Hindawi
Complexity
Volume 2020, Article ID 6747985, 14 pages
https://doi.org/10.1155/2020/6747985

WILEY | Hindawi

Research Article

NECTAR-An Agent-Based Dynamic Task Allocation Algorithm in the UAV Swarm

Chao Chen, Weidong Bao, Tong Men, Xiaomin Zhu, Ji Wang, and Rui Wang

College of Systems Engineering, National University of Defense Technology, Changsha 410073, Hunan, China

Correspondence should be addressed to Weidong Bao; wdbao@nudt.edu.cn

Received 2 August 2020; Accepted 29 August 2020; Published 16 September 2020

Academic Editor: Shi Cheng

The advancement of UAV technology makes the use of UAVs more and more widespread, and the swarm is the main mode of UAV applications owing to its robustness and adaptability. Meanwhile, task allocation plays an essential role in a swarm to obtain overall high performance and unleash the potential of each UAVs owing to the complexity of the large-scale swarm. In this paper, we pay attention to the real-time allocation problem of dynamic tasks. We design models for the task assigning problem to construct the constraints model and assigning objectives. In addition, we introduce a novel agent-based allocating mechanism based on the auction process, including the design for three kinds of agents and the cooperation mechanism among different agents. Moreover, we proposed a new algorithm to calculate the bidding values of UAVs, by which the messages passed between UAVs can be reduced. On the basis of the assigning mechanism, we put up with a novel agent-based real-time task allocation algorithm named NECTAR for dynamic tasks in the UAV swarm. Furthermore, we conduct extensive experiments to evaluate the performance of our NECTAR, and the results indicate that NECTAR is able to solve the real-time task allocation for dynamic tasks and achieve high performance of the UAV swarm.

1. Introduction

With the rapid development of artificial intelligence, high-end manufacturing, and robotics, the performance of Unmanned Aerial Vehicles (UAVs) is getting better and better. At present, UAVs have replaced humans in various complex situations, e.g., public security investigation, environmental protection detection, terrain exploration, and transmission line maintenance [1–3]. In addition, owing to UAVs' good stealth performance, strong autonomy, and reusability, UAVs are also widely used in battlefield environments by various countries. The main application mode of UAVs is the swarm, in which the UAVs would collaborate with each other to achieve operational performance.

It should be noted that the tasks undertaken by UAVs usually have strict time constraints in which the tasks' success depends not only on the execution but also on the time instants at which the execution finished. Therefore, the tasks should be assigned to suitable UAVs of the swarm as soon as possible to enhance their success probability. Consequently, there come two critical issues, one is how to find a suitable UAV in the swarm to avoid resource wasting, and the other is how to shorten the response time to promote the efficiency of the swarm.

Due to the characteristics of tasks, the allocation method plays an essential role in obtaining high performance for the swarm. Only by realizing reasonable task allocation can the cost of completing tasks be as small as possible, and coordination between different UAVs can be formed to achieve higher performance of the swarm. Besides, task allocation must try to save communication bandwidth as the communication capabilities of UAVs are usually limited [4]. To date, lots of allocation algorithms for UAVs have been studied. However, these algorithms do not sufficiently consider the characteristics of the UAV swarm, such as requirements of real-time response for dynamic tasks, low-bandwidth communication environment, and continuous task response.

In order to select proper UAVs from the UAV swarm to undertake tasks and make the response time short, we

comprehensively consider the characteristics of the UAV swarm, designing a novel agent-based assigning mechanism and, then, developing the auction-based task allocation algorithm for dynamic tasks.

The main contributions of our work can be summarized as follows:

(i) We designed models for the task-assigning problem in the UAV swarm and constructed the assigning constraints and objectives.

(ii) We propose an agent-based allocation mechanism based on the auction process. In addition, a novel calculation method of the bidding value was developed to reduce communication.

(iii) We investigate three selection strategies, the M strategy, P strategy, and D strategy, to determine the winner bidder.

(iv) We put up with a novel intelligent task allocation method, named NECTAR, for dynamic tasks in the UAV swarm.

The rest of the paper is organized as follows. We summarized the related works in Section 2. The problem description and model are given in Section 3. Section 4 is the designing of the agent-based mechanism. Also, we would give a detailed illustration of the NECTAR algorithm in Section 5. The experimental evaluation and analysis would be presented in Section 6. In Section 7, we concluded the paper with a summary and future work.

2. Related Work

Generally speaking, the models for task allocation problem for UAVs are mainly modeled into the following three categories: the mixed-integer linear programming model [5], dynamic network flow optimization model [6], and multiagent system [7]. These models have their own advantages and adaptation scenarios, but in general, the agent-based model has wider adaptability and better performance. Consequently, we choose the multiagent system to be the basic model. As for the solutions to the problem, many scholars have made attempts. We conclude them into two categories: the centralized task allocation method and distributed task allocation method.

Centralized solution schemes mainly include heuristic methods, such as genetic algorithm, particle swarm algorithm, simulated annealing algorithm, ant colony algorithm, and so on. Zhu et al. [8] focus on the problem of heterogeneous target assigning and UAVs' task sequence optimization, and they use a genetic algorithm to minimize the task execution time and UAVs' total consumption. As discussed in [9], the authors designed an improved particle swarm optimization method based on standard particle swarm optimization and evolutionary game theory, and they adopted a novel self-adaptive strategy in the task allocation process; the results show their advantages over other peers. Another heuristic method to solve the task allocation problem is the ant colony algorithm [10]; the authors considered the features such as being highly nonlinear, dynamic, highly adversarial, and multimodal, and they proposed a dynamic ant colony's labor division model based on the classic fixed response threshold model to finish task assignment. The simulation results identified its effectiveness in self-organization, flexibility, and real-time response to dynamic environments. Chun et al. [11] focused on a scenario where multiple air vehicles are required to prosecute geographically dispersed targets, aiming at solving the multirobot task allocation problems, and they developed an optimal task assignment algorithm based on a mixed-integer linear programming model. Also, the existence of the solution can be guaranteed. Other typical centralized algorithms such as wolf pack algorithm [12] and search algorithm [13] also show effectiveness in assigning tasks for multirobot systems. Owing to the fact that centralized methods have global information, they can obtain an ideal solution when dealing with task allocation problems in small-scale swarms. However, when the scale is large enough, it is also easy to be inefficient and difficult to obtain solution results quickly, and the reason is that the task assignment problem is an NP-hard problem. At the same time, since the centralized method is based on the status information of UAVs, communication bandwidth is required to meet the requirements of obtaining the real-time status of the UAV. However, the bandwidth of UAV swarms is one of the biggest limitations in current swarm applications. In addition, the centralized method also relies heavily on the central node, which makes this method not robust and adaptable.

When the scale of swarms becomes larger and the requirements for flexibility become higher and higher, the advantages of distributed methods are becoming more prominent. Thus, decentralized algorithms are receiving increasing attention. For example, Mehdi and Jonathan [14] improved the basic implicit coordination approach to finish the task allocation process, but the method is based on data synchronization among UAVs. However, synchronization is a difficult problem for swarms with limited communication bandwidth. Later, this work was extended by Whitten et al., and they considered coupled constraints to address complex mission characteristics in the allocation process. Besides, they designed the notion of pessimistic or optimistic bidding strategies and the relative timing constraints between tasks, and the results showed that it provided measurable performance improvement to the baseline CBBA [15]. As discussed in [16], the authors take limited bandwidth into consideration, they extended CBBA by duplicating cooperative tasks to modify the task list and added a judgment mechanism, and the simulation presented that their algorithm can achieve a better conflict-free allocation with fewer communications. In addition, Yang et al. [17] applied reinforcement learning in the task allocation for UAVs, they designed a networking scheme based on the expansion strategy, based on which the UAVs can improve the assigning through autonomous learning, and they conducted experiments on the UAV swarm to obtain optimized assigning in specific cases. Besides, many other algorithms such as sequential auctions [18], the human-agent collaboration method [19], and the bid-based grouping method

[20] are used. Though these distributed algorithms have solved the task allocation problem in UAVs, the communication limitation, real-time response, and continuous allocation are not sufficiently considered, making these methods not able to adapt to the actual environment.

In this research, we put up with a novel agent-based task allocation algorithm based on an improved auction mechanism to achieve real-time task allocation for dynamic tasks in the UAV swarm. The proposed agent-based assigning mechanism can quickly respond to the dynamic tasks with little communication, and it is able to continuously assign because the agents collaborate with each other based on their local status.

3. Problem Depictions

To study the dynamic task assignment problem, we mainly consider three kinds of UAV tasks, which are reconnaissance [21], striking [22], and damage assessment [23]. For dynamic tasks, they would arrive aperiodically, and the arrival time of them is unknown in advance. Besides, the tasks should be assigned as soon as possible owing to its deadline and the efficiency of the UAV swarm. Therefore, tasks should be assigned one by one rather than in a batch mode. Another fact that should not be ignored is that all UAVs would take off at the base station before its execution of tasks, and it should return to the base station before reaching its endurance time.

3.1. Models and Notations. In this paper, we mainly focus on the distributed task allocation method for the UAV swarm. Also, the tasks considered are uncoupled, and the reason is that coupled tasks can be decomposed by commanders into uncoupled tasks with other constraints such as time sequence and interaction order. Other features the tasks have include being nonpreemptive, deadline-sensitive, and aperiodic. We use a set $T = \{t_1, t_2, \ldots\}$ to represent the dynamic tasks. It should be noted that the count of tasks is not known because the tasks are appearing continuously as the confrontation progresses. For each task $t_i \in T$, it can be modeled as $t_i = \{ty_i, ta_i, td_i, tp_i, ts_i\}$, of which ty_i, ta_i, td_i, tp_i, ts_i denote t_i' type, arrival time, deadline time, position, and task supplementary information, respectively. Note that task type ty_i can be reconnaissance task, striking task, or damage assessment task.

We consider a set $U = \{u_1, u_2, \ldots, u_m\}$ of UAVs that have different capabilities. A UAV u_j of the swarm can be modeled by a collection of parameters, i.e., $u_j = \{uv_j, ut_j, ud_j, ltf_j, ltp_j, rsr_j, tft_j\}$, where uv_j, ut_j, ud_j, ltf_j, ltp_j, rsr_j, and tft_j are the u_j' velocity, ability type, duration time, finish time of the last execution task, position of the last execution task, remained striking resource, and total flying time, respectively. It is worth noting that each UAV has one or two kinds of ability among ability collection {*reconnaissance, striking, damage assessment*}. Besides, tft_j is the time from its take-off to its landing at the base station.

In Table 1, we summarize the main notations of this study for future reference. It should be noted that ltf_j and ltp_j represent the finish time and position of the last task in its task list. If a new task t_n has been assigned to u_j before the existing last task's completion, then the value of ltp_j would change to tp_n, and the value of ltf_j changes to f_{nj}. However, on the condition that no tasks have been allocated to u_j before the existing last task's completion, the position of u_j would be kept constant, which is because the UAV would keep hovering to save energy. Besides, we use rsr_j to denote the remained striking resource, which is used for task assignment. When the value of rsr_j equals to 0, it means UAV u_j can no longer undertake any striking tasks any more.

Let pt_{ij} be the preparation time of UAV u_j to execute task t_i, which refers to the flying time from u_j's last execution position to t_i' position, and the last execution position of u_j would be changed to t_i' position if t_i has been assigned to u_j. In addition, sp_{ij} and se_{ij} are used to denote the preparation start time and execution start time. Also, $U(t_i)$ refers to the UAV which undertakes task t_i. The allocation matrix $X = \{x_{ij}\}$ denotes the assigning relations between tasks and UAVs. In addiyion, the value of element x_{ij} should be "1" or "0", which means task t_i is allocated to UAV u_j or not.

3.2. Constraints. As is assumed before, the tasks discussed in this paper are not splittable, and a task can only be assigned to one UAV. On the other hand, different tasks usually have different position requirements; thus, each UAV can only undertake one task at any time instant. Therefore, we can conclude the constraint C_1 as follows:

$$C_1: \begin{cases} \sum_{j=1}^{m} x_{ij} = 0 \text{ or } 1, \quad i \in [1, 2, \cdots], \\ x_{ij} \cdot x_{kj} = 0, \quad \text{if } [sp_{ij}, f_{ij}] \cap [sp_{kj}, f_{kj}] \neq \varnothing. \end{cases} \tag{1}$$

For the preparation start time sp_{ij}, execution start time se_{ij}, and finish time f_{ij}, they are determined by the t_i's arrival time and finish time of u_j's last task. As is depicted in Figure 1, the preparation start time sp_{ij} can be calculated as follows:

$$sp_{ij} = \max\{ta_i + tb_i,\ f_{kj}\}, \tag{2}$$

where tb_i represents the time spent for the auction process. Furthermore, the execution start time se_{ij} can be determined by formula (3), and the finish time f_{ij} can be calculated by (4):

$$se_{ij} = sp_{ij} + pt_{ij}, \tag{3}$$

$$f_{ij} = se_{ij} + e_{ij}. \tag{4}$$

Whether a task t_i can be assigned to a certain UAV u_j or not is mainly determined by time constraint and ability constraint, which can be concluded as C_2:

$$C_2: \begin{cases} f_{ij} \leq td_i, & \text{if } x_{ij} = 1, \\ \Phi(t_i) \in ut_j, & \text{if } x_{ij} = 1, \end{cases} \tag{5}$$

where $\Phi(t_i)$ means the required ability of task t_i. Time constraint mainly refers to the fact that the finish time f_{ij} must satisfy t_i' deadline. Ability constraint mainly indicates that the UAV must have the ability to process task t_i.

Table 1: Definition of main notations.

Notation	Definition
t_i	The i^{th} task in task set T
ty_i, ta_i, td_i, tp_i, and ts_i	Task t_i' type, arrival time, deadline time, position, and task supplementary information
u_j	The j^{th} UAV in UAV set U
uv_j, ut_j, ud_j, ltf_j, ltp_j, rsr_j, and tft_j	UAV u_j' velocity, ability type, duration time, finish time of the last execution task, position of the last execution task, remained striking resource, and total flying time
pt_{ij}	The preparation time of UAV u_j to execute task t_i
sp_{ij} and se_{ij}	The preparation start time and execution start time of task t_i on UAV u_j
e_{ij} and f_{ij}	The execution time and finish time of t_i on u_j
$U(t_i)$	The UAV to which t_i is assigned
x_{ij}	x_{ij} is "1" if t_i is assigned to u_j; otherwise, x_{ij} is "0"

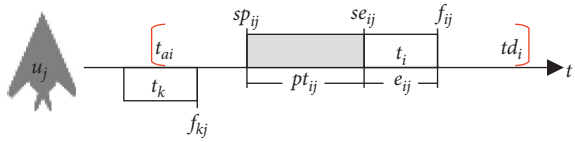

Figure 1: The task assigning parameters.

Another constraint comes to the striking tasks, and they need the remained striking resource to be not null at the time of task assignment. Also, the constraint is modeled as

$$C_3: \begin{cases} x_{ij} = 0, & \text{if } t_i = \text{striking}, rsr_j = 0, \\ x_{ij} = 1 \text{ or } 0, & if\ t_i = \text{striking}, rsr_j \neq 0, \end{cases} \tag{6}$$

and note that such constraint is only considered in t_i's assigning process because rsr_j may be equal to 0 after the task's execution.

3.3. Assigning Objectives. For the UAV swarm, it should complete as more dynamic tasks as possible to realize swarm efficiency. Besides, each UAV should try to increase the proportion of execution time in the total flying time as much as possible to realize a single UAV's efficiency. Consequently, we take the task success ratio and execution time ratio as two main assigning objectives.

Task success ratio:

$$\max\left\{\frac{\sum_{i=1}^{|T|}\sum_{j=1}^{m} x_{ij}}{|T|}\right\}. \tag{7}$$

Execution time ratio:

$$\max\left\{\frac{\sum_{i=1}^{|T|}\sum_{j=1}^{m} x_{ij}\cdot e_{ij}}{\sum_{j=1}^{m} tft_j}\right\}. \tag{8}$$

The abovementioned equations indicate that our agent-based assignment method aims at allocating more tasks to UAVs for successful execution within complex constraints and realizes higher UAV utilization.

4. Agent-Based Assigning Mechanism Design

On the designing of the agent-based assigning method for the UAV swarm, we mainly try to employ an auction mechanism [24] to complete task allocation. The auction mechanism allows agents to cooperate to finish the task assignment process in a distributed way, which makes it ideal for the problem of UAV task allocation. Note that, although the bidding process is managed by the central management agent, there is no central control [25]. More importantly, we devise the calculation of bidding values to local agents obeying global uniform rules, by which reducing the communication load among agents.

4.1. Agent Design. To realize agent-based task assigning, three kinds of agents are designed, i.e., the manager agent, task agent, and UAV agent. Each of them works based on its own rules, and they would cooperate with each other to finish the auction process. We use a triple tuple $<M^A, T^A, U^A>$ to represent the agents used in this paper.

(i) M^A represents the manager agent. The manager agent is yield with the construction of the swarm and would exist all the time. In addition, the manager agent is responsible for the management of all task agents and UAV agents.

(ii) $T^A = \{t_1^A, t_2^A, \ldots\}$ is a task agent set, and t_1^A denotes the i^{th} task agent in T^A. A task agent is mainly responsible for the auction and execution of the task. For each task agent, it would be constructed with the arrival of its corresponding task and die out with the finish. Note that t_1^A locates on a UAV which is responsible for the task t_i, and the system resources occupied by t_1^A would be reclaimed by the UAV after the finish of t_i. As for which UAV t_1^A is located, it depends on how the mission appears. If the task is discovered by a UAV u_j and approved by the commander, t_1^A would locate on u_j; if the task is discovered by the commander and assigned to the UAV u_k, t_1^A would locate on u_k.

(iii) $U^A = \{u_1^A, u_2^A, \ldots, u_m^A\}$ is a UAV agent set, and u_j^A represents the j^{th} UAV agent in U^A. A UAV agent is responsible for the management of UAV resources and bidding for tasks. For each UAV agent, it would exist all the time with the related UAV. Besides, each

UAV agent would update its status information and report to the manager agent in the following conditions: (1) after the construction of the UAV agent; (2) after a striking task has been assigned to it; and (3) encountering an abnormal situation.

4.2. Design of the Auction Mechanism. In the auction process, there are mainly two kinds of roles, i.e., announcers and bidders. The work of announcers would be mainly undertaken by task agents, and that of bidders would be mainly finished by UAV agents. Consequently, we would focus on the interactions among different agents to illustrate the auction mechanism.

Considering the problem of task assigning in the UAV swarm, the key point is to allocate each arriving task to a proper UAV. In this work, we use an auction mechanism to determine which UAV would be the proper one for a certain task. As is the case in auction activities, the first step is to analyze task requirements and select candidate UAVs, and we conclude these works into the preparation stage. The following stages are the announcing, bidding, awarding, and execution stage. Figure 2 depicts the basic activities and interactions of the agents, and we would give a more detailed illustration for the auction mechanism as follows.

4.2.1. Preparation Stage. In this stage, there are mainly three types of work. The first is that UAV agents report basic capacity information to the manager agent when coming to the conditions mentioned in the Agent Design part. The second is that the task agent sends basic information to the manager agent when the task agent is generated. The third is that the manager agent finishes the matching process and sends information of candidate UAVs to the task agents.

4.2.2. Announcing Stage. The main work of this stage is announcing to candidate UAVs the task agent would receive the information of candidate UAVs and send basic task information to these matched UAVs for the invitation. As for UAV agents, they would receive the bidding invitation and task information.

4.2.3. Bidding Stage. In this stage, each UAV agent would calculate the bidding value based on its status parameters (including position, earliest finish time, and duration time) and task information. Also, the calculation should obey the predefined common rules. After getting the bidding value, UAV agents would send the values to the task agent for bidding. It should be noted that the bidding process does not conflict with the task execution, which means that a UAV can bid to a new task when executing an assigned task. The reason is the announcing task is to be executed in the future time and has no influence on the current execution process.

4.2.4. Awarding Stage. After the task agent has received bidding values from bidders, it would select a proper UAV to award the contract according to predefined selection strategies. Also, the complete task information would be sent by the task agent to the winner UAV agent.

4.2.5. Execution Stage. In this stage, the winner UAV would execute the task and report the result to the task agent after the execution.

In Figure 3, we give an example of assigning a task to a proper UAV based on the auction mechanism. In this example, task t_i is to be assigned, and we assume that there are three UAV agents: u_1^A, u_2^A, and u_3^A. Task agent t_i^A firstly sends its basic task information to the manager agent M^A. Then, M^A would select UAVs that satisfy t_i's requirements to be the matched UAVs, and u_1, u_2, u_3 are the selected ones in this example. After the matching process, M^A sends the UAVs' information to t_i^A. Thirdly, t_i^A sends bidding invitation and necessary task information to u_1^A, u_2^A, and u_3^A, and the UAVs would calculate the bidding values about t_i on their local devices. Next, the three UAVs bid to t_i by sending their own bidding values to t_i^A. After receiving the bidding values, t_i^A selects u_2^A as the winner UAV based on bidding values and selection strategy. Finally, t_i^A sends the complete task information to u_2^A and awarding to it. After the task assigning, task execution and reporting would be performed by u_2^A.

4.3. Bidding Value. In the auction mechanism mentioned before, the task agent selects a proper UAV based on the bidding values. Consequently, the calculation of bidding values is significant to the efficiency. In the bidding process, whether a candidate UAV can satisfy the deadline is the most important factor, and we describe this effect by a step function.

$$\varphi\left(td_i - f_{ij}\right) = \begin{cases} 1, & \text{if } td_i - f_{ij} > 0, \\ 0, & \text{if } td_i - f_{ij} \le 0. \end{cases} \tag{9}$$

Another factor that should be taken into consideration is the execution start time, and it should be as early as possible. Besides, the preparation time would be better to be short. In addition, it would be better to have a short execution time. Therefore, we calculate the bidding values of UAV u_j to task t_i by the following equation:

$$b_{ij} = \frac{\varphi\left(td_i - f_{ij}\right) \cdot e_{ij}^{-1}}{pt_{ij} \cdot \left(se_{ij} - ta_i\right)}. \tag{10}$$

4.4. Selection Strategies. In the auction process, the task agent would select a bidding UAV agent to award a contract if there is, at least, one bidding value which is not 0. Note that different selection strategies may have distinctive performances. In this work, we present three selection strategies, which are the M strategy, D strategy, and P strategy. Also, we would give detailed illustrations later.

When task t_i ($i = 1, 2, 3, \ldots$) arrives, there would be a UAV responsible for its auction work, and we use *AN* (short for announcer) to denote the UAV. As for the UAVs bidding to t_i, we use $BID = \{bid_j, j = 1, 2, \ldots\}$ to represent. What we need to pay attention to is that when the bidding value is 0,

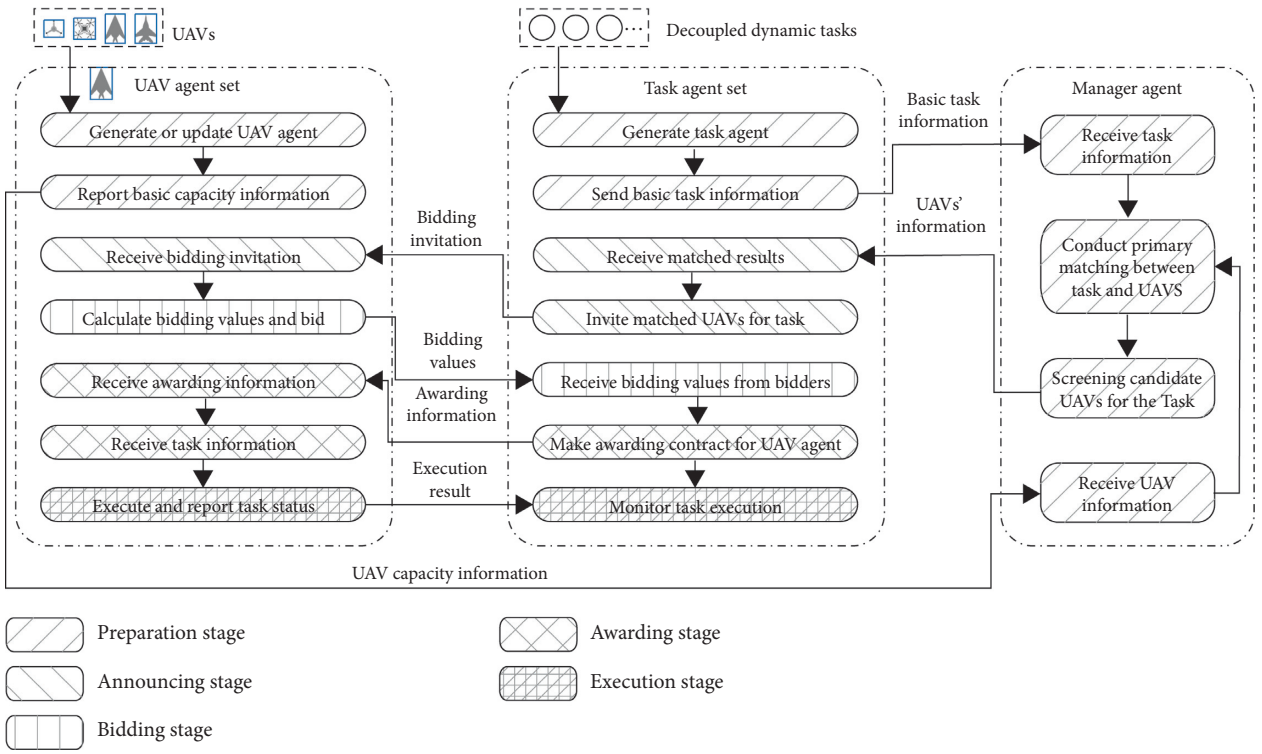

Figure 2: The basic activities and interactions of agents.

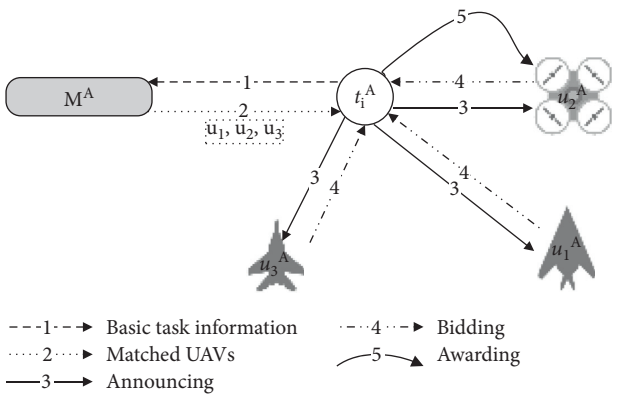

Figure 3: An example of task assigning.

the corresponding UAV will not be treated as a valid bidder, and it will not be added into the set *BID*. Besides, BV = $\{bv_j, j = 1, 2, \ldots\}$ is the bidding value set, in which bv_j represents the bidding value of bidder bid_j for *AN*. It should be noted that when there is only one bidder in *BID*, it would be awarded directly, so the selection strategy is only applicable in the condition of more than one bidder.

4.4.1. M Strategy. Under this strategy, the *AN* selects the bidder with maximal bidding value. If bid_k is selected, it must meet the following constraint:

$$\forall bid_j \in BID, \quad bv_k \geq bv_j. \tag{11}$$

4.4.2. P Strategy. Under this strategy, the *AN* chooses a bidder according to probability policy. The winning probability wp_j of bidder bid_j would be determined by

$$wp_j = \frac{bv_j}{\sum_{i=1}^{|BID|} bv_i}. \tag{12}$$

To select the bidders based on probability, we let *wp* be a random number, and wp ∈ (0, 1). Without loss of generality, we let $wp_0 = 0$. When the value of *wp* satisfies equation (13), bidder bid_k is chosen as a winner.

$$\sum_{m=0}^{k-1} wp_m < wp \leq \sum_{n=0}^{k} wp_n. \tag{13}$$

4.4.3. D Strategy. Considering that the distance between the task and the UAV is a key factor that affects the execution efficiency, the D strategy tries to select a bidder with the condition that the bidding value is larger than 0 and has the shortest distance among all bidders. It is worth noting that the bidders should send the value of distance with the bidding value to the announcer. We let d_j be the distance between the *AN* and bidder bid_j. If bid_k is selected, then the following constraint must be satisfied:

$$\forall bid_j \in BID, \quad d_k \leq d_j. \tag{14}$$

5. NECTAR Method

Based on the mechanism mentioned above, we propose a novel agent-based dynamic task allocation algorithm, NECTAR, for the UAV swarm. In this section, we would give detailed illustrations for NECTAR by presenting the algorithms of the manager agent, task agents, and UAV agents.

In Algorithm 1, we present the pseudocode of the algorithm for the manager agent. The manager agent initializes its operation in line 1, after which the existing UAV agents would be added into the agent set U^A. In line 3, the function *Waiting( )* is running in an independent thread, and the manager agent would jump out of the waiting state only when there are requests from task agents or reports from UAV agents. Line 4 means that there is a new request from task agent t_i^A. Also, the manager agent would let the candidate UAV agent set to be null in line 5. Then, the manager agent polls all the agents in U^A (line 6). If the task is not a striking task, all the UAV agents satisfying task t_i's ability requirement would be added into set $U^A_{candidate}$ (line 7). However, if the task is a striking task, the selected agents must satisfy additional resource constraints, i.e., the rsr_j should not be 0 (lines 9–10). Line 12 means that there is a new report from UAV agent u_m^A. If the agent is newly constructed and not in U^A, the manager agent would add it to U^A and update the related state. Otherwise, the status information of u_m^A would be updated directly (lines 12–15). After the response for a request or a report, the manager agent would return to function *Waiting( )* (line 16). Also, the manager agent would jump out of the waiting state and start processing at line 4 when the next request or report is coming.

The pseudocode of the algorithm for the task agent is given in Algorithm 2. After the construction of the task agent, it would initialize and analyze the task information (line 1). Lines 2 and 3 show that the task agent would send the task requirements to the manager agent and receive the candidate UAV agents. It should be noted that the basic requirements are mainly the task type and its additional information, such as resolution and firepower type. The task agent would announce to the candidate UAV agents for the task (lines 4-5). In lines 6 and 7, the task agent would initial the valid bidding value set to be an empty set, after which the bidding value of each bidder would be received. If a UAV agent has the bidding value bigger than 0, the value would be added to the valid bidding value set *BidValue* (lines 9–10). In lines 11 and 12, the task agent would select a winner bidder based on selection strategies. However, if the set *BidValue* is null, that means there is no UAV agent that can satisfy task demands, and the task should be rejected (lines 13-14).

Algorithm 3 depicts the pseudocode of the algorithm for UAV agents. Line 1 means that the UAV agent is constructed, and it would initialize. Then, the UAV agent would acquire status parameters and report to the manager agent for registration, and the reporting information mainly includes abilities and remaining resources. In line 3, the function *UagentWaiting()* would run in an independent thread, and the UAV agent would jump out of the waiting state in the following three conditions: bidding invitation from task agents, awarding message from task agents, and the UAV encounters an abnormal situation. Note that abnormal situations mainly refer to device errors which lead to the loss of abilities. Lines 5-7 describe the first situation that causes the UAV agent to jump out of the waiting state. The UAV agent would analyze the task information and calculate the bidding value, after which the bidding value is to be sent to the corresponding task agent. Besides, the awarding message would also lead the UAV agent to jump out of the waiting state (lines 8–12). The UAV agent would add the task to the UAV task list and update status parameters. If the task is a striking task, the UAV agent should lock the striking resource and report to the manager agent. Another situation that would lead to jumping out of the waiting state is an abnormal situation. If this situation occurs, the UAV agent would acquire the device states and report to the manager agent M^A (lines 13–15).

As illustrated above, NECTAR would allocate the task at its arrival rather than allocate the tasks in batch mode, so the NECTAR achieves real-time allocation for dynamic tasks. In addition, the calculation of bidding values would locate on the UAV agent rather than the manager agent, by which the status messages passed between each other will be greatly reduced, so it is able to be used in a limited bandwidth environment.

6. Experimental Evaluation

As illustrated in Section 4, there are four kinds of selection strategies. Thus, four kinds of task assigning algorithms can be generated by adopting different bidder selection strategies, i.e., NECTAR-M, NECTAR-P, and NECTAR-D. It should be noted that NECTAR-M employs the M strategy when selecting the winner bidder, and NECTAR-P and NECTAR-D adopt the P and D strategy, respectively. In addition, we would compare them with a baseline algorithm- RANDOM. For the sake of avoiding central control to reduce communications, RANDOM uses an agent-based mechanism (similar to NECTARs) to get candidate UAVs and randomly chooses the winner UAV to allocate the task.

To verify the effectiveness of our proposed algorithms and ensure the repeatability of the experiments, we conduct simulation experiments to evaluate the performance of NECTAR. Also, the performance metrics by which we evaluate algorithms' efficiency mainly include the following.

6.1. Task Success Ratio (TSR). It can be defined as *TSR* = Total count of successful tasks/Total count of arrived tasks, and the mathematical expression can be found in (7).

6.2. Execution Time Ratio (ETR). The definition of the *ETR* can be described as *ETR* = Total execution time of all tasks/Total flying time of all UAVs, and the mathematical expression can be found in (8).

(1) Initialize and let UAV agent set $U^A \longleftarrow NULL$;
(2) $U^A \longleftarrow$ existing UAV agents;
(3) *Waiting( )*;
(4) **if** *There is a new request from* t_i^A **then**
(5) Let $U^A_{candidate} \longleftarrow NULL$;
(6) **foreach** u_j^A *in* U^A **do**
(7) **if** $\Phi(t_i)\, u_j^t$ *and* $t_y^j \neq striking$ **then**
(8) Add u_j^A into $U^A_{candidate}$;
(9) **else if** $\Phi(t_i)\, u_j^t$ *and* $rsr_j > 0$ **then**
(10) Add u_j^A into $U^A_{candidate}$;
(11) Send $U^A_{candidate}$ to t_i^A
(12) **else if** *There is a new report from* u_m^A **then**
(13) **if** u_m^A is not in U^A **then**
(14) Add u_m^A into U^A;
(15) Update related status of u_m^A in U^A;
(16) Return to *Wainting( )*;

Algorithm 1: The algorithm for the manager agent.

(1) Initialize and analyze task information;
(2) Send basic task requirements to M^A;
(3) Receive candidate UAV agent set $U^A_{candidate}$;
(4) **foreach** u_j^A *in* $U^A_{candidate}$ **do**
(5) Send announcement information to;
(6) Let valid bidding value set *BidValue* $\longleftarrow$ *NULL;*
(7) Receive bidding value from UAV agents;
(8) **foreach** u_j^A *in* $U^A_{candidate}$ **do**
(9) **if** $bid_j > 0$ **then**
(10) Add bid_j into set *BidValue*;
(11) **if** *BidValue is not null* **then**
(12) Select a winner bidder Bid_{win} based on the selection strategies;
(13) **else**
(14) Reject task;

Algorithm 2: The algorithm for the task agent.

(1) Initialize and analyze UAV status;
(2) Acquire parameters of status;
(3) Report status parameters to the manager agent M^A;
(4) *UagentWaiting()*;
(5) **if** *There is a bidding invitation from* t_i^A **then**
(6) Analyze the task information;
(7) Calculate bidding value and send value to t_i^A;
(8) **if** *There is an awarding message* **then**
(9) Add task to UAV task list;
(10) Update status parameters;
(11) **if** *Awarded task is a striking task* **then**
(12) Report remained striking resource to M^A;
(13) **if** *The UAV encounters an abnormal situation* **then**
(14) Acquire device ability parameters;
(15) Report status parameters to M^A;
(16) Return to *UagentWaiting()*;

Algorithm 3: The algorithm for the UAV agent.

6.3. Simulation Parameters. For the parameters mentioned above, we give a detailed setting as follows:

(i) The tasks arrive dynamically, following the Poisson distribution with the average internal time $1/\lambda$.

(ii) Parameter *BaseDeadline* is used to set the deadline of task t_i, and the formula is as follows:

$$td_i = ta_i + \text{BaseDeadline}, \tag{15}$$

where parameter ta_i represents the arrival time of t_i and *BaseDeadline* follows a uniform distribution in the interval $[40, 40 + a \times 10]$.

(iii) The initial number of UAVs is set to be 150.

(iv) The tasks are located in a rectangle area, the length of which is 3000 m and the width is 2000 m. At the time instant of 0, the UAVs are randomly distributed in the mission area.

(v) The UAVs' maximum velocity is set to be *60* m/s according to current technologies [26].

(vi) The set of abilities possessed by each UAV is a true subset of set {*reconnaissance ability, striking ability, damage assessment ability*}. For the striking resources, the initial number is 10, which means that the maximum number of strike tasks the UAVs can undertake is 10. Also, the UAVs of the striking type would return to their base when the striking resources have been used up.

(vii) The execution time follows a uniform distribution pattern, and the value of execution time is determined by their types. For reconnaissance tasks, the value of e is between [15, 25], striking tasks e~[25, 35], and assessment tasks e~[35, 45].

We mainly carry out four groups of experiments to study the effectiveness of the proposed algorithms, NECTARs, and analyze the impact of time, task arrival rate, task deadline, and the number of UAVs on the performance. The intervals of these parameters are listed in Table 2.

Table 2: Parameters for experiment studies.

Parameter	Value (fixed) – (varied)
Experiment Time	(500) – (200 900)
λ	(3) – (1,2,3,4,5,6,7)
a	(4) – (1,2,3,4,5,6,7)
UAV numbers	(150) – (60, 90, 120, 150, 180, 210, 240)

6.4. Performance with Time Variation. First, we study the performance of NECTAR algorithms when time varies. It is noted that time mainly affects the number of striking UAVs because strike resources are consumable, so the later the time, the fewer the striking UAVs available. In this group of experiments, λ equals 5, the value of a is 4, the number of UAVs is 150, and the statistical time varies from 200 to 900. The result can be seen in Figure 4.

From Figure 4(a), we can see that all algorithms have achieved a task success ratio of over 80% in the beginning, which can verify the effectiveness of the agent-based auction process. By comparing the *TSR* of the four algorithms, we can find that NECTAR-M has the highest success rate, and RANDOM achieves the lowest success rate. As for the reason that RANDOM has the lowest *TSR*, we attribute this to the random selection strategy. Due to the strategy, it will select a winner UAV randomly from candidate UAVs who meet the task's deadline. However, the selected UAV may need long transition time and wait a period of time before the start of the task's execution, which greatly affects the efficiency of the UAV, resulting in resource wasting, so the *TSR* is low. From the comparison of NECTAR-D and RANDOM, we would see that the performance gap between them is about 6%, so we can conclude that the distance is a main factor affecting the task assigning efficiency. Consequently, it would be easy to understand that NECTAR-D has slightly better performance than NECTAR-P, and the reason is that NECTAR-P only has a larger probability to select the UAVs with larger bidding value, but NECTAR-D would choose the nearest UAV to undertake the task, leading to the time spent for transition being the shortest. As for the reason that NECTAR-M has the highest *TSR*, it can be concluded to the employment of maximum bidding values selection strategy, which takes the distance, execution time, and UAV waiting time into consideration, comprehensively. Therefore, it would choose a UAV that is relatively suitable for the task and obtains higher effectiveness. From the comparison between NECTAR-M and NECTAR-D, we can find NECTAR-M achieves higher performance than NECTAR-D for about 4%, which indicates the waiting time and execution time would be another factor that should not be ignored in the assigning process.

Another phenomenon observed in Figure 4(a) is that the performance of all algorithms would decrease with time variations. The reason is that the striking resource is limited, and more striking tasks may be rejected due to the exhaustion of striking resources. Furthermore, the performance gap between NECTAR-M and NECTAR-D has become smaller and smaller over time. The reason can be attributed to the rejection of striking tasks, which makes the advantage of NECTAR-M can only be reflected in reconnaissance tasks and damage assessment tasks, leading to the gaps becoming smaller.

We can observe from Figure 4(b) that NECTAR-M achieves the highest *ETR* among the four algorithms, and the reason is that the M strategy would select the most suitable UAV to reduce the idle time and flying distance in a comprehensive way, and thus, the execution time ratio would be higher. The performance of other algorithms on the *ETR* is NECTAR-P, NECTAR-D, and RANDOM in order, and the explanation of this fact is similar to that on the *TSR*. An interesting observation is that the *ETR* of NECTAR-M, NECTAR-P, and NECTAR-D would slightly decrease with time variation, and the descending speed would increase after the time 500s. That is because after 500s, the resources of some striking UAVs are exhausted and the UAVs would return to the base, the available striking UAVs become less, so the UAVs undertaking striking tasks will

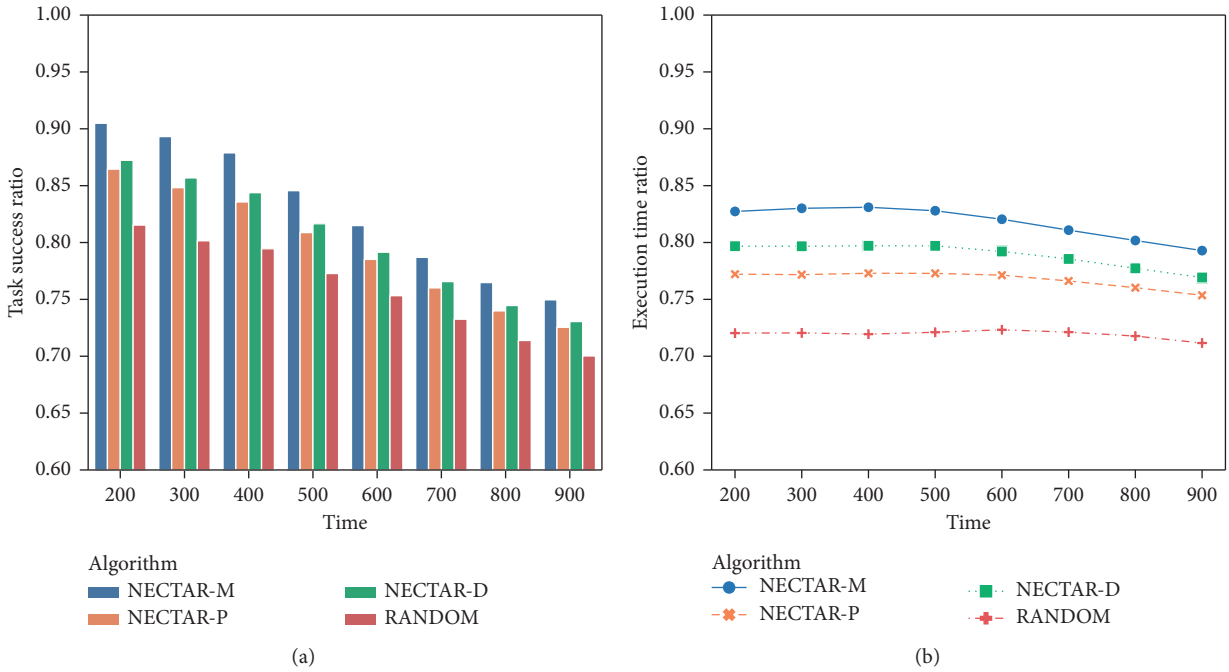

Figure 4: Performance with time variation.

have to fly longer distances, resulting in a lower *ETR*. However, the *ETR* of RANDOM is to first keep stable and, then, fall slightly. The reason for the smaller decline on the *ETR* is that RANDOM does not save the transition time, waiting time, and execution time in the beginning, so the reduction in the number of available UAVs has less impact on RANDOM.

We can conclude from Figure 4 that the proposed agent-based auction mechanism is effective in assigning tasks to the UAV swarm, and the main factors affecting the assigning efficiency are the transition distance, waiting time, and execution time. In addition, the performance verifies the effectiveness of the designed NECTARs.

6.5. Performance Impact of the Task Arrival Rate. In this group of experiments, we pay attention to the impact of the arrival rate on the performance of NECTARs and RANDOM. The parameter λ varies from 2 to 8 with an increment of 1. Note that the parameter λ mainly affects the number of arrival tasks in a certain unit of time. Also, the experiment results are presented in Figure 5.

It can be seen from Figure 5(a) that when the value of λ is 2, all algorithms have achieved a 100% *TSR*. The reason is that the number of arriving tasks is small and all tasks can be successfully executed as the UAV resources are sufficient. When the value of λ is larger than 3, the task success ratio of the four algorithms will decrease with the increment of λ. This can be explained by the fact that when the task arrival rate becomes large enough, all UAVs have reached a heavy burden, and they have no choice but to reject some tasks due to the limited resources, and thus, the *TSR* would decrease.

An interesting phenomenon found in Figure 5(a) is that, with the increment of λ, the performance gap between NECTARs and RANDOM would first become larger and, then, smaller. The reason is that as the number of tasks increases, NECTAR's advantages become more and more manifest, so the performance gap becomes larger. Also, when the task load reaches a certain level, the advantage of NECTARs over RANDOM would keep stable as the potential of UAVs has been realized, but the total tasks would keep increasing with λ; thus, the proportion of the advantage in total tasks would decrease, so the performance gap becomes smaller.

From Figure 5(b), we can observe that the *ETR* would increase with the increment of λ. At first, it would increase rapidly, which is because the UAVs are in light load, and more tasks would help UAVs improve the utilization. However, when the UAVs are in heavy load, the improvement of utilization would slow down or nearly to be 0, so the *ETR* would grow slowly or keep stable.

It can be concluded from Figure 5 that the parameter λ mainly affects the workload state for UAV swarms, and the proposed NECTARs can adapt to the environment of different workloads.

6.6. Performance Impact of Task Deadline. We study the impact of the task's deadline on the performance of the NECTARs in this group of experiments. The parameter a varies from 1 to 7 with an increment of 1. It should be noted that the parameter a mainly affects the deadline, and when the value of a becomes larger, the deadline of tasks would be looser. Also, the results are depicted in Figure 6.

We can observe from Figure 6(a) that the *TSR* of NECTARs and RANDOM would grow with the increasing of parameter a's value. This is because when the task deadline is looser, tasks can be more flexibly assigned to

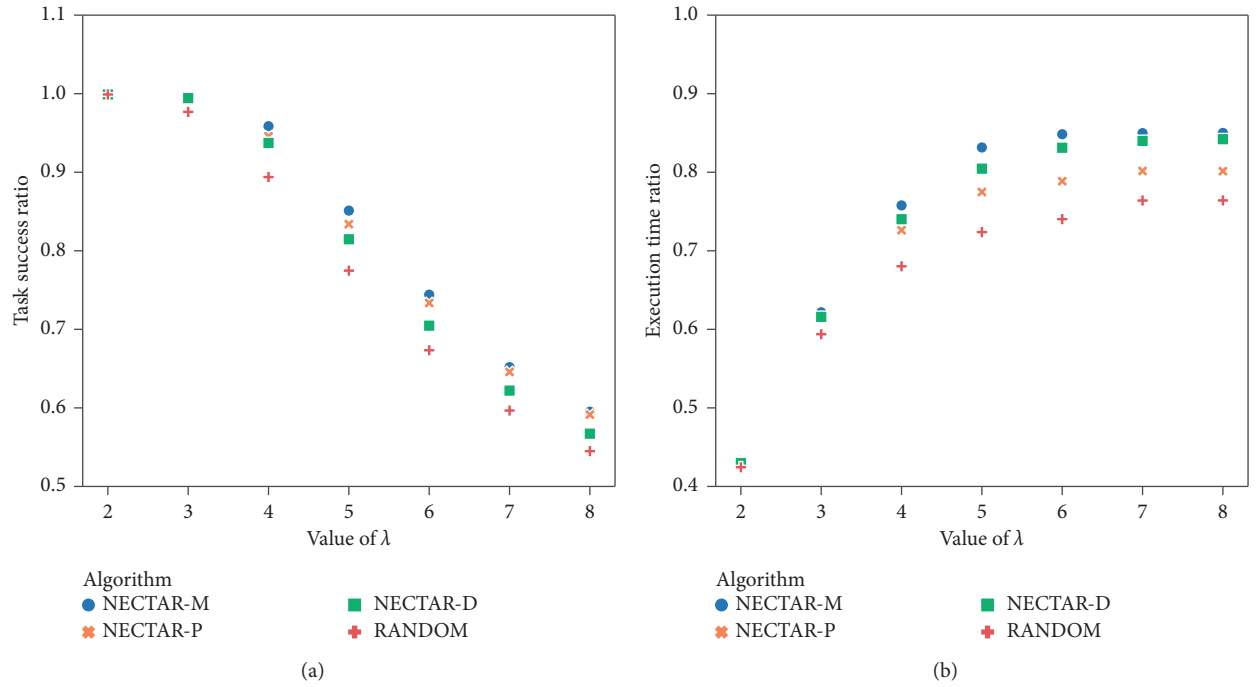

Figure 5: Performance impact of the task arrival rate.

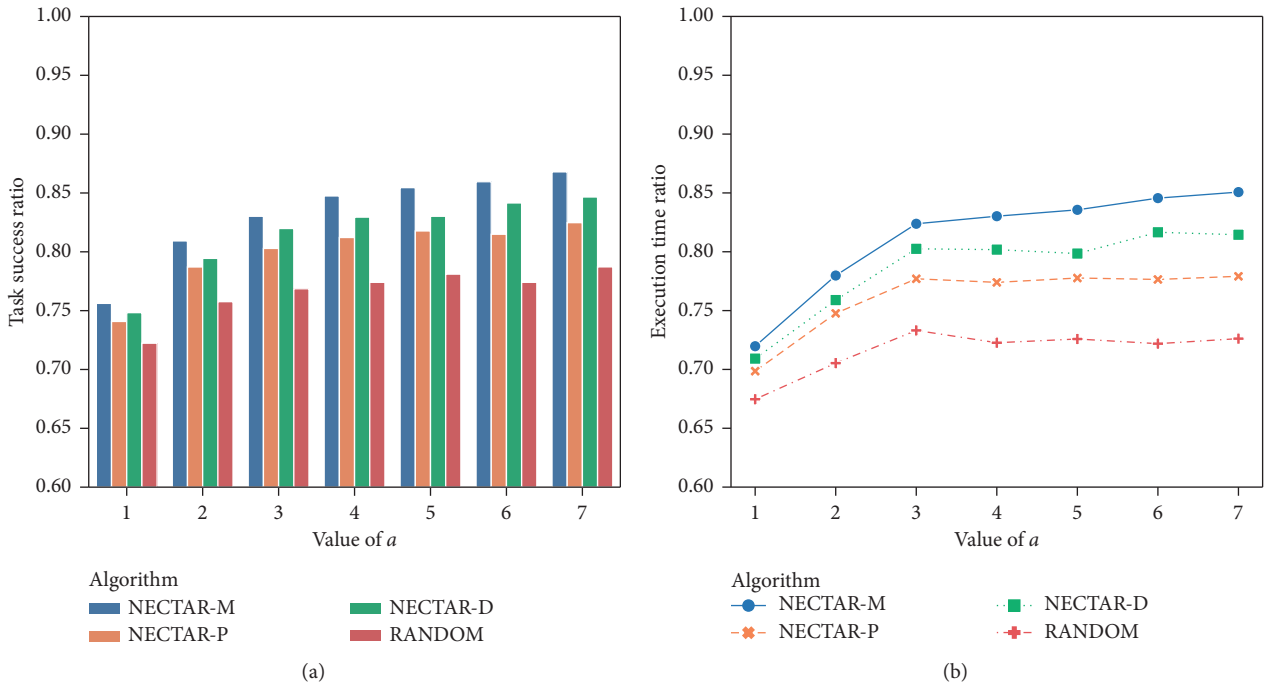

Figure 6: Performance impact of task deadline.

UAVs, which improves the utilization of UAVs. Another fact that can be concluded is that the growth of the *TSR* is fast at first and slows down later. The reason is that the success of tasks is mainly affected by the number of available UAVs, looser deadline would help to promote the utilization of a single UAV, but the promotion is limited. Consequently, the growth of the *TSR* would slow down and the *TSR* would keep stable at last. Besides, we can observe from Figure 6(a) that the growth of NECTAR-M is the largest, and this verifies the efficiency of the proposed mechanism. We conclude the reason to be that NECTAR-M has taken distance and time into consideration in a

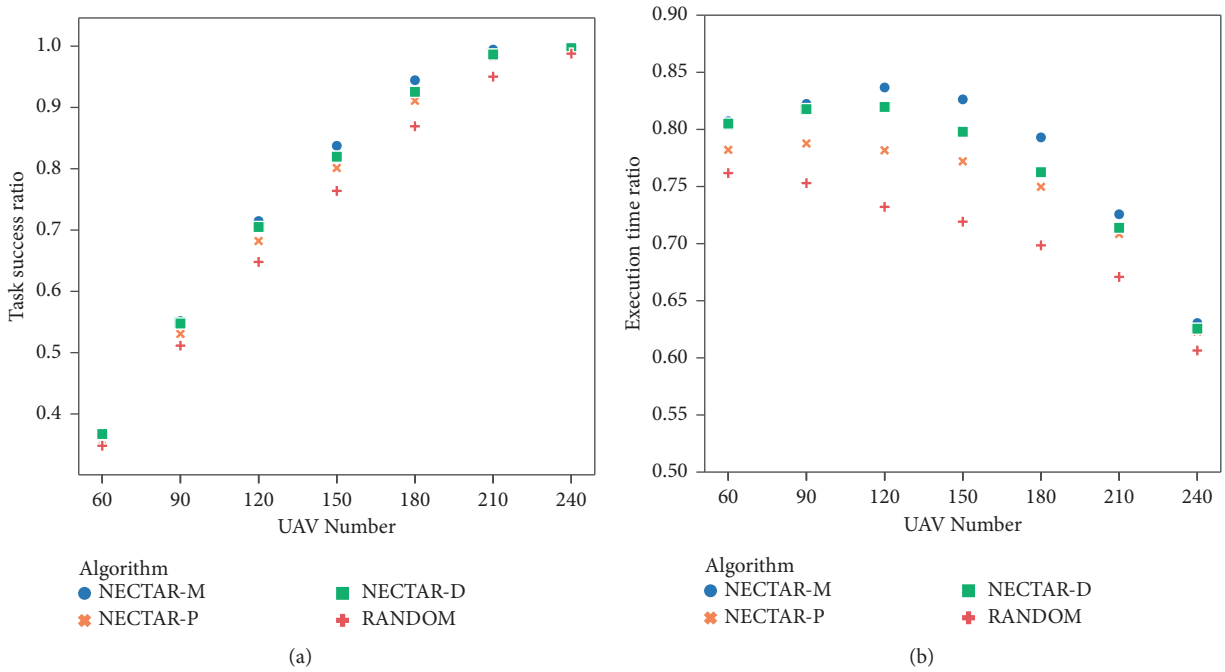

Figure 7: Performance impact of the UAV number.

comprehensive way, so the improvement of UAV's utilization would be more significant.

From Figure 6(b), we can find that the *ETR* of NECTARs and RANDOM would grow with an increment of *a*. Another fact is that the performance gap on the *ETR* between different algorithms would become larger with looser deadline. Also, the explanations would be similar to those of the *ETR*.

It can be summarized from Figure 6 that when tasks have looser deadline, the performance of assigning algorithms would get improved, but the improvement is limited. In addition, our proposed NECTARs achieve better improving on both the *TSR* and *ETR*.

6.7. Performance Impact of Task Deadline. In this group of experiments, the impact of the UAV number would be studied. Also, the UAV numbers mainly affect the available UAV resources in the swarm. The results are presented in Figure 7.

As depicted in Figure 7(a), the *TSR* of NECTARs will continue to grow with the increase of the UAV number until it reaches 100%. The reason is that there are more UAVs that can respond to tasks, so more tasks can be successfully executed. Besides, the performance gap on the *TSR* between NECTARs and RANDOM would first become larger and, then, smaller. The explanation is that the tasks are arriving dynamically and must be responded to at its arrival, when the number of UAVs is too small, the range of options is small, and the optimization effect is limited, so the advantages of NECTARs are not significant. While the number of UAVs increases, the NECTARs would achieve obvious performance advantages over RANDOM; thus, the performance gap would become larger. However, when the number of UAVs is larger enough, NECTARs have achieved 100% *TSR* and the increase of UAV number would only promote the performance of RANDOM, so the performance gap would become smaller. Besides, NECTAR-M has achieved the highest *TSR* among all algorithms, and NECTARs perform better than RANDOM, which illustrates that our proposed mechanism and algorithms are robust.

From Figure 7(b), we can find that the *ETR* of NECTERs would slightly rise and, then, decline. This can be explained that the tasks arriving first need to respond first, and the number of candidate UAVs is too small, so the distance between the winner UAV and the task may not be short, so the time spent in transition may be too long, resulting in low efficiency. With the UAV number grows, the NECTARs would realize a higher *ETR*. As for the decline of the *ETR*, the reason is that when the UAVs are enough to undertake the arrived tasks, more UAVs would only bring resource idling, and thus, the *ETR* would decline. As for the direct decline of RANDOM's *ETR*, we attribute this to the random selection strategy. Because RANDOM has not taken the distance and time into comprehensive consideration such as NECTARs, the UAV resources are not sufficiently used. Thus, the *ETR* would decline.

We can conclude that the NECTARs would perform better than the baseline algorithm regardless of the number of UAVs. Consequently, the proposed mechanism and algorithms have good adaptability.

7. Conclusions and Future Work

In this study, we investigated the problem of dynamic real-time task assigning for the UAV swarm and proposed a novel distributed task assigning algorithm, NECTAR, based on agent intelligence and auction mechanism. The intelligent agent technology is employed to realize self-management for

UAVs. Also, the auction mechanism is used to complete the task assigning process in a distributed way. Our contributions include designing the task assigning models, and auction-based task assigning activities, as well as the collaboration processing stage and flows. In addition, we devised the calculation of bidding values on local devices to reduce communications. Besides, three selection strategies, i.e., the M strategy, D strategy, and P strategy, were proposed to determine the winner UAVs. Furthermore, we described NECTAR in detail by giving the algorithms for different agents. Finally, we conducted extensive simulation experiments to verify the effectiveness and efficiency, of which the results indicate that NECTAR is an effective task allocation algorithm for assigning dynamic tasks in the UAV swarm, and it is adaptable and robust when the parameters change.

Our NECTAR is the first of its kind found in the literature because NECTAR adopts intelligent agents and auction mechanism to address the dynamic tasks assigning problem in a distributed way for the UAV swarm, rather than allocate tasks in batch mode. In our future work, we would like to address the problems as follows. Firstly, we want to enhance NECTAR by taking weak communication conditions into consideration and study how they could complete the assigning process through some effective rules. Secondly, we plan to apply NECTAR into real UAV systems to accomplish task allocation.

Data Availability

The experiment data can be obtained by contacting the corresponding author (wdbao@nudt.edu.cn).

Conflicts of Interest

The authors declare that there are no conflicts of interest about the publication of this paper.

Authors' Contributions

(a) Chao Chen was involved in comprehensive processing of thesis writing, experiment, and data. (b) Weidong Bao was involved in idea guidance, paper guidance and revision, rewriting the Abstract and Introduction, and revision of the experiment code. (c) Tong Men conducted the experiment. (d) Xiaomin Zhu was involved in algorithm design guidance, pseudocode writing, and polishing the writing of the whole paper. (e) Ji Wang sorted out the related literature and helped Chao Chen in experimental analysis. (f) Rui Wang redrew the data graphs and corrected the syntax errors of the resubmitted version.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Nos. 61872378 and 61773390), the Science Fund for Distinguished Young Scholars in Hunan Province (No. 2018JJ1032), and the Scientific Research Project of National University of Defense Technology (No. ZK19-03).

References

[1] T. Huang, D. Huang, Z. Wang, and A. Shah, "Robust tracking control of a quadrotor UAV based on adaptive sliding mode controller," *Complexity*, vol. 23, 2019.

[2] F. Sun, X. Wang, and Z. Rui, "Task scheduling system for UAV operations in agricultural plant protection environment," *Journal of Ambient Intelligence and Humanized Computing*, vol. 23, 2020.

[3] H. Meng and Y. Guo, "Automatic safety routing inspection of the electric circuits based on UAV light detection and ranging," *DEStech Transactions on Engineering and Technology Research*, vol. 23, 2017.

[4] L. Zhou, H. Ma, Z. Yang, S. Zhou, and W. Zhang, "Path loss modeling and evaluation: unmanned aerial vehicle communications," *IEEE Vehicular Technology Magazine*, vol. 34, 2020.

[5] M. Darrah, W. Niland, and B. Stolarik, "Multiple UAV dynamic task allocation using mixed integer linear programming in a SEAD mission," *Infotech*, vol. 34, 2006.

[6] E. Kendall, R. Phillip, and P. Meir, "Dynamic network flow optimization model for air vehicle resource allocation," in *Proceedings of the American Control Conference*, pp. 1853–1858, New York, NY, USA, 2001.

[7] D. Zhou, A. Zhang, and P. Yang, "Fixed-time output feedback consensus of second-order multi-agent systems with settling time estimation," *International Journal of Control Automation and Systems*, vol. 45, 2020.

[8] W. Zhu, L. Li, L. Teng, and Y. Wen, "Multi-UAV reconnaissance task allocation for heterogeneous targets using an opposition-based genetic algorithm with double-chromosome encoding," *Chinese Journal of Aeronautics*, vol. 31, no. 2, pp. 339–350, 2018.

[9] Z. Zhu, B. Tang, and J. Yuan, "Multi-robot task allocation based on an improved particle swarm optimization approach," *International Journal of Advanced Robotic Systems*, vol. 14, no. 3, pp. 1–22, 2017.

[10] H. Wu, H. Li, R. Xiao, and J. Liu, "Modeling and simulation of dynamic ant colony's labor division for task allocation of UAV swarm," *Physica A: Statistical Mechanics and Its Applications*, vol. 491, pp. 127–141, 2018.

[11] C. Shumacher, P. Chandler, M. Pachter, and L. Pachter, "UAV task assignment with timing constraints via mixed-integer linear programming," in *Proceedings of the AIAA 3rd Unmanned Unlimited Systems Technical Conference*, p. 6410, New York, NY, USA, 2004.

[12] H. Wu and F. Zhang, "Wolf pack algorithm for unconstrained global optimization," *Mathematical Problems in Engineering*, vol. 91, 2014.

[13] S. Rasmussen, T. Shima, J. Mitchell, A. Sparks, and P. Chandler, "State-space search for improved autonomous UAVs assignment algorithm," in *Proceedings of the Conference on Decision and Control*, New York, NY, USA, 2004.

[14] A. Mehdi and P. Jonathan, "Decentralized task assignment for unmanned aerial vehicles," in *Proceedings of the European Control Conference Cdc-ecc 05 IEEE Conference on Decision & Control*, New York, NY, USA, 2005.

[15] A. Whitten, L. Choi, L. Johnson, and P. How, "Decentralized task allocation with coupled constraints in complex missions," in *Proceedings of the American Control Conference*, New York, NY, USA, 2011.

[16] X. Fu, J. Pan, X. Gao, B. Li, and K. Zhang, "Task allocation method for multi-UAV teams with limited communication bandwidth," in *Proceedings of the 2018 15th International*

Conference on Control, Automation, Robotics and Vision (ICARCV), New York, NY, USA, 2018.

[17] J. Yang, X. You, G. Wu, M. M. Hassan, A. Almogren, and J. Guna, "Application of reinforcement learning in UAV cluster task scheduling," *Future Generation Computer Systems*, vol. 95, no. 3, pp. 140–148, 2019.

[18] P. Sujit and R. Beard, "Distributed sequential auctions for multiple UAV task allocation," in *Proceedings of the American Control Conference IEEE*, New York, NY, USA, 2007.

[19] S. Ramchurn, J. Fischer, Y. Ikuno, F. Wu, J. Flann, and A. Waldock, "A study of human-agent collaboration for multi-UAV task allocation in dynamic environments," in *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, New York, NY, USA, 2015.

[20] K.-S. Kim, H.-Y. Kim, and H.-L. Choi, "A bid-based grouping method for communication-efficient decentralized multi-UAV task allocation," *International Journal of Aeronautical and Space Sciences*, vol. 21, no. 1, pp. 290–302, 2020.

[21] H.-X. Chen, Y. Nan, and Y. Yang, "Multi-UAV reconnaissance task assignment for heterogeneous targets based on modified symbiotic organisms search algorithm," *Sensors*, vol. 19, no. 3, p. 734, 2019.

[22] Z. Zhen, D. Xing, and C. Gao, "Cooperative search-attack mission planning for multi-UAV based on intelligent self-organized algorithm," *Aerospace Science and Technology*, vol. 76, pp. 402–411, 2018.

[23] G. Fernandez, N. Kerle, and M. Gerke, "UAV-based urban structural damage assessment using object-based image analysis and semantic reasoning," *Natural Hazards and Earth System Sciences*, vol. 15, no. 6, pp. 1087–1101, 2015.

[24] S. Adhau, M. L. Mittal, and A. Mittal, "A multi-agent system for distributed multi-project scheduling: an auction-based negotiation approach," *Engineering Applications of Artificial Intelligence*, vol. 25, no. 8, pp. 1738–1751, 2012.

[25] P. Ezhilchelvan and G. Morgan, "A dependable distributed auction system: architecture and an implementation framework," in *Proceedings of the 5th International Symposium on Autonomous Decentralized Systems*, pp. 3–10, New York, NY, USA, 2001.

[26] UAV, "The 10 longest range unmanned aerial vehicles (UAVs)," 2020.