Optimal Searching Time Allocation for Information Collection Under Cooperative Path Planning of Multiple UAVs

Yanmin Li, Lihua Liu, Jibing Wu, Mao Wang, Haohao Zhou, and Hongbin Huang

Abstract—Unmanned Aerial Vehicles (UAVs) play an essential role in information collection where routing and time scheduling are two critical factors and are considered sequentially. Given the paths of UAVs, metaheuristic algorithms are explored to allocate searching time of UAVs, which are hard to guarantee the optimal solution for time allocation. In this paper, a novel searching time allocation under cooperative path planning (GENERAL) is proposed to solve the optimal solution of the time allocation given paths of UAVs. GENERAL adopts a semi-greedy construction and a repair procedure to initialize and amend the routing solutions during iterations. Motivated by the Newton's method from convex optimization domains, we introduce a new Perturbed Parametric Nonlinear Complementarity Problem function (PPNCP-function), which reformulates the time allocation problem as a smoothing system of equations according to Karush-Kuhn-Tucker (KKT) theorem. Then a smoothing Newton method is introduced to obtain the optimal time allocation solution with superlinear convergence. Experimental results empirically indicate the GENERAL's effectiveness compared to Ant Colony Optimization with Simulated Annealing (ACO-SA) and Genetic Algorithm (GA). Besides, some numerical results indicate that the smoothing Newton method with PPNCP function is promising. The emerging aspects of this paper include: 1) it integrates the path planning with time allocation problem for maximizing the information collection reward; 2) a new PPNCP-function and a smoothing Newton method have been proposed to solve the optimal time allocation under path planning of UAVs; 3) the theoretical convergence analysis of the smoothing Newton method has been provided in this paper.

Index Terms—GENERAL, information searching, path planning, time allocation, unmanned aerial vehicles.

I. INTRODUCTION

Unmanned aerial vehicle (UAV) can provide a quickly deployable infrastructure for collecting information in some dull, dirty, or dangerous situations [1], such as disaster reliefs [2] and forest fire detections [3]. Multiple UAVs are often sent out to search for and collect information since multiple UAVs can cover a larger area in a shorter time than a single UAV, especially when it is bad weather or the situation is cluttered [4]. For example, following the outbreak of the COVID-19, researchers are attempting to deploy UAVs equipped with infrared cameras to collect the large-scale temperature measurements information [5].

The information collection of UAVs is usually regarded as the integration of combinatorial optimization and nonlinear programming, where routing and time allocation decisions are the critical decisions, the optimization of which can significantly improve the efficiency of UAVs. Traditional methods for UAVs scheduling follow the paradigm that develops a two-stage solution to consider paths and time sequentially.

This first stage focuses on the routing of UAVs regardless of searching difficulties and information values in different regions. The exact methods are first used to find competitive solutions. But the problems with a large group of variables or constraints are usually intractable by these algorithms [6]. Those heuristics and metaheuristics, including greedy search [7], Ant Colony Optimization (ACO) [9], are widely used because of their more comfortable operating and lower computational complexity. The semi-greedy algorithm combines both the greedy and random methods to balance the diversity and efficiency of solutions. Existing semi-greedy algorithms have been verified on the Steiner tree problem [10], maximum satisfiability problem [8], and Traveling Salesman Problem (TSP) [11], but little work is applied to multiple UAVs path planning.

In the second stage, the UAVs’ searching time allocation is studied to maximize UAVs’ reward (information collected) on the scheduled paths in the first stage. The strategies of the searching time allocation can usually be divided into two categories: pre-scheduled and real-time schedules. Pre-schedule of time allocation is suitable for situations where there is no need to temporarily interrupt or change the scheduled plan when UAVs conduct their missions. The real-time schedule of time allocation can provide better performance than the pre-schedule since the scheduled plan can be improved by real-time observations [12]. However, it requires online communication, real-time process, and dynamic control of all UAVs, which lacks robustness and is challenging to implement efficiently [13], [14].

The pre-schedule of time allocation is focused on in this paper since it is easy to deploy [15]. Metaheuristic methods, which rely on local improvement operators or population evolution
processes, are the most common approaches in solving time allocation problem, i.e., Simulated Annealing (SA) [16], Genetic Algorithm (GA) [17], and Particle Swarm Optimization (PSO) [18]. However, these metaheuristic approaches are hard to guarantee the optimal solution for time allocation given paths of UAVs, since they are easy to fall into local optimal solution [8].

To alleviate the above issue, a novel searchinG timE alloca-tioN undEr coopeRative pAth pLaning (GENERAL) of multiple UAVs is proposed to find the optimal solution of searching time allocation given paths of UAVs. GENERAL first constructs some candidate paths by the semi-greedy construction in the first stage. In the second stage, GENERAL transforms the time allocation problem into a semi-smooth equation according to Karush-Kuhn-Tucker (KKT) theorem [19]. Then the optimal solution of this equation is solved by the smoothing Newton method based on the Perturbed Parametric Nonlinear Complementarity Problem function (PPNCP-function) introduced in this paper. The diagram and four crucial parts of the GENERAL are shown in Fig. 1.

The main contributions of this paper are present as follows:

- A practically-motivated information searching model is developed, where routing and time allocation decisions are the two most critical decisions. Based on the model, a novel routing and searching time allocation method, GENERAL is proposed to solve the optimal solution of the time allocation given paths of UAVs.
- The PPNCP-function is introduced, which covers the Fischer-Burmeister and other well-known functions in the special cases. Based on the function, a novel smoothing Newton method is proposed to solve the equation of the time allocation which is transformed by the KKT theorem. The theoretical convergence analysis of this method has been provided in some detail.
- Extensive experiments on multiple UAVs show GENERAL achieves better performance than the compared methods, which empirically indicates the effectiveness of the proposed method.

The paper is structured as follows. Section 2 presents the literature review, and Section 3 illustrates the problem and the model. Section 4 introduces the GENERAL algorithm and the smoothing Newton method with PPNCP-function. The experiments under different instances are carried out in Section 5. Finally, Section 6 concludes this paper.

II. LITERATURE REVIEW

In the two-stage method, the first phrase is always described as the Multiple Traveling Salesman Problem (MTSP) extension. In [20], the information collection problem involved planning the search of a transportation graph in the minimum possible time and was described as a variant of MTSP. Stump et al. [21] also considered the first stage as finding travel sequences of discrete sites, which was solved by those methods developed in the operational research community. The MTSP model was also used by Ergezer et al. [22] to determine the visiting sequence of desired regions while avoiding flying over forbidden regions and reaching the destination.

Other exact methods, including Depth-First Search (DFS) [23], Breadth-First Search (BFS) [24], and Branch and Bound (BB) [25], were also used to find competitive solutions, even not the best within a reasonable time. Furthermore, the big graph with a large group of variables or constraints is usually intractable by these algorithms [6]. On the other hand, the computational intelligence methods [26], as well as software-in-the-loop simulations [27], were widely used because of their more comfortable operating and lower computational complexity [28]–[30].

In [9], two optimization methods, including ACO and SA, are used to find the shortest possible path between different nodes. In [31] and [32], the GA was implemented to quickly calculate a feasible and quasi-optimal path while the destinations are changed during its mission. Also, the Network-based Hetero-geneous Particle Swarm Optimization (NHPSO) was presented in [33] and outperformed other PSO algorithms on their test
Fig. 2. Routing a fleet of UA Vs to search for and collect information in a set of disjoint regions.

cases. In [34] and [35], a novel evolutionary search algorithm was used to capture the structured knowledge from earlier solutions, which can be further applied to schedule paths whenever faced with a similar task. Experiments in these papers showed that metaheuristic methods could find promising solutions faster than the exact algorithms, especially when the number of variables is sizeable [36].

The time allocation problem, in the second stage, was initiated by Koopman [37] in 1980. He claimed and proved that any optimal resource allocation problem must involve an exponential function. Xia et al. [38] viewed the search problem as a finite horizon partially observable decentralized Markov decision process for the real-time schedule. However, its theories only hold for the scenario where there are no more than two UA Vs.

On the other hand, a promising way is to seek computational intelligence to optimize the resource allocation problem in advance. For example, Moskal [15] maximizes global information gain using data from each region’s sampled points. In [16] and [39], the SA algorithm was introduced to optimally allocate tasks and solve the nonlinear problem, respectively. In [40], a kernel matrix and probability model was introduced to balance the relationship between diversity and convergence during the high-dimensional optimization process. In [17] and [41], the evolutionary optimization method was applied to solve resource allocation problem in the presence of various uncertainties.

Besides, Zhang [42] transformed the search problem to a semi-smooth equation, then proposed a Newton-type method algorithm under the simplest smoothing function, i.e., the Fischer-Burmeister function. The function is also used in [43] to map the optimality conditions to a nonlinear system of equations which is solved using Newton’s method. However, the properties of other smoothing functions were not studied in these literatures.

From the above review, it is clear that some pioneering works about the routing problem in graphs and the time allocation method have been done. However, these current methods are hard to guarantee the optimal solution for time allocation given paths of UA Vs, since they are easy to fall into local optimal solution [8]. To alleviate these issues, a mathematical model is developed, and a novel searching time allocation under cooperative path planing of multiple UA Vs is proposed to find the optimal solution of searching time allocation given paths of UA Vs.

III. PROBLEM DESCRIPTION AND MODEL DEVELOPMENT

A. Problem Description

In some cases, such as searching for wreckage, the UA Vs’ mission area is very far from the airport. Therefore, to simplify the model, we only consider the schedules when UA Vs fly in the mission area, not in the whole flight process.

As illustrated in Fig. 2, the mission area is divided into many equal-sized, tightly connected regions. Its center coordinates represent the location of a region. Only a part of the regions contains valuable information. These regions must be searched, and each of them should be searched only once. Assumes that all UA Vs are assigned with the same start and end point. Each UAV needs to travel between regions and spends some time in regions for information searching.

B. Notations

The notations used in the model development and solution are presented in Table I.

Besides, the following notations will be helpful in the part of smoothing Newton method. All vectors are column ones, and the superscript $T$ denotes transpose. $H'$ denotes the derivative of the function $H$. $\partial H$ is the generalized Jacobian of function $H$. $\mathbb{R}^n$ (respectively, $\mathbb{R}$) denotes the space of $n$-dimensional real column vectors (respectively, real numbers). $\mathbb{R}_+^n$ and $\mathbb{R}_{++}^n$ denote the nonnegative and positive orthants of $\mathbb{R}^n$. $\mathbb{R}_+$ (respectively,
TABLE I  
NOTATIONS AND IMPLICATIONS

| Sets and Matrix: |  |
|------------------|---|
| $\mathcal{U}$ | set of UAVs, $\mathcal{U} = \{1, 2, \ldots, M\}$. |
| $\mathcal{A}$ | set of regions to search, $\mathcal{A} = \{1, 2, \ldots, L\}$. |
| $T_m^m$ | travel time matrix of UAV $m$. |

| Parameters: |  |
|------------|---|
| $0, L + 1$ | indices represent the start and end points, respectively. |
| $t_{ij}^m$ | a given integer provides the travel time for UAV $m$ between region $i$ and $j$. |
| $t_{0,j}^m$ | a given integer provides the time for UAV $m$ to travel from start point to region $j$. |
| $t_{i,L+1}^m$ | a given integer provides the time for UAV $m$ to travel from region $i$ to end point. |
| $c_i$ | a given float provides the priority probability for one piece of information exists in region $i$. |
| $-b_i$ | a given float provides the difficulty of searching for information in region $i$. |
| $\lambda_i$ | a given integer provides the value of information in region $i$. |
| $t_{i}^{\text{max}}$ | a given integer provides the maximum time for UAV $m$ to fly in the mission area. |

| Variables: |  |
|-----------|---|
| $m$ | index of a UAV. |
| $d_{ij}^m$ | a binary variable represents whether UAV $m$ travels from region $i$ to $j$. |
| $d_{0,j}^m$ | a binary variable represents whether UAV $m$ travels from start point to region $j$. |
| $d_{i,L+1}^m$ | a binary variable represents whether UAV $m$ travels from region $i$ to end point. |
| $d_{0,1}^m$ | a binary variable represents whether UAV $m$ travels from start point to end point. |
| $x_i^m$ | an integer variable represents the search time of region $i$ for UAV $m$. |
| $k_i^m$ | an integer variable represents the order index of region $i$ for UAV $m$. |

$\mathbb{R^+}$ denotes the nonnegative (respectively, positive) orthant of $\mathbb{R}$. The symbol “:=” means “define”. Let $N := \{1, 2, \ldots, n\}$. For any vector $v \in \mathbb{R}^n$, we denote by $\text{diag}\{v_i : i \in N\}$ the diagonal matrix whose $i$th diagonal element is $v_i$ and $\text{vec}\{v_i : i \in N\}$ the vector of $v$. We use $(u, v)$ for the column vector $(u^T, v^T)^T$. The matrix $I$ stands for the identity matrix of arbitrary dimensions. The $\| \cdot \|$ represents the 2-norm. Lastly, for any $p, q \in \mathbb{R}^+$, $p = O(q)$ (respectively, $p = o(q)$) means $p/q$ is uniformly bounded (respectively, tends to zero) as $q \to 0$.

C. Model Development

Consider a search problem in which unmanned aerial vehicles $\mathcal{U} = \{1, 2, \ldots, M\}$ are assigned to search information from a set of regions $\mathcal{A} = \{1, 2, \ldots, L\}$. All UAVs share the same start and end points. Each UAV $m \in \mathcal{U}$ flies from the start point, then visits a subset of regions in $\mathcal{A}$ and leaves the mission area within a given flight time $t_{i}^{\text{max}}$. All regions in $\mathcal{A}$ must be searched, and any region $\alpha \in \mathcal{A}$ is searched only once. At most one piece of information exists in region $\alpha \in \mathcal{A}$ with the probability of $e_\alpha$ and the proportional difficulty of $-b_\alpha$. The value of information in region $\alpha$ is $\lambda_\alpha$.

Consider a UAV $m$ flies to region $\alpha$ and spends $x$ units of time to search for information. $p(x|e_\alpha)$ is the conditional probability of finding the information in the $x$th unit time given information existence probability $e_\alpha$. The UAV may also fail to collect any information in region $\alpha$ after spending $x$ units of time with probability $1 - P(x|e_\alpha)$. We call $P(x|e_\alpha)$ as the detection function of the region, which satisfies

$$P(x|e_\alpha) = \sum_{x=1}^{P(x)} p(z|e_\alpha)$$ (1)

We assume that all UAVs take random searches and the detection function follows the search theory by Koopman [37]. So $P(x|e_\alpha)$ takes the form

$$P(x|e_\alpha) = e_\alpha \left(1 - \exp(-b_\alpha x)\right)$$ (2)

We present the formulation that maximizes the total benefits of collecting information from regions.

$$\max \sum_{m=1}^{M} \sum_{i=1}^{L} c_i \lambda_i \left(1 - \exp(-b_i x_i^m)\right)$$ (3)

s.t. $\sum_{m=1}^{M} \sum_{i=1}^{L} d_{ij}^m = 1, j = 1, \ldots, L,$ (4)

$$\sum_{i=1}^{L} d_{0,i}^m + d_{0,L+1}^m = 1, m = 1, \ldots, M,$$ (5)

$$\sum_{j=1}^{L} d_{0,i}^m + d_{0,L+1}^m = 1, m = 1, \ldots, M,$$ (6)

$$d_{0,i}^m + \sum_{j=1}^{L} d_{ij}^m = d_{i+1}^m + \sum_{j=1}^{L} d_{ij}^m, m = 1, \ldots, M, i = 1, \ldots, L,$$ (7)

$$\sum_{i=0}^{L} \sum_{j=1}^{L} d_{ij}^m (v_i^m + x_j^m) + \sum_{i=0}^{L} d_{i+1}^m (v_i^m + x_j^m) \leq t_{i}^{\text{max}}, m = 1, \ldots, M,$$ (8)

$$k_j^m - k_i^m \geq d_{ij}^m (1 - d_{ij}^m)(L - 1), m = 1, \ldots, M, i, j = 1, \ldots, L,$$ (9)

$$d_{ij}^m \in \{0,1\}, m = 1, \ldots, M, i, j = 0, \ldots, L + 1,$$ (10)

$$k_i^m, x_j^m \in \mathbb{Z}^+, m = 1, \ldots, M, i = 1, \ldots, L.$$ (11)

In the formulation, the objective function (3) is the total search benefits obtained from all UAVs. Constraint (5) ensures that each UAV arrives at the end point only once, while constraint (6) requires that each UAV leaves the start point only once. Constraint (7) enforces that the number of UAVs arriving at a region is equal to the number of UAVs leaving the region. Constraint (8) requires that the total flight time of each UAV
Algorithm 1: GENERAL.

Input: number of UAVs $M$, travel time matrices of UAVs $\{T^m : m \in U\}$, maximum flight time of UAVs $\{t^m_{\max} : m \in U\}$, information values $\{\lambda_i : i \in A\}$, information searching difficulties $\{-b_i : i \in A\}$.

Output: travel sequences of vehicles $S^*$, time allocation strategies $X^*$.

Initialization:
1: Initialize the value of the incumbent: $h^* \leftarrow 0$.
Compute and update:
2: while stopping criterion not satisfied do
3: $S \leftarrow$ Semi-greedy Construction;
4: if $S$ is not feasible then
5: $S \leftarrow$ Repair($S$);
6: end if
7: for $S_m \in S$ do
8: $S_{m} \leftarrow$ Tabu-Search ($S_{m}$, $T^m$, $t^m_{\max}$);
9: $X_{m} \leftarrow$ smoothing Newton method ($S_{m}$);
10: end for
11: if $h(X) > h^*$ then
12: $S^* \leftarrow S$;
13: $X^* \leftarrow X$;
14: $h^* \leftarrow h(X)$;
15: end if
16: end while
17: return $S^*$, $X^*$.

However, semi-greedy construction may not always generate a feasible solution. For instance, the total travel time exceeds the UAV’s maximum flight time in the mission area. When this situation happens, a repair procedure should be applied in line 5 to amend $S$. If feasibility cannot be reached, we should discard it and generate a new solution.

Once obtaining a feasible solution, the routing and time allocation solutions can be solved in lines 7 to 10. In line 8, the tabu search is applied to improve the quality of solution $S$ during its local iterations. In line 9, the best time schedule solution can be generated by the smoothing Newton method. If the current total search benefits $h(X)$ is better than the best benefits $h^*$ in the previous iterations, then the route sequences $S$ and time allocations $X$ are recorded as $S^*$ and $X^*$ in lines 12 and 13. The value $h(X)$ is recorded as $h^*$ in line 14. The best sequences and allocations found over all iterations are returned in line 17 as the whole procedure’s solution.

2) Semi-greedy construction: In a greedy construction, solutions are progressively and greedily built from scratch. A new element is added into its partial solution under construction at each iteration until there is no element in the candidate set. The selection of the next element $i^*$ follows

$$i^* \leftarrow \arg\min \{c_i : i \in FCS\}$$

where $FCS$ is the feasible candidates set and $c_i$ is the increment of the cost associated with the element $i \in FCS$ to be incorporated into the partial solution. In the greedy construction, all candidates’ increments are constant, so it cannot build different solutions in different runs.

To date, existing semi-greedy algorithms have been verified on the Steiner tree problem [10], maximum satisfiability problem [8], and Traveling Salesman Problem (TSP) [11], but little work is applied to the MTSP. In this paper, the semi-greedy construction is modified and developed for multiple UAVs’ path planning problem. The pseudo-code for the semi-greedy construction is shown in Algorithm 2. The present algorithm distinguishes from other algorithms mainly in the following two aspects. First, unlike the semi-greedy in solving TSP, such as in [10], where the number of the latest visited nodes is only one, the UAVs’ path planning problem to be tackled in this work is subject to different latest visited regions of UAVs. Second, the
Algorithm 2: Semi-greedy Construction.

Input: number of UAVs \( M \), random threshold \( \eta \), travel time matrices of UAVs \( \{T^m : m \in \mathcal{U}\} \).

Output: travel sequences of UAVs \( S \).

Initialization:

1: Initialize the latest travel regions list: \( LTR \leftarrow \emptyset \);
2: Initialize the feasible candidate set: \( FCS \leftarrow \emptyset \);
3: for \( m = 1 : M \) do
4: Initialize travel sequence of UAV \( m \): \( S_m \leftarrow \emptyset \);
5: Add start point to the latest travel regions list: \( LTR \leftarrow LTR \cup \{(m, 0)\} \);
6: end for

Solution Construction:

7: while \( FCS \neq \emptyset \) do
8: \( c^{\text{min}} \leftarrow \min \{t_{ij}^m : (m, i) \in LTR, j \in FCS\} \);
9: \( c^{\text{max}} \leftarrow \max \{t_{ij}^m : (m, i) \in LTR, j \in FCS\} \);
10: \( RCL \leftarrow \{(m, i, j) : t_{ij}^m \leq c^{\text{min}} + \eta(c^{\text{max}} - c^{\text{min}}), (m, i) \in LTR, j \in FCS\} \);
11: Choose \( (m', i', j') \) at random from \( RCL \);
12: \( S_{m'} \leftarrow S_{m'} \cup \{(i', j')\} \);
13: \( FCS \leftarrow FCS \setminus \{(i', j')\} \);
14: \( LTR \leftarrow LTR \setminus \{(m', i', j')\} \);
15: end while
16: for \( (m, i) \) in \( LTR \) do
17: \( S_m \leftarrow S_m \cup \{(i, L + 1)\} \);
18: end for
19: Update the set of travel sequences: \( S \leftarrow \{S_1, \ldots, S_M\} \);
20: return \( S \).

Restricted Candidate List (\( RCL \)) in this paper is determined by both the Latest Travels Regions (\( LTR \)) and \( FCS \) (in line 10), compared to that is only determined by \( FCS \), e.g., in [8], [11], and [45].

Firstly, the initialization procedure is carried out from lines 1 to 6. The tuple \( (m, i) \) in \( LTR \) represents that the latest travel region of UAV \( m \) is region \( i \). Then, the solution construction is processed until the feasible candidate set \( FCS \) is empty. In lines 8 and 9, \( c^{\text{min}} \) and \( c^{\text{max}} \) stand for the minimum and maximum travel time increments of adding the regions from \( FCS \) to current sequences. In line 10, the tuple \( (m, i, j) \) represents that UAV \( m \) flies from region \( i \) to region \( j \). In this paper, \( RCL \) is associated with a threshold parameter \( \eta \in (0, 1) \). The case \( \eta = 0 \) corresponds to a fully greedy construction, while \( \eta = 1 \) is equivalent to a random construction. Those tuples whose travel time under the upper bound \( c^{\text{min}} + \eta(c^{\text{max}} - c^{\text{min}}) \) are incorporated into the \( RCL \). \( (m', i', j') \) is chosen randomly from \( RCL \) in line 11. Then the tuple \( (i', j') \) is added to the sequence \( S_{m'} \) in line 12. The \( FCS \) and \( LTR \) are updated in lines 13 and 14, respectively. The while loop ends until no element in \( FCS \).

An example of the greedy and semi-greedy constructions is showed in Fig. 3. Suppose we have 2 UAVs. Both of them have the same travel time matrix.

Firstly, we apply the greedy construction to this problem. From the start point, regions 1 and 4 are selected as the next regions because \( t_{01} = t_{04} = 3 \) are the two smallests in the first row of the matrix except the element \( t_{00} \). From region 1, the travel time to regions 2, 3, and 5 are 1, 6, and 3, respectively. From region 4, the travel time to regions 2, 3 and, 5 are 6, 2, and 8, respectively. Since \( t_{32} = 1 \) is the smallest, region 2 is selected as the next region for UAV 1. As \( t_{32} = 2 \) is smaller than traveling from region 4 to any unselected regions, region 3 is chosen as next for UAV 2. Consequently, both UAVs' full tours are \( 0 \rightarrow 1 \rightarrow 2 \rightarrow 5 \rightarrow 6 \) and \( 0 \rightarrow 4 \rightarrow 3 \rightarrow 6 \). The total travel time of the two UAVs is 37.

Now apply the semi-greedy construction to the same problem. Suppose the threshold \( \eta \) is 0.8. Initially, the \( FCS \) is \( \{(1, 0), (2, 0)\} \). Since the restricted candidate set is \( \{(1, 0, 1), (1, 0, 2), (1, 0, 3), (1, 0, 4), (2, 0, 1), (2, 0, 2), (2, 0, 3), (2, 0, 4)\} \), either element could be selected with the probability of 0.125. Suppose \( (1, 0, 2) \) is chosen. So region 2 is incorporated into the sequence of UAV 1. Thus, \( FCS \) is \( \{(1, 3, 4, 5) \) and \( LTR \) is \( \{(1, 3), (2, 0)\} \). \( t_{21}, t_{23}, t_{24}, t_{25} \) are, respectively, 1, 7, 6 and 8. \( t_{01}, t_{03}, t_{04}, t_{05} \) are, respectively, 4, 3, 8 and 5. So the \( c^{\text{min}} \) is 1 and \( c^{\text{max}} \) is 8. The \( RCL \) is \( \{(1, 2, 1), (1, 2, 4), (2, 0, 1), (2, 0, 3), (2, 0, 4)\} \). Suppose \( (2, 0, 4) \) is chosen. So region 4 is added to the tour of UAV 2. Consequently, the full tours are \( 0 \rightarrow 2 \rightarrow 1 \rightarrow 5 \rightarrow 6 \) and \( 0 \rightarrow 3 \rightarrow 4 \rightarrow 6 \). The total travel time of two UAVs is 27.

From above, we can see, the total travel time produced by semi-greedy construction is much smaller than that obtained by the pure greedy method. As considering randomization and selecting elements from \( RCL \) randomly, semi-greedy construction is more likely to find a higher-quality solution than the greedy algorithm.

3) Repair: In the GRASP algorithm, the semi-greedy may not always construct a feasible solution. At this time, we should use a repair procedure to amend it. The core of infeasible solutions is the imbalanced assignments between different UAVs. Specifically, some UAVs may have to visit most regions and have little time to search for information or even exceed the maximum flight time in the mission area. So the repair procedure is proposed to modify the infeasibility.

In the instance of Fig. 3, we wish to find two tours with balanced travel time. Suppose the two routes are \( S \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow E \) and \( S \rightarrow 5 \rightarrow E \).

The most time-consuming regions should be moved to another route when UAVs have unbalanced tasks. In the example of Fig. 3, UAV 1 consumes 7 units of time from region 2 to 3, more than other travel time in the route. So we consider moving region 3 to the route of UAV 2. The region 3 can be inserted into two positions which can produce two different routes \( S \rightarrow 3 \rightarrow 5 \rightarrow E \) and \( S \rightarrow 5 \rightarrow 3 \rightarrow E \). The total travel time of both are, respectively, 22 and 26. So we finally accept the first position.

4) Tabu search: After the semi-greedy construction and repair procedure, we can get a feasible solution for UAVs’ routes. Then we apply the tabu search [46] to optimize the traveling
order of every UAV. We use the so-called best improvement method even if this may increase the algorithm’s running time.

B. Smoothing Newton Method

According to the aim function and constraints in the proposed model, we can derive a time allocation model for UAVs

\[
\max_{x \in X_m} \sum_{i=1}^{n} a_i \left(1 - \exp(-b_i, x_i)\right)
\]

(13)

where \(X_m = \{x \in \mathbb{R}^n : 1^T x = c_m\}, a, b \in \mathbb{R}^+\), \(c > 0\), and \(1\) is the vector of ones. Specifically, \(n\) is the number of regions in \(S_m\) and we define \(N := \{1, 2, \ldots, n\}\). The total information collection time is \(c_m\), which equals the maximum flight time \(t_{max}\) minus its total travel time between regions. Besides, \(a_i\) equals the product of information existence probability \(e_i\) and information value \(\lambda_i\). Moreover, \(-b_i\) is the information searching difficulty of region \(i\).

1) Preliminaries: We first briefly recall some definitions and results.

**Definition 1.** Let \(F\) be a mapping from \(\mathbb{R}^n\) into itself. The *nonlinear complementarity problem*, denoted by NCP(F), is to find a vector \(x^*\) such that

\[
x^* \in \mathbb{R}^n, F(x^*) \in \mathbb{R}^n \text{ and } F(x^*)^T x^* = 0,
\]

(14)

where \(F(x)\) is a nonlinear and continuously differentiable function of \(x\).

**Definition 2.** A function \(\varphi : \mathbb{R}^2 \rightarrow \mathbb{R}\) is called an NCP function if

\[
\varphi(x, y) = 0 \iff x \geq 0, y \geq 0, xy = 0.
\]

(15)

The two probably most well-known NCP functions are the minimum function \([47]\)

\[
\varphi(x, y) = \min \{x, y\},
\]

and the Fischer-Burmeister function \([48]\)

\[
\varphi(x, y) = \sqrt{x^2 + y^2} - a - b.
\]

**Definition 3.** Let \(G : \mathbb{R}^n \rightarrow \mathbb{R}^n\) be locally lipschitzian at \(x \in \mathbb{R}^n\). \(G\) is *semismooth* at \(x\) if

\[
\lim_{g' \rightarrow 0} \frac{V g'}{V \in \partial G(x + tg')},
\]

(16)

exists for all \(g \in \mathbb{R}^n\).

**Definition 4.** Function \(G_\mu(x) : \mathbb{R}^n \rightarrow \mathbb{R}^n (\mu > 0)\) is called a *smoothing function* of a semismooth function \(G(x)\) if it is continuously differentiable everywhere and there is a constant \(\mu\) independent of \(\mu\) such that

\[
\|G_\mu(x) - G(x)\| \leq \mu, \quad \forall x.
\]

(17)

2) Algorithm description: In this paper, we use the following one-parametric NCP function:

\[
\varphi_\xi(x, y) := \sqrt{(x - y)^2 + \xi xy} - x - y,
\]

(18)

where \(\xi \in (0, 4)\).

From \([49]\), we know that the \(\varphi_\xi\) is not only an NCP function but also is semismooth. Clearly, in the case of \(\xi = 2\), the function \(\varphi_\xi\) reduces to the Fischer-Burmeister function, and in the limiting case \(\xi \rightarrow 0\), the function \(\varphi_\xi\) becomes a variation of the minimum function.

Based on the \(\varphi_\xi\), we study the perturbed parametric NCP function, denoted by PPNCP-function:

\[
\varphi_\xi(\mu, x, y) := \sqrt{\mu^2 + (x - y)^2 + \xi xy} - x - y, \quad \xi \in (0, 4).
\]

(19)

It is obvious that for any \(\mu > 0\) and \(\xi \in (0, 4)\), \(\varphi_\xi\) is differentiable everywhere. For all \(\mu \geq 0\) and \(\xi \in (0, 4)\), we have

\[
|\varphi_\xi(\mu, x, y) - \varphi_\xi(x, y)| \leq \mu, \quad \forall (x, y) \in \mathbb{R}^2.
\]

(20)

When \(\mu = 0\), we have \(\varphi_\xi(0, x, y) = \varphi_\xi(x, y)\) for all \((x, y) \in \mathbb{R}^2, \xi \in (0, 4)\). Hence, the PPNCP-function in Eq. (19) is a smoothing function of \(\varphi_\xi\) in Eq. (18).

According to the Karush-Kuhn-Tucker theorem, the model in Eq. (13) can be transformed to

\[
1^T x = c_m,
\]

\[
x_i \geq 0, \quad y_i \geq 0, \quad \forall i \in N,
\]

\[
x_i y_i = 0, \quad \forall i \in N,
\]

(21)

where \(y_i := s - a_i b_i \exp(-b_i, x_i), i \in N, s \in \mathbb{R}\). Now, we define the equation operator \(\Psi_\xi : \mathbb{R}^{n+1} \rightarrow \mathbb{R}^{n+1}\) by

\[
\Psi_\xi(s, x) := \begin{pmatrix}
\varphi_\xi(x_1, s - a_1 b_1 \exp(-b_1 x_1)) \\
\vdots \\
\varphi_\xi(x_n, s - a_n b_n \exp(-b_n x_n))
\end{pmatrix}.
\]

(22)

If we define the semismooth function \(H_\xi : \mathbb{R}^{n+1} \rightarrow \mathbb{R}^{n+1}\) by

\[
H_\xi(s, x) := \begin{pmatrix}
1^T x - c_m \\
\Psi_\xi(s, x)
\end{pmatrix},
\]

(23)

then we may rewrite the problem in Eq. (21) as the following semismooth equation:

\[
H_\xi(s, x) = 0.
\]

(24)

Based on the PPNCP-function, we obtain the following smooth equation:

\[
G_\xi(z) := G_\xi(\mu, s, x) := \begin{pmatrix}
1^T x - c_m \\
\Psi_\xi(\mu, s, x)
\end{pmatrix} = 0,
\]

(25)

where

\[
\Psi_\xi(\mu, s, x) := \begin{pmatrix}
\varphi_\xi(\mu, x_1, s - a_1 b_1 \exp(-b_1 x_1)) \\
\vdots \\
\varphi_\xi(\mu, x_n, s - a_n b_n \exp(-b_n x_n))
\end{pmatrix}.
\]

(26)

By the definition of the NCP-function, we know that the problem in Eq. (21) is equivalent to the Eq. (25) in the sense that their solutions are coincident.
Algorithm 3: Smoothing Newton Method.

Input: the number of regions in the sequence n, the smooth function $G_\xi$.

Output: the suboptimal time allocation solution $x^k$.

Initialization:
1: Choose $\xi \in (0, 4)$, $\delta, \sigma \in (0, 1)$;
2: Initialize $\mathcal{U} = (\mu^0, 0, 0) \in \mathbb{R}_{++} \times \mathbb{R} \times \mathbb{R}^n$, $s^0 \in \mathbb{R}$ and $x^0 \in \mathbb{R}^n$;
3: $z^0 \leftarrow (\mu^0, s^0, x^0)$;
4: Choose $\gamma \in (0, 1)$ such that $\gamma \|G_\xi(z^0)\| < 1$ and $\gamma \mu^0 < 1$;
5: $k \leftarrow 0$.

Computation and Updation:
6: while $G_\xi(x^k) > 0$ do
7: \[ \rho_k \leftarrow \rho(z^k); \]
8: Compute $\Delta x^k \in \mathbb{R}^{n+2}$ by
\[ G_\xi(z^k) + G_\xi'(z^k) \Delta x^k = \rho_k \mathcal{U}; \]
9: \[ (\Delta \mu^k, \Delta s^k, \Delta x^k) \leftarrow \Delta x^k; \]
10: Let $k_0$ be the smallest nonnegative integer such that $\Theta_\xi(z^k + \delta^{k_0} \Delta x^k) \leq [1 - \sigma (1 - \gamma \mu^0) \delta^{k_0}] \Theta_\xi(z^k)$;
11: $\tau_k \leftarrow \delta^{k_0}$;
12: $z^{k+1} \leftarrow z^k + \tau_k \Delta x^k$;
13: $k \leftarrow k + 1$.
14: end while
15: return $x^k$.

Now, we give some properties of the function $G_\xi$, which will be used in the following analysis. The proofs of them are given in Appendix.

Property 1: Let $G_\xi$ be defined by Eq. (25). Then $G_\xi$ is semismooth on $\mathbb{R}^{n+2}$ and continuously differentiable at any $z = (\mu, s, x) \in \mathbb{R}_{++} \times \mathbb{R}^{n+1}$ with its Jacobian
\[ G_\xi'(z) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1^T \\ \frac{\partial \Psi_{\xi}}{\partial s} & \frac{\partial \psi_{\xi}}{\partial s} & \frac{\partial \psi_{\xi}}{\partial x} \end{pmatrix} \]
where
\[ \frac{\partial \Psi_{\xi}}{\partial \mu} := \text{vec} \left\{ \frac{\mu}{\omega_i} : i \in N \right\}, \]
\[ \frac{\partial \psi_{\xi}}{\partial s} := \text{vec} \left\{ \frac{2y_i + (\xi - 2)x_i}{2\omega_i} - 1 : i \in N \right\}, \]
\[ \frac{\partial \psi_{\xi}}{\partial x} := \text{diag} \left\{ \frac{2y_i + (\xi - 2)x_i}{2\omega_i} - 1 + \theta_i \left( \frac{\partial \psi_{\xi}}{\partial s} \right)_i : i \in N \right\}, \]
with
\[ y_i := s - a_i b_i \exp(-b_i x_i), i \in N, \]
\[ \omega_i := \sqrt{(x_i - y_i)^2 + x_i y_i + \mu^2}, i \in N, \]
\[ \theta_i := a_i b_i^2 \exp(-b_i x_i), i \in N. \]

Property 2: Let $G_\xi'(z)$ be defined by Eq. (27). Then the matrix $G_\xi'(z)$ is nonsingular on $\mathbb{R}_{++} \times \mathbb{R}^{n+1}$.

We now propose a smoothing Newton method for solving Eq. (25). It is a modified version of the Newton-type method proposed in [42]. The difference is that we use the PPNCP-function instead of taking the classical Fischer-Burmeister function.

Let $z = (\mu, s, x) \in \mathbb{R}^{n+2}, \gamma \in (0, 1)$ and define $\Theta : \mathbb{R}^{n+2} \rightarrow \mathbb{R}_+$ by
\[ \Theta_\xi(z) := \|G_\xi(z)\|^2. \]

Let $\rho : \mathbb{R}^{n+2} \rightarrow \mathbb{R}_+$ be
\[ \rho(z) := \gamma \min \{1, \Theta_\xi(z)\}. \]

Algorithm 3 shows the precise statement of our smoothing Newton method. Following the proof of Theorem 6 in [42], we can obtain that the sequence $\{z^k\}$ generated by Algorithm 3 is bounded and its solution is optimal to the problem described in 13.

3) Convergence analysis: Now we analyze the rate of convergence for Algorithm 3. Firstly, we give a lemma and complete its proof.

Lemma 1: Suppose that $z^* = (\mu^*, s^*, x^*)$ is an accumulation point of its iteration $z^k$ generated by Algorithm 3. Let Q be a matrix in $\partial H_\xi(s^*, x^*)$. Then the matrix $Q$ is nonsingular.

Proof: Define the index sets:
\[ N_1 := \left\{ i \in N : (x_i^*)^2 + (s^* - a_i b_i) \exp(-b_i x_i^*)^2 \neq 0 \right\}, \]
\[ N_2 := \left\{ j \in N : x_j^* = 0, s^* = a_j b_j \right\}. \]

According to the definition of function $H_\xi$ in Eq. (23), we get
\[ Q = \begin{pmatrix} 0 & 1^T \\ \alpha & A \\ \beta & B \end{pmatrix} \]
where $A$ is a matrix of $|N_1| \times n$ dimension. All elements in $A$ are zero except for the $(i, j)$th as $A_{ij}$ for $i \in N_1, B$ is a matrix of $|N_2| \times n$ dimension. All elements in $B$ are zero except for the $(j, j)$th as $B_{jj}$ for $j \in N_2$. $\alpha, \beta, A_i$ and $B_j$ are shown below:
\[ \alpha := \text{vec} \left\{ \frac{2y_i + (\xi - 2)x_i}{2\omega_i} - 1 : i \in N_1 \right\}, \]
\[ A_i := \frac{2x_i + (\xi - 2)y_i}{2\omega_i} - 1 + a_i b_i^2 \exp(-b_i x_i) \alpha_i, i \in N_1, \]
\[ (\beta_j, B_j) \in \left\{ (p + 1, q + 1, a_j b_j^2 p) : \frac{(p + q)^2}{\xi} + \frac{(p - q)^2}{4 - \xi} \leq 1 \right\}, j \in N_2, \]
where
\[ y_i^* := s^* - a_i b_i \exp(-b_i x_i^*), i \in N_1, \]
\[ \omega_i := \sqrt{(x_i^* - y_i^*)^2 + x_i^* y_i^*}, i \in N_1. \]

From Eq. (57), we have
\[ \alpha_i < 0, A_i < 0, i \in N_1, \]
\[ \beta_j \leq 0, B_j < 0, j \in N_2. \]
Let $Q(r, w) = 0$, Then we get
\[
\sum_{i \in N_1} w_i + \sum_{i \in N_2} w_i = 0, \\
\alpha_i r_i + A_i w_i = 0, \quad i \in N_1, \\
\beta_j r_j + B_j w_j = 0, \quad j \in N_2.
\]
Thus,
\[
w_i = -\frac{\alpha_i}{A_i} r_i, \quad i \in N_1, \quad w_j = -\frac{\beta_j}{B_j} r_j, \quad j \in N_2.
\]
Therefore, by Eqs. (39), (40) and (41), we have $r = 0$ and $w = 0$, so the matrix $Q$ is nonsingular.

**Theorem 1:** Suppose that $\{z^k\}$ is the iteration solutions generated by Algorithm 3. Then $\{z^k\}$ converges to $z^*$ superlinearly, i.e., $\|z^{k+1} - z^*\| = o(\|z^k - z^*\|)$.

**Proof:** According to Theorem 6 in [42], the sequence $\{z^k = (\mu^k, s^k, x^k)\}$ generated by Algorithm 3 is bounded. Let $z^* = (\mu^*, s^*, x^*)$ represent an accumulation point of $\{z^k\} = (\mu^k, s^k, x^k)$. Then $\mu^*$ is, and $x^*$ is the optimal solution.

From Proposition 3.1 in [50], for all $z^k$ sufficiently close to $z^*$,
\[
\|G_\xi(z^k)^{-1}\| = O(1).
\]
From Property 1, we know that $G_\xi$ is smootismooth. Based on Lemma 2.1 in [51], for all $z^k$ sufficiently close to $z^*$, we have
\[
\|G_\xi(z^k) - G_\xi(z^*) - G_\xi'(z^*)(z^k - z^*)\| = o(\|z^k - z^*\|).
\]
From Eqs. (42) and (43), for all $z^k$ sufficiently close to $z^*$, we have
\[
\|z^k + \Delta z^k - z^*\| = \|z^k + G_\xi(z^k)^{-1}(-G_\xi(z^k) + \rho k \bar{u}) - z^*\|
\]
\[
= O(\|G_\xi(z^k) - G_\xi(z^*) - G_\xi'(z^*)(z^k - z^*)\| + \rho k \|\bar{u}\|)
\]
\[
= o(\|z^k - z^*\|) + O(\Theta_\xi(z^*)
\]
\[
= o(\|z^k - z^*\|).
\]
Because $G_\xi$ is semismooth at $z^*$, $G_\xi$ is locally Lipschitz continuous near $z^*$, for all $z^k$ sufficiently close to $z^*$,
\[
\Theta_\xi(z^*) = \|G_\xi(z^*)\|^2 = O(\|z^k - z^*\|^2).
\]
Therefore, from Eqs. (44) and (45), for all $z^k$ sufficiently close to $z^*$,
\[
\|z^k + \Delta z^k - z^*\| = o(\|z^k - z^*\|).
\]
Following the proof of Theorem 3.1 in [52], for all $z^k$ sufficiently close to $z^*$, we have
\[
\|z^k - z^*\| = O(\|G_\xi(z^*) - G_\xi(z^*)\|).
\]
Hence, for all $z^k$ sufficiently close to $z^*$, we have
\[
\Theta_\xi(z^k + \Delta z^k) = \|G_\xi(z^k) - G_\xi(z^*)\|^2
\]
\[
= o(\|z^k + \Delta z^k - z^*\|)
\]
\[
= o(\|z^k - z^*\|)
\]
Therefore, for all $z^k$ sufficiently close to $z^*$ we have
\[
z^{k+1} = z^k + \Delta z^k
\]
which, together with Eq. (46), proves $\|z^{k+1} - z^*\| = o(\|z^k - z^*\|)$.

### C. Complexity Analysis

In Algorithm 2, the number of tuple $(m, i, j), m \in U, i, j \in A$ is $ML(L - 1)/2$, so the time complexity of finding the maximum and minimum is upper bounded by $O(ML^2/2)$. As we computer the while loop $L$ times, so Algorithm 2 has a time complexity of $O(ML^3/2)$.

In Algorithm 3, to compute the Newton step $\Delta z^k \in \mathbb{R}^{n+2}$, we first form and evaluate the smooth equation $G_\xi(z^*)$ and its Jacobian $G_\xi'(z^*)$. Then we solve the system of linear equations $G_\xi'(z^*) + G_\xi'(z^*)\Delta z^k = \rho k \bar{u}$. While a general linear equation solver can be used, it is better to use $LU$ factorization that takes advantage of the nonsingularity in Property 2. So the total time complexity in the worst case is $O((2/3)L^3)$.

In Algorithm 1, the time complexity of the GENERAL can be estimated as
\[
T(I_w, M, L) = O(I_w \cdot [T_S + T_R + M \cdot (T_T + T_N)]),
\]
where $T_S, T_R, T_T, T_N$ denote the time complexity of semi-greedy construction, repair procedure, tabu search, and smoothing Newton method, respectively, while $I_w$ is the number of iterations in the internal loop of Algorithm 1. Through the analysis of repair procedure, we can get its time complexity $T_R$ of $O(ML)$. Moreover, $T_T$ increases quadratically with $L$. Hence, the time complexity of GENERAL can be expressed as
\[
T(I_w, M, L) = O \left( \frac{7}{6} I_w M L^3 \right),
\]
which obeys a polynomial order.

Similarly, the storage complexity of semi-greedy construction and smoothing Newton method are $O(ML^2/2)$ and $O(L^2)$, respectively. The storage complexity of GENERAL can be expressed as
\[
S(M, L) = O(S_S + S_R + S_T + S_N),
\]
where the $S_S, S_R, S_T, S_N$ denote the storage complexity of semi-greedy construction, repair procedure, tabu search, and smoothing Newton method, respectively. The storage complexity of repair procedure is upper bounded by $O(L)$. Moreover, $S_T$ increases quadratically with $L$.

Therefore, the storage complexity of GENERAL can be estimated as
\[
S(M, L) = O \left( \left( 2 + \frac{M}{2} \right) L^2 \right).
\]
V. EXPERIMENTS AND RESULTS

In this section, experiments based on randomly generated instances are carried out to explore the performance of GENERAL algorithm, comparing with ACO-SA and GA. Besides, we compare the numerical performance of the smoothing Newton method and the algorithm in [42] on three randomly generated problems. Tests are performed on a personal computer with i5 CPU and 16 G RAM under the macOS Big Sur system. All numerical experiments and simulations are developed by using Python 3.7.9.

A. Parameters on Random Instances

1) Parameter setup and experiment design: A rectangular area of size \( W_1 \times W_2 \) has a start point and an end point located at the middle of the left and right edges. \( M \) UAVs are assigned to search \( L \) regions, which are uniformly distributed throughout the rectangular area. The maximum mission time of all UAVs is \( T \) and the travel time between two regions is calculated by Euclidean distance. Parameters associated with regions are generated as follows:

- \( \epsilon_i \), the prior probability of information in region \( i \), is generated from a sub-interval \([\epsilon_i L, \epsilon_i U]\) according to normal distribution with mean \( \theta \) and variance \( \zeta \).
- \( \lambda_i \), the information value in region \( i \), is an integer generated in interval \([1, 10]\).
- \( -b_i \), is the difficulty of searching for information in region \( i \). According to the Koopman search theory in [37], \( b_i \) is uniformly generated from \([b_i L, b_i U] \subset (0, 1)\).

We investigated the performance of the GENERAL for a given group instances, using the following parameters: \( W_1 \times W_2 = 40 \times 50 \), \([\epsilon_i L, \epsilon_i U] = [0.3, 0.8]\), \( \theta = 0.5 \), \( \zeta = 0.7 \), \([b_i L, b_i U] = [0.05, 0.025]\). Three information search scenarios were designed. In the small scenario, 2 or 3 UAVs were assigned to search 20 regions. In the middle scenario, we had 30 regions and the number of UAVs is 2 or 3. Moreover, we had to schedule 3 or 5 UAVs to search 40 regions in the large scenario.

In the GENERAL algorithm, the iteration is fixed to 100 as the stopping criterion. The parameters used in smoothing Newton method were \( \mu^0 = 0.001 \), \( \delta = 0.8 \), \( \sigma = 0.2 \) and \( \gamma = \min\{1/||G(y)||, 0.9\} \).

We compared the GENERAL algorithm with the ACO-SA and GA. We chose ACO and GA because they are often used in the first phase of the two-stage method to solve path planning problem. In addition, the SA and GA had been improved for solving nonlinear programming and widely used in many fields, such as [32] and [53]. The parameters of these algorithms are set up as follows. In the ACO algorithm, the information volatileization factor is 0.03, the information heuristic factor is 1, the expected heuristic factor is 3, and the number of ants is 50. In the SA, the maximum number of iterations is 1000, the initial temperature is 1, the end temperature is 0.0001, and the cooling rate is 0.9. In the GA, the number of population is 100, and the number of iterations is 1000. The crossover probability is 0.6, and the mutation probability is 0.001.

2) Experiment results: The results of three scenarios under different UAVs number \( (M) \) and mission time \( (T) \) are presented in Table II. For each algorithm, the \( \min \), \( \max \), and \( \text{avg} \) provide the minimum, maximum, and average values of information searching benefits among ten runs, respectively. The significant differences among various algorithms are shown in the last two columns. According to the p-values of GEN-AS and GEN-GA, which are less than 0.01, it can be claimed that GENERAL achieves significantly better performance than ACO-SA and GA.

We can see that GENERAL has a significant advantage in terms of information benefits. In the small scenario, the increase of average information searching benefits between solutions of GENERAL and those of ACO-SA is from 0.08 to 1.28. All the minimum information benefits solved by GENERAL are much better than the average benefits produced by GA. Specifically, in the cases of \((20, 2, 200)\) and \((20, 2, 300)\), the minimum benefits of GENERAL are much larger than the maximum benefits of ACO-SA and GA.

It can be observed that the adoption of GENERAL can also increase the information benefits in the middle scenario. The most significant gap of average information searching benefits between GENERAL and GA is around 10. Furthermore, the performance of GENERAL is more stable with larger instances. For example, in the cases of \((40, 5, 150)\) and \((40, 5, 200)\), the minimum values of GENERAL are much larger than the maximum values produced by ACO-SA and GA.

Although using the Newton-type algorithm to optimize the time allocation in every GENERAL iteration seems time-consuming, it can obtain more promising solutions despite accepting longer flying distances. For example, as illustrated in Fig. 4, the total tour time of 3 UAVs in Fig. 4(a) and Fig. 4(b)
TABLE II
EXPERIMENTS RESULTS OF GENERAL, ACO-SA [9], [39], AND GA [31]. THE P-VALUES OF GEN-AS AND GEN-GA DENOTE THE SIGNIFICANT DIFFERENCES AMONG GENERAL, ACO-SA, AND GA, RESPECTIVELY. THE OUTSTANDING RESULTS ARE MARKED BY BOLD FONT

| (L, M, T) | ACO-SA [9], [39] | GA [31] | GENERAL | P-Value |
|-----------|------------------|---------|---------|---------|
|           | min   | max   | avg   | min   | max   | avg   | GEN-AS | GEN-GA |
| (20, 2, 150) | 35.05 | 36.03 | 35.25 | 31.46 | 36.79 | 34.73 | 36.03 | 36.92 | 36.53 | 5E-05 | 3E-05 |
| (20, 2, 200) | 43.86 | 44.04 | 43.97 | 41.56 | 44.40 | 42.94 | 44.62 | 44.79 | 44.68 | 1E-04 | 5E-06 |
| (20, 2, 300) | 50.09 | 50.35 | 50.18 | 49.26 | 50.60 | 49.95 | 50.70 | 50.72 | 50.70 | 2E-07 | 6E-07 |
| (20, 3, 150) | 44.02 | 46.38 | 45.33 | 44.64 | 46.42 | 45.66 | 45.71 | 46.43 | 46.31 | 2E-03 | 2E-03 |
| (20, 3, 300) | 49.66 | 50.39 | 49.99 | 49.23 | 50.46 | 50.21 | 50.30 | 50.46 | 50.43 | 4E-04 | 5E-04 |
| (20, 3, 300) | 52.15 | 52.23 | 52.19 | 52.12 | 52.29 | 52.22 | 52.25 | 52.29 | 52.27 | 2E-07 | 7E-03 |
| (30, 2, 150) | 18.43 | 41.75 | 32.87 | 21.03 | 37.44 | 30.73 | 37.92 | 42.94 | 40.65 | 9E-03 | 3E-05 |
| (30, 2, 200) | 50.37 | 58.17 | 53.85 | 54.16 | 58.34 | 55.93 | 57.98 | 60.44 | 59.62 | 5E-06 | 5E-06 |
| (30, 2, 300) | 72.31 | 75.33 | 73.33 | 70.30 | 75.55 | 73.95 | 75.73 | 77.05 | 76.39 | 2E-08 | 8E-04 |
| (30, 3, 150) | 56.30 | 63.28 | 59.10 | 43.91 | 60.26 | 55.89 | 58.50 | 63.29 | 61.40 | 8E-05 | 5E-06 |
| (30, 3, 200) | 70.87 | 75.43 | 73.57 | 68.61 | 74.57 | 71.55 | 71.42 | 77.06 | 75.32 | 2E-07 | 1E-08 |
| (30, 3, 300) | 83.28 | 85.83 | 84.49 | 81.72 | 85.66 | 84.06 | 84.00 | 85.80 | 85.43 | 2E-03 | 5E-04 |
| (40, 3, 150) | 52.99 | 55.63 | 54.07 | 45.40 | 47.25 | 43.94 | 51.81 | 56.27 | 54.51 | 4E-04 | 2E-08 |
| (40, 3, 200) | 72.16 | 72.83 | 72.62 | 64.40 | 69.62 | 67.06 | 72.36 | 73.50 | 72.86 | 4E-04 | 7E-06 |
| (40, 3, 300) | 88.27 | 88.53 | 88.45 | 83.83 | 87.43 | 86.20 | 85.76 | 88.74 | 87.71 | 4E-03 | 4E-04 |
| (40, 5, 150) | 72.51 | 76.70 | 73.93 | 67.73 | 73.18 | 71.05 | 77.68 | 78.51 | 78.33 | 2E-06 | 3E-05 |
| (40, 5, 200) | 74.37 | 88.77 | 86.66 | 86.35 | 86.63 | 86.47 | 88.77 | 89.27 | 89.09 | 2E-05 | 1E-06 |
| (40, 5, 300) | 94.86 | 96.33 | 95.93 | 82.00 | 96.35 | 94.95 | 96.27 | 96.37 | 96.34 | 7E-04 | 7E-07 |

TABLE III
SENSITIVITY ANALYSIS OF THE SEMI-GREEDY THRESHOLD η

| η  | 0  | 0.2 | 0.4 | 0.6 | 0.8 | 1.0 |
|----|----|-----|-----|-----|-----|-----|
| Searching benefits | 55.84 | 55.88 | 55.97 | 55.76 | 55.21 | 55.27 |
| Flight time | 222 | 220 | 208 | 229 | 257 | 263 |

is 241.7 and 245.6, respectively. Nevertheless, the total information benefit of the former is larger than that of the latter. So if we produce the shortest path and then solve the best time allocation, we would not seek the more promising solution like that in Fig. 4(a).

3) Sensitivity analysis: The sensitivity analysis of the semi-greedy threshold η is shown in Table III. It can be observed that when the parameter turns from 0 to 1, the changes of both the total searching benefits and total flight time of UAVs are relatively small. It also can be seen that when the parameter value is around 0.4, the maximum information benefit can be obtained.

B. Experiments of Smoothing Newton Method

1) Experiments design: First, we devised a dynamic smoothing Newton method where the parameter ξ may change during iterations. The ξ was updated at each iteration using the following rules:

- Set ξ := 2 at the beginning of each iteration.
- If ΘΣ(zk) ≥ ε1, then set ξ := min{κ1 ΘΣ(zk), ξ}, else if ε1 ≥ ΘΣ(zk) ≥ ε2, set ξ := ΘΣ(zk).
- If ΘΣ(zk) ≤ ε2, then set ξ := min{κ2, ξ}.

where

ε1 = 10⁻², ε2 = 10⁻⁴, κ1 = 10, and κ2 = 10⁻⁶.

Second, we compared the performance of smoothing Newton, dynamic smoothing Newton, and Newton-type method on three randomly generated problems with 5, 50, and 200 regions, respectively. The parameters used in them were μ0 = 0.001, δ = 0.8, σ = 0.2, and γ = min{1/∥G(z)∥, 0.9}.

2) Experiments results: Each problem runs 50 times, and their numerical results are summarized in Table IV and Fig. 5. Here, NTM denotes the Newton-type method proposed in [42].
Then a smoothing Newton method is presented to solve the corresponding equations.

The GENERAL method is evaluated on random instances. Experimental results empirically indicate the effectiveness of the GENERAL method compared to the ACO-SA and GA. The smoothing Newton method’s numerical results indicate that the Fischer-Burmeister function gives the shortest running time with a small number of regions. The dynamic smoothing Newton method runs significantly faster than the other two algorithms when the number of regions is huge.

### Appendix A

Here we list the proofs of the properties in Section 4.

**Property 1:** Let \( G_\xi \) be defined by Eq. (25). Then \( G_\xi \) is semismooth on \( \mathbb{R}^{n+2} \) and continuously differentiable at any \( \mathbf{z} = (\mu, s, \mathbf{x}) \in \mathbb{R}_{++} \times \mathbb{R}^{n+1} \) with its Jacobian

\[
G'_\xi(\mathbf{z}) = \begin{pmatrix}
0 & 0 & 0 \\
0 & 0 & 1^T \\
\partial \Psi_\xi / \partial \mu & \partial \Psi_\xi / \partial s & \partial \Psi_\xi / \partial \mathbf{x}
\end{pmatrix}
\]

where

\[
\frac{\partial \Psi_\xi}{\partial \mu} := \text{vec} \left\{ \frac{\mu}{\omega_i} : i \in N \right\},
\]

\[
\frac{\partial \Psi_\xi}{\partial s} := \text{vec} \left\{ \frac{2y_i + (\xi - 2)x_i}{2\omega_i} - 1 : i \in N \right\},
\]

\[
\frac{\partial \Psi_\xi}{\partial \mathbf{x}} := \text{diag} \left\{ \frac{2x_i + (\xi - 2)y_i}{2\omega_i} - 1 + \theta_i \left( \frac{\partial \Psi_\xi}{\partial s} \right)_i : i \in N \right\},
\]

with

\[
y_i := s - a_i b_i \exp(-b_i x_i), \quad i \in N
\]

\[
\omega_i := \sqrt{(x_i - y_i)^2 + \xi x_i y_i + \mu^2}, \quad i \in N
\]

\[
\theta_i := a_i b_i^2 \exp(-b_i x_i), \quad i \in N.
\]

**Proof:** Since the \( \varphi_\xi(0, x, y) \) is strong semismooth on \( \mathbb{R}^2 \), \( G_\xi \) is semismooth on \( \mathbb{R}^{n+2} \). It is obvious that \( \Psi_\xi \) is continuously differentiable at any \( \mathbf{z} = (\mu, s, \mathbf{x}) \in \mathbb{R}_{++} \times \mathbb{R}^{n+1} \). So the \( G_\xi \) defined in Eq. (25) is continuously differentiable on \( \mathbb{R}_{++} \times \mathbb{R}^{n+1} \).

**Property 2:** Let \( G'_\xi(\mathbf{z}) \) be defined by Eq. (54). Then the matrix \( G'_\xi(\mathbf{z}) \) is nonsingular on \( \mathbb{R}_{++} \times \mathbb{R}^{n+1} \).

**Proof:** For any \( \mu > 0 \), the Jacobian matrix of \( G'_\xi \) can be obtained by a straightforward calculation from Eq. (25). We have for any \( \mu \in \mathbb{R}_{++} ,
\[
\left( \frac{\partial \Psi_\xi}{\partial s} \right)_i = \frac{2y_i + (\xi - 2)x_i}{2\omega_i} - 1
\]

\[
= \frac{\sqrt{4(x_i - y_i)^2 + 4\xi x_i y_i + \xi (\xi - 4)x_i^2}}{4(x_i - y_i)^2 + 4\xi x_i y_i + 4\mu^2} - 1.
\]

**VI. Conclusion**

A practically-motivated information searching model is developed, where routing and time allocation decisions are the two most critical decisions. From the perspective of practical setting, we develop an information searching model, where routing and time allocation decisions are the two most critical decisions. Then we propose a novel GENERAL to solve the optimal solution of the time allocation given paths of UAVs. During the GENERAL iterations, we devise a semi-greedy construction and repair procedure to initialize and improve solutions. Furthermore, we apply the PPNCP-function to reformulate the time allocation problem as a smoothing system of equations.
because $\xi \in (0, 4)$, $x_i \geq 0$ and $y_i \geq 0$, we have for any $\mu \in \mathbb{R}^{n+1}$,

$$\left( \frac{\partial \Psi_i}{\partial s} \right)_i < 0. \tag{58}$$

Similarly, we have for any $\mu \in \mathbb{R}^{n+1}$,

$$2x_i + (\xi - 2)y_i - 1 < 0. \tag{59}$$

On the other hand, we have

$$q_i := a_i b_i^2 \exp(-b_i x_i) > 0, \quad \forall i \in N. \tag{60}$$

Hence, by Eqs. (58), (59) and (60), we obtain for any $\mu \in \mathbb{R}^{n+1}$ that

$$\left( \frac{\partial \Psi_i}{\partial x} \right)_i < 0, \quad i \in N. \tag{61}$$

Let

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u \\ v \\ w \end{pmatrix} = 0. \tag{62}$$

Then, we have $u = 0$ and

$$\sum_{i=1}^{n} w_i = 0, \quad i \in N. \tag{63}$$

The second equality in Eq. (63) indicates

$$w_i = -\left( \frac{\partial \Psi_i}{\partial x} \right)_i v, \quad \forall i \in N. \tag{64}$$

Because $-\left( \frac{\partial \Psi_i}{\partial x} \right)_i v > 0$ and $\left( \frac{\partial \Psi_i}{\partial x} \right)_i v < 0$ for all $i \in N$, the equalities in Eq. (63) yields $v = 0$ and $w_i = 0$. Therefore, the matrix $G'_{ii}(z)$ defined by Eq. (54) is nonsingular on $\mathbb{R}^{n+1}$.

REFERENCES

[1] L. N. Mascarell and F. Quagliotti, “Challenges and safety aspects of a disaster relief exercise,” in Proc. Int. Conf. Unmanned Aircr. Syst., 2016, pp. 840–848.

[2] B. Wang, Y. Sun, N. Zhao, and G. Gui, “Learn to coloring: Fast response to perturbation in UAV-assisted disaster relief networks,” IEEE Trans. Veh. Technol., vol. 69, no. 3, pp. 3505–3509, Mar. 2020.

[3] K. Harikumar, J. Senthilnath, and S. Sundaram, “Multi-UAV oxyrrhis marina-inspired search and dynamic formation control for forest firefighting,” IEEE Trans. Automat. Sci. Eng., vol. 16, no. 2, pp. 863–873, Apr. 2019.

[4] X. Li, H. Yao, J. Wang, S. Wu, C. Jiang, and Y. Qian, “Rechargeable multi-UAV aided seamless coverage for QoS-guaranteed IoT networks,” IEEE Internet Things J., vol. 6, no. 6, pp. 10902–10914, Dec. 2019.

[5] X. Wang and L. Wang, “A literature survey of the robotic technologies during the COVID-19 pandemic,” J. Manuf. Syst., vol. 60, pp. 823–836, Aug. 2021.

[6] J. Maddliuk, “New shades of the vehicle routing problem: Emerging problem formulations and computational intelligence solution methods,” IEEE Trans. Emerg. Top. Comput. Intell., vol. 3, no. 3, pp. 230–244, Jun. 2019.

[7] Y. Bian, Y. Wang, Y. Yao, and H. Chen, “Ensemble pruning based on objective maximization with a general distributed framework,” IEEE Trans. Neural Netw. Learn. Syst., vol. 31, no. 9, pp. 3766–3774, Sep. 2020.

[8] M. Gendreau and J.-Y. Potvin, Handbook of Metaheuristics. Cham, Switzerland: Springer, 2019.

[9] H. Daryanavard and A. Harifi, “UAV path planning for data gathering of IoT nodes: Ant colony or simulated annealing optimization,” in Proc. 3rd Int. Conf. Internet Things Appl., 2019, pp. 1–4.

[10] M. C. Resende and C. C. Ribeiro, Optimization by GRASP. New York, NY, USA: Springer, 2016.

[11] R. Interian and C. C. Ribeiro, “A grasp heuristic using path-relinking and restarts for the steiner traveling salesman problem,” Int. Trans. Oper. Res., vol. 24, no. 6, pp. 1307–1323, 2017.

[12] Y. Yang, L. Liao, H. Yang, and L. Li, “An optimal control strategy for multi-UAVs target tracking and cooperative competition,” IEEE/CAA J. Automat. Sinica, to be published, doi: 10.1109/JAS.2020.1003012.

[13] X. Li, H. Yao, J. Wang, X. Xu, C. Jiang, and L. Hanzo, “A near-optimal UAV-aided radio coverage strategy for dense urban areas,” IEEE Trans. Veh. Technol., vol. 68, no. 9, pp. 9098–9109, Sep. 2019.

[14] J. Wang, C. Jiang, Z. Wei, C. Pan, H. Zhang, and Y. Ren, “Joint UAV hovering altitude and power control for space-air-ground IoT networks,” IEEE Internet Things J., vol. 6, no. 2, pp. 1741–1753, Apr. 2019.

[15] I. Moskal and D. Michael, “Adaptive unmanned aerial vehicle routing methods for tactical surveillance operations,” Ph.D. dissertation, State Univ. New York, Buffalo, NY, USA, 2016.

[16] H. Yuan, M. Zhou, Q. Liu, and A. Abusorrah, “Fine-grained resource provisioning and task scheduling for heterogeneous applications in distributed green clouds,” IEEE/CAA J. Automat. Sinica, vol. 7, no. 5, pp. 1380–1393, Sep. 2020.

[17] R. Cheng, T. Rodemann, M. Fischer, M. Olhofer, and J. Yin, “Evolutionary many-objective optimization of hybrid electric vehicle control: From general optimization to preference articulation,” IEEE Trans. Emerg. Top. Comput. Intell., vol. 1, no. 2, pp. 97–111, Apr. 2017.

[18] Z. H. Zhan et al., “Matrix-based evolutionary computation,” IEEE Trans. Emerg. Top. Comput. Intell., pp. 1–14, 2021, doi: 10.1109/TETCI.2020.3047410.

[19] X. Li, H. Yao, J. Wang, X. Xu, C. Jiang, and L. Hanzo, “A near-optimal UAV-aided radio coverage strategy for dense urban areas,” IEEE Trans. Veh. Technol., vol. 68, no. 9, pp. 9098–9109, Sep. 2019.

[20] R. Cheng, T. Rodemann, M. Fischer, M. Olhofer, and J. Yin, “Evolutionary many-objective optimization of hybrid electric vehicle control: From general optimization to preference articulation,” IEEE Trans. Emerg. Top. Comput. Intell., vol. 1, no. 2, pp. 97–111, Apr. 2017.

[21] E. Stump and N. Michael, “Multi-robot persistent surveillance planning with mixed-integer optimization,” in Proc. Int. Conf. Emerg. Topics Comput. Intell., 2017.

[22] H. Ergezer and K. Leblebicioglu, “Path planning for UAVs for maximum information collection,” IEEE Trans. Aerial Electron. Syst., vol. 49, no. 1, pp. 502–520, Jan. 2013.

[23] Y. H. Chen and C. M. Wu, “An improved algorithm for searching maze based on depth-first search,” in Proc. IEEE Int. Conf. Consum. Electron. - Taiwan, 2020, pp. 1–2.

[24] G. Lopes Andrade and D. Hoffmann Thomas, “An optimized breadth-first search algorithm for routing in optical access networks,” IEEE Latin Amer. Trans., vol. 17, no. 7, pp. 1088–1095, Jul. 2019.

[25] A. Mallek and M. Boudhar, “A branch-and-bound algorithm for the problem of scheduling with a conflict graph,” in Proc. Int. Conf. Decis. Aid Sci. Appl., 2020, pp. 778–782.

[26] Y.-S. Ong and A. Gupta, “Air5: Five pillars of artificial intelligence research,” IEEE Trans. Emerg. Top. Comput. Intell., vol. 3, no. 5, pp. 411–415, Oct. 2019.

[27] J. Yang, X. Wang, S. Baldi, S. Singh, and S. Fari, “A software-in-the-loop implementation of adaptive formation control for fixed-wing UAVs,” IEEE/CAA J. Automat. Sinica, vol. 6, no. 5, pp. 1230–1239, Sep. 2019.

[28] C. Lin, Y. Bi, G. Han, J. Yang, H. Zhao, and Z. Liu, “Scheduling for time-constrained big-file transfer over multiple paths in cloud computing,” IEEE Trans. Emerg. Top. Comput. Intell., vol. 2, no. 1, pp. 25–40, Feb. 2018.

[29] W. Zheng, H. Chen, and X. Yao, “Analysis of evolutionary algorithms on fitness function with time-linkage property,” IEEE Trans. Evol. Comput., vol. 25, no. 4, pp. 696–709, Aug. 2021.

[30] X. Guo, S. Liu, M. Zhou, and G. Tian, “Disassembly sequence optimization for large-scale products with multiresource constraints using scatter search and Petri nets,” IEEE Trans. Cybern., vol. 46, no. 11, pp. 2435–2446, Nov. 2016.

[31] M. Yousefikhoshbakht, “An effective genetic algorithm for solving the multiple traveling salesman problem,” J. Optim. Ind. Eng., vol. 8, pp. 73–79, Mar. 2011.

[32] B. Yuan, B. Li, H. Chen, and X. Yao, “A new evolutionary algorithm with structure mutation for the maximum balanced biclique problem,” IEEE Trans. Cybern., vol. 45, no. 5, pp. 1054–1067, May 2015.
[33] W. Du et al., “Network-based heterogeneous particle swarm optimization and its application in UAV communication coverage,” IEEE Trans. Emerg. Top. Comput. Intell., vol. 4, no. 3, pp. 312–323, Jun. 2020.

[34] L. Zhou et al., “Solving dynamic vehicle routing problem via evolutionary search with learning capability,” in Proc. IEEE Congr. Evol. Comput., 2017, pp. 890–896.

[35] A. Gupta, Y.-S. Ong, and L. Feng, “Insights on transfer optimization: Because experience is the best teacher,” IEEE Trans. Emerg. Top. Comput. Intell., vol. 2, no. 1, pp. 51–64, Feb. 2018.

[36] B. Yuan, H. Chen, and X. Yao, “Toward efficient design space exploration for fault-tolerant multiprocessor systems,” IEEE Trans. Evol. Comput., vol. 24, no. 1, pp. 157–169, Feb. 2020.

[37] B. Koopman, Search and Screening: General Principles and Historical Applications. New York, NY, USA: Pergamon Press, 1980.

[38] Y. Xia, R. Batta, and R. Nagi, “Routing a fleet of vehicles for decentralized reconnaissance with shared workload among regions with uncertain information,” Operations Res., vol. 65, no. 3, pp. 674–692, 2017.

[39] B. W. Wah and T. Wang, “Constrained simulated annealing with applications in nonlinear continuous constrained global optimization,” in Proc. Int. Conf. Tools Artif. Intell., 1999, pp. 381–388.

[40] P. Zhang, J. Li, T. Li, and H. Chen, “A new many-objective evolutionary algorithm based on determinantal point processes,” IEEE Trans. Evol. Comput., vol. 25, no. 2, pp. 334–345, Apr. 2021.

[41] J. Ding, C. Yang, Q. Xiao, T. Chai, and Y. Jin, “Dynamic evolutionary multiobjective optimization for raw ore allocation in mineral processing,” IEEE Trans. Emerg. Top. Comput. Intell., vol. 3, no. 1, pp. 36–48, Feb. 2019.

[42] L. Zhang, “A newton-type algorithm for solving problems of search theory,” Adv. Operations Res., vol. 2013, pp. 1–7, 2013.

[43] D. Liao-McPherson, M. Huang, and I. Kolmanovsky, “A regularized and smoothed fischer-burmeister method for quadratic programming with applications to model predictive control,” IEEE Trans. Autom. Control, vol. 64, no. 7, pp. 2937–2944, Jul. 2019.

[44] M. G. C. Resende and C. C. Ribeiro, GRASP: Greedy Randomized Adaptive Search Procedures. Boston, MA, USA: Springer, 2014, pp. 287–312. [Online]. Available: https://doi.org/10.1007/978-1-4614-6944-3_11

[45] M. Prais and C. C. Ribeiro, “Reactive grasp: An application to a matrix decomposition problem in TDMa traffic assignment,” INFORMS J. Comput., vol. 12, no. 3, pp. 164–176, 2000.

[46] R. Guo, W. Zhang, W. Guan, and B. Ran, “Time-dependent urban customized bus routing with path flexibility,” IEEE Trans. Intell. Transp. Syst., vol. 22, no. 4, pp. 2381–2390, Apr. 2021.

[47] J.-P. Dussault, M. Frappier, and J. C. Gilbert, “A lower bound on the iterative complexity of the harker and Pang globalization technique of the newton-min algorithm for solving the linear complementarity problem,” EURO J. Comput. Optim., vol. 7, pp. 359–380, Aug. 2019.

[48] A. Fischer, “A special newton-type optimization method,” Optimization, vol. 24, no. 3-4, pp. 269–284, 1992.

[49] C. Kanzow and H. Kleinmichel, “A new class of semismooth newton-type methods for nonlinear complementarity problems,” Comput. Optim. Appl., vol. 11, no. 3, pp. 227–251, 1998.

[50] L. Qi and J. Sun, “A nonsmooth version of newton’s method,” Math. Prog., vol. 58, pp. 353–367, Jan. 1993.

[51] L. Zhang, S.-Y. Wu, and T. Gao, “Improved smoothing newton methods for P0 nonlinear complementarity problems,” Appl. Math. Comput., vol. 215, no. 1, pp. 324–332, 2009.

[52] L. Qi, “Convergence analysis of some algorithms for solving nonsmooth equations,” Math. Operations Res., vol. 18, no. 1, pp. 227–244, 1993.

[53] L. Yu, H. Chen, S. Wang, and K. K. Lai, “Evolving least squares support vector machines for stock market trend mining,” IEEE Trans. Evol. Comput., vol. 13, no. 1, pp. 87–102, Feb. 2009.

Yamin Li received the B.S. degree in information and communication engineering from Information Engineering University, Zhengzhou, China, in 2018. He is currently pursuing the M.S. degree in management science and engineering with the National University of Defense Technology, Changsha, China. His research interests include data processing and computational intelligence.

Lihua Liu received the B.S. and Ph.D. degrees in information and communication engineering from the National University of Defense Technology (NUDT), Changsha, China, in 2004 and 2012, respectively. She is currently an Associate Professor with the Science and Technology on Information Systems Engineering Laboratory, NUDT. Her research interests include target recognition, data fusion, data engineering, and intelligent decision.

Jibing Wu received the B.S. and M.S. degrees in information system engineering, and the Ph.D. degree in management science and engineering from the National University of Defense Technology (NUDT), Changsha, China, in 2010, 2012, and 2017, respectively. He is currently an Associate Professor with the Science and Technology on Information Systems Engineering Laboratory, NUDT. His research interests include heterogeneous information network, information management, and computational intelligence.

Mao Wang received the B.S. degree in computer science and technology from the Beijing Institute of Technology, Beijing, China, in 2011, and the M.S. and Ph.D. degrees in computer science and technology from the National University of Defense Technology (NUDT), Changsha, China, in 2013 and 2018, respectively. He is currently a Lecturer with the Science and Technology on Information Systems Engineering Laboratory, NUDT. His research interests include image retrieval and evolutionary computation.

Haohao Zhou received the B.S. and M.S. degrees in information system engineering, and the Ph.D. degree in management science and engineering from the National University of Defense Technology (NUDT), Changsha, China, in 2009, 2011, and 2016, respectively. He is currently a Researcher with the Science and Technology on Information Systems Engineering Laboratory, NUDT. His research interests include integer programming, network scheduling, and information management.

Hongbin Huang received the B.S. and Ph.D. degrees in information system engineering from the National University of Defense Technology (NUDT), Changsha, China, in 1996 and 2007, respectively. He is currently a Professor with the Science and Technology on Information Systems Engineering Laboratory, NUDT. His research interests include cyber-physical system, intelligent decision, computational intelligence, and information management.