. Wang, C. Jiang, B. Zhang, C. Qin, and Y. Ren, "QLACO: Q-learning aided ant colony routing protocol for underwater acoustic sensor networks," in *2020 IEEE wireless communications and networking conference (WCNC)*, Seoul, Korea (South), 2020.

[9] D. Zhu, X. Cao, B. Sun, and C. Luo, "Biologically inspired self-organizing map applied to task assignment and path planning of an AUV system," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 10, no. 2, pp. 304–313, 2017.

[10] J. C. Amorim, V. Alves, and E. P. de Freitas, "Assessing a swarm-GAP based solution for the task allocation problem in dynamic scenarios," *Expert Systems with Applications*, vol. 152, article 113437, 2020.

[11] A. Shamsoshoara, M. Khaledi, F. Afghah, A. Razi, J. Ashdown, and K. Turck, "A solution for dynamic spectrum management in mission-critical UAV networks," in *2019 16th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, Boston, MA, USA, 2019.

[12] Z. Cai and Y. Wang, "A multiobjective optimization-based evolutionary algorithm for constrained optimization," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 658–675, 2006.

[13] J. Edmonds, "Matroids and the greedy algorithm," *Mathematical Programming*, vol. 1, no. 1, pp. 127–136, 1971.

[14] J. Zhou, X. Zhao, D. Zhao, and Z. Lin, "Task allocation in multi-agent systems using many-objective evolutionary algorithm NSGA-III," in *International Conference on Machine Learning and Intelligent Communications*, Springer, Cham, 2019.

[15] D. R. Prescott, J. D. Andrews, and C. G. Downes, "Multiplatform phased mission reliability modelling for mission planning," *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, vol. 223, no. 1, pp. 27–39, 2009.

[16] J. D. Andrews, J. Poole, and W.-H. Chen, "Fast mission reliability prediction for unmanned aerial vehicles," *Reliability Engineering & System Safety*, vol. 120, pp. 3–9, 2013.

[17] R. Remenyte-Prescott, J. D. Andrews, and P. W. H. Chung, "An efficient phased mission reliability analysis for autonomous vehicles," *Reliability Engineering & System Safety*, vol. 95, no. 3, pp. 226–235, 2010.

[18] S.-i. Minato, N. Ishiura, and S. Yajima, "Shared binary decision diagram with attributed edges for efficient Boolean function manipulation," in *27th ACM/IEEE Design Automation Conference*, Orlando, USA, 1990.

[19] M. A. Friedl and C. E. Brodley, "Decision tree classification of land cover from remotely sensed data," *Remote Sensing of Environment*, vol. 61, no. 3, pp. 399–409, 1997.

[20] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, 2014.

[21] W. Mkaouer, M. Kessentini, A. Shaout et al., "Many-objective software remodularization using NSGA-III," *ACM Transactions on Software Engineering and Methodology*, vol. 24, no. 3, pp. 1–45, 2015.

[22] H. Qiu and H. Duan, "A multi-objective pigeon-inspired optimization approach to UAV distributed flocking among obstacles," *Information Sciences*, vol. 509, pp. 515–529, 2020.

[23] J. Sánchez-García, D. G. Reina, and S. L. Toral, "A distributed PSO-based exploration algorithm for a UAV network assisting a disaster scenario," *Future Generation Computer Systems*, vol. 90, pp. 129–148, 2019.