Multi-UAV Task Assignment with Parameter and Time-sensitive Uncertainty using Modified Two-part Wolf Pack Search Algorithm

Yongbo Chen, Di Yang and Jianqiao Yu

***Abstract*—This paper presents a systematical framework to solve the multiple unmanned aerial vehicles (multi-UAV) cooperative task assignment problem. Based on a combinatorial optimization model, it is solved by a digraph-based method and a novel meta-heuristic optimization method, named modified two-part wolf pack search (MTWPS) algorithm. When the number of UAVs/targets is large, in order to reduce the simulation time, we also present a new solution framework based on an easy-computing objective function. Additionally, the parameter and time-sensitive uncertainty are considered in the extended task assignment problem. For the problem with parameter uncertainty, it is formulated by a robust optimization method and solved by a novel combined algorithm, including the classical interior point method and our MTWPS algorithm. For the problem with time-sensitive uncertainty, it is solved by a practical online hierarchical planning algorithm. Finally, numerical simulations and physical experiments demonstrate that the proposed methods can provide a flyable solution for the UAVs and achieve outstanding performance in comparison with other algorithms.**

***Index Terms*—Multiple unmanned aerial vehicles (multi-UAV) cooperative task assignments problem, modified two-part wolf pack search (MTWPS) algorithm, robust optimization method, online hierarchical planning algorithm**

I. Introduction

MULTIPLE unmanned aerial vehicles (multi-UAV) task assignment helps to assign some heterogeneous UAVs to perform many tasks for some given targets in a period of decision-making time, so as to form coordination and minimize the system cost to reach optimal efficiency. These UAVs can replace the manned vehicles to execute a variety of complex tasks, mainly including: classification, response, and verifying targets. We consider it as one of the most challenging problems of the multi-UAV system.

Different from the formation control in [1], the multi-UAV task assignment is suitable for loose formulation structure. In general, it is a sophisticated combinatorial optimization problem with complex constraints, in consideration of task precedence and coordination, timing constraints, and flyable trajectories. In the real environment, the appropriate mathematical description of this problem is difficult, particularly for the complex dynamic environment with the uncertainties. Essentially, the task assignment problem is also a non-deterministic polynomial hard (NP-hard) problem [2]. So the size of the problem (*e.g.* the number of UAVs, targets, and tasks) impacts the complexity of the solving process. In a word, the main challenges of the multi-UAV task assignment are to build up an appropriate mathematical model and to find an efficient solving algorithm. Aim at this open problem, we try to show a systematic solution framework which focuses on its modeling process and the solving process under different environments.

In this paper, we first set up the optimization model of the task assignment problem of multiple heterogeneous UAVs based on a combinatorial optimization model in the complicated multi-tasking scene. Then, a novel graph-based method for solving deadlock detection and resolution problem is presented to ensure the validity of the solution. Next, a modified two-part wolf pack search (MTWPS) algorithm updated by the two-part individual encoding approach as well as the transposition and extension (TE) operation is proposed for solving this problem. After that, in order to balance the accuracy of the model with the computational efficiency, a new solution frame is presented based on two different objective functions. Finally, a robust optimization theory and online re-planning approach are respectively introduced to solve the problem with parameter and time-sensitive uncertainties.

The main contributions of our paper include: the introduction of the graph-based algorithms for the deadlock problem, a proposed MTWPS algorithm, a new solution frame for the task assignment problem, two methods for the task assignment problem with the parameter and time-sensitive uncertainties. Compared with previous literature, this paper shows a novel, effective and systematic framework to solve the task assignment problem with different environments.

The structure of this paper is as follows: In Section II, related work is addressed and discussed. The descriptions, the analysis, the preparatory work and the rigorous mathematical model of the task assignment problem of multiple heterogeneous UAVs are introduced in Section III. In Section IV, a novel digraph-based method is proposed to solve the deadlock

Manuscript received * *, 2017; This work was supported by the ***. This paper was recommended by **.

Y. Chen, D. Yang and J. Yu are with the School of Aerospace Engineering, Beijing Institute of Technology, Beijing, People Republic of China (e-mail: bit_chenyongbo@ 163.com). Y. Chen is with School of Electrical, Mechanical and Mechatronic Systems, Faculty of Engineering and Information Technology (FEIT), University of Technology, Sydney, PO Box 123, NSW 2007, Australia (e-mail: Yongbo. Chen@student.uts.edu.cn).

detection and deadlock problem. In Section V, a novel MTWPS algorithm is presented. Then, in Section VI, a new solution frame is presented to solve the small-scale and large-scale problems uniformly. The robust optimization method and the practical online hierarchical planning algorithm are used to solve the problems respectively with parameters and time-sensitive uncertainty in Section VII. In Section VIII, the simulations and experiments are carried out. The results verify the correctness of the proposed algorithms. Finally, further discussions and conclusions are presented in Section IX.

II. RELATED WORK

Viewed from the basic mathematics model, the models of the multi-UAV task assignment problems mainly include: mixed integer linear programming model (MILP), dynamic network flow optimization model (NFO) and multiple intelligent agent-based systems [3-5]. Besides, Alighanbari, M. and How, J. P. formulate this issue as a Dynamic Programming (DP) problem, which is shown to be computationally more tractable than the previous MILP solution approaches [6]. In recent years, with the advances of the cross discipline, some concepts in other research areas have been introduced to describe this problem. For example, Karaman, S. *et al.* introduce the Process Algebra into the modeling procedures of this problem [7]. The authors firmly grasp the time-sequence of these tasks, and then, apply the composition, which is an important concept of the Process Algebra, to express all actions of the multiple UAVs. Similarly, Karaman, S. and Frazzoli, E. introduce the linear temporal logic language to describe this problem and then convert it into two widely-used MILP formulations [8]. Even though different models have their own advantages, in general, the classical MILP model, which is used in this paper, has the general applicability.

In the view of the solving algorithm, as an NP-hard problem, the perfect solving methods cannot be found at the present stage. However, many scholars try to solve it in small-scale and medium-scale situations. The solving methods can be divided into the centralized and distributed algorithms. The centralized algorithms mainly include Genetic Algorithm (GA), Tabu search algorithm, particle swarm optimization algorithm, ant colony algorithm and state-space best-first search algorithm [9-14]. Most of these algorithms have the global optimizing capability which can be very helpful in dealing with the small-scale NP-hard problem. The prior work of the distributed algorithms is significantly fewer than the ones of the centralized algorithms [15]. Long, T. puts forward the contract network protocol (CNP) method to solve the multi-UAV deadlock problem [16]. The main advantage of the distributed algorithms is good robustness. If there are some failures of one UAV, the distributed algorithms can still work successfully based on the other UAVs. However, the distributed algorithms cannot ensure the optimization of the results.

Beyond that, there are two significant problems, which may be ignored in much prior work: the deadlock problem and the uncertainty problem. Both of them will limit the effectiveness, optimality, and practicability of the results.

As for scenarios involving targets with multiple sequential tasks, some UAVs might have to wait for other vehicles if a vehicle that executes a previous or simultaneous task has not arrived at a related target [17]. The deadlock problem, which means two or more vehicles are waiting for each other to finish and thus fall into endless waiting, will happen and make the solution invalid. The deadlock problem is derived from computer science in the design of multi-programming systems [18]. Lemaire, T. *et al.* present a swapping method, which can avoid deadlocks by swapping two children tasks in a UAV's local plan if they are not in the chronological order [19]. However, this method becomes invalid, if the time constraints become more complex and deadlocks involve two or more UAVs. Deng, Q. B., *et al.* introduce a graph-based method to analyze solutions and detect deadlocks. Nevertheless, their method needs to finish a series of matrix transpose operations, which greatly reduce the computational efficiency. The detection and unlocking of the deadlocks are NP-hard [20]. In this paper, we present a simpler and efficient deadlock-free algorithm.

Except for the deadlock problem, the uncertainty which occurs in the task assignment process also needs to be taken into account seriously. The uncertainty mainly includes: the uncertainty of the environment, the uncertainty of the communication and the uncertainty of the UAVs. In solving the uncertainty problem, all methods can be divided into two kinds: Forethought and post-processing.

Forethought, which means considering the uncertainty before problems of the system occur, is usually used in the planning stage of the task. This method is suitable for the parameter uncertainty problem. The parameter uncertainties may include the uncertainty of the flying velocity and the uncertainty of task effectiveness. There are three main forethought approaches: stochastic optimization, stochastic programming, and robust optimization [21]. After considering the characteristics of this task assignment problem, the robust optimization method is the most suitable way to establish the model. Its purpose is to compute a result which always meets the constraints in every situation, and ensures that this result is the optimal solution for the worst case. Lanah, E. *et al.* introduce a new version of the orienteering problem that considers the uncertainty of the fuel parameters already in the modeling stage by applying techniques from the robust optimization method [22]. However, it is different from the robust optimization method we used in our paper. There are several kinds of frameworks for the robust optimization theory. For example, the method of [23] is built on the conic quadratic framework provided by [24], which is different from the earlier linear models built by Soyster, A. L. [25]. The practical drawback of Ben-Tal, A. *et al.*'s framework is that it leads to nonlinear, although convex, models, which are more demanding computationally. So ours is based on the framework presented by [26], whose approach for robust linear optimization retains the advantages of the linear framework of Soyster, A. L. [25]. More importantly, their approach offers full control of the degree of conservatism for every constraint.

Post-processing, which means solving the accident situations according to the emergency mechanism, is commonly applied in the task execution process. This method is used to deal with online uncertainties, such as the communication problem, the UAVs malfunction problem and time-sensitive (new) target problem [27-28]. Although these uncertainties are different, many of them can be converted into each other. For example, the communication problem can lead to the loss of data of some UAVs, which can be regarded as the UAVs malfunction problem. By that analogy, the UAVs malfunction problem, which makes that their corresponding tasks are unfinished, can be seen as some new tasks for the important time-sensitive targets. Hence, this paper only considers the uncertainties of the time-sensitive targets. Because of the real-time requirement, in general, the post-processing approach is based on the distributed algorithms. The most common distributed algorithm is the CNP [29-31]. Except for these distributed algorithms, the re-planning method based on the small-scale centralized algorithms can also be used. Of course, its planning consumed time needs to be limited and minimum.

III. PROBLEM DESCRIPTION AND FORMULATION

A. Problem description

The problem considered in this paper is to assign N_v UAVs to complete the classification, response, and verification tasks on N_t targets, where N_v<3N_t and N_t, $N_v \in \{1, 2,\ldots\}$, for every target, its three tasks have the sequence request: classification prior to response and response prior to verification. By the mirror representation of the targets [10], which means to transform the one target- multiple tasks problem into one target- one task problem, the multi-UAV task assignment problem is defined as a multiple Traveling Salesmen Problem (m-TSP). And then, it can be formulated by a complete digraph G= (V, E), where vertex set V= {1, 2 ,…, N_v+3N_t} numbers the starting points of the UAV and the mirrored target points; and each edge in (i, j)$\in E$ represents the shortest flying path between two target points (named city in m-TSP) i and j, where $i \neq j$. The UAVs fly at separate altitudes so as to have collision-free paths. For the fixed-wing UAV, the shortest feasible path between two cities using the Dubins Car model has been proved to be a combination of arcs with the minimum turning radius and straight lines [10]. We follow the same notations used in [10] where C represents a circular arc and L represents a straight-line tangent to C, and focus on the CLC and CC paths. For the rotor UAV, the shortest path is the straight line between two cities.

B. Problem formulation

The multiple heterogeneous UAVs task assignment problem can be formulated as a combinatorial optimization problem:

$$\min_{\{X_{i,j}^k\}} \sum_{i=1}^{N_v+3N_t} \sum_{j=1}^{N_v+3N_t} \sum_{k=1}^{N_v} P_{k,mod\left((j-N_v),3\right)} \left(L_{i,j}^k + f_{i,j}^k(\{X_{i,j}^k\})\right) X_{i,j}^k \quad (1)$$

$$s.t. \quad \begin{array}{l} \sum_{j=1}^{N_v+3N_t} \sum_{k=1}^{N_v} X_{i,j}^k = 1, \quad \forall i \\ \sum_{i=1}^{N_v+3N_t} \sum_{k=1}^{N_v} X_{i,j}^k = 1, \quad \forall j \end{array}, \quad (2)$$

$$\begin{array}{l} \sum_{j=1}^{N_v+3N_t} X_{i,j}^k = 1, \quad if \quad \sum_{j_1=1}^{N_v+3N_t} X_{j_1,i}^k = 1, \quad \forall i,k \\ X_{i,j}^k = 0, \quad if \quad \sum_{j_1=1}^{N_v+3N_t} X_{j_1,i}^k = 0, \quad \forall i,j \end{array}, \quad (3)$$

$$\begin{array}{l} X_{i,j}^k = 0, \quad if \quad (i \le N_v \cap i \neq k) \quad or \quad \cdots \\ (j \le N_v \cap j \neq k) \quad or \quad (i \le N_v \cap j \le N_v), \quad \forall i,j,k \end{array}, \quad (4)$$

$$\begin{array}{l} u_{i-N_v} - u_{j-N_v} + 3N_t \sum_{k=1}^{N_v} X_{i,j}^k \le 3N_t - 1, \\ \forall i,j \text{ meets } 1 \le i - N_v \neq j - N_v \le 3N_t. \end{array} \quad (5)$$

$$\sum_{i=1}^{N_v+3N_t} \sum_{j=1}^{N_v+3N_t} \left(L_{i,j}^k + f_{i,j}^k(\{X_{i,j}^k\})\right) X_{i,j}^k < L_{\text{Lim}}^k, \quad \forall k \quad (6)$$

$$\sum_{i=1}^{N_v+3N_t} \sum_{j=1}^{N_v+3N_t} X_{i,j}^k \le N_{\text{Lim}}^k, \quad \forall k \quad (7)$$

where $X_{i,j}^k$ is a 0-1 decision variable, which means that the k-th UAV flies from the i-th city to the j-th city to complete the l= mod((j-N_v),3) task; $P_{k,l}$ means the failure rate of the k-th UAV performing the l-th task type, where l=1 corresponds to classification task, l=2 corresponds to response task, and l=0 corresponds to verification tasks; mod(•,†) is a function, which means taking the remainder when • is divided by †; It is noted that when j is less than or equal to N_v, it means the UAV performing the landing task, whose $P_{k,l}$ is set as 0; $L_{i,j}^k$ represents the length of the shortest path of the k-th UAV flying from the i-th city to the j-th city; $f_{i,j}^k(\bullet)$ is a function which is the equivalent waiting length of the k-th UAV flying from the i-th city to the j-th city under the solution •; L_{Lim}^k and N_{Lim}^k are respectively the maximum flying distance and the maximum limitation of the task number of the k-th UAV.

The objective function (1) minimizes the total weighted cost which represents two main components: failure rates and flying costs. We will further show how to compute it in Section III.C. For $\forall$ i, $j \in$ {1, 2, …, N_v+3N_t} and $\forall$ $k \in$ {1, 2, …, N_v}, there are 6 constraints: The constraint (2) ensures that every city is visited for once, which is the traditional constraint of the m-TSP problem. The constraint (3) ensures that the UAVs come in and the same UAVs go out for each visited city. The constraint (4) indicates that the i-th UAV can only take off and land at its own starting points, and also ensure that it cannot fly from i-th city to the same one. The constraint (5) is the other sub-tour elimination constraints (SEC) condition of the m-TSP problem. It is similar to the *node potentials*, which is introduced by Miller, C. E. *et al*. [32]. The effectiveness of the SECs is proved in Appendix A. These SECs introduce $O(n^2)$ auxiliary continuous variables u_m (m=1, 2, 3,…, 3N_t). In order to improve the computation speed, they had better meet $u_m \ge 0$. In a word, the above four constraints (2)-(5) can ensure that the paths of all UAVs are connected and flyable. The other two constraints (6)-(7) are the limit of the flying distance and the task number respectively. According to this model, its variable dimension is N_v (N_v+3N_t)^2, which means that this problem will become very difficult to solve, when N_v and N_t increase.

C. Computation of the objective function

In (1), the shortest feasible path $L_{i,j}^{k}$ is easy to be computed by the coordinates of the *i*-th city to the *j*-th city based on the CLC and CC paths. The equivalent waiting length $f_{i,j}^{k}(\{X_{i,j}^{k}\})$ of the *k*-th UAV flying from the *i*-th city to the *j*-th city is obtained by:

$$f_{i,j}^{k}\left(\{X_{i,j}^{k}\}\right)=v_k\hat{t}_k\left(\left\{X_{i,j}^{k}\right\}\right) \tag{8}$$

$$\hat{t}_k\left(\left\{X_{i,j}^{k}\right\}\right)=T_j\left(\left\{X_{i,j}^{k}\right\}\right)-T_i\left(\left\{X_{i,j}^{k}\right\}\right)-\frac{L_{i,j}^{k}}{v_k} \tag{9}$$

where v_k means the nominal velocity of the *k*-th UAV, $\hat{t}_k$ is the waiting time of the *k*-th UAV, $T_i(\{X_{i,j}^{k}\})$ and $T_j(\{X_{i,j}^{k}\})$ are respectively performing time of the *i*-th and *j*-th cities.

If two UAVs perform different tasks of the same target, one UAV always waits for the other one. So the waiting time of the *k*-th UAV is not independent, it is calculated based on the whole solution $\{X_{ij}^{k}\}$ coordinating the tasks of the other UAVs. For the large-scale problem, the waiting situations are intricate, involving many different UAVs. The key to computing the waiting time is to put all UAVs in a uniform timeline. The computing steps are summarized as follows: First, if the performing task sequence of the UAV is $<a_i, a_{i+1}, \ldots, a_j>$, their performing time will initially be: 0, …, $\sum_{m=0}^{l} L_{a_{i+m},a_{i+m+1}}^{k}/v_k$, …, l=0, 1, …, j-i-1. And then, by a circulatory test operation, all tasks of the same target are found out. Meanwhile, their performing time is coordinated. The minimum interval of every two tasks is set as T_c, which is the fixed performing time of the previous task and does not include the flying time. As an example, if the performing time of the response task of one target is $T_i(\{X_{i,j}^{k}\})$, the verification task of the same target will be later than $T_i(\{X_{i,j}^{k}\})+T_c$. If the performing time of one task changes because of this limit, the same changed time will also be added to all the performing time of the tasks of the same UAV after this task. With the finite circulation, the performing time of all UAVs can be obtained. The pseudo-code of this process is shown in Algorithm 1.

Algorithm 1 whole computing process of the objective function

```
Input: an candidate solution.
Output: the objective function.
1:  Based on the candidate solution, build a matrix $C_{Nv\times(Nt-Nv+1)}=[c_{i,j}]_{Nv\times(Nt-Nv+1)}$ whose
    rows mean the order of the UAVs and columns represent its tasks by order;
2:  Obtain a mirror matrix $C'_{Nv\times(Nt-Nv+1)}$ of $C_{Nv\times(Nt-Nv+1)}$ whose elements $c'_{i,j}$ is the
    corresponding initial performing time by the general term formula in Section III.
3:  $test$= 0; %%a variable is used to mark whether the coordinated process is finished.
4:  while($test$==0)
5:      $test$= 0;
6:      Find out the elements whose target is the same by traversing search;
7:      if $c'_{i,j}$ and $c'_{i',j'}$ represent the performing time of two different tasks of one target
        and $c'_{i,j}$ prior to $c'_{i',j'}$
8:          if $c'_{i,j}+nT_c>c'_{i',j'}$
9:              $c'_{i',j}=c'_{i,j}+nT_c$, $n$=1, 2;
10:             Add $c'_{i,j}+nT_c-c'_{i',j'}$ to all non-zero elements in this row after this element.
11:             $test$=1;
12:         end
13:     end
14: end
15: %%Update the performing time of the first task of each UAV which can be
modified by changing take-off time.
16: for $i$=1:1: $N_v$
17:     if $c_{i,1}$ is the classification or response task and $c_{i,2}$ is the other task of the
        same target
18:         $c'_{i,1}=\min(c'_{i,2}-L_{c_{i,1},c_{i,2}}^{k}/v_k, c'_{i,2}-nT_c)$;
19:     else
20:         $c'_{i,1}=c'_{i,2}-L_{c_{i,1},c_{i,2}}^{k}/v_k$;
21:     end
22: end
23: Obtain the take-off time of the $k$-th UAV by $c'_{i,1}-L_{c_{i,1},c_{i,2}}^{k}/v_k$;
24: Obtain the landing time of the $k$-th UAV similarly;
25: Compute the objective function by equation (1), (8) and (9);
```

Notes 1: The operational characters max(), min() and %% respectively mean the maximum, minimum and annotation.

Notes 2: The value of the variable n is based on the types of the tasks. If they are classification and verification tasks, n=2. And the other situations, n=1.

The above approach helps to obtain the objective function of a solution. In fact, a deadlock problem, which makes the objective function infinite, is ignored in this approach. In next section, we will present a novel graph-based method to solve this problem.

IV. Novel Digraph-Based Method for Solving Deadlocks

The constraint (6) is indeed to limit the flight time of all UAVs. However, if the candidate solutions fall into deadlock situations, without this prerequisite or implicit condition of the task assignment problem, the optimization results will be invalid. To solve this problem, a novel digraph-based method is presented in this part.

First, we need to judge a candidate solution whether the deadlock situations exist or not. As mentioned previously, the candidate solutions can be seen as a digraph G= (V, E). It is noted that in this part the vertexes corresponding to the starting points of the UAVs of each solution are deleted, because the complete UAV flying process of taking off and landing at its own starting points will not lead to the deadlock problem. Hence, it can be expressed as adjacency list and adjacency matrix. Each candidate solution can correspond to a digraph G_V based on different UAVs. Meanwhile, the task-executing orders (classification is prior to response and response is prior to verification) of each target, which is independent of all candidate solutions, can also generate a target-based digraph G_T. This target-based digraph, which is inherent and fixed, is defined based on the mathematical model of the task assignment problem. So, essentially, the whole digraph G_A of the task-executing orders for a candidate solution meets $G_A=G_V\bigcup G_T$. Based on Property 2 of [17], we know that if a solution is not deadlocked, its corresponding whole digraph G_A is acyclic. Hence the detection of the deadlock problem is equivalent to the ringed detection of its whole digraph G_A. In graph theory, the famous Depth First Search (DFS) algorithm is often used to detect ring structures in a digraph. If the digraph G_A is deadlocked, the DFS algorithm can at least output one ring structure and save the adjacency list $<v_i,\ldots, v_k,\ldots, v_i>$ into a vector, where $< v_i, v_j>$ means the time-adjacent relation of the vertexes v_i and v_j. By this way, the deadlock detection is finished by its digraph and the DFS algorithm.

Property 1: If a solution is deadlocked, its deadlock circuit at least includes an edge which is not in the target-based digraph G_T [17].

Then, we need to resolve the deadlock problem. The types of the deadlocks include self-locks, multiple deadlocks, and a mixture of both. The self-lock means that a latter task is sorted in front of the former one when more than one task of a target is assigned to a UAV. The self-lock is easy to solve, so we need to solve it first. In most situations, the deadlocks of a solution are multiple deadlocks and the mixture of both. After solving all these self-lock problems, we need to solve them.

Finally, by the help of the DFS algorithm, one random deadlock circuit is saved in a vector. Based on **Property 1**, we can get that each deadlock circuit at least includes an edge which is not in the target-based digraph G_T. These edges, named free edges, are adaptable, and then one free edge in the first deadlock, which is obtained by the DFS algorithm, is picked out randomly. This free edge will be transposed. By this way, the main structure of the solution is retained, and meanwhile, this deadlock circuit is resolved. It is noted that in this process a new deadlock circuit caused by this transposition operation may appear. Even though the deadlock problem is an NP-hard problem, if the number of the targets is determined, the deadlock problem can be solved by finite above operation [18]. Therefore, the transposition operation needs to be repeated many times until the deadlock problem is solved.

The whole process is given in Algorithm 2. It is similar to the algorithm proposed in [17]. The main difference is that the matrix operation is replaced by several new edge operations in order to reduce the difference between the deadlock initial solutions and the deadlock-free solution.

Because of this algorithm, the problem (1)-(9) becomes solvable. In the next section, we will present a new algorithm to solve this combinatorial optimization problem.

V. New Modified Two-part WPS Algorithm

A. Two-part Individual Representation Technique

Based on [33], the original WPS algorithm cannot be applied to the multi-UAV task assignment problem directly. In this section, we propose a novel WPS algorithm with a two-part individual representation, named TWPS algorithm, to solve this problem. The two-part individual representation technique is inspired by the two-part chromosome technique in [34] and [35]. In this technique, the wolf is divided into two parts: the first part is a permutation of $3N_t$ cities and the second part is the number of cities assigned to the corresponding salesman [35]. The example is shown as follows (see Fig. 1):

Algorithm 2 whole deadlock detection process and deadlock-free process

```
Input: an initial assignment.
Output: an unlocked solution.
1: construct $G_V$, $G_T$, $G_A$;
2: use DFS algorithm to check $G_A$, output the deadlock matrix;
3: if the deadlock matrix is not deadlocked
4:     while (deadlock matrix is not empty)
5:         Check every UAV task sequence for solving self-locks of an input solution, and switch
           self-lock vertices;
6:         Update $G_V$ and $G_A$;
7:         Find all free edges, and record their corresponding numbers of occurrences;
8:         Sort the numbers of occurrences and output the most frequent one and its occurrence
           number;
9:         if the occurrence number is bigger than 1
10:            Transpose the order of the two vertexes of the edge;
11:        else
12:            Create random number and transpose the order of the two vertexes of this random edge;
13:        end
14:        Update $G_V$ and $G_A$;
15:        Use DFS algorithm to check $G_A$, output the deadlock matrix;
16:    end
17: end
```

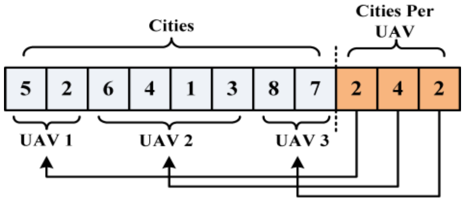


Fig. 1 Example of the two-part individual representation technique for 8 cities with 3 UAVs

Property 2: The mapping from the candidate solutions to the two-part individuals is a one-to-one mapping.

Proof: The proof of this result is shown in Appendix B.

By this way, this representation which is exclusive for each valid solution reduces the redundant solution in the solution space furthest. Without the limitation of the number of tasks, the size of the solution space for the two-part individual is $(3N_t)!\binom{3N_t-1}{N_v-1}$ [34]. If there is a limitation of the number of tasks and the flight distance, the size of the solution space will become smaller. The reduction is decided by the maximum limitation of the number of tasks N_{Lim}^k and the maximum flying distance L_{Lim}^k. Because their effects are similar, in the following part, we will only discuss the reduction caused by N_{Lim}^k:

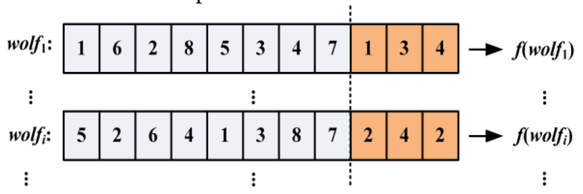
If $N_{Lim}^k \geq 3N_t - N_v + 1$, this limitation will become inoperative. The size of the solution space will remain unchanged. If $N_{Lim}^k < 3N_t/N_v$, the limitation of the number of tasks will cause that the result of the model is insoluble because of the constraint (2). If $N_{Lim}^k \in [3N_t/N_v,\ 3N_t-N_v+1)$, l=1, 2,…, N_v-1, the size of the solution space will be greatly affected based on N_{Lim}^k.

B. Repair operations for Two-part WPS algorithm

In this section, we repair the original WPS algorithm by the two-part individual representation. The original WPS algorithm imitating the hunting activity of a wolf group consists of the following steps: 1) Initialization – an initial wolf pack; 2) Fitness – evaluate the objective functions of the whole pack; 3) Elitism – select a more appropriate small group, whose members are more experienced and stronger than the others, in the pack; 4) Safari – optimization in a small scale around the members of the elite group; 5) Update – every wolf gets close to the best wolf; 6) Replacement – replace old weak wolves by new ones; 7) Loop – go back to step 2 [33]. The circulation will stop if an ending condition is satisfied.

An example is used below to illustrate the process of using TWPS algorithm to search the optimal solution with 8 cities and 3 UAVs. First, after the initialization step, there are N wolves, which are one-dimensional arrays $\{wolf_i$, i=1, 2, ..., $N\}$, satisfying the structure of the two-part individual representation technique. Secondly, the objective function of each wolf is computed and sorted. We can obtain a small elite wolf group with N' wolves. Thirdly, these elites make small local optimizations around themselves. The local search scale R is defined as the number of repetitions of the safari process. The safari process of the elites is defined by the following two steps: a. As shown in following Fig. 2-6, we randomly select an element {4} from the first part of i-th wolf. Two new wolves will be generated by changing the order between {4} and its two adjacent elements {6} and {1}, then the objective functions of these three wolves will be computed and sorted. The best one will replace the original one; b. Then, two elements in the second part are chosen randomly to add <1, -1> and <-1, 1> to them. If the original element is 1 or $3N_t$-N_v+1, they only need to add <1, -1> or <-1, 1> instead of both of them. Similarly, the objective functions of these three wolves will be computed to obtain the optimal one to replace the best wolf *Gbest*. The best wolf *Gbest* is assumed to be <1, 2, 3, 4, 5, 6, 7, 8, 2, 2, 4>. Fourthly, every wolf gets close to the best wolf *Gbest* by a rate *Step*. The approach rate *Step* means repeating the following operation for *Step* times in a generation. Let *Step* be 1. This specific operation is divided into two parts: the approach of the first part and the second part. In the first part of the i-th wolf, an element {3} is chosen randomly to seek its position in the best wolf *Gbest*. Its order in i-th wolf is 6, which is bigger than its order in the best wolf *Gbest*, so the approach of the first part means changing the order <1, 3, 8> into <3, 1, 8>. Subsequently, in order to finish the approach of the second part, the numbers of cities assigned to the corresponding UAVs are classified into 3 groups (bigger, smaller, and equal) compared with the corresponding elements in the best wolf *Gbest*. The approach way is to add 1 to a random element in the smaller group and to subtract 1 from a random element in the bigger group. Just like the example, the equal group, which has no operation, is {2} for the first UAV. The smaller group is {2} for the third UAV and the bigger group is {4} for the second UAV. So the original sequence changes from <2, 4, 2> to <2, 3, 3> by adding <0, -1, 1>. Fifthly, the objective functions of all new wolves are computed and compared with the original ones. Next, the original best wolf *Gbest* and the last N^* old weak wolves are replaced respectively by a new wolf $wolf_{new}$ whose objective function $f(wolf_{new})$ is better than $f(Gbest)$ and N^* new random wolves. Finally, repeat step 2 to step 5 until the ending conditions are met [36].

Step 1 and 2: The initialization and objective function evaluation of the wolf pack.



wolf1: 1 6 2 8 5 3 4 7 1 3 4 → f(wolf1)
wolfi: 5 2 6 4 1 3 8 7 2 4 2 → f(wolfi)

Fig. 2 Example of step 1 and 2

Step 3.a: The safari of the first part of the elites.

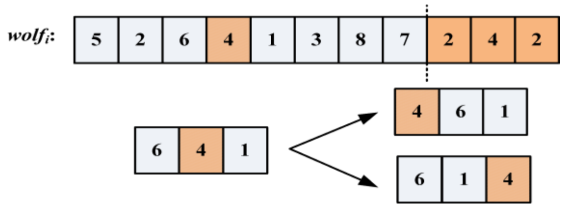


Fig. 3 Example of step 3.a

Step 3.b: The safari of the second part of the elites.

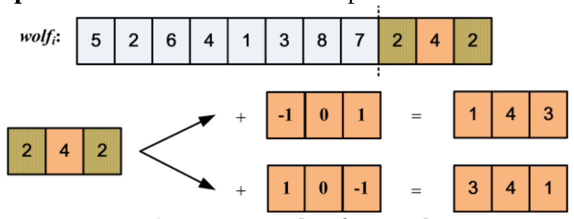


Fig. 4 Example of step 3.b

Step **4.a**: The approach of the first part of each wolf.

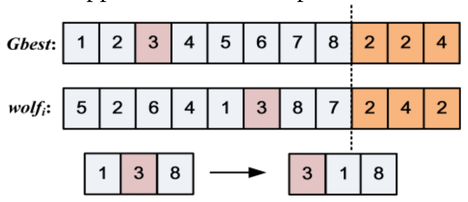


Fig. 5 Example of step 4.a

Step **4.b**: The approach of the second part of each wolf.

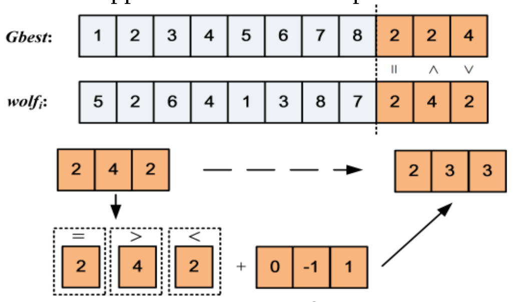


Fig. 6 Example of step 4.b

In general, the computation of the objective function is the time-consuming operation in most optimization problems. In a generation, the TWPS algorithm needs to compute the objective functions of the wolves (N+2RN*, N+4RN*) times. In order to compare with other algorithms fair, the termination generation of the TWPS needs to be reduced.

C. *Modified Two-part WPS algorithm*

In this section, a modified two-part wolf pack search (MTWPS) algorithm with similar computational complexity is presented to improve the diversity of the wolf pack. Even though the safari operation is a good local optimization operation, its search effect is still too small. So we present two thoughts to modify the MTWPS algorithm. The first one is that a new transposition and extension (TE) operation is introduced to replace the safari operation of the TWPS algorithm, named MTWPS1. The other one is to add the TE operation after the safari operation, named MTWPS2. Then, we will introduce the transposition operation and the extension operation.

For each wolf, the transposition operation is used in the first part of the elites, which is shown in Fig. 7. It is to obtain a new wolf *wolf'*$_i$ by inverting the order of elements between two random elements <2, 6, 4, 1, 3> into <3, 1, 4, 6, 2>. Then the cost functions of the old wolf *wolf*$_i$ and the new wolf *wolf'*$_i$ are computed and compared to choose the better one. Finally, the better one will replace the old wolf *wolf*$_i$. This simple operation can easily break through the limitation of the approach operation to enhance the diversity of the first part of the elites. Meanwhile, this operation can easily untie the intersectant route and maintain the tour structure [36].

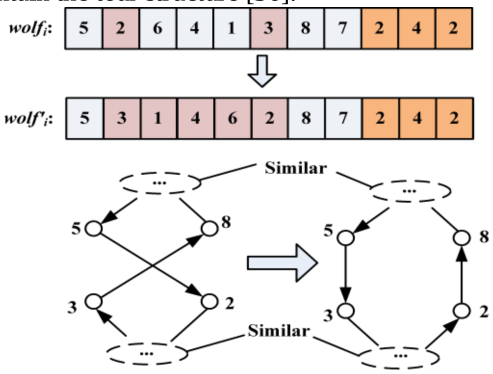

wolf$_i$: 5 2 6 4 1 3 8 7 2 4 2
wolf'$_i$: 5 3 1 4 6 2 8 7 2 4 2
Similar
Similar
...
5
8
3
2

Fig. 7 The transposition operation of the MTWPS algorithm

For the second part of each elite, an extension operation is introduced to greatly enlarge the search space, which is shown in Fig. 2 as an example. First, a set A={2, 3, 4} which means the limitation of the second part is generated by a random element {4} of an elite *wolf*$_i$ and its corresponding element {2} of the best wolf *Gbest*. The universal set U in the second part based on the constraint of m-TSP is {1, 2, 3, 4, 5}. Subsequently, the complementary set $C_U{}^A$={1, 5}, which means the lost solution space, is obtained by A and U. Then, an element {5} in $C_U{}^A$ is selected out randomly to create a new second part for the new wolf *wolf'*$_i$. The other elements in the second part are generated randomly as well as meeting the constraint $b_1+b_2+\ldots+b_{Nv}=3N_t$. Finally, the new wolf *wolf'*$_i$. is compared with the original wolf *wolf*$_i$ to replace it by the better one.

Compared with the TWPS algorithm, the objective functions of the wolves in the MTWPS1 and MTWPS2 algorithm respectively need to be computed N+2RN* times and (N+4RN*, N+6RN*) times in a generation.

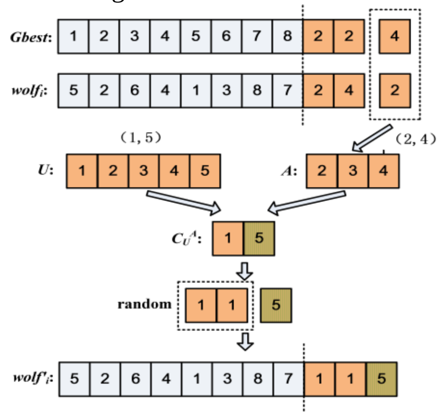

Gbest: 1 2 3 4 5 6 7 8 2 2 4
wolf$_i$: 5 2 6 4 1 3 8 7 2 4 2
(1, 5)
(2, 4)
U: 1 2 3 4 5
A: 2 3 4
$C_U{}^A$: 1 5
random 1 1 5
wolf'$_i$: 5 2 6 4 1 3 8 7 1 1 5

Fig. 8 The extension operation of the MTWPS algorithm

VI. New Solution Frame for the UAV Task Assignment Problem

Even though using the previous methods we can solve the task assignment problem, in this section we try to present a new solution framework to ensure that the system has a good timeliness, which helps to improve its practicability.

Based on the objective function shown in (8)-(9), we have:

Property 3: If the candidate solution is not deadlock-free, the coordination circulations and the objective function of this solution using Algorithm 1 will be infinite.

This property is obvious. So if you want to use this objective function, the deadlock-free process is indispensable. However, when the scale of the problem grows, this process will be extremely time-consuming. Hence, it is badly in need of a new solution frame based on two different objective functions for dealing with the small and large-scale problem. The new solution frame is shown in Fig. 9.

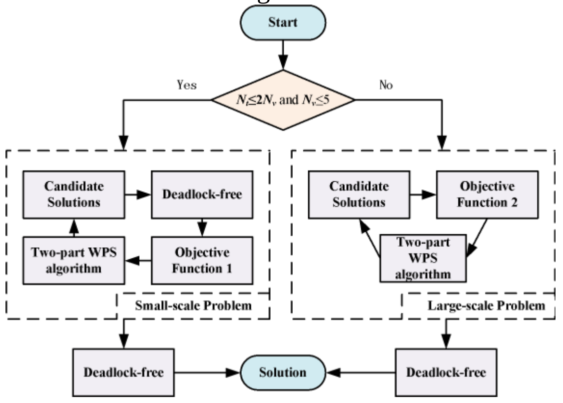


Fig. 9 The new solution framework

As shown in Fig. 9, the new framework is presented based on the size of the problem. The small and large-scale problems are classified by the inequalities: $N_t \le 2N_v$ and $N_v \le 5$.

For the small-scale problem, the original objective function (1), (8), and (9) are applied in the optimization process. For the large-scale problem, the new objective function is defined as:

$$\min_{\{X_{i,j}^k\}} \sum_{i=1}^{N_v+3N_t} \sum_{j=1}^{N_v+3N_t} \sum_{k=1}^{N_v} P_{k,mod((j-N_v),3)} \left(L_{i,j}^k + \tilde{f}_{i,j}^k(\{X_{i,j}^k\}) \right) X_{i,j}^k \quad (10)$$

$$\tilde{f}_{i,j}^k\left(\{X_{i,j}^k\}\right) = l' \sum_{i=1}^{N_v+3N_t} \sum_{j=1}^{N_v+3N_t} X_{i,j}^k, \quad , \quad (11)$$

$$\forall i, j \text{ meets } |i-j| \ge 3 \cap i > N_v \cap j > N_v$$

where l' means the penalty length which is used to imitate the average of the equivalent waiting length.

In general, the deadlock problem occurs when three tasks of one target are finished by different UAVs. In this case, one UAV always needs to wait for other vehicles. We can see that the new objective function (10) and (11) is to obtain the times of the cases when more than one UAV performs the same target. It is easy to find that the computational process of this new function is independent of the deadlock-free process, which helps to improve the timeliness of the algorithm.

Let's set the average solving time of the deadlock-free process as $f_{st}(N_t, N_v)$, where f_{st} is not a polynomial order function, then we can save O(N_{max}(N+6RN*)f_{st}(N_t, N_v))

simulation time using this new framework and MTWPS2 algorithm, N_{max} means the maximum iterations.

Because the deadlock-free process just is used for the final solution, there exists a performance degradation phenomenon, which is the inherent drawback of the new frame. In short, the new framework can pertinently solve the offline task assignment problem with some compromises in performance.

VII. The Uncertainty Problem in UAV Task Assignment problem

The previous methods can only solve the deterministic problem without uncertainty. In this section, we will introduce the robust optimization method into the multi-UAV task assignment problem with different kinds of uncertainty.

A. Robust Optimization Method for the multiple UAVs Task Assignment Problem

As a forethought way, the robust optimization method is to consider the parameter uncertainty before the startup of the real system. For real task environments, the failure rates $P_{k,l}$ of the UAVs change with the task environment and the working condition. Therefore, it is very difficult to obtain its constant value and probability distribution. We can only obtain an approximate range of this coefficient without an accurate probability distribution by the statistical approach or the expert system. So if we regard $P_{k,l}$ as a random variable, the problem modeled as (1)-(9) is an optimization problem with uncertain objective function coefficients, and meanwhile, the probability distributions of coefficients are unknown.

For discrete optimization problems, when both the cost coefficients and the data in the constraints of an integer programming problem are subject to uncertainty, the approach in [26], they propose a robust integer programming problem of moderately larger size that allows controlling the degree of conservatism of the solution in terms of probabilistic bounds on constraint violation. It's obvious that the objective functions of the model in this paper are both not linear. In order to use the robust optimization framework presented by Bertsimas and Sim, the equivalent waiting length $f_{i,j}^k(\{X_{ij}^k\})$ is simplified into a random number, which is in a small range with an unknown probability distribution. Under these circumstances, the robust optimization method is suitable for this problem.

Let: $\bar{X}_m = X_{i,j}^k$, $c_m = P_{k,l}\,(L_{i,j}^k + f_{i,j}^k(\{X_{i,j}^k\}))$, equation (1)-(9) can be simplified as:

$$\min_{\{\bar{X}_m\}} \sum_{m=1}^{(N_v+3N_t)^2 N_v} c_m \bar{X}_m$$
$$s.t. \begin{cases} g_p(\bar{X}_m) \le 0, \\ h_q(\bar{X}_m) = 0. \\ m \in IJK \end{cases}, \tag{12}$$

where $IJK=\{1, 2, \ldots, (N_v+3N_t)^2 N_v\}$; p, q are respectively the serial numbers of the inequality and equality constraints; $g_p(\bar{X}_m) \le 0$ and $h_q(\bar{X}_m) = 0$ respectively represent the inequality and equality constraints in (2)-(7).

Because $P_{k,l}$ and $f_{i,j}^k(\{X_{i,j}^k\})$ are reduced to two random numbers, we can get that c_m takes values in $[c'_m, c'_m + dc_m]$, where $dc_m>0$. Using the robust optimization and Langrange duality theory, we can get an equivalent mixed integer programming formulation is shown as follows:

$$\min_{\{\bar{X}_m, \nu_m, \lambda\}} \sum_{m=1}^{(N_v+3N_t)^2 N_v} c_m \bar{X}_m + \sum_{m\in I_0} \nu_m + \lambda\Gamma_0$$
$$s.t. \begin{cases} g_p(\bar{X}_m) \le 0, \\ h_q(\bar{X}_m) = 0, \\ \lambda + \nu_m \ge dc_m \bar{X}_m, & \forall m \in I_0 \\ \lambda \ge 0, \\ \nu_m \ge 0, & \forall m \in I_0 \\ m \in IJK. \end{cases}, \tag{13}$$

where $I_0=\{m|dc_m>0\}$, $\Gamma_0 \in [0, |I_0|]$, λ, μ_m and ν_m are the Lagrange multiplicative derivations, and they meet $\lambda\ge0$, $\mu_m\ge0$ and $\nu_m\ge0$. The derivation of (13) from (12) is given in Appendix C.

By this way, the original problem (1)-(7) with parameter uncertainty is transformed into a new, constant and solvable mixed integer programming problem. However, we can see that there are $|I_0|+1$ new variables being introduced into the new optimization problem, which increases the solving difficulty of the new problem. For the small-scale problem, the branch and bound method in the Lingo software is applied to compute this problem. For the large-scale problem, we present a novel combined algorithm including the classical interior point method and our MTWPS algorithm.

Because the new problem is a mixed integer programming problem, if the part of the integer programming problem is separated, it will degrade into a linear programming model, which can be easily computed by the interior point method. According to this idea, the degraded linear programming model and the classical interior point method are applied to compute the objective function of the candidate solution in the MTWPS updated process. Once the candidate solution is determined, the degraded linear programming model is:

$$\min_{\{\nu_m, \lambda\}} \sum_{m\in I_0} \nu_m + \lambda\Gamma_0$$
$$s.t. \begin{cases} \lambda + \nu_m \ge dc_m \bar{X}_m, & \forall m \in I_0 \\ \nu_m \ge 0, & \forall m \in I_0 \\ \lambda \ge 0. \end{cases} \tag{14}$$

We know that the accurate optimal solution of this new problem can be obtained by the interior point method, even for the large-scale problem. Then, the objective function of the candidate solution is available, which makes the whole updated process complete. Finally, this problem (13) can be solved no matter in large-scale or small-scale situations (See Fig.10).

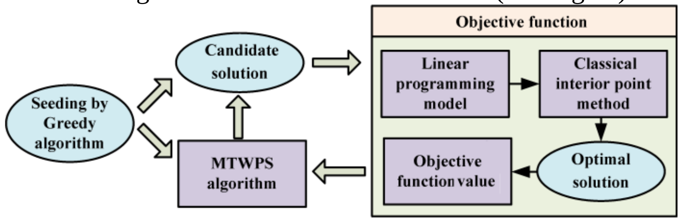


Fig. 10 The solution frame of the robust optimization problem

Because of the solving of the degraded linear programming problem, the progress of the objective function becomes much more time-consuming. In order to improve its convergence rate, the solutions of the greedy algorithm are regarded as the initial solutions and the candidate solutions of the replacement step in the MTWPS algorithm, named seeding by the greedy algorithm. The greedy algorithm used in our paper will be further presented in Section VIII. Of course, for the large-sized task assignment problem (1)-(7) without uncertainty, if you cannot tolerate the performance degradation of the new framework, you can also use this seeding way.

B. Online Hierarchical Planning Algorithm for the UAV Task Assignment Problem

When the time-sensitive targets appear, the robust optimization method is invalid. A practical online hierarchical planning algorithm is presented in this part. It is composed of a small-scale centralized planning algorithm and an exchange-bidding algorithm. The key point of this algorithm is to reduce the planning time and improve the real-time performance. The optimality of the online result, which cannot be guaranteed in most online algorithms, is considered furthest based on a decision variable for the remaining time.

As time-sensitive targets, their appearance and discovery are unpredictable in the flying process. Hence, if they are found in the initial stage, we will have enough time to obtain the optimal or sub-optimal solution. Otherwise, we had better offer a high-efficiency algorithm without considering the optimality.

The starting time and the finish time of the whole UAV system and the detection time of the time-sensitive targets are respectively set as 0, T' and t_1, which meet $0 \le t_1 < T'$. If t_1 meets $t_1 \in [0, 0.3T']$, the online centralized planning algorithm will be used. The planning time T_p of this algorithm is limited to $min(T'-t_1, 0.3T')$. For every time-sensitive target, it has three new tasks: the classification, response, and verification tasks. Similar to the mirror representation of the original targets in Section III, every time-sensitive target is regarded as three cities in the m-TSP.

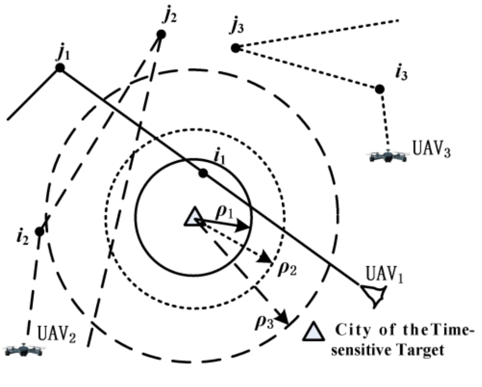


Fig. 11 The simplified judgment way

Because the UAVs need to finish their original tasks, the resources of the UAVs have decreased, which leads the result that some UAVs do not have enough resources to complete the tasks of these new cities. We present a judgment way to judge whether the UAV has the ability to finish the new tasks.

For every city, a simplified judgment way is shown in Fig. 11. In Fig. 11, the cities i_1, i_2, i_3, j_1, j_2 and j_3, which have not been visited yet before t_1+0.3T', are the off-line planning results. Three different paths including one full line and two dash lines are the unexecuted paths of three UAVs, which will be followed by the UAVs after t_1+0.3T'. ρ_1, ρ_2 and ρ_3 are the remaining working radii of three UAVs, meeting ρ_k=0.5 L^k_{Lim} - 0.5v_kT_c- 0.5 $\sum_{i=1}^{N_v+3N_t}\sum_{j=1}^{N_v+3N_t}\left(L^k_{i,j}+f^k_{i,j}(\{X^k_{i,j}\})\right)X^k_{i,j}$. With these three radii, draw three circles around the new city. If some parts of the paths of the k-th UAV locate in these circles and this UAV has enough task ability $N^k_{\text{Lim}} - \sum_{i=1}^{N_v+3N_t}\sum_{j=1}^{N_v+3N_t} X^k_{i,j} > 0$, it means that the k-th UAV has enough resources to visit the new cities. By repeating this judgment way 3N_t' times, for N_t' time-sensitive target, all UAVs, which have enough resources to solve all cities, can be found out by a simple program in a short time, and the number of them N_v' meets $N_v' \le N_v$. If too many UAVs meet this judgment way ($N_v' > N_v^*$), in order to limit the problem size, only the closest N_v^* UAVs are considered. Because of the limitation of the resources, if there are too many time-sensitive targets, we can only solve the earlier detected N_t^* targets. According to Property 4 and the objective function (8)-(9), there are N_v' UAVs and $min(N_t', N_t^*)$ cities in the new online solvable centralized planning problem.

Property 4: If the UAVs meet the above judgment way, there is at least one solution for the new task assignment problem with this new city.

Proof: The proof of this result is shown in Appendix D.

If t_1 meets $t_1 \in (0.3T', T']$, the online exchange-bidding algorithm will be used. The original bidding algorithm imitates the auction behavior based on the CNP. The main advantage of the bidding algorithm is its distribution and real-time performance. But because of the resource constraints of the UAV system, some UAVs cannot take part in the bidding process and cannot draw on each other's strengths, which leads to the reducing of the global efficiency. We introduce an online exchange-bidding algorithm which adds an exchange contract to the traditional bidding algorithm considering the time sequence requirement.

The process of the exchange-bidding algorithm is shown as follows: First, one UAV as a toastmaster of the auction puts the cities of the time-sensitive targets for auction one by one. In order to avoid the deadlock problem, the new city of the time-sensitive targets is assigned later than the original off-line tasks and the assigned on-line tasks. Meanwhile, the order of the assigned cities of one target meets the sequence request presented in Section VI. The city for bidding is denoted as $T_{i'} \in S$, $|S|$=3N_t', which means i'-th city of the universal set S of the cities for N_t' time-sensitive targets. Let us suppose that the original task sequence of the j-th UAV is < a_1,…, a_{bj}>, and then if it wins the auction, its new sequence will be < a_1,…, a_{bj}, $T_{i'}$>.

After that, all UAVs begin to show their abilities $U^{k'}(T_{i'})$, k'=1, 2,…, N_v for the i'-th city. The ability $U^{k'}(T_{i'})$ is defined as:

$$
\begin{aligned}
&U^{k'}(T_{i'}) = U_1^{k'}(T_{i'})U_2^{k'}(T_{i'})U_3^{k'}(T_{i'})\\
&U_1^{k'}(T_{i'}) = P_{k',mod(i',3)}\left(L_r^{i'} - \widehat{L}_{i'}^{k'}(T_{i'})\right)\\
&U_2^{k'}(T_{i'}) = \begin{cases} 1 & L_{\text{Lim}}^{k'} - \sum_{i=1}^{N_n}\sum_{j=1}^{N_n}\left(L_{i,j}^{k'} + f_{i,j}^{k'}(\{X_{i,j}^{k'}\})\right)X_{i,j}^{k'} \ge \widehat{L}_{i'}^{k'}(T_{i'}) \\ 0 & L_{\text{Lim}}^{k'} - \sum_{i=1}^{N_n}\sum_{j=1}^{N_n}\left(L_{i,j}^{k'} + f_{i,j}^{k'}(\{X_{i,j}^{k'}\})\right)X_{i,j}^{k'} < \widehat{L}_{i'}^{k'}(T_{i'}) \end{cases},\\
&U_3^{k'}(T_{i'}) = \begin{cases} 1 & N_{\text{Lim}}^{k'} > \sum_{i=1}^{N_n}\sum_{j=1}^{N_n} X_{i,j}^{k'} \\ 0 & N_{\text{Lim}}^{k'} = \sum_{i=1}^{N_n}\sum_{j=1}^{N_n} X_{i,j}^{k'} \end{cases}\\
&L_r^{i'} = \sum_{k'=1}^{N_v} L_{\text{Lim}}^{k'} - \sum_{i=1}^{N_n}\sum_{j=1}^{N_n}\sum_{k'=1}^{N_v}\left(L_{i,j}^{k'} + f_{i,j}^{k'}(\{X_{i,j}^{k'}\})\right)X_{i,j}^{k'}\\
&N_n = N_v + 3N_t + i',\ \ i' = 1,2,\ldots,3N_t'
\end{aligned}
\tag{15}
$$

where i' means that the complete auction progress is repeated i' times, which means that i' cities of the time-sensitive targets have been assigned. $\widehat{L}_{i'}^{k'}(T_{i'})$ is the additional flying cost to finish the i'-th city of the time-sensitive target. We can see that the UAV can only use the surplus resource to compete for the new city. For the novel exchange-bidding algorithm, the UAVs, whose last tasks are the verification tasks, are allowed to give up their original last verification tasks, and then, use the obtained and surplus resource to compete for the new city. Of course, the triggering condition of the exchange operation is that the ability of the whole UAVs system needs to be improved to some extent. Meanwhile, because the original last task is abandoned, this task needs to be auctioned in i'+1 auction progress and add one to the repeated number of the auctions. Finally, all abilities of the UAVs are sorted and the largest one obtains the performing right of the i'-th city. It is noted that, in order to avoid too many times of the exchange progress, the total times of the exchange progress are limited to N_v times.

VIII. Simulation and Experimentation

A. *The deterministic UAV Task Assignment Problem*

In this part, the simulations of the offline deterministic UAV deadlock problem are performed. First, the effectiveness of the deadlock detection and deadlock-free method is presented focusing on the increase of the problem size. Then, several optimization methods are applied and compared. Finally, the simulation results of the new framework are compared for the problems with different sizes.

8.1.1 Deadlock detection and deadlock-free simulation

The first part of the experiments is to straightforward examine our deadlock detection and deadlock-free method in a small-sized problem with N_v=5 UAVs and N_t=10 targets. The simulation results are stated in Table. 1.

Table 1. Deadlock detection and deadlock-free test results

Amount	Success rate	deadlock-free	self-lock	Average similarity degree	Unlocked solutions
100	100%	1701	442	48.03%	0
200	100%	3313	947	47.58%	0
500	100%	8599	2310	49.13%	1
1000	100%	17255	4653	49.23%	1
5000	100%	87164	22552	49.08%	9

We randomly generate different amounts (100, 200, 500, 1000, and 5000) of the initial solutions for the deadlock detection and deadlock-free test. In Table. 1, we can see that all the deadlock solutions are found out and transformed into unlocked solutions successfully, which adequately certifies the effectiveness of our algorithm. The total times of the deadlock-free operations and the times of the self-lock operations are shown in the third and fourth columns. It shows that the operations for self-locks are only a small part of all deadlock-free operations. Beyond that, because the deadlock-free operation will change the sequence of the solution, in order to show the difference between the initial solutions and the deadlock-free solutions, the average similarity degree results are shown in the fifth column. The similarity degree is defined as the ratio of the number of the same cities between the initial solutions and the deadlock-free solutions to $3N_t$ in the two-part individual representation. For example, if one initial solution is <1, 2, 3, 4, 6, 5, 3, 3> which has N_v=2 UAVs and N_t=2 targets and its corresponding deadlock-free solution is <1, 2, 3, 4, 5, 6, 3, 3>, the similarity degree will be 4/(3N_t)≈66.66%. In Table. 1, the average similarity degree approximates 50%. In fact, it is a very high similarity degree in the deadlock problem, which means that our algorithm can keep the original features of the solution as many as possible. Finally, the numbers of the unlocked solutions before the deadlock-free progress are shown in the final column. We can learn that the deadlock problem is almost universal for this problem. Almost all of the randomly generated solutions are deadlocked, which means that this problem cannot be ignored in the optimization progress.

The next experiment is to discuss the effect of the problem sizes. The amounts of the initial random solutions of all situations with different N_v and N_t are set as 1000. Then, the simulation results are shown in Fig. 12.

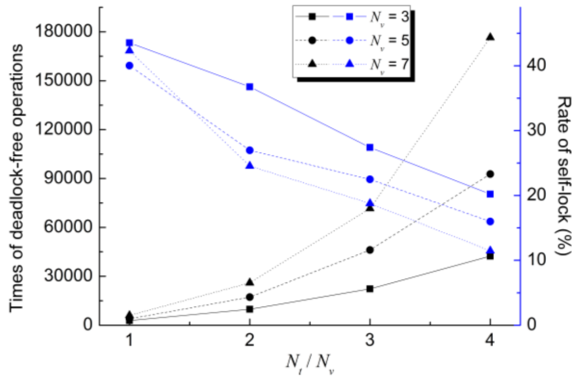


Fig. 12 The deadlock detection and deadlock-free method

In Fig. 12, it directly indicates that with the growth of the problem size, the deadlock-free operations become much more frequent and time-consuming. Meanwhile, the rate of the self-lock becomes lower when the size of the problem grows. In other words, the share of the multiple deadlocks, which is complicated, increases. Beyond that, all solutions are un-locked after using our algorithm. So the success rate is 100%.

8.1.2 Simulation for several optimization methods

In a small-sized problem with N_v=5 fixed-wing UAVs and N_t=10 targets, it can be regarded as the m-TSP problem with 35 cities. Every city has its phase, including coordinates and the best task angle, the whole planning space is shown in Fig. 13. The velocities of these UAVs are respectively 100m/s, 50m/s, 80m/s, 30 m/s and 50 m/s, as well as their radii are all 100m. Except for the 5-th UAV, the failure rates of others for the classification, response and verification tasks are 30%, 20% and 10%. The failure rates of the 5-th UAV are set as 10%, 10% and 10%. The maximum flying distances of all UAVs L_{Lim}^k are respectively set as 20000, 20000, 20000, 20000, and 25000. The maximum task number of the UAVs N_{Lim}^k is limited to being equal to or less than 10. If these two conditions are not satisfied, the penalty function will be added in the objective function.

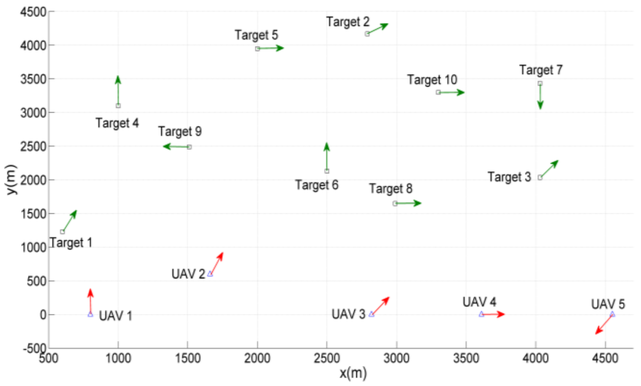


Fig. 13 The initial phases of the UAVs

Based on this planning space, the TWPS method, the GA with TCX method [35], the modified GA with multi-type genes (MGA) [10], the MTWPS1 method, the MTWPS2 method, the greedy algorithm, and the random search (RS) way are introduced to solve this task assignment problem. The greedy algorithm is similar to [35]. The first non-home cities of the salesmen in these solutions are stochastic, and then the greedy solutions are generated based on the present location of all the salesmen to find the unassigned city that has the optimal objective function for one of the salesmen. The optimal unassigned city is then assigned to the corresponding salesman. This process continues until all cities are assigned.

The initialization of the parameters of all search algorithms is as follows: the population size is set as 40(N=40); The maximum iteration is 10000. Except for these common parameters, their unique initial parameters are shown as follows: The mutation probability of the GA with TCX method and the MGA is 5%. The approach rate *Step*, the population of the small elite wolf group N' and the old weak wolves N* are 1, 10, and 4 respectively. Because of the randomness of the parameters, the simulation results are usually different even for the same method. If the above factors are taken into consideration, the simulations need to be repeated many times. The comparative experiments repeatedly run (20 times) for every method. The simple statistical results are shown in Fig. 14.

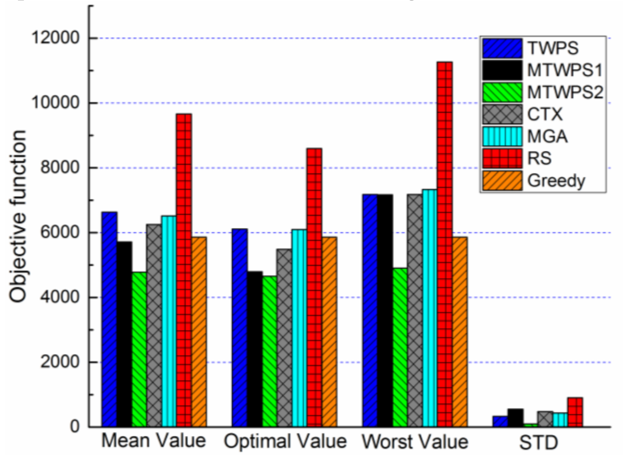


Fig. 14 The statistical results of these algorithms

We can see that the optimal values, the worst values and the mean values of the objective functions of the MTWPS2 algorithm are better than the ones of the other algorithms obviously. For the standard deviation (STD), except that the STD of the greedy algorithm is 0, the STD of the MTWPS2 algorithm is smaller than the ones of others. Thus, it can be seen that the results of the MWPS2 method are much more stable than the others. So, the MTWPS2 algorithm has the good global search ability for the multi-UAV task assignment problem.

The best objective function curves without the penalty function value in 20 experiments for each method are shown in Fig. 15. Meanwhile, the best result of the MTWPS2 algorithm, whose objective function is 4650.5, is presented in Fig.16.

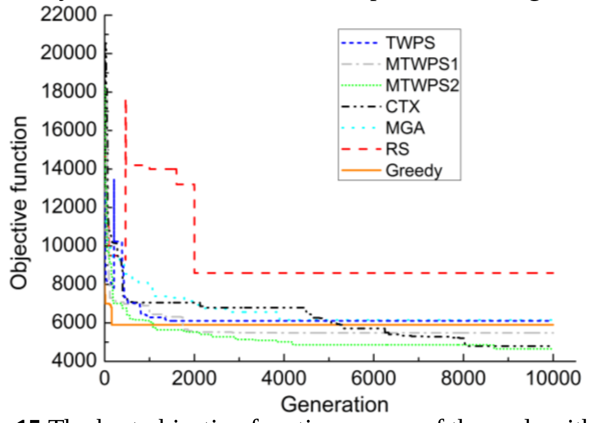


Fig. 15 The best objective function curves of these algorithms

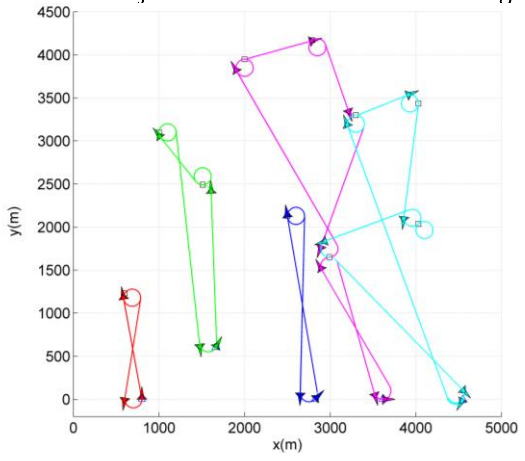


Fig. 16 The planning paths of the best task assignment result

In Fig.15, the convergence rate of the MTWPS2 method is between the TWPS method and the MTWPS1 method. This is due to the MTWPS2 method is the combination of the other two algorithms. From Fig.16, we can see that the best task assignment result is short and reasonable. Because the failure rates of the 5-th UAV are set as 10%, 10%, and 10%, it has the best ability to solve the classification and response tasks, which leads the fact that the 5-th UAV needs to reach its number limitation of the task N^{5}_{Lim}=10. Meanwhile, the other UAVs mainly satisfy the nearby principle. It is really a good solution which may be close to the optimal solution.

All the simulations are coded by MATLAB R2014a in the Windows 7 professional 64 bits system. All experiments are performed on a computer using Intel(R) Core(TM) i7-4790K CPU @ 4.0GHz with 32 GB of DDR3 1600MHz Memory.

Under these conditions, the average simulation time of these algorithms is 3677s (TWPS algorithm), 3418s (MTWPS1 algorithm), 4310s (MTWPS2 algorithm), 2912s (GA with CTX algorithm), 3120s (MGA algorithm), 1949s (RS way), and 2791s (Greedy algorithm). We can find that the MTWPS2 algorithm needs to use the longest time to finish the update of 10000 generations. But if the termination generation of the MTWPS2 algorithm is reduced to 6067, which is similar to reducing the average simulation time to 2615s, the objective function and the convergence rate of the MTWPS2 algorithm are still better than the ones of the other algorithms. In conclusion, the MTWPS2 algorithm is the best algorithm in these methods even in the same limitation of simulation time.

8.1.3 New solution frame

Using the MTWPS2 method, the main purpose of this section is to verify that the new solution frame shown in Fig. 9 can reduce the simulation time and maintain the performances based on two different objective functions. Some optimization problems with different sizes using two different objective functions are solved and shown in Table. 2:

Table. 2 The differences of the simulation time and the objective function

N_v=5	Objective function differences	Reduced simulation time
N_t/N_v=3	2268 (25.5%)	86.88%
N_t/N_v=4	3582 (23.3%)	98.26%

The way to compute the objective function difference is shown as follows: First, the solutions are obtained by two different objective functions using the MTWPS2 method repeatedly. And then, their objective function values are computed uniformly based on the objective function (8)-(9). Finally, the average differences, including the value and the proportion relative to the old frame, are obtained and put forward in Table. 2. We can see that the new frame can reduce the simulation time greatly at a price. Because of the lack of the deadlock-free operations in objective function computing, we can at least save 85% simulation time. The new objective function is not strictly equivalent to the old function, and this framework also exists a performance degradation phenomenon, so we can find a gap (About 20%).

B. The UAV Task Assignment Problem with uncertainty using Robust Optimization Method

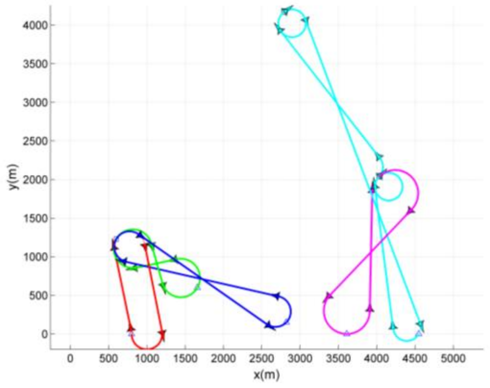
These simulations are to verify the effectiveness of the robust optimization model and the novel combined algorithm presented in Section VII.A for the problem under parameter uncertainty. These two simulations are respectively solved based on the branch and bound method in the Lingo software and the novel combined algorithm.

First of all, let's consider the following scenario: The targets are reduced to the first three targets. The coordinates and the starting angles of the UAVs are the same as the previous section. But the velocities of these UAVs are respectively set as 200m/s, 250m/s, 200m/s, 300 m/s, and 180 m/s, as well as their radii are set as 70m, 80m, 70m, 90m, and 60m.

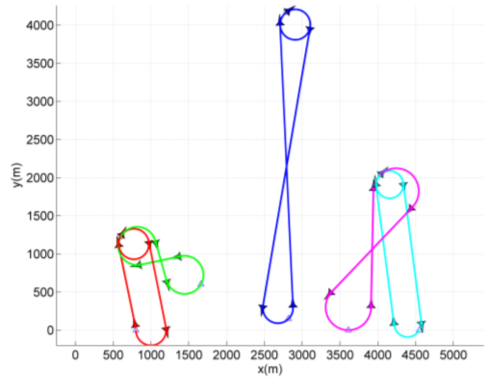
The uncertainties of the parameters are summarized as c_m, in the meantime, the uncertainty of $f^{k}_{i,j}(\{X^{k}_{i,j}\})$ is transferred to the failure rates of UAV $P_{k,l}$. By this way, the simulation can be simplified. For the failure rates, except the 5-th UAV, others for the classification, response and verification tasks are set as constant 30%, 20%, and 10%. For the normal optimization model, the failure rates of 5-th UAV are set as [10%, 10%, 10%]. For the robust optimization model, its failure rates are set between [10%, 10%, 10%] and [30%, 30%, 30%].

The coefficients of the objective function are coded in MATLAB. Secondly, these coefficients are stored in a text file by the form of a column vector. Then, these whole optimization models are solved by the Lingo.11 software. The linearization degree option of the Lingo software is set to be high.

By the above setting, the trajectories of the optimal solutions of the two different models are illustrated in Fig. 17.



(a) The original optimization model



(b) robust optimization model

Fig. 17 The results of the two different optimization models

As shown in Fig. 17a, Target 1 is classified by UAV 1;

Target 1 is attacked by UAV 2; The verification of Target 1 is finished by UAV 3; The verification of Target 3 is completed by UAV 4; Because UAV 5 has higher classification and attack effectiveness than the other UAVs, all the rest tasks are prosecuted by UAV 5. The cost time of the whole mission is 119.6 seconds. Note that the obtained solution has no appending maneuvers. In Fig. 17b, we can know that the tasks of UAV 5 are reduced, because the failure rate of UAV 5 is unstable. The robust optimization model focuses on the worst situation, so UAV 5 has the small classification advantage, the same attack ability but a disadvantage of small verification. With the effect of the flying length, the classification and attack tasks of Target 2 are solved by UAV 3 instead of UAV 5. This result shows that the result is right and reasonable. The time spent on the whole mission is 80.8652 seconds.

The robust optimization theory is to compute a result which always meets the constraints in every situation, and ensures that this result is the optimal solution for the worst case. The decisive factor of the planning results is the worst case. So, we can see that the mission planning results change greatly based on different low bounds instead of the same upper bounds. Those results verify the correctness of the obtained model.

Then, let's consider the following task assignment scenario, and solve it by our novel combined algorithm including the classical interior point method and the MTWPS algorithm: Except that the failure rates of the 5-th UAV are set between [10%, 10%, 10%] and [30%, 30%, 30%], the other settings are the same as the situation of Section 8.1.2.

The whole program is coded by MATLAB. The degraded linear programming model (14) is solved in the computation of the objective function by the library function "linprog". The maximum number of iterations is set as 1000. The simulation result is shown in Fig. 18.

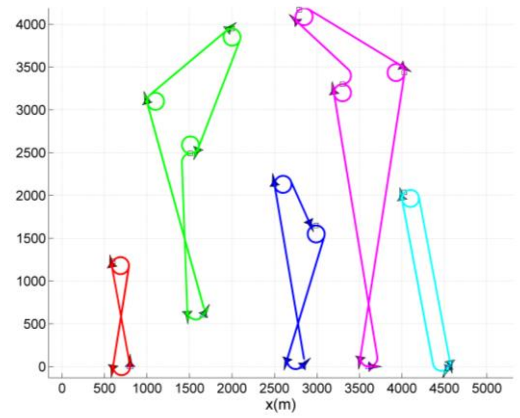


Fig. 18 The first results of the robust optimization model

In Fig.18, compared with Fig. 16, because the failure rates of the 5-th UAV change into an unstable situation, by the help of the robust optimization model, we can see that the tasks of UAV 5 are reduced. The new objective function is 5443.8. Meanwhile, the simulation time is 721s, which is acceptable.

In order to further verify its effectiveness, we try to change the failure rates of all aircraft for the above second task assignment scenario. They are respectively set as: 1-st and 3-rd UAVs (between [30%, 20%, 10%] and [30%, 20%, 15%]), 2-nd UAV (between [30%, 20%, 10%] and [30%, 30%, 30%]), 4-th UAV (between [30%, 20%, 10%] and [30%, 20%, 30%]), 5-th UAV (between [10%, 10%, 10%] and [30%, 30%, 30%]) with unknown probability distributions. The simulation result is shown in Fig.19.

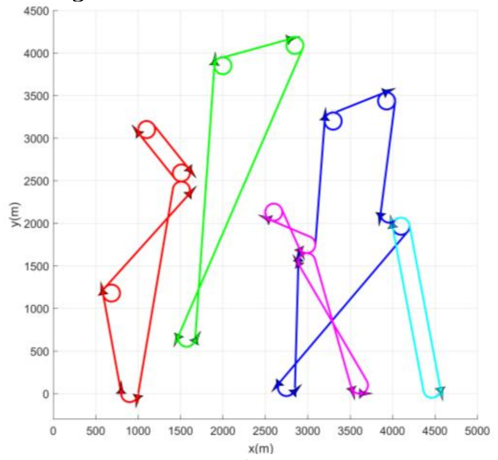


Fig. 19 The second results of the robust optimization model

In Fig.19, compared with Fig. 16, the tasks of all UAVs are changed because of the new unstable failure rates. The 1-st and 3-rd UAVs have the best abilities, so they perform most tasks. The new objective function of this result based on the unchanged failure rates is 7140.9. Meanwhile, the simulation time is 807s, which is also acceptable.

These results show that the novel combined algorithm for the optimization problems is reasonable and effective.

C. *Online Hierarchical Planning Algorithm for the UAV Task Assignment Problem*

The application of the real system is the best option to verify the effectiveness of the online hierarchical planning algorithm. The quad-rotor UAVs system with two UAVs using practical series PID controllers is applied in a small indoor environment based on a vision localization system.

The whole hardware system (Fig. 21) is composed of three main parts, including the vision localization system, the central computer workstation and the UAVs. The vision localization system can obtain the attitude information and the location information of the UAVs based on the markers, the camera and the smart switches. The central computer workstation can deal with the information obtained from the vision localization system, compute the outer-loop position control instruction of the UAV and obtain the expected attitude angle. On the quad-rotor UAVs, our own development flight controller is composed of the gyro, the accelerometer and the magnetometer, and the actuators, including the motors and the electronic speed controller, can finish the control of the inner attitude loop.

The scenario of the online task assignment is set as: there are two UAVs performing nine tasks of three targets in a 2m×2m space. The velocities of two UAVs v_k are both set as 1 m/s. The initial coordinates of the UAVs are [1.8m -1.5m] and [-1.5m 1.5m]. The coordinates of three targets are [1.2m, 1.2m], [1.2m, 1.8m], [-0.8m, 1.7m]. The minimum interval time of two tasks is set as 0.2s and the performing time T_c of the tasks is set as 1s. The initial offline task scheme using the two-part individual representation is [1, 2, 3, 4, 5, 6, 7, 8, 9, 6, 3]. The decision variable T' and the finish time T_a are respectively 5s and 9.9513s. Two different detection time t_1 of the time-sensitive targets which can touch off two different online algorithms are respectively set as 2s and 5s. The maximum flying distances of the two UAVs are respectively 28m and 12m.

Based on these hardware systems and scene-settings, the flying results of two new online schemes are shown as follows (See Fig. 20-22):

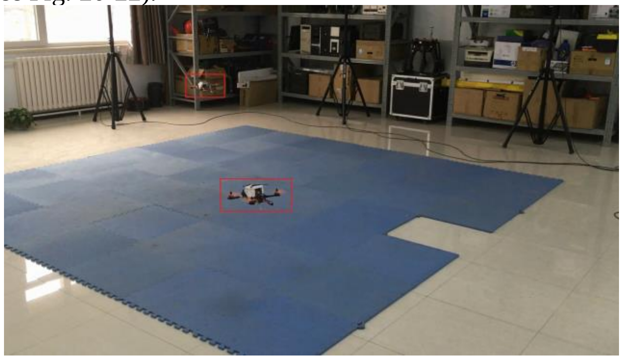
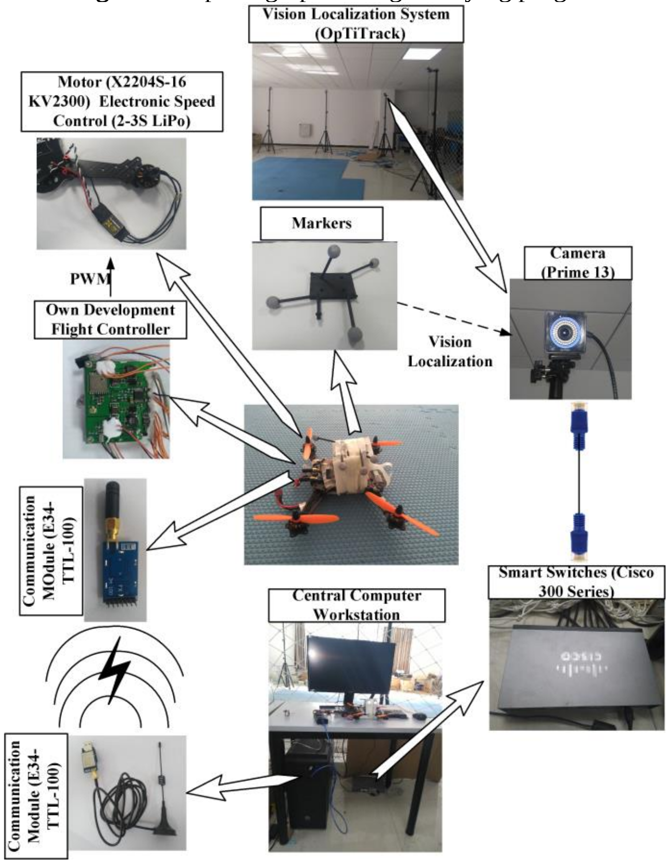
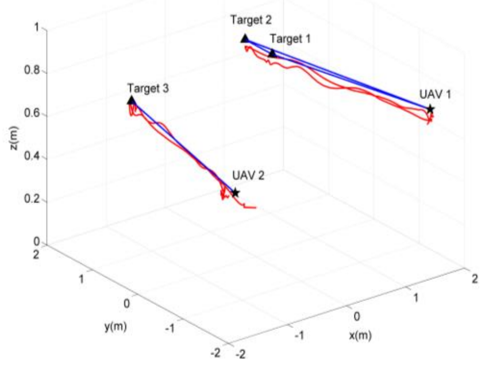


Fig. 20 The photograph during the flying progress

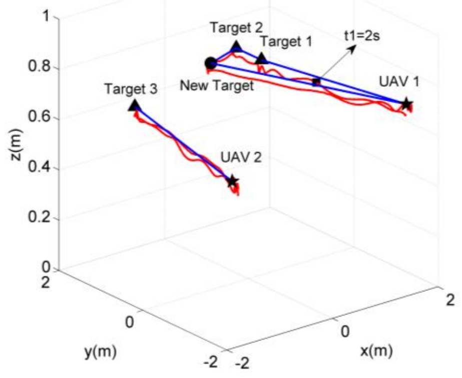


Vision Localization System (OpTiTrack)
Motor (X2204S-16 KV2300) Electronic Speed Control (2-3S LiPo)
Markers
Camera (Prime 13)
PWM
Own Development Flight Controller
Vision Localization
Communication MOdule (E34-TTL-100)
Smart Switches (Cisco 300 Series)
Central Computer Workstation
Communication Module (E34-TTL-100)

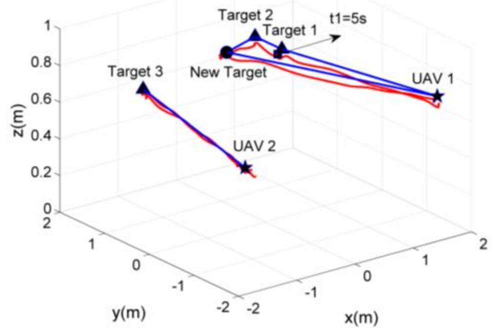
Fig. 21 The whole hardware system



(a) The result of the original offline task scheme



(b) The result of the online task scheme 1



(c) The result of the online task scheme 2

Fig. 22 The flying results of three task schemes

In these results, we can see that the tasks of the time-sensitive targets are finished no matter when t_1=2s or t_1=5s, which can certify that this online algorithm is feasible. When t_1=2s ($t_1 \in [0, 0.3T']$), the small centralized planning problem is solved easily based on MTWPS method with greedy seeding and 5 steps limitation. When t_1=5s, the online exchange-bidding algorithm is applied.

It is noted that the planning results of two different online algorithms are the same. In fact, for the small-sized problem, the exchange-bidding algorithm, which is similar to the greedy algorithm, always can obtain a similar result or the same result as the one of the centralized planning algorithm.

IX. Conclusion

The multi-UAV task assignment problems considering different situations, which include the traditional offline centralized situation, the offline one with parameter uncertainty and the online situation with the time-sensitive uncertainty, are fully modeled and solved. A novel digraph-based deadlock-free algorithm and a novel MTWPS algorithm are presented to solve the deterministic offline problem efficiently. For the problem with parameter uncertainty, the robust optimization method, the duality theory and a novel combined algorithm including the classical interior point method and our MTWPS algorithm are introduced. A practical online hierarchical planning algorithm, which is composed of a small-scale centralized planning algorithm and an exchange-bidding algorithm, is presented to solve the online problem with the time-sensitive uncertainty. Finally, several numerical simulations and physical experiments are finished to verify the effectiveness of the presented algorithms.

This work considers some complicated scenarios with multiple consecutive tasks, heterogeneous vehicles and uncertainty. The solution frameworks are practical, systematic and universal. Not restricted to the UAV application, they can be used in many similar or simpler scenes, which can be formulated as the m-TSP problem with the resource constraints and task sequence restriction. Compared with some existing results, these algorithms have some advantages in the scope of application and efficiency.

Appendix A

The effectiveness of the SECs:

$$u_{i-N_v} - u_{j-N_v} + 3N_t \sum_{k=1}^{N_v} X_{i,j}^k \le 3N_t - 1, \quad 1 \le i - N_v \ne j - N_v \le 3N_t \tag{16}$$

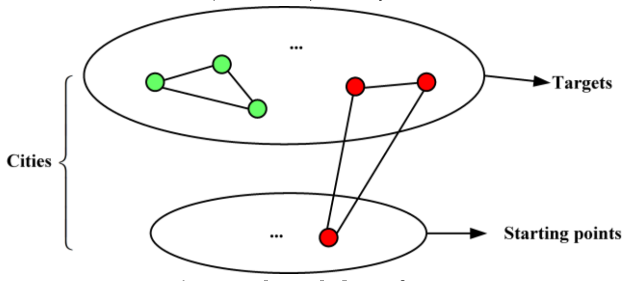


Fig. 23 The sub-loop figure

In order to ensure that (22) can obtain the desired effects, there are two conclusions which need to be proved: a. Any results including the sub-loop will be excluded by (22); b. Any non-closed broken line paths will not be excluded by (22) (See Fig. 23).

Proofs by contradiction:

It is assumed that the sub-loops exist in the targets set. The cities of one sub-loop can be defined as $i_1 \to i_2 \to \cdots \to i_k \to i_1$, so we can get:

$$\begin{aligned} u_{i_1} - u_{i_2} + 3N_t &\le 3N_t - 1, \\ u_{i_2} - u_{i_3} + 3N_t &\le 3N_t - 1, \\ &\cdots \\ u_{i_k} - u_{i_1} + 3N_t &\le 3N_t - 1. \end{aligned} \tag{17}$$

Add up these *k* in equations, we have:

$$3kN_t \le 3kN_t - 3k, \text{contradiction!}. \tag{18}$$

In a word, this assumption is wrong. So the conclusion (a) is proved.

The **proof** of (b) conclusion (Proofs by structured approach):

For arbitrary non-closed broken line path $i_1 \to i_2 \to \cdots \to i_k$, u_i is defined as the sequence number of *i*-th city (Special case: If the non-closed broken line path only has one city, u_i equal to 1). So the value range of u_i is {1, 2, 3,…, 3N_t}.

The arbitrary non-closed broken line path $i_1 \to i_2 \to \cdots \to i_k$ meets:

$$\begin{aligned} u_{i_1} - u_{i_2} + 3N_t &= 3N_t - 1 \le 3N_t - 1, \\ u_{i_2} - u_{i_3} + 3N_t &= 3N_t - 1 \le 3N_t - 1, \\ &\cdots \\ u_{i_{k-1}} - u_{i_k} + 3N_t &= 3N_t - 1 \le 3N_t - 1, \\ u_{i_k} - u_{i_1} &= k - 1 \le 3N_t - 1. \end{aligned} \tag{19}$$

So the conclusion (b) is proved.

Appendix B

Property 2: The mapping from the candidate solutions to the two-part individuals is a one-to-one mapping.

Proof: First of all, we need to prove that this mapping is a surjection. Without loss of generality, one of the two-part individuals with N_t targets and N_v UAVs is written as: <a_1, a_2, a_3, …, a_{3Nt}, b_1, $b_{2,\ldots}$, b_{Nv}>, where b_1, $b_{2,\ldots}$, b_{Nv} meet $b_1+b_{2+\ldots}+b_{Nv}=3N_t$. Its corresponding element in the candidate solutions set will be:

$$\{X_{i,j}^k \mid X_{i,j}^k = \begin{cases} 1 & i = a_{i_1}, j = a_{i_1+1}, k \text{ meets } \sum_{l=0}^{k-1} b_l < i_1 \le \sum_{l=1}^{k} b_l \\ 0 & else \end{cases}, \cdots, \text{ for } i_1 = 1, 2, \ldots, 3N_t - 1\} \tag{20}$$

where b_0=0. So this mapping is a surjection.

Then, we need to certify that this mapping is injective. Without loss of generality, any two different solutions in the candidate solutions set respectively are $\{X_{i,j}^k\}$ and $\{X_{i_1,j_1}^{k_1}\}$, where $\{X_{i,j}^k\} \ne \{X_{i_1,j_1}^{k_1}\}$. Hence, there exist at least two elements in these two sets, meeting $X_{i,j}^k=1$ and $X_{i_1,j_1}^{k_1}=0$, where $k=k_1$, $i=i_1$ and $j=j_1$. This situation will lead that their corresponding elements in the two-part individuals set are different. The first corresponding element include a segment <*i*, *j* > in the position between $\sum_{l=0}^{k-1} b_l$ and $\sum_{l=1}^{k} b_l$. The other one is different. Therefore, this mapping is injective.

In short, if a mapping is both injective and surjective, it will be a one-to-one mapping. **Property 2** is proved.

Appendix C

In this section we show how to derive Problem (13) from Problem (12).

Based on (12), we can get its robust counterpart as follows:

$$\min_{\{\bar{X}_m\}} \sum_{m=1}^{(N_v+3N_t)^2 N_v} c_m \bar{X}_m + \beta(\bar{X}_m) \quad s.t. \begin{cases} g_p(\bar{X}_m) \le 0, \\ h_q(\bar{X}_m) = 0. \\ m \in IJK \end{cases}, \tag{21}$$

where I_0={*m*| dc_m>0}, $\beta(\bar{X}_m, \Gamma_0) = \max_{\{S_0|S_0 \subseteq I_0, |S_0| \le \Gamma_0\}} \left\{ \sum_{m \in S_0} dc_m \bar{X}_m \right\}$ and $\Gamma_0 \in$ [0, |I_0|]. Except for $\beta(\bar{X}_m)$, the coefficients of the other parts of (13) are all constant. So only $\beta(\bar{X}_m)$ needs to be transformed. The conversion process is shown as follows:

$$\begin{aligned}&\beta(\bar{X}_m)\\&=\max\left\{\sum_{m\in S_0} dc_m\bar{X}_m : S_0\subseteq I_0, |S_0|\le\Gamma_0\right\}\\&=\min\left\{\sum_{m\in I_0}\left(-dc_m\bar{X}_m z_{0m}\right): \sum_{m\in I_0} z_{0m}\le\Gamma_0, 0\le z_{0m}\le 1\right\}\end{aligned}. \quad (22)$$

Based on the problem (21) and (22), its Lagrange function:

$$\begin{aligned}&L(\bar{X}_m, z_{0m},\lambda,\mu,\nu)=\sum_{m\in I_0}\left(-dc_m\bar{X}_m z_{0m}\right)+\cdots\\&\lambda\left(\sum_{m\in I_0} z_{0m}-\Gamma_0\right)+\sum_{m=1}^{|I_0|}\mu_m\left(-z_{0m}\right)+\sum_{m=1}^{|I_0|}\nu_m\left(z_{0m}-1\right)\end{aligned}, \quad (23)$$

where λ, μ_m and ν_m are the Lagrange multiplicative derivations, and they meet $\lambda\ge0$, $\mu_m\ge0$ and $\nu_m\ge0$. Its Lagrange dual function is:

$$\begin{aligned}&g(\lambda,\mu,\nu)\\&=\inf_{\{z_{0m}\}} L(\bar{X}_m, z_{0m},\lambda,\mu,\nu)\\&=\inf_{\{z_{0m}\}}\sum_{m\in I_0}\left(-dc_m\bar{X}_m+\lambda-\mu_m+\nu_m\right)z_{0m}-\lambda\Gamma_0-\sum_{m\in I_0}\nu_m\end{aligned}, \quad (24)$$

It can be written as:

$$g(\lambda,\mu,\nu)=\begin{cases}-\lambda\Gamma_0-\sum_{m\in I_0}\nu_m & -dc_m\bar{X}_m+\lambda-\mu_m+\nu_m=0\\-\infty & \text{otherwise}\end{cases}, \quad (25)$$

Then, we can get that the Lagrange dual problem of (22) is:

$$\begin{aligned}&\beta(\bar{X}_m)\\&=\max\left\{-\lambda\Gamma_0-\sum_{m\in I_0}\nu_m : -dc_m\bar{X}_m+\lambda-\mu_m+\nu_m=0,\right.\\&\left.\cdots\lambda\ge0,\mu_m\ge0,\nu_m\ge0\right\}\\&=\min\left\{\sum_{m\in I_0}\nu_m+\lambda\Gamma_0 : \lambda+\nu_m-dc_m\tilde{X}_m\ge0,\lambda\ge0,\nu_m\ge0\right\}\end{aligned}, \quad (26)$$

Finally, the equivalent mixed integer programming formulation is shown as follows:

$$\begin{aligned}&\min_{\{\bar{X}_m,\nu_m,\lambda\}}\sum_{m=1}^{(N_v+3N_t)^2N_v} c_m\bar{X}_m+\sum_{m\in I_0}\nu_m+\lambda\Gamma_0\\&s.t.\begin{cases}g_p(\bar{X}_m)\le0,\\h_q(\bar{X}_m)=0,\\\lambda+\nu_m\ge dc_m\bar{X}_m, & \forall m\in I_0\\\lambda\ge0,\\\nu_m\ge0, & \forall m\in I_0\\m\in IJK.\end{cases}\end{aligned}. \quad (27)$$

Consequently, Problem (13) is equivalent to Problem (12).

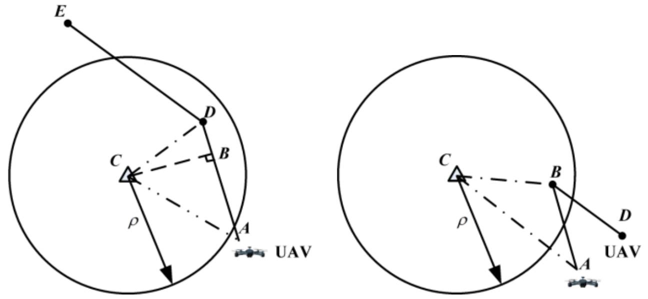
Appendix D

Property 4: If the UAVs meet the above judgment way, there is at least one solution for the new task assignment problem with this new city.

Proof: There are two general situations in this problem. We will prove them in the following parts:

a. As shown in the Fig. 24a, the original cities of the UAV are marked as *E* and *D*. And the city of the time-sensitive target is marked as *C*. By this way, the original off-line path of the UAV is *ADE*. *CB* is the auxiliary line perpendicular to *AD*.

If one UAV meets the above judgment way, we can get that *BC*<ρ and *AD*>0, where ρ is the remaining working radiuses. We can see that there at least exists one new path *ACDE* for the new problem with a new city. In order to ensure the reasonability of the new path, we need to ensure that the sum of the length of *ACDE* and the task performing cost v_kT_c are shorter than the source of the UAV L^k_{Lim}. Based on the triangle inequality, we can get *AC*+*CD*<*AD*+2*BC*. Further, because ρ=0.5L^k_{Lim} -0.5*AD*-0.5*DE*-0.5v_kT_c, we have *AC*+*CD*+*DE*< *AD*+2*BC*+*DE*< *AD*+*DE*+2ρ= *AD*+*DE*+L^k_{Lim}-*AD*-*DE*- v_kT_c=L^k_{Lim} - v_kT_c. At last, we can get that *AC*+*CD*+*DE*+v_kT_c< L^k_{Lim}.



(a) Situation 1 (b) Situation 2

Fig. 24 The general situation

b. In the Fig. 24b, the original cities of the UAV are marked as *B* and *D*. And the city of the time-sensitive target is marked as *C*. By this way, the original off-line path of the UAV is *ABD*.

If one UAV meets the above judgment way, we can get that *BC*<ρ, where ρ is the remaining working radiuses. We can see that there at least exists one new path *ACDE* for the new problem with a new city. In order to ensure the reasonability of the new path, we need to ensure that the sum of the length of *ACDE* and the task performing cost v_kT_c are shorter than the source of the UAV L^k_{Lim}. Based on the triangle inequality, we can get *AC*< *AB*+ *BC*. Further, because ρ=0.5L^k_{Lim}- 0.5*AB*- 0.5*BD* - 0.5v_kT_c, we have *AC*+ *CB*+ *BD*< *AB*+ 2*BC*+ *BD*< *AB*+ *BD*+ 2ρ= *AB*+ *BD*+ L^k_{Lim}- *AB*- *BD*- v_kT_c=L^k_{Lim}- v_kT_c. At last, we can get that *AC*+ *CB*+ *BD*+ v_kT_c< L^k_{Lim}.

In short, **Property 4** is proved.

References

[1] W. Xing, Y. Zhao and H. R. Karimi, "Convergence analysis on multi-AUV systems with leader-follower architecture," *IEEE Access*, vol. 5, no. 1, pp. 853-868, Jan. 2017.

[2] T. Shima, S. J. Rasmussen, A. G. Sparks and M. P. Kevin, "Multiple task assignments for cooperating uninhabited aerial vehicles using genetic algorithms," *Computers & Operations Research*, vol. 33, no. 11, pp. 3252-3269, Nov. 2006.

[3] E. J. Forsmo, E. I. Grøtli, T. I. Fossen and T. A. Johansen, Optimal Search Mission with Unmanned Aerial Vehicles using Mixed Integer Linear Programming, in *International Conference on Unmanned Aircraft Systems*, Grand Hyatt Atlanta, Atlanta, GA, 2013, pp. 253-259.

[4] E. N. Kendall, R. C. Phillip and P. Meir, Dynamic network flow optimization model for air vehicle resource allocation, *Proceedings of the American Control Conference*, Arlington, Massachusetts, USA, 2001, pp. 1853-1858.

[5] J. Peng, M. F. Wen, G Q. Xie, X. Y. Zhang and K. C. Lin, "Coordinated dynamic mission planning scheme for intelligent multi-agent systems," Journal of Central South University, vol. 19, no. 11, pp 3170-3179, Nov. 2012.

[6] M. Alighanbari and J. P. How, Cooperative task assignment of unmanned aerial vehicles in adversarial environments, in *American Control Conference*, Portland, OR, USA, 2005, pp. 4661-4666.
[7] S. Karaman, T. Shima and E. Frazzoli, “A process algebra genetic algorithm,” *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 4, pp. 489-503, Oct. 2011.
[8] S. Karaman and E. Frazzoli, “Linear temporal logic vehicle routing with applications to multi-UAV mission planning,” *International Journal of Robust & Nonlinear Control*, vol. 21, no. 12, pp. 1372-1395, May 2011.
[9] T. Shima, J. R. Steven and G. S. Andrew, UAV cooperative multiple task assignments using genetic algorithms, in *American Control Conference*, Portland, OR, USA, 2005, pp. 2989-2994.
[10] Q. B. Deng, J. Q. Yu and N. F. Wang, “Cooperative task assignment of multiple heterogeneous unmanned aerial vehicles using a modified genetic algorithm with multi-type genes,” *Chinese Journal of Aeronautics*, vol. 26, no. 5, pp. 1238–1250, Oct. 2013.
[11] M. Alighanbari, Y. Kuwata and J. P. How, Coordination and control of multiple UAVs with timing constraints and loitering, *Proceedings of the American Control Conference*, Denver, Colorado, USA, 2003, pp. 5311-5316.
[12] Z. X. Zhu, B. W. Tang and J. P. 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, May 2017.
[13] H. P. Wang, C. G. Liu and W. J. Li, “On optimizing UAV (unmanned aerial vehicle) mission planning with ANT algorithm,” *Journal of Northwestern Polytechnical University*, vol. 23, no. 1, pp. 98-101, Apr. 2005.
[14] S. J. Rasmussen, T. Shima, J. W. Mitchell, A. G. Sparks and P. Chandler, State-space search for improved autonomous UAVs assignment algorithm, in *43rd IEEE Conference on Decision and Control December*, Atlantis, Paradise Island, Bahamas, 2004, pp. 2911-2916.
[15] W. Q. Zhao, Q. G. Meng and P.W. H. Chung, “A heuristic distributed task allocation method for multivehicle multitask problems and its application to search and rescue scenario,” *IEEE Transactions on Cybernetics*, vol. 46, no. 4, pp. 902-915, Apr. 2016.
[16] T. Long, “Research on Distributed task allocation and coordination for multiple UCAVs cooperative mission control,” Ph.D. dissertation, Graduate School, National University of Defense Technology, Changsha, Hunan, China, 2006.
[17] Q. B. Deng, J. Q. Yu and Y. S. Mei, “Deadlock-free consecutive task assignment of multiple heterogeneous unmanned aerial vehicles,” *Journal of Aircraft*, vol. 51, no. 2, pp. 596-605, Feb. 2014.
[18] E. G. Coffman, M. Elphick and A. Shoshani, “System deadlocks,” *ACM Computing Surveys*, vol. 3, no. 2, 1971, pp. 67-78. Jun. 1971.
[19] T. Lemaire, R. Alami and S. Lacroix, A distributed tasks allocation scheme in multi-UAV context, in *Proceedings of the International Conference on Robotics and Automation*, 2004, pp. 3622-3627.
[20] P. M. Stephen, “Static detection of deadlocks in polynomial time,” Ph.D. dissertation, Rutgers University, New Brunswick, NJ, USA, Rutgers Univ, 1993.
[21] Z. Lu, “Task assignment under uncertainty: stochastic programming and robust optimization approaches,” *International Journal of Production Research*, vol. 53, no. 5, pp. 1487-1502, Jun. 2015.
[22] E. Lanah, D. Twan, I. B. Ana and M. Herman, “Robust UAV mission planning,” *Annals of Operations Research*, vol. 222, no. 1, pp. 293-315, Nov. 2014.
[23] L. F. Bertuccelli, M. Alighanbari and J. P. How, Robust planning for coupled cooperative UAV missions, in *43rd IEEE Conference on Decision and Control*, Atlantis, Paradise Island, Bahamas, 2004. pp. 2917-2922.
[24] A. Ben-Tal and A. Nemirovski, “Robust solutions of linear programming problems contaminated with uncertain data,” *Math. Program*ming, vol. 88, no. 3, pp. 411-424, Sep. 2000.
[25] A. L. Soyster, “Convex programming with set-inclusive constraints and applications to inexact linear programming,” *Operations Research*, vol. 21, no. 5, pp. 1154-1157, Oct. 1973.
[26] D. Bertsimas and M. Sim, “Robust discrete optimization and network flows,” *Mathematical Programming*, vol. 98, no. 1-3, pp. 49–71, Sep. 2003.
[27] L. Evers, A. I. Barros, H. Monsuur and A. Wagelmans, “Online stochastic UAV mission planning with time windows and time-sensitive targets,” *European Journal of Operational Research*, vol. 238, no. 1, pp. 348-362, Oct. 2014.
[28] Y. Tang, X. Xing, H. R. Karimi, L. Kocarev and J. Kurths. “Tracking control of networked multi-agent systems under new characterizations of impulses and its applications in robotic systems,” *IEEE Transactions on Industrial Electronics*, vol. 63, no. 2, pp. 1299-1307, Feb. 2016.
[29] N. Nigam, Dynamic replanning for multi-UAV persistent surveillance, in *AIAA Guidance, Navigation, and Control (GNC) Conference*, 2013, pp. 2013-4887.
[30] G. Oh, Y. Kim and J. Ahn, Market-based decentralized task assignment for cooperative UAV mission including rendezvous, in *AIAA Guidance, Navigation, and Control (GNC) Conference*, Boston, MA, USA, 2013, pp. 1-18.
[31] P. B. Sujit and R. Beard, Distributed sequential auctions for multiple UAV task allocation, in *Proceedings of the American Control Conference*, New York, USA, 2007, pp. 2525-2530.
[32] C. E. Miller, A. W. Tucker and R. A. Zemlin, “Integer programming formulation of traveling salesman problems,” *Journal of Association for Computing Machinery*, vol. 7, no. 4, pp. 326-329, Oct. 1960.
[33] C.G. Yang, X.Y. Tu, J. Chen, Algorithm of marriage in honey bees optimization based on the wolf pack search, In: *International Conference on Intelligent Pervasive Computing*, Washington, DC, USA, 2007, pp. 462-467.
[34] A. E. Carter and C.T. Ragsdale, “A new approach to solving the multiple traveling salesperson problem using genetic algorithms,” *European Joumal of Operational Research*, vol. 175, no. 1, pp. 246-257, Nov. 2006.
[35] S. Yuan, S. Bradley, S. D. Huang, D. K. Liu, “A new crossover approach for solving the multiple travelling salesmen problem using genetic algorithms,” *European Journal of Operational Research*, vol. 228, no. 1, pp. 72-82, Jul. 2013.
[36] Y. B. Chen, Z. Y. Jia, X. L. Ai, D. Yang and J. Q. Yu. “A modified two-part wolf pack search algorithm for the multiple traveling salesmen problem,” *Applied Soft Computing*, vol. 61, no. 1, pp. 714-725, Dec. 2017.

Yongbo Chen received his B.S. degree in Beijing Institute of Technology in 2012. He is currently working toward a dual doctoral degree at Beijing Institute of Technology, Beijing, China and University of Technology Sydney, Sydney, Australia. His research interest is active SLAM, UAV path planning and UAV task assignment.

Di Yang received the B.S degree in mechanical design from Shenyang Aerospace University, Shenyang, China, in 2014, and the M.S. degree in aerospace engineering from Beijing Institute of Technology, Beijing, China, in 2017, respectively. She is currently a Ph.D. candidate with School of Aerospace Engineering, Beijing Institute of Technology. Her current research interests include UAV path planning, formation control and cooperative control of multi-agent.

Jianqiao Yu received BS, MS and PhD degrees from Beijing Institute of Technology in 1994, 1997 and 2007, respectively, and now is a professor there. His main research interests include flight dynamics and control, cooperative control, flight vehicle system design and robust control.