Swarm and Evolutionary Computation 69 (2022) 101005

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo

Coverage path planning of heterogeneous unmanned aerial vehicles based on ant colony system

Jinchao Chen [a,*], Fuyuan Ling [a], Ying Zhang [a], Tao You [a], Yifan Liu [a], Xiaoyan Du [a]

School of Computer Science, Northwestern Polytechnical University, Xi'an 710072, China

ARTICLE INFO

Keywords:
Ant colony system
Coverage path planning
Unmanned aerial vehicle
Multiple regions
Heterogeneous UAVs

ABSTRACT

Unmanned aerial vehicle (UAV) has been extensively studied and widely adopted in practical systems owing to its effectiveness and flexibility. Although heterogeneous UAVs have an enormous advantage in improving performance and conserving energy with respect to homogeneous ones, they give rise to a complex path planning problem. Especially in large-scale cooperative search systems with multiple separated regions, coverage path planning which seeks optimal paths for UAVs to completely visit and search all of regions of interest, has a NP-hard computation complexity and is difficult to settle. In this work, we focus on the coverage path planning problem of heterogeneous UAVs, and present an ant colony system (ACS)-based algorithm to obtain good enough paths for UAVs and fully cover all regions efficiently. First, models of UAVs and regions are built, and a linear programming-based formulation is presented to exactly provide the best point-to-point flight path for each UAV. Then, inspired by the foraging behaviour of ants that they can obtain the shortest path between their nest and food, an ACS-based heuristic is presented to seek approximately optimal solutions and minimize the time consumption of tasks in the cooperative search system. Experiments on randomly generated regions have been organized to evaluate the performance of the new heuristic in terms of execution time, task completion time and deviation ratio.

1. Introduction

With the rapid development of automatic control and artificial intelligence, unmanned aerial vehicle (UAV) has been extensively used in large-scale practical applications, such as traffic monitoring [1], disease surveillance [2] and target tracking [3]. In contrast with manned aerial vehicles, UAV frees human from dangerous, boring and burdensome work [4], and provides a flexible and efficient way to military and civilian activities. Due to the limited computing, communication and storage capacities [5], a single UAV has great difficulty in carrying out complicated missions. To cope with the increasing scale and complexity of real-world applications, multi-UAV systems have attracted tremendous attention and become one of the most active research hotspots for their significant benefit of adaptability and flexibility [6].

Heterogeneous UAVs can be operating as a group to improve system performance and conserve energy consumption, by sharing information and cooperating with others [7]. In large-scale search or surveillance systems, heterogeneous UAVs with different flight, scan and load-carrying capabilities are widely applied to make systems more cost-effective and fault tolerant. For example, in the military field, heterogeneous UAVs carrying different devices are ordered to perform the searching task and track dangerous targets. In the agriculture field, heterogeneous UAVs would be adopted in terraces separated by mountains to collect the growth information of vegetation and plants.

Although heterogeneous UAVs can effectively reduce costs and enhance flexibility in several application domains, they make the systems harder to perform tasks of path planning [8], collaborative control [9], intelligent decision-making [10] and performance optimization [11]. Especially in systems where concerns involve all the information of regions [12], coverage path planning of heterogeneous UAVs, which seeks optimal paths between start and goal points to scan all regions while satisfying a large number of various constraints [13], is very challenging. Not only the multi-variable optimization property gives it a NP-Hard computational complexity [14], but also the heterogeneity compounds difficulty in establishing the timing and security constraints.

Faced with the increasing numbers of heterogeneous UAVs and separated regions, system developers begin to adopt decision-support tools to achieve a reasonable point-to-point path for each UAV, such that missions would be carried out efficiently and highly reconfigurable systems become more intelligent. Many mature approaches, especially on the basic of bio-inspired optimization algorithms [15], such as particle swarm optimization (PSO) and ant colony optimization (ACO), have

* Corresponding author.
E-mail addresses: cjc@nwpu.edu.cn (J. Chen), ling_fy@mail.nwpu.edu.cn (F. Ling), ying_zhang@nwpu.edu.cn (Y. Zhang), youtao@nwpu.edu.cn (T. You), 2018100732@mail.nwpu.edu.cn (Y. Liu), 18729865562@mail.nwpu.edu.cn (X. Du).

https://doi.org/10.1016/j.swevo.2021.101005
Received 12 December 2020; Received in revised form 17 September 2021; Accepted 15 October 2021
Available online 22 October 2021

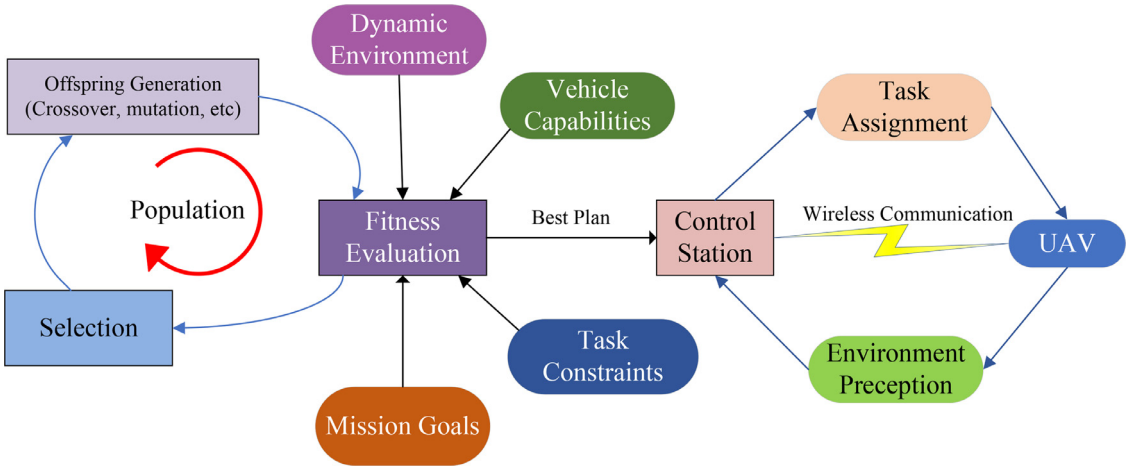

Fig. 1. Evolutionary process of bio-inspired optimization algorithms adopted in path planning problems.

been proposed for various applications with different requirements. The evolutionary process of bio-inspired optimization algorithms is one of the most important factors determining the efficiency of path planning solutions and has three major phases: fitness evaluation phase, selection phase, and offspring generation phase [16]. As shown in Fig. 1, fitness evaluation phase evaluates the fitness of generated paths with objective functions considering the constraints of dynamic environment, vehicle capabilities, task constraints and mission goals. Selection phase chooses and holds paths with the best fitness, and the next generation of paths is produced with the selection paths thought various crossover and mutation mechanisms in the offspring generation phase. Although these algorithms provide reasonable flight paths for UAVs effectively, most of them address the path planning problem from a geometrical point of view, without considering the collaboration capabilities of UAV swarms.

In this work, we study the coverage path planning problem of heterogeneous UAVs with different flight and scan capabilities, and try to find an optimal flight path for each UAV from the start region to the end region such that all regions of interest would be sequentially covered in the shortest time. We mainly focus on the region assignment and path optimization of UAVs, without involving the collision, obstacle avoidance, equipment failure and other problems. The main contributions are as follows:

1. Constraints of the path planning problem are described, and a linear programming-based formulation is proposed to exactly settle this problem. The proposed formulation not only obtains the best flight paths for UAVs, but also helps guide the design of multiple region coverage systems.

2. Inspired from the foraging behaviours of ants, an ant colony system (ACS)-based algorithm is designed to allocate regions to proper UAVs and to adjust the visiting orders of regions so as to reduce the completion time of the coverage task as much as possible.

Since ant colony system can provide excellent guidance in searching of optimal solutions by using the shared pheromone and heuristic information, it is one of the most frequently used heuristic algorithms in solving the multi-objective optimization problems. Using an ACS-based optimization strategy, the proposed approach hasn't to completely explore the solution space and would provide good enough paths for UAVs in short time. Although this ACS-based method is not optimal or exact, it takes into consideration of the influence of distances and scan areas of regions, and has better global search ability in dealing with the coverage path planning problem.

The remainder of this work is organized as follows. Related work is introduced in Sect. 2. Sect. 3 models and formulates the coverage path planning problem. Sect. 4 presents the ACS-based approach and obtain reasonable paths for UAVs. In Sect. 5, the performance of the proposed approach is tested from several aspects and compared with five existing methods. Finally, Sect. 6 summarizes this work and introduces the further work.

2. Related work

As one of the most important aspects of mission planning in search and exploration systems, coverage path planning has played a vital role in enhancing the survivability and mission ability [12] by seeking valid paths to visit all of regions. It has been widely studied by researchers, and takes into account the influence of several factors, such as the region shape, formidable obstacle, vehicle performance, and energy supply. By addressing the relevance between flying speeds and power consumption of UAVs, Franco et al. [17] built an energy model, and gave an energy-aware algorithm to generate a flight path for a single UAV. In [18], Zhao et al. formally formulated the coverage path planning as a travelling salesman problem, and adopted an improved genetic algorithm to provide shortest flight paths in irregular regions. By analysing the constraints of UAV energy and maneuverability, Gramajo et al. [19] proposed an efficient path planning formulation to meet the demands of high degree autonomy and strong maneuver capability. However, only one UAV is used to perform the searching or driving tasks in all of the above researches, and the proposed approaches cannot directly accommodate to large-scale applications where multiple heterogeneous UAVs are adopted.

At present, most of studies pay attention to the path optimization of multiple autonomous UAVs on a single region. In general, there are three major steps to settle the coverage path planning of UAVs: UAV performance evaluation, region division, and path generation. In [20], Maza and Ollero divided an entire region into several convex polygons according to the flight and energy capabilities of UAVs, and adopted a back-and-forth method to generate flight paths such that all subregions would be covered. In [21], Vincent and Rubin proposed a cooperative search mechanism to plan the paths and motions of UAVs, and enhanced the sensing capability to avoid being affected by the unexpectedly failure of UAVs. Although the mentioned researches can solve the coverage path planning problem efficiently, they worked in applications where the number of regions of interest is only one, and had a limited range of applications. Their results cannot be directly used in multiple region coverage systems where a number of regions are required to visit by heterogeneous UAVs.

Compared with the case when only one region is covered, the coverage path planning of UAVs on multiple regions is less studied. Generally, this problem can be abstracted as combinatorial optimization models with different constrains and settled by optimization approaches [22]. Ye et al. [23] proposed an integer programming model to distribute coverage tasks to multiple cooperative UAVs, by analysing the timing and

security constraints. Batsoyol et al. [24] built ad-hoc networks to connect all regions of interest in outside environment, and adopted the Dijkstra's algorithm to provide collision-free paths for multiple autonomous UAVs. Ding et al. [25] abstracted autonomous UAVs as the Dubins vehicles that can fly at a certain high altitude, and used motion primitives to produce optimal paths and offered convoy protection to vehicles moving on ground. Coverage path planning on multiple separated regions has a large computation complexity, and should deal with lots of optimization objectives and logical constraints. So as to obtain a feasible solution for each UAV, traditional optimization approaches have to consider all of possible flight paths of UAVs and spend quite significant time in fully searching the solution space. Particularly in practical applications where strong real-time requirements are required [26], it is extremely time-consuming and inefficient to tackle the coverage path planning problem by adopting traditional optimization-based approaches.

In Chen et al. [27], the authors also dealt with the path planning problem of UAVs with different abilities in multi-region systems. Inspired from density-based clustering approaches, they first classified regions into clusters on the basis of their spatial-temporal densities computing via relative distances, and then provided approximately optimal paths for UAVs to effectively visit all of regions. Although the approach proposed in Chen et al. [27] was efficient and helped designers in planning reasonable point to point paths for UAVs, it did not optimize the visiting orders of regions that were assigned to the same UAV, leading to unsatisfactory results in some situations.

In this research, we analyze the constraints and objective of the path planning problem of UAVs, and propose an efficient ACS-based approach to achieve an approximately optimal path for each UAV such that the time consumption of tasks in the cooperative search system would be reduced. Compare with the method presented in [27], we allocate regions to UAVs according to their *effective time ratios* instead of the densities, and adopt an optimization strategy to adjust the visiting orders of regions so as to minimize the task completion times. The approach proposed in this work has a wider application range and can obtain relatively better solutions for the coverage path planning problem in cooperative search systems.

3. System model and exact formulation

3.1. System models

In this work, we use a set of UAVs $U = \{U_1, U_2, ., U_n\}$ to perform a mission of fully searching m regions $R = \{R_1, R_2, \dots, R_m\}$. UAVs are heterogeneous and have various types of modules, such as calculation modules, communication modules, control modules, on-broad devices, etc., and each module type is designed specifically to perform a special functionality efficiently. In this paper, we have only focused on the flight and scan performance, ignoring other capabilities and attributes of UAVs.

Each UAV is characterized as $U_i = \langle V_i^{\max}, H_i, W_i \rangle$ where $V_i^{\max}$ and H_i respectively represent the maximum flight velocity and the maximum flight altitude, and W_i denotes the scan width of cameras installed in the UAV. We assume that all UAVs take off from the same training base, and their battery capacities, payload weights, fuselage sharps, system architecture and on-board communications are neglected.

An m-row m-column matrix $D = \{D_{j,k}\}$ is used to denote the distances between each two regions. Each element $D_{j,k}$ represents the flight distance from region R_j and region R_k. If R_j and R_k are the same, their distance is zero, i.e., $D_{j,j} = 0$; otherwise, the spatial distance of R_j and R_k should be computed through via their coordinates.

An $n \times m$ matrix $V = \{V_{i,j}\}$ is adopted to represent the maximum permitted flight velocity of UAVs when they are used to scan the regions. If U_i cannot perform the coverage task in R_j, we assume the flight velocity of U_i when it scans R_j is zero, i.e., $V_{i,j} = 0$. We adopt $TS_{i,j}$ to denote the time cost of U_i when it is adopted to scan the region R_j, and $TS_{i,j}$ could be calculated via,

$$TS_{i,j} = \begin{cases} +\infty, & \text{if } V_{i,j} = 0 \\ \frac{A_j}{V_{i,j} W_i}, & \text{otherwise} \end{cases} \tag{1}$$

where A_j represents the scan area of region R_j.

Meanwhile, $TF_{i,j,k}$ is adopted to denote the time cost of UAV U_i in flying from one region R_j to another one R_k, therefore,

$$TF_{i,j,k} = \frac{D_{j,k}}{V_i^{max}} \tag{2}$$

Example 1 shows the system models used in a three-region coverage task.

Example 1. Consider a scenario where a UAV U_1 is adopted to completely search three regions R_1, R_2 and R_3 for detecting the growth of vegetation. If the maximum flight velocity, flight altitude and scan width of U_1 are 5 m/s, 30 m and 2 m, this UAV can be denoted by $U_1 = \langle 5, 30, 2 \rangle$. We assume that the scan area of region R_1 is 20 m^2 and the maximum permitted flight velocity of U_1 on R_1 is 1 m/s, then $A_1 = 20$, $TS_{1,1} = 20/(1 \times 2) = 10$. If the center coordinates of the three regions are (0,0), (0,50) and (50,50), the distances from R_1 to R_2 and R_3 are 50 m are 70.7 m, respectively. Therefore $D_{1,2} = 50$, $D_{1,3} = 70.7$, $TF_{1,1} = 50/5 = 10$ and $TF_{1,3} = 70.7/5 = 14.14$.

3.2. Exact formulation

Coverage path planning, which seeks good enough paths to scan all of regions, is widely adopted in practical applications such as earthquake rescue, emergency search and fire monitoring. It can be formulated as minimizing an objective function (e.g., task completion time of UAVs) subject to a set of linear constraints (e.g., energy consumption constraints of UAVs). When regions are fully covered, flight paths of UAVs should satisfy the following constraints:

(C1): Once a UAV is selected to perform the coverage task, it cannot return back and land at the training base until all of regions have been covered.

(C2): The number of UAVs adopted to perform the coverage task is less than or equal to the total number of UAVs.

(C3): Each UAV should take off from the training base to perform the coverage task, or stay at the base.

(C4): Every region should be scanned by UAVs once and only once, in order to ensure that no region is scanned repeatedly.

An array $X = \{x_{i,j,k} \mid 1 \le i \le n, 0 \le j \le m, 1 \le k \le m\}$ is adopted to denote the planned paths of UAVs, in which $x_{i,j,k}$ is a boolean variable, representing whether U_i has a flight from R_j to R_k. If U_i goes direct from region R_j to region R_k, $x_{i,j,k} = 1$; otherwise, $x_{i,j,k} = 0$.

Constraint (C1) implies that all of UAVs could take off and land at most once. Since $x_{i,0,j}$ and $x_{i,j,0}$ denote whether U_i takes off from the initial location to the region R_j and whether U_i flies back from R_j, respectively, Constraint (C1) could be represented as:

$$\forall i \in [1, n], \sum_{j=1}^{m} x_{i,0,j} \le 1, \sum_{j=0}^{m} x_{i,j,0} \le 1 \tag{3}$$

Constraint (C2) implies the number of UAVs performing the task and leaving the base is equal to or less than the total UAV number, i.e.,

$$\sum_{i=1}^{n} \sum_{j=1}^{m} x_{i,0,j} \le n \tag{4}$$

When a given U_i is selected to perform the coverage task, it must take off from and land to the base, i.e., $\sum_{j=0}^{m} x_{i,0,j} = 1, \sum_{k=0}^{m} x_{i,k,0} = 1$. Meanwhile, if U_i keeps staying at the training base during the coverage task execution, it should not fly from one region to another, in other words, $\forall k, l \in [0, m], x_{i,k,l} = 0$. Hence, Constraint (C3) could be written

as:

$$\begin{cases} \sum_{k=0}^{m} x_{i,k,0} = 1, & \text{if } \sum_{j=0}^{m} x_{i,0,j} = 1 \\ \forall k, l \in [0, m], x_{i,k,l} = 0, & \text{otherwise} \end{cases} \tag{5}$$

Constraint (C4) shows that each region is covered once and only once, which implies that one and only one UAV can fly to and leave from a given region. This constraint can be written as

$$\forall j \in [1, m], \sum_{i=1}^{n} \sum_{k=1}^{m} x_{i,j,k} = 1, \sum_{i=1}^{n} \sum_{k=0}^{m} x_{i,k,j} = 1 \tag{6}$$

Since the goal of the problem is to seek a best point to point path for each UAV so as to carry out the search tasks efficiently, the objective function of this problem can be formulated as reducing the task completion time of UAVs. We adopt $F(U_i)$ to denote the total time cost of UAV U_i in taking off from the training base and scanning all of regions allocated to it, then $F(U_i)$ could be computed via,

$$F(U_i) = \sum_{j=0}^{m} \sum_{k=1}^{m} x_{i,j,k}(TS_{i,j} + TF_{i,j,k}) \tag{7}$$

Note that the task completion time considered in this paper neglects the time spent by UAVs in flying back to the base, which is different from that optimized in [27]. Coverage path planning of UAVs concerned in our work is formulated as finding an optimal array X to minimize task completion times of UAVs. Given that the unknown variables in this programming are a mixture of integer (e.g., elements of X) and real (e.g., the maximum time cost of UAVs) variables and all constraints are linear, this formulation is an MILP one and stated as:

$$\begin{aligned} &min \quad \max_{1 \le i \le n} F(U_i) \\ &s.t. \quad (1), (2), (3), (4) \end{aligned} \tag{8}$$

computation times of UAVs. It will completely explore the solution space and achieve best paths for UAVs if no time limit is set. Unfortunately, the solution space is exploding along with the scale and complexity of systems, making the application of this formulation inefficient because of the required time consumption. Especially in self-organizing systems where path planning should be real-time achieved, the efficiency of the linear programming-based formulation is unacceptable. Inspired by the foraging behaviour of ants, a more efficient approach is presented in the following sections.

4. ACS-Based path planning algorithm

In this section, an ant colony system (ACS)-based path planning algorithm (APPA) is presented to deal with the coverage problem of UAVs on multiple regions. ACS is one of nature-inspired algorithms and was first proposed in Dorigo and Gambardella [28] to settle a travelling salesman problem. Inspired by the foraging behaviour of ants that they can obtain the shortest path between their nest and food with help of shared information passed through pheromone [29], ACS adopts pheromone and heuristic information to record the accumulated historical experience and guides the searching process. Experiments on Amazon EC2 cloud platform [30] show that ACS has better performance than not only the multi-objective optimization approaches but also the constrained optimization ones, providing excellent guidance in searching of optimal solutions by using the shared pheromone and heuristic information.

The proposed algorithm first divides regions to be covered into several groups and generates an initial sequence of regions to visit, then optimizes the visiting orders of regions in the sequence to minimize the time cost. Therefore two main phases are contained in our algorithm: a *region allocation phase* for allocating regions to proper UAVs with an objective that regions with large areas or near to free UAVs are covered first, and an *order optimization phase* for adopting an ACS-based optimization strategy to adjust the visiting order of regions and reduce the time cost of UAVs. Pseudo-codes of the region allocation phase and the order optimization phase are shown in Algorithm 1 and Algorithm 2 , respectively.

Algorithm 1: The pseudo-code of the region allocation phase.

Input: region set R and UAV set U
Output: regions assigned to each UAV

$LeftRegion \leftarrow R$;
$\forall i \in [1, n]$, $Region[i] \leftarrow \emptyset$, $FinishTime[i] \leftarrow 0$;
repeat
  $i \leftarrow$ index of the UAV which would complete the coverage task at the earliest time;
  $j \leftarrow$ index of the last region that would be visited by U_i;
  Calculate effective time ratios of regions in $LeftRegion$;
  $k \leftarrow$ index of the region that has the largest effective time ratio;
  Remove R_k from $LeftRegion$;
  Add R_k into $Region[i]$;
  $FinishTime[i] += TS_{i,k} + TF_{i,j,k}$;
until $LeftRegion \neq \emptyset$;

Algorithm 2: Pseudo-code of the order optimization phase.

Input: number of ants N, evolution algebra of populations G
Output: optimized coverage sequence for each UAV

for $r = 1$ *to* n **do**
  $Region[r] \leftarrow$ the set of regions assigned to U_r;
  Place M ants in the regions belonging to $Region[r]$ randomly;
  $L \leftarrow$ tour length obtained from the nearest neighbour approach;
  $\forall i, j \in [0, m]$, $\tau(i, j) \leftarrow \frac{1}{m \times L}$, $\eta(i, j) \leftarrow \frac{1}{D_{i,j}}$;
  for $t = 1$ *to* G **do**
    for $a = 1$ *to* N **do**
      $R_i \leftarrow$ the current region in which the ant a is placed;
      $\Pi_a \leftarrow Region[r] \setminus \{R_i\}$, $Order[a] \leftarrow \{R_i\}$;
      repeat
        $R_i \leftarrow$ the current region in which a is placed;
        $q \leftarrow$ a randomly generated number ranging in [0,1];
        $R_k \leftarrow$ the next region selected according to Eq. (6);
        Move the ant a from R_i to R_k;
        Put R_k into $Order[a]$;
        Remove R_k from Π_a;
        Update the local pheromone according to Eq. (10);
      until $\Pi_a = \emptyset$;
    end
    Select the best ant that finds the shortest path in current solutions;
    Update the global pheromone according to Eq. (11);
  end
end

4.1. Region allocation phase

In this phase, the regions are assigned to free UAVs according to their effective time ratios. Assume a UAV U_i is located at R_j and tries to fly to and scan a region R_k. The time spent by U_i in flying from R_j to R_k and in scanning R_k is denoted by $TF_{i,j,k}$ and $TS_{i,k}$, respectively. Since the purpose of U_i is to cover R_k, the time spent in scanning R_k is meaningful and cannot be discarded. The time cost in scanning a region is referred to as an effective time. *Effective time ratio* is defined as the ratio of an effective time to the total time, can be calculated via

$$ETR_{i,j,k} = \frac{TS_{i,k}}{TF_{i,j,k} + TS_{i,k}} = \frac{A_k V_i^{max}}{A_k V_i^{max} + v_{i,k} W_i D_{j,k}} \tag{9}$$

As can be seen in Eq. (9), effective time ratios grow not only with increase of the scan areas of the target regions, but also with decrease of the distances between regions and UAVs. Regions with large scan ar-

eas and near to free UAVs are more likely selected and covered first. Algorithm 1 demonstrates the pseudo-code of this region allocation phase. Once a UAV U_i finishes its current scan task, the last region visited by U_i is chosen first. Then, effective time ratios of assigned regions are computed. Finally, the region which has the largest effective time ratio, would be chosen and allocated to U_i.

4.2. Order optimization phase

In the phase presented in Sect. 4.1, every UAV that is selected to perform the search task, obtains a sequence of regions waiting to be visited by this UAV one by one. However, the visiting orders of regions assigned to UAVs are usually not optimal and need to be optimized. An ACS-based optimization policy is used to adjust the orders such that the time consumption of UAVs could be minimized.

4.2.1. Solution construction

In this optimization policy, ants would construct solutions gradually by selecting and visiting regions one by one, and try to seek the shortest path to cover all of regions with the help of pheromone and heuristic information. Before the construction process starts, ants should be randomly placed to increase the diversity of their initial positioning and to avoid getting stuck at locally optimal value. In each step, a new region is selected according to Eq. (10).

$$R_k = \begin{cases} \underset{R_j \in \Pi_a}{arg\,\max} \tau(i,j)^{\alpha} \eta(i,j)^{\beta}, & \text{if } q \le q^0 \\ \text{Roulette Wheel Selection}, & \text{otherwise} \end{cases} \tag{10}$$

In Eq. (10), α ($\alpha > 0$) and β ($\beta > 0$) are predefined parameters, controlling the effect of the pheromone and heuristic information on ants selection. q_0 ($0 \le q_0 \le 1$) is predefined to control the exploitation and exploration behaviours of ants (Liu et al. [29]). Π_a represents the region set that an ant a is permitted to visit. $\tau(i,j)$ and $\eta(i,j)$ denote the pheromone and heuristic information accumulated between regions R_i and R_j, respectively.

When an ant tries to move from one region to another, q is randomly generated first. If $q \le q^0$, this ant goes to the region with the maximum pheromone and heuristic information, i.e., the region which has the largest value of $\tau(i,j)^{\alpha}\eta(i,j)^{\beta}$ is chosen to be visited next. Otherwise, this ant chooses a region by the roulette wheel selection with help of a predefined probability denoted by $p(i,j)$, which represents a probability that ants move from R_i to R_j, and could be computed via Eq. (11).

$$p(i,j) = \begin{cases} \frac{\tau(i,j)^{\alpha}\eta(i,j)^{\beta}}{\sum_{\forall k \in \Pi_a} \tau(i,k)^{\alpha}\eta(i,k)^{\beta}}, & \text{if } R_j \in \Pi_a \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

4.2.2. Pheromone update

A) Initialization. Derived from the ACS parameters proposed in Dorigo and Gambardella [28], the initial values of pheromones between each two regions are the same and set to τ_0 according to Eq. (12), in which L is an optimal tour length obtained from the nearest neighbour approach proposed in Rosenkrantz et al. [31].

$$\forall i, j \in [0, m], \tau(i,j) = \tau_0 = \frac{1}{m \times L} \tag{12}$$

Since the nearest neighbour first strategy is frequently used in path planning to optimize the visiting order, we initialize the heuristic information according to the distance between regions. The heuristic information is initialized as Eq. (13), and indicates that the closer region is more likely chosen to visit next by the ants during the construction process.

$$\forall i, j \in [0, m], \eta(i,j) = \frac{1}{D_{i,j}} \tag{13}$$

B) Local updating rule. In this work, ACS adopts both a local and a global pheromone update to take full advantage of the pheromone information. In the solution construction process of each ant, local update of pheromones occurs once an ant moves, and the rule can be expressed as:

$$\tau(i,j) = (1-\rho)\tau(i,j) + \rho\tau_0 \tag{14}$$

in which ρ is a predefined decay parameter ranging in (0,1).

When an ant moves from R_i to R_j, a local update occurs immediately according to Eq. (14). Since $\tau(i,j)$ is no less than τ_0 in most cases, the value of $\tau(i,j)$ decreases when update is finished. The probability that other ants directly move from R_i to R_j is reduced, and other paths are more likely chosen. Therefore, the local updating rule increases the diversity of solutions and avoids premature convergence [30].

C) Global updating rule. The global update of pheromones occurs after all ants have finished their tours and solutions have been constructed. Only the ant that has found the shortest path is permitted to update the pheromone. The updating rule can be written as:

$$\tau(i,j) = (1-\varepsilon)\tau(i,j) + \varepsilon \triangle \tau(i,j) \tag{15}$$

where

$$\triangle \tau(i,j) = \begin{cases} \frac{1}{L_{gb}}, & \text{if } (R_i, R_j) \in P_{gb} \\ 0, & \text{otherwise} \end{cases} \tag{16}$$

In Eq. (15), ε is the predefined decay parameter ranging in (0,1), L_{gb} represents the length of current global best path, and P_{gb} denotes the set of edges in the current global best path. Eq. (15) implies that only the pheromone on edges of current global best path would be enhanced, and the others will be weakened when the global update occurs. The global updating rule promotes spreading of the experience information, and efficiently improves the convergence speed of ACS-based algorithms. The order optimization phase terminates after the predefined evolution algebra of populations (i.e., the maximum number of evolutionary generations) is reached. Algorithm 2 shows the pseudo-code of this order optimization phase.

4.3. Running time analysis

Running time complexity of the ACS-based algorithm is analysed in this section. As we pointed out previously, the proposed algorithm contains two main phrases: the region allocation phase shown in Algorithm 1 and the order optimization phase shown in Algorithm 2. We first analyze the complexity of the first phase named region allocation.

As Algorithm 1 shows, the region allocation phase stops when all of regions are allocated, and at most circulates m times. In every cycle, the effective time ratios for unallocated regions are computed directly, and the region with the largest value is chosen and allocated to free UAVs. Since the maximum number of unallocated regions is m, the time complexity of each cycle is $O(m)$. Then, the running time cost of this phase is $O(m^2)$.

Now the time cost of the second phase named order optimization is analysed. From Algorithm 2 we can find that, the order optimization phase contains a structure of three nested loops. In lines from 7 to 19, the inner loop seeks an optimal visiting order for each ant. Since the number of ants used in the ACS is N, the inner loop circulates N times. In each cycle, a solution is constructed by the ant (from line 8 to 18). For each region that the ant is allowed to visit, $\tau(i,j)^{\alpha}\eta(i,j)^{\beta}$ is computed and the computation complexity is $O(M)$, where the parameter M represents the maximum number of regions assigned to the same UAV. The running time of the solution construction process is $O(M^2)$. Since the inner loop circulates N times, computation complexity of the inner part is $O(NM^2)$.

In the middle loop (from line 6 to 22), the evolution algebra of populations G controls the repetition times of the inner loop. Therefore, the middle loop runs in $O(GNM^2)$. Since the outer loop circulates at most n times, the complexity of the order optimization phase is $O(GNM^2n)$.

Through the above analysis, we can find that the complexity of the ACS-based approach is $O(m^2) + O(GNM^2n)$. Since the computation complexity depends not only on the region number m and the UAV number n, but also on the colony size N, the evolution algebra of populations G, and the maximum number of regions assigned to the same UAV M, our approach has a pseudo-polynomial time complexity. This complexity explosively grows with the increasing of m and M. However, the computation complexity of our approach is obtained based on the most pessimistic estimates, in simulation experiments we can find that our approach normally performs better in achieving good enough solutions.

5. Experiments and results

The correctness and efficiency of the proposed approach are verified in this section. Results of the proposed approach would be compared with those of five solutions including the MILP formulation built in Sect. 3.2, spatial-temporal clustering-based algorithm with the nearest-to-any strategy (STCA_NA) and with the nearest-to-end strategy (STCA_NE) presented in [27], genetic algorithm (GA) proposed in [32], and the short distance first algorithm (SDF) presented in [33]. Since all of the above methods proposed in the original literatures were adopted to settle the same kind of path planning problems (i.e., optimizing the flight paths and minimizing the task completion time of the working unmanned vehicles) as our approach, their parameter settings in the following experiments are the same as those used in the original literatures. All algorithms are programmed in C++ language on a development platform from the 2015 version of Microsoft Visual Studio. The machine we adopt has an Intel(R) Core(TM) i5-3320M CPU 2.60 GHz, 4.00 GB RAM and 64-bit Windows 8.1 operating system. In following paragraphs, regions with random parameters are generated first, then performances of the six approaches are tested in terms of execution time, task completion time and deviation ratio.

5.1. Region generation

We first introduce four definitions used in the generation process. Area ratio u_j is used to represent the ratio of the area of a given region R_j to the total area A_{total} of the flight range. System area ratio u is defined as the sum of area ratios of all regions, i.e., $u = \sum_{1 \le j \le m} u_j$. Similarly, a drag factor $d_{i,j}$ ranging in [0,1] is used to control the scan speed of UAV u_i, and expressed as the ratio of a UAV's scan speed to its maximum flight velocity, i.e., $d_{i,j} = V_{i,j}/V_i^{\max}$. System drag factor d is used to denote the average value of the drag factors.

Three major parameters are adopted to generate the regions: the region number m, the system area ratio u, and the system drag factor d. They directly determine peculiar features of regions and the scale of the coverage path planning problem studied in this work. The generation steps are the same as those adopted in [27]. First, UUniFast algorithm [34] is applied to generate a random area ratio u_j for every region R_j. The scan area of R_j is calculated via $A_j = u_j A_{total}$. Then, two numbers $\bar{x}_j$ and $\bar{y}_j$ are chosen randomly in the search range, and $(\bar{x}_j, \bar{y}_j)$ is identified as the center point coordinate of region R_j. Finally, for a given region R_j, drag factor $d_{i,j}$ is randomly chosen from $d - \delta$ to $d + \delta$, where δ is the smaller value of $1 - d$ and d. Scan speeds of UAVs are constructed as $V_{i,j} = d_{i,j} V_i^{\max}$.

5.2. Execution time evaluation

The performance of the six methods is first tested in terms of execution time, i.e., the time consumption of each method in finding a feasible solution. With a logarithmic scale, Fig. 2 demonstrates the average execution time of methods when they are used to seek paths on multiple regions. The regions have been generated when the parameters u and d are 0.05 and 0.9, respectively. The six curves named "MILP", "APPA", "GA", "SDF", "STCA_NA" and "STCA_NE" represent the experimental results of the MILP formulation, our ACS-based approach, GA method, SDF algorithm, STCA_NA algorithm and STCA_NE algorithm, respectively. A 3600 seconds time limit is set when the MILP formulation is adopted to solve the coverage problem, for MILP solvers almost never stops even after ten or more hours (Chen et al. [35]). When the time limit is exceeded, MILP solvers terminate the current runs and produce the best solutions that have been found, no matter those solutions are optimal or not. In Fig. 2 (also in all of figures used in this section), every point denotes the average value of results on 100 instances.

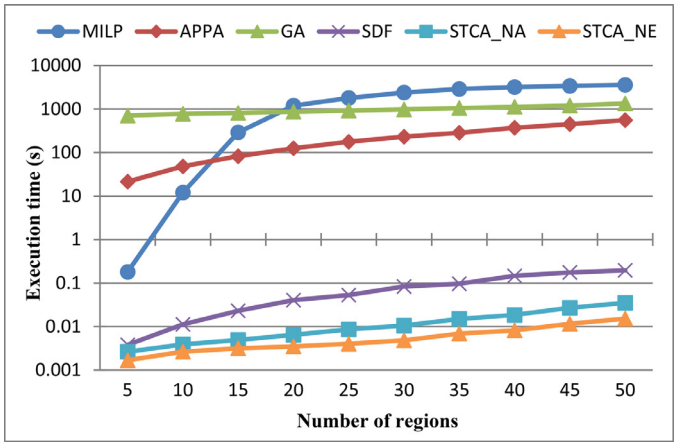

Fig. 2. Execution time of the mentioned six approaches on randomly generated regions.

As we can see in Fig. 2, the execution time of the six methods goes up gradually when the region number increases. Our method denoted by APPA, has a relatively lower time consumption than the MILP formulation and GA in most cases (when the region number is more than 10, to be more precise). Unfortunately the time cost of our method is larger than that of SDF, STCA_NA and STCA_NE, and the disadvantage is obvious. This is because SDF, STCA_NA and STCA_NE ignore the optimization process of visiting orders of regions, and produce poorer outcomes with respect to our method and GA. When the region number is 20, the time consumption of our method is 125.4 seconds, which is 8.0 and 2.1 times less than that of MILP and GA. Statistical results of the average time consumption of the six approaches are shown in Table 1.

In Fig. 3, the speed of our approach is tested by varying the numbers of regions and UAVs. As expected, the time consumption of our approach increases along with the number of regions. Meanwhile, time consumption of our approach gradually deceases along with the UAV number. One possible reason is that, when a larger number of UAVs are used to perform a coverage task, the maximum number of regions assigned to the same UAV becomes smaller, and our ACS-based approach could find good enough flight paths more quickly. Therefore, time consumption of our approach goes down with the increase of the UAV number. Results shown in Fig. 2 and Fig. 3 conform to the computational complexity analyzed in Sect. 4.3. Table 2 shows the time consumption statistic (i.e., the minimum time, average time, maximum time and standard deviation) of our approach when the UAV number is 4 and the region number increases from 5 to 50.

5.3. Task completion time evaluation

Now the performance of methods is verified in terms of task completion time, i.e., the time cost in covering the regions obtained by each method. Since each method could produce a different path for UAVs to visit the regions, the task completion times obtained by the six methods are different even on the same region sets.

In Fig. 4, task completion times of the six methods are verified by varying the region number. Regions have been generated when $d = 0.9$ and $u = 0.02$. As Fig. 4 shows, the six curves, respectively representing task completion times achieved by approaches, have a similar changing

Table 1
Time consumption statistic (in seconds) of the six approaches when the number of regions increases from 5 to 50.

Number	MILP	APPA	GA [32]	SDF [33]	STCA_NA [27]	STCA_NE [27]
5	0.18	21.52	705.96	0.004	0.0026	0.0017
10	12.05	48.07	776.79	0.011	0.0039	0.0027
15	290.83	82.31	814.89	0.023	0.0049	0.0032
20	1201.74	125.38	863.71	0.041	0.0065	0.0035
25	1822.93	176.54	923.04	0.053	0.0086	0.0040
30	2461.12	231.30	980.41	0.084	0.0105	0.0048
35	2900.35	284.79	1056.80	0.097	0.0150	0.0069
40	3213.61	370.64	1124.72	0.146	0.0184	0.0082
45	3401.29	449.40	1201.08	0.175	0.0271	0.0116
50	3599.82	557.66	1352.36	0.196	0.0351	0.0153

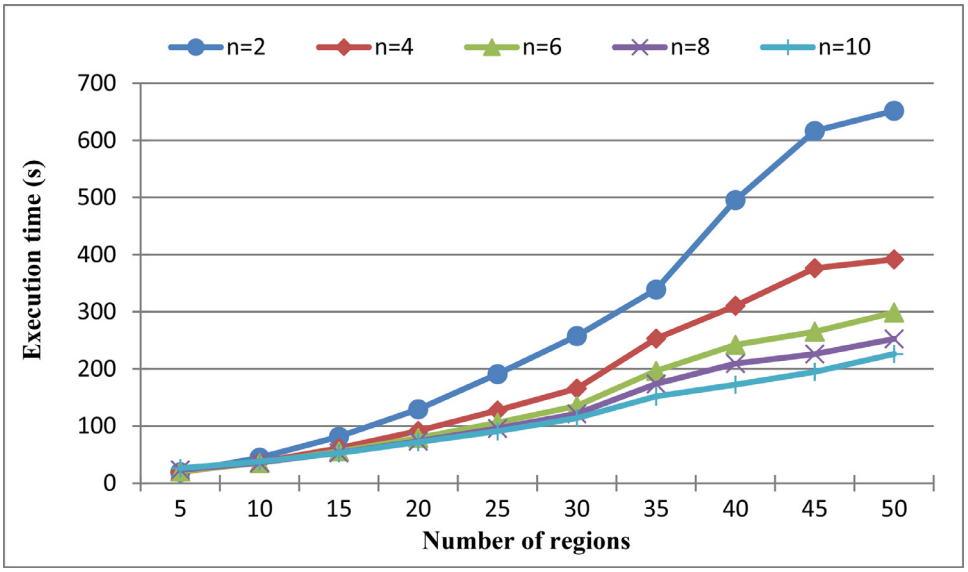

Fig. 3. Time cost of our method when the numbers of regions and UAVs respectively vary from 5 to 50 and from 2 to 10.

Table 2
Time cost statistic of our approach when the UAV number is 4.

Region number	5	10	15	20	25	30	35	40	45	50
Minimum time (s)	17.6	33.7	50.2	85.1	118.6	154.2	199.7	262.4	344.0	364.6
Average time (s)	19.0	37.0	61.3	91.5	127.3	165.8	253.1	310.5	376.4	391.8
Maximum time (s)	26.7	46.8	74.5	111.4	141.8	198.1	405.4	356.8	439.8	450.9
Standard deviation	1.71	1.88	3.19	4.35	4.95	8.41	43.66	14.82	15.42	16.69

trend that they first drop to minimum values and then gradually go up with increase of the region number. This is because the scan areas of regions are generated with a given system utilization. A smaller region number results in the generation of regions with larger scan areas. Since regions are allocated unevenly to UAVs, task completion times decrease in the first phase. Meanwhile, as the number of regions grows, the total length of flight paths between regions is increasing, and more time is required to fly from one region to another. Then the task completion times go up in the second phase.

As can be seen in Fig. 4, our approach obtains the second smallest task completion time, and the differences between our approach and MILP are small when the region number is more than 35. When 50 regions are used during the simulation experiments, the task completion time found by our approach is 145 minutes, which is 2 minutes more than that of MILP but 5 minutes, 17 minutes, 11 minutes and 14 minutes less than those of GA, SDF, STCA_NA and STCA_NE, respectively. Statistical results of the average task completion times obtained by the six approaches are shown in Table 3.

From Fig. 4 and Table 3 we can find that, when the number of regions is 10, the values of average task completion time obtained by our approach are close to those obtained by other heuristic algorithms. Additional tests are required to determine whether there is a significant difference between the performances of our approach and other algorithms. Since Wilcoxon's test is one of popular non-parametric statistical hypothesis test methods and designed for comparing two related data samples [36], we use a Wilcoxon signed-rank test in this section to determine the significance of differences between the task completion times obtained by path planning solutions.

In Table 4, 20 pairs of task completion times obtained by our approach and the GA method are tabulated in columns named "APPA" and "GA". We first calculate the differences and absolute differences between the paired date samples, then list the rank of the absolute differences while filling a blank if the absolute difference is 0. Columns "Negative Rank" and "Positive Rank" report the rank values where the differences in column "Diff" are negative and are positive, respectively. We sum the columns "Negative Rank" and "Positive Rank" to obtain T^- of 132 and T^+ of 39. The smaller of these values is the test statistic $T = 39$. Here we use a significance level $\alpha = 0.05$ and the sample size $n = 18$ (i.e., the number of data pairs with non-zero differences). From the Wilcoxon signed-ranks table, we find that the critical value $T_{crit} = 40$.

Table 3
Task completion time statistic (in minutes) of the six approaches on regions generated when $u = 0.02$ and $d = 0.9$.

Number	MILP	APPA	GA [32]	SDF [33]	STCA_NA [27]	STCA_NE [27]
5	133.3	147.9	151.8	161.2	156.9	158.7
10	120.0	127.3	129.3	137.9	133.8	136.9
15	121.6	130.0	131.0	139.0	132.3	135.0
20	126.5	133.0	134.1	141.6	136.4	137.5
25	131.6	135.7	137.0	145.7	140.5	141.9
30	135.5	139.0	140.1	148.8	143.3	145.6
35	138.1	141.1	142.5	151.1	145.2	147.3
40	140.4	142.4	145.0	154.2	149.2	150.6
45	142.5	144.5	148.2	158.0	152.6	155.1
50	143.2	145.5	150.8	162.6	156.4	159.6

Table 4
Wilcoxon signed-rank test for paired data samples coming from the task completion times provided by our approach and GA.

APPA	GA	Diff	Abs Diff	Rank of Abs Diff	Negative Rank	Positive Rank
127.14	130.51	-3.37	3.37	12	12	
125.33	124.15	1.17	1.17	3		3
117.00	119.78	-2.77	2.77	9	9	
133.10	130.14	2.96	2.96	11		11
132.58	130.10	2.48	2.48	8		8
128.41	131.84	-3.44	3.44	13	13	
124.71	134.66	-9.95	9.95	18	18	
122.73	122.73	0	0			
126.07	130.90	-4.83	4.83	15	15	
126.89	132.04	-5.16	5.16	16	16	
119.67	129.00	-9.33	9.33	17	17	
135.54	134.42	1.13	1.13	2		2
126.86	125.91	0.95	0.95	1		1
126.82	126.82	0	0			
132.75	134.38	-1.63	1.63	4	4	
125.57	121.77	3.80	3.80	14		14
129.89	131.66	-1.76	1.76	5	5	
133.32	136.12	-2.80	2.80	10	10	
125.21	127.31	-2.09	2.09	6	6	
127.48	129.63	-2.15	2.15	7	7	

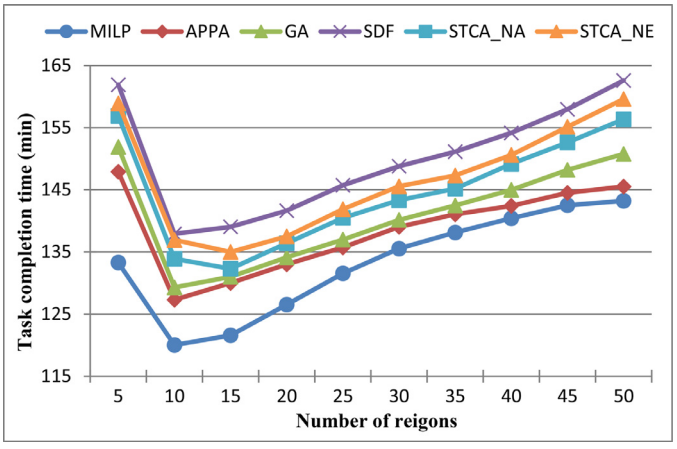

Fig. 4. Task completion times of methods on regions generated when $u = 0.02$ and $d = 0.9$.

Since $T_{crit} > T$, we accept the hypothesis that there is a significant difference between the performances of our approach and GA.

In Table 5, the Wilcoxon signed-rank test is used to determine whether the task completion times obtained by our approach and by other algorithms have significant differences. As pointed in [36], the accuracy of the Wilcoxon signed-rank test obvious depends on the size of data samples, and too small or too large sample size has adverse influence on analysing the degree of dependence between the confidence intervals and the p-values. The sample size n of each test can be calculated with the exact variance approach proposed in [37]. In this table, the significance level α is 0.05, and each pair of data sample contains a task completion time obtained by our approach and that obtained by another method. "GA", "SDF", "STCA_NA" and "STCA_NE" represent the comparison results between our approach and GA, SDF, STCA_NA and STCA_NE, respectively. Note that when the effective sample size n is sufficiently large, the test statistic T is equal to the absolute of the difference between the measured T^+ and T^-, i.e., $T = \mid T^+ - T^- \mid$, and the distribution of test statistic T can be approximated by a normal distribution with a mean μ and a variance σ^2 [38]:

$$\mu = \frac{n(n+1)}{4} \qquad \sigma^2 = \frac{n(n+1)(2n+1)}{24} \tag{17}$$

From Table 5 we can find that, in each statistical test, the p-value is smaller than 0.05, demonstrating that there is a significant difference

Table 5
Statistical test results on task completion times by using the Wilcoxon signed-rank test when the significance level α is 0.05.

Key variable	GA [32]	SDF [33]	STCA_NA [27]	STCA_NE [27]
sample size (n)	95	100	98	99
test statistic (T)	2844	4132	3145	1803
mean (μ)	2280	2525	2425.5	2475
variance (σ^2)	72580	84587	79637	82087
std dev (σ)	269.4	290.8	282.2	286.5
z-score (z_score)	2.093	5.525	2.550	2.345
p-value	0.0363	3.29E-08	0.0107	0.0190

Table 6
The task completion time statistic (in minutes) of the six approaches when the system drag factors is varying from 0.1 to 0.9.

Drag factor	MILP	APPA	GA [32]	SDF [33]	STCA_NA [27]	STCA_NE [27]
0.1	504.7	548.3	583.1	698.8	630.2	668.3
0.2	278.2	298.1	327.9	438.0	370.1	391.4
0.3	190.9	204.3	224.8	305.9	250.4	285.0
0.4	147.5	159.6	175.6	242.2	205.2	220.3
0.5	124.9	139.0	152.9	207.4	176.1	189.9
0.6	99.23	107.8	118.7	170.0	141.0	155.1
0.7	84.56	95.32	102.3	144.9	125.3	134.9
0.8	75.67	82.81	91.07	130.9	115.5	120.2
0.9	69.48	79.57	87.52	126.2	99.98	104.1

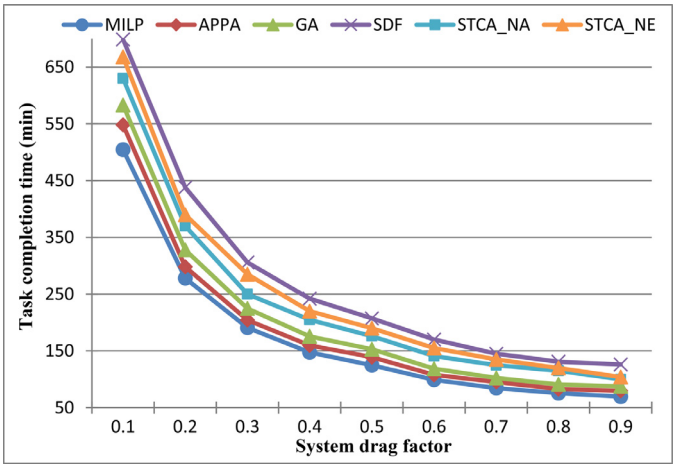

Fig. 5. Task completion times found by approaches under different system drag factors.

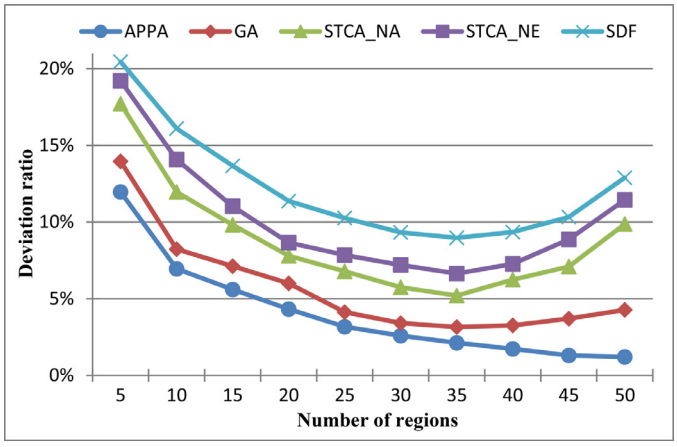

Fig. 6. The deviation ratios of the five inexact approaches on different numbers of regions.

between task completion times obtained by our approach and by GA, SDF, STCA_NA and STCA_NE. Since Fig. 4 has shown that our approach has the smallest task completion time in the heuristic algorithms, we can conclude that our approach has a significant better performance than the above four algorithms in terms of task completion time.

In the experiments of Fig. 5, we evaluate the task completion times of the six approaches by changing the system drag factors from 0.1 to 0.9. The region number is 15 and the system area ratio is 0.01. Since a large system drag factor leads to the generation of regions with high drag factors, task completion times found by approaches decrease when system drag factors increase. Our approach has the second best results on a given system drag factor. When $d = 0.2$, the task completion times achieved by MILP, our approach APPA, GA, SDF, STCA_NA and STCA_NE are 278 minutes, 298 minutes, 327 minutes, 438 minutes, 370 minutes and 391 minutes, respectively. The results indicate that our approach performs a better performance than SDF, GA, STCA_NA and STCA_NE. Table 6 lists the average task completion time statistic of approaches when the system drag factors is varying.

5.4. Deviation ratio evaluation

In this subsection, we begin to test the performance of methods in terms of deviation ratio [39], which is defined as a ratio of the relative difference between task completion times found by an inexact method (our approach APPA, GA, SDF, STCA_NA and STCA_NE here) and found by an exact method (the MILP formulation here), i.e., $dr = (t_{inexact} - t_{exact})/t_{exact}$.

In the experiments of Fig. 6, regions have been generated when $u = 0.02$ and $d = 0.9$, and the deviation ratios of the five inexact approaches are tested on different numbers of regions. We can find that the deviation ratios of GA, STCA_NA, STCA_NE and SDF have the same change trend

Table 7
Deviation ratio statistic of the five inexact approaches on different numbers of regions.

Number	APPA	GA [32]	STCA_NA [27]	STCA_NE [27]	SDF [33]
5	11.96%	13.96%	17.71%	19.21%	20.45%
10	6.95%	8.23%	11.95%	14.07%	16.10%
15	5.60%	7.12%	9.81%	11.02%	13.66%
20	4.32%	6.00%	7.79%	8.66%	11.37%
25	3.17%	4.13%	6.78%	7.85%	10.26%
30	2.59%	3.41%	5.75%	7.20%	9.32%
35	2.12%	3.15%	5.20%	6.64%	8.97%
40	1.73%	3.26%	6.24%	7.26%	9.34%
45	1.30%	3.70%	7.10%	8.86%	10.32%
50	1.21%	4.27%	9.87%	11.44%	12.89%

that they first go down from more than 14% to less than 8.9% and then grow up gradually. However, the deviation ratio of our approach just decreases with increase of the region number. Our approach has the best deviation ratio on a given number of regions. When the region number is 50, the deviation ratio of our approach is 1.2%, which is 2.5, 7.2, 8.5 and 9.7 times smaller than those of GA, STCA_NA, STCA_NE and SDF, respectively. Table 7 shows the average deviation ratio statistic of the five approaches.

In Fig. 7, we verify the deviation ratios of the five inexact approaches by tuning the UAV number from 2 to 10. Regions have been generated with $u = 0.01$ and $d = 0.8$. The deviation ratios first grow up and then go down gradually with the UAV number. One of the most possible reason is that all of the five approaches are non-exact and only produce approximated solutions for the planning problem. When the UAV number increases, more and more valid flight paths should be considered, thus in the first phase the deviation ratios increase. While, as the number of UAVs grows, the solution space that approaches should seek grows

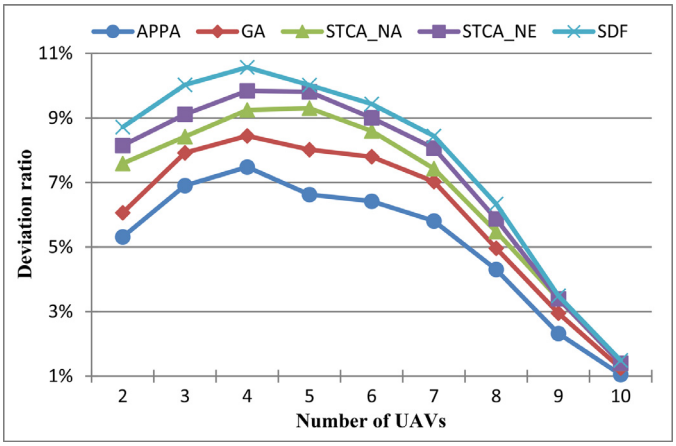

Fig. 7. The deviation ratios of approaches when 20 regions are covered.

Table 8
Deviation ratio statistic of the five inexact approaches when the number of UAVs increases from 2 to 10.

Number	APPA	GA [32]	STCA_NA [27]	STCA_NE [27]	SDF [33]
2	5.32%	6.06%	7.59%	8.15%	8.72%
3	6.90%	7.92%	8.42%	9.11%	10.03%
4	7.44%	8.44%	9.25%	9.84%	10.54%
5	6.63%	8.02%	9.30%	9.81%	10.02%
6	6.42%	7.80%	8.60%	9.01%	9.43%
7	5.81%	7.03%	7.43%	8.06%	8.44%
8	4.30%	4.96%	5.48%	5.87%	6.34%
9	2.32%	2.95%	3.40%	3.39%	3.49%
10	1.05%	1.24%	1.39%	1.40%	1.48%

dramatically. More and more optimal solutions cannot be found by the MILP formulation when a time limit is set. Therefore, the deviation ratios decrease in the second phase.

As can be seen in Fig. 7, when four UAVs are adopted in these experiments, the deviation ratios of SDF, STCA_NE, GA and our approach reach a maximum, and their values are 10.5%, 9.8%, 8.4% and 7.4%, respectively. The maximum deviation ratio of STCA_NA appears when the UAV number is five, and its value is 9.3%. Our approach has the smallest deviation ratio, and the results indicate that our approach performs better than both other methods in terms of deviation ratio. Statistical results of the average deviation ratios of the five methods are shown in Table 8.

6. Conclusion

In this research, we tried to settle the coverage path planning problem for UAVs with different flight velocity, altitude and scan width. We first analysed the constraint conditions and objective function of the problem and presented an exact linear programming-based formulation to solve this problem. Then, inspired by the foraging behaviours of ants, we proposed an ACS-based approach to allocate regions and optimize the visiting orders so as to minimize the task completion time in large-scale cooperative search systems. The proposed approach takes into consideration of the influence of the distances and scan areas of regions, and provides reasonable paths for UAVs with an objective of visiting and scanning all of regions correctly and efficiently.

Although the proposed approach can obtain reasonable solutions in affordable times, there are still lots of aspects that need to deserve further investigation. For example, we want to adopt some more advanced models and technologies to enhance the efficiency and accuracy of the proposed approach, and try to see whether some of results proposed in this paper could be adopted to support other application scenarios.

Declaration of Competing Interest

The authors declare no conflict of interest.

Acknowledgment

This research is supported in part by the National Natural Science Foundation of China No. 62106202, the Aeronautical Science Foundation of China No. 2020Z023053004, the National Key Research and Development Program No. 2018YFB2101304, and the Special Civil Aircraft Research Program No. MJ-2017-S-39.

References

[1] V. Saxena, J. Jalden, H. Klessig, Optimal UAV base station trajectories using flow-Level models for reinforcement learning, IEEE Trans. Cognit. Commun. Networking 5 (4) (2019) 1101–1112.
[2] A.V. Savkin, H. Huang, A method for optimized deployment of a network of surveillance aerial drones, IEEE Syst. J. 13 (4) (2019) 4474–4477.
[3] X. Zhang, Y. Fang, X. Zhang, J. Jiang, X. Chen, A novel geometric hierarchical approach for dynamic visual servoing of quadrotors, IEEE Trans. Ind. Electron. 67 (5) (2020) 3840–3849.
[4] C. Ramirez Atencia, J. Del Ser, D. Camacho, Weighted strategies to guide a multi-objective evolutionary algorithm for multi-UAV mission planning, Swarm Evol Comput 44 (2019) 480–495.
[5] Y. Peng, H. Wang, Z. Su, Cooperative path planning with applications to target tracking and obstacle avoidance for multi-UAVs, Aerosp. Sci. Technol. 54 (2016) 10–22.
[6] A. Ollero, K. Kondak, 10 years in the cooperation of unmanned aerial systems, in: 2012 IEEE International Conference on Intelligent Robots and Systems, 2012, pp. 5450–5451.
[7] X. Zhao, Q. Zong, B. Tian, B. Zhang, M. You, Fast task allocation for heterogeneous unmanned aerial vehicles through reinforcement learning, Aerosp. Sci. Technol. 92 (2019) 588–594.
[8] P.K. Das, H.S. Behera, B.K. Panigrahi, A hybridization of an improved particle swarm optimization and gravitational search algorithm for multi-robot path planning, Swarm Evol Comput 28 (2016) 14–28.
[9] Z. Chen, S. Wang, J. Wang, K. Xu, T. Lei, H. Zhang, X. Wang, D. Liu, J. Si, Control strategy of stable walking for a hexapod wheel-legged robot, ISA Trans 108 (2021) 367–380.
[10] M. Mavrovouniotis, C. Li, S. Yang, A survey of swarm intelligence for dynamic optimization: algorithms and applications, Swarm Evol Comput 33 (2017) 1–17.
[11] J. Chen, P. Han, Y. Liu, X. Du, Scheduling independent tasks in cloud environment based on modified differential evolution, Concurrency and Computation: Practice and Experience (2021) e6256.
[12] Y. Li, H. Chen, J.E. Meng, X. Wang, Coverage path planning for UAVs based on enhanced exact cellular decomposition method, Mechatronics 21 (5) (2011) 876–885.
[13] W. Yao, N. Qi, N. Wan, Y. Liu, An iterative strategy for task assignment and path planning of distributed multiple unmanned aerial vehicles, Aerosp. Sci. Technol. 86 (2019) 455–464.
[14] R.J. Szczerba, P. Galkowski, I.S. Glicktein, N. Ternullo, Robust algorithm for real–time route planning, IEEE Trans Aerosp Electron Syst 36 (3) (2000) 869–878.
[15] H. Duan, P. Li, Y. Shi, X. Zhang, C. Sun, Interactive learning environment for bio-inspired optimization algorithms for UAV path planning, IEEE Trans. Educ. 58 (4) (2015) 276–281.
[16] J. Colito, Autonomous mission planning and execution for unmanned surface vehicles in compliance with the marine rules of the, road (2007).
[17] C.D. Franco, G. Buttazzo, Coverage path planning for UAVs photogrammetry with energy and resolution constraints, J. Intell. Rob. Syst. 83 (3–4) (2016) 1–18.
[18] C. Zhao, Y. Liu, J. Zhao, Path planning method of UAV area coverage searching based on pega, Science and Technology Review 32 (22) (2014) 85–90.
[19] G. Gramajo, P. Shankar, An efficient energy constraint based UAV path planning for search and coverage, International Journal of Aerospace Engineering 2017 (2017) 1–13.
[20] I. Maza, A. Ollero, Multiple UAV cooperative searching operation using polygon area decomposition and efficient coverage algorithms, in: R. Alami, R. Chatila, H. Asama (Eds.), Distributed Autonomous Robotic Systems 6, Springer Japan, Tokyo, 2007, pp. 221–230.
[21] P. Vincent, I. Rubin, A framework and analysis for cooperative search using UAV swarms, in: ACM Symposium on Applied Computing, 2004, pp. 79–86.
[22] S. Aggarwal, N. Kumar, Path planning techniques for unmanned aerial vehicles: a review, solutions, and challenges, Comput Commun 149 (2020) 270–299.
[23] Y. Yuan Yuan, C.P. Min, H.Y. Zhu, L.C. Shen, Multiple UCAV mission assignment based on integer programming, Information and Control 34 (5) (2005) 548–552.
[24] N. Batsoyol, Y. Jin, H. Lee, Constructing full-coverage 3D UAV ad-hoc networks through collaborative exploration in unknown urban environments, in: 2018 IEEE International Conference on Communications (ICC), 2018, pp. 1–7.
[25] X. Chu Ding, R.R. Amir, M. Egerstedt, Multi-UAV convoy protection: an optimal approach to path planning and coordination, IEEE Trans. Rob. 26 (2010) 256–268.
[26] J. Chen, C. Du, P. Han, X. Du, Work-in-progress: Non-preemptive scheduling of periodic tasks with data dependency upon heterogeneous multiprocessor platforms, in: 2019 IEEE Real-Time Systems Symposium (RTSS), 2019, pp. 540–543.

[27] J. Chen, C. Du, Y. Zhang, P. Han, W. Wei, A clustering-based coverage path planning method for autonomous heterogeneous UAVs, IEEE Trans. Intell. Transp. Syst. (2021) 1–11., doi:10.1109/TITS.2021.3066240.
[28] M. Dorigo, L.M. Gambardella, Ant colony system: a cooperative learning approach to the traveling salesman problem, IEEE Trans. Evol. Comput. 1 (1) (1997) 53–66.
[29] X. Liu, Z. Zhan, J.D. Deng, Y. Li, T. Gu, An energy efficient ant colony system for virtual machine placement in cloud computing, IEEE Trans. Evol. Comput. 22 (1) (2018) 113–128.
[30] Z. Chen, Z. Zhan, Y. Lin, Y. Gong, T. Gu, F. Zhao, H. Yuan, X. Chen, Q. Li, Multiobjective cloud workflow scheduling: a multiple populations ant colony system approach, IEEE Trans Cybern 49 (8) (2019) 2912–2926.
[31] D. Rosenkrantz, R. Stearns, P. Lewis, An analysis of several heuristics for the traveling salesman problem, SIAM J. Comput. 6 (3) (1977) 563–581.
[32] B. Li, S. Patankar, B. Moridian, N. Mahmoudian, Planning large-scale search and rescue using team of UAVs and charging stations, in: IEEE International Symposium on Safety, Security, and Rescue Robotics, 2018, pp. 1–8.
[33] J. Chen, C. Du, X. Lu, K. Chen, Multi-region coverage path planning for heterogeneous unmanned aerial vehicles systems, in: IEEE International Conference on Service-Oriented System Engineering, 2019, pp. 356–361.
[34] E. Bini, M. Di Natale, G. Buttazzo, Sensitivity analysis for fixed-priority real-time systems, Real-Time Systems 39 (1–3) (2008) 5–30.
[35] J. Chen, C. Du, F. Xie, B. Lin, Scheduling non-preemptive tasks with strict periods in multi-core real-time systems, J. Syst. Archit. 90 (2018) 72–84.
[36] E. Osaba, E. Villar-Rodriguez, J. Del Ser, A.J. Nebro, D. Molina, A. LaTorre, P.N. Suganthan, C.A. Coello Coello, F. Herrera, A tutorial on the design, experimentation and application of metaheuristic algorithms to real-world optimization problems, Swarm Evol Comput 64 (2021) 100888.
[37] G. Shieh, S.-L. Jan, R. Randles, Power and sample size determinations for the wilcoxon signed-rank test, J Stat Comput Simul 77 (8) (2007) 717–724.
[38] J.W. Pratt, J.D. Gibbons, Concepts of nonparametric theory, New York: Springer-Verlag. Springer series in statistics, 1981.
[39] J. Chen, C. Du, F. Xie, Z. Yang, Schedulability analysis of non-preemptive strictly periodic tasks in multi-core real-time systems, Real-Time Systems 52 (3) (2016) 239–271.