2019 International Conference on Unmanned Aircraft Systems (ICUAS)
Atlanta, GA, USA, June 11-14, 2019

Robust multi-UAV route planning considering UAV failure

Ruchir Patel[1], Eliot Rudnick-Cohen[1], Shapour Azarm[1], Jeffrey W. Herrmann[1,2]

Abstract—**This paper describes a robust multi-UAV route planning problem in which any one of the vehicles could fail during plan execution at any visited location. The UAVs must visit a set of fixed locations; if one UAV fails, the other vehicles must cover any unvisited locations. The objective is to optimize the worst-case cost. This paper formulates the problem with a min-sum objective (minimizing the total distance traveled by all vehicles) and a min-max objective (minimizing the maximum distance traveled by any vehicle). A Genetic Algorithm (GA) was used to find approximate robust optimal solutions on seven instances. The results show that the GA was able to find solutions that have better worst-case cost than the solutions generated by other approaches that were tested.**

I. Introduction

Teams of Unmanned Aerial Vehicles (UAVs) have come into focus over recent years due to their advantages in performance and robustness over single vehicle systems. These systems are commonly seen in both commercial and military applications including drone delivery, search and rescue, fighting wildfires, surveillance, and hazard clearance. Planning routes for multi-UAV systems is critical to accomplishing such tasks in the best way possible. In all of these applications there is uncertainty in the environment, hazards, and vehicles (e.g., sensor or reliability uncertainty). Planning under uncertainty has been considered in the past assuming knowledge of probability distributions for uncertain parameters [1-3]. However, such distributions may not be known in practice. In this case, planning methods need to produce plans that are robust against these sources of uncertainty with guarantees on worst-case performance. Here, robustness (or robust optimality) is defined as the performance under worst-case realizations of uncertainty, which is especially relevant when there is no probability distribution for the uncertainty.

This paper discusses an approach for solving a multi-UAV failure-robust route planning problem where the goal is to plan the most robust route for vehicles to visit a given set of locations of interest. Here, robustness is considered with respect to uncertainty in vehicle failure: a UAV could fail at any location that it visits. This planning problem is motivated by hazardous applications and scenarios in which an adversary may seek to destroy a vehicle. As aircraft, UAVs are more at risk to a catastrophic, mission-ending accident due to the potential of a crash if something onboard fails, winds become dangerous, or some other interference. The objective is to minimize the worst-case cost, where the cost includes the cost of the initial plan up to the failure and the cost of the recovery plan generated to complete the mission after the failure, i.e., the locations that the failed UAV did not visit must be visited by the surviving UAVs. Fig. 1 shows an example of this where the unvisited locations along the failed UAV's route would not be covered by the initial plan. Two objectives are considered and optimized: min-sum, where the total distance traveled by all vehicles is minimized, and min-max, the maximum distance traveled by any vehicle is minimized. The min-sum objective is useful when optimizing fuel consumption, while the min-max objective corresponds to minimizing total mission time.

The rest of this paper is organized as follows: Section II positions the proposed approach with respect to related work. Section III defines the multi-UAV failure-robust routing planning problem. Section IV describes the problem formulations for the min-sum and min-max objectives. Section V describes the proposed Genetic Algorithm (GA) solution approach. Section VI describes the experimental setup. Results are presented in Section VII, and Section VIII discusses the results. Finally, Section IX concludes this paper with final remarks and a brief discussion on future directions.

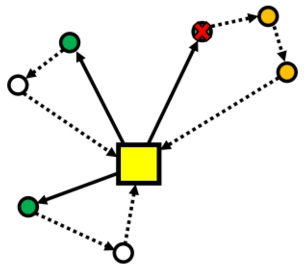

Figure 1. Example set of routes for 3 UAVs with a failure scenario. The square is the depot and circles are locations of interest. Traveled segments are denoted with solid lines and visited locations are filled green. UAV failure is shown by the red "x" over one location. The orange locations are the ones that the failed vehicle did not visit.

II. Related Work

The multi-UAV failure-robust route planning problem can be considered an extension of the well-known multiple Traveling Salesman Problem (mTSP), where the goal is to route multiple salesmen to visit a set of cities. In this paper, the salesmen are the UAVs and the cities are the locations of interest (or locations to be visited). The classical mTSP has been studied extensively in the past. Bektas [4] discussed various formulations and solution procedures used to solve the mTSP exactly and approximately. Singh [5] and Kumar and Panneerselvam [6] reviewed meta-heuristic algorithms that have been used to solve the mTSP and the Vehicle Routing Problem (VRP).

[1]Department of Mechanical Engineering, University of Maryland, College Park, MD 20742 {rpatel18, erudnick, azarm, jwh2}@umd.edu
[2]Institute for Systems Research, University of Maryland, College Park, MD 20742

Ritzinger et al. [1] presented a survey of stochastic VRPs and described the most commonly studied sources of uncertainty, including stochastic travel times, demands, customers, and combinations of these. The objective in stochastic VRPs and/or mTSPs is often to minimize the expected costs where a probability distribution is available for uncertain parameters. Sundar et al. [2] studied such a problem in the context of path planning for multiple heterogeneous unmanned vehicles with uncertain service times. They formulated the problem with an expected value objective and solved it using a branch-and-cut algorithm.

Dulai et al. [3] studied the "fault-tolerant" VRP, which is similar to the problem considered in this paper, and solved it using a heuristic method based on the Clarke and Wright savings algorithm. They assumed that the failure scenarios are equally likely and optimized the expected value. In their problem, only one surviving vehicle is used to visit the locations on the failed vehicle's route. However, in applications such as multi-UAV search, using multiple vehicles to cover the unvisited locations can lead to lower costs.

Problems that utilize expected value require knowledge of probability distributions for uncertain parameters. Robust variants of these problems such as the robust VRPs studied in [7] and [8], however, do not require distributions to be known because the objective is to optimize the worst-case performance over all possible realizations of uncertain parameters.

Vehicle failure has been considered by researchers of autonomous multi-agent systems. Habib et al. [9] studied an Open Multiple Depot VRP (OMDVRP) in the context of cooperative UAV path planning in which each vehicle starts at its own depot and visits a set of locations without returning to the initial depot. They used a branch-and-bound algorithm to find solutions to a Mixed-Integer Linear Programming (MILP) formulation of the problem in which a UAV fails. Giger et al. [10] focused on mission replanning for unmanned underwater vehicles (UUVs) after vehicle failure. They modeled the problem as a Multiple Depot mTSP (MDmTSP) and used a GA for replanning. Although multiple approaches for replanning have been studied, previous work did not study the problem of generating initial route plans that are robust against vehicle failure.

To extend this line of research, our study formulates a new multi-UAV route planning problem that seeks to maximize robustness against vehicle failure and evaluates the performance of a GA and other solution techniques. Robustness is the cost under the worst-case realization of uncertainty. Therefore, we do not utilize a probability distribution for uncertain parameters, unlike Ritzinger et al. [1], Sundar et al. [2], and Dulai et al. [3].

III. Problem Definition

The multi-UAV failure-robust route planning problem can be defined as follows: a team of m UAVs is initially stored at a depot but is tasked with visiting n-1 specified locations in a region of interest (n equals the total number of locations, including the depot). The goal is to plan robust optimal routes for these UAVs such that every location is visited once and surviving vehicles end their routes at the depot. The objective function is the worst-case cost, where the cost includes the cost of the initial plan up to the failure and the cost of the recovery plan generated to complete the mission after the failure. Maximizing robustness requires minimizing the worst-case cost. Two cost functions were considered: the total distance traveled by all vehicles (the min-sum objective) and the maximum distance traveled by any vehicle (the min-max objective). The recovery plan is created at the time of failure; this replanning step involves determining routes for the remaining UAVs so that they visit the unvisited locations and return to the depot.

The problem formulation is based on the following assumptions: A UAV can fail only at locations visited upon arrival. The location at which a UAV fails is considered visited and does not need to be visited by any other vehicle. At most one UAV can fail. Once one UAV fails, the remaining vehicles will not fail. At each location, each UAV spends a fixed amount of time to perform the required task (such as search) and a variable amount of idle time. UAV dynamics (such as acceleration and deceleration) are ignored. UAVs travel at a constant speed (v) from location to location.

IV. Formulation

This problem can be represented as a two-stage robust optimization problem where the first stage determines the initial plan and the second stage is for replanning or recourse after failure uncertainty is realized. (In practice, the failure-robust problem could be solved again to mitigate the impact of another failure, if it could be solved quickly enough.)

The first stage can be thought of as an mTSP where the UAVs are the salesmen and the depot and the locations to visit are cities. The problem can be modeled using a graph in which the nodes represent the locations of interest and the depot and the edges represent the paths between these nodes. The second stage is an MDmTSP where the surviving UAVs visit the locations that have yet to be visited after a vehicle failure.

A. Classical min-sum formulation

For the min-sum mTSP formulation, let the binary variable x_{ij} = 1, if the edge from node i to node j is included in the solution, otherwise x_{ij} = 0. Let u_i denote the position of node i in a salesman's tour, and let p denote the maximum number of nodes that can be visited by any salesman. Let c_{ij} be the cost (distance) of the edge from i to j. The first stage solution for the min-sum formulation is the initial plan $\boldsymbol{x_1}$ which is comprised of x_{ij} and u_i for all i and j. Based on [4], the classical integer programming formulation for the min-sum mTSP is the following:

$$\min_{\boldsymbol{x_1}} \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \tag{1}$$

$$\text{s. t.:} \sum_{j=2}^{n} x_{1j} = m \tag{2}$$

$$\sum_{i=2}^{n} x_{i1} = m \tag{3}$$

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad j = 2, \ldots, n \tag{4}$$

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad i = 2, \ldots, n \tag{5}$$

$$x_{ij} + x_{ji} \leq 1 \quad i, j = 2, \ldots, n \tag{6}$$

$$u_i - u_j + p x_{ij} \leq p - 1 \quad i, j = 2, \ldots, n \tag{7}$$

$$x_{ij} \in \{0, 1\} \qquad i, j = 1, \ldots, n \tag{8}$$

$$u_i \in \{1, 2, \ldots, p - 1\} \quad i = 2, \ldots, n \tag{9}$$

The min-sum objective (or cost) function, as given by (1), represents the total distance traveled by all salesmen. Equations (2) - (7) represent the constraints. Equations (2) and (3) ensure that all *m* salesmen exit and enter the depot. Equations (4) and (5) ensure that every other node is visited exactly once since it constrains the tours such that only one edge enters and exits each intermediate node. Equation (6) ensures that no symmetry occurs between edges. This means if an edge from node *i* to node *j* is used, then the edge from node *j* to node *i* cannot be used. Equation (7), which is known as the Miller-Tucker-Zemlin (MTZ) constraint [11], eliminates the possibility of sub-tours (cycles not connected to the depot). Equations (8) and (9) specify the domain of x_{ij} and u_i.

B. *Classical min-max formulation*

For the min-max formulation, let the binary variable $x_{ijk} = 1$, if the edge from node *i* to node *j* is included in the solution of salesman *k*, otherwise $x_{ijk} = 0$. Let u_i denote the position of node *i* in a salesman's tour, and let *p* denote the maximum number of nodes that can be visited by any salesman. Edge costs are denoted by c_{ij}. For the min-max formulation the initial plan $\boldsymbol{x_1}$ is comprised of x_{ijk} and u_i for all *i, j*, and *k*. The classical three-index integer programming formulation for the min-max mTSP [12] is the following:

$$\min_{\boldsymbol{x_1}} \max_{k} \left\{ \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ijk} \right\} \tag{10}$$

$$\text{s. t.:} \sum_{j=2}^{n} x_{1jk} = 1 \qquad k = 1, \ldots, m \tag{11}$$

$$\sum_{i=2}^{n} x_{i1k} = 1 \qquad k = 1, \ldots, m \tag{12}$$

$$\sum_{i=1}^{n} \sum_{k=1}^{m} x_{ijk} = 1 \qquad j = 2, \ldots, n \tag{13}$$

$$\sum_{j=1}^{n} \sum_{k=1}^{m} x_{ijk} = 1 \qquad i = 2, \ldots, n \tag{14}$$

$$\sum_{i=1}^{n} x_{ijk} - \sum_{i=1}^{n} x_{jik} = 0 \quad j = 2, \ldots, n \quad k = 1, \ldots, m \tag{15}$$

$$\sum_{k=1}^{m} x_{ijk} + \sum_{k=1}^{m} x_{jik} \leq 1 \quad i, j = 2, \ldots, n \tag{16}$$

$$u_i - u_j + p \sum_{k=1}^{m} x_{ijk} \leq p - 1 \quad i, j = 2, \ldots, n \tag{17}$$

$$x_{ijk} \in \{0, 1\} \qquad i, j = 1, \ldots, n \quad k = 1, \ldots, m \tag{18}$$

$$u_i \in \{1, 2, \ldots, p - 1\} \quad i = 2, \ldots, n \tag{19}$$

Equation (10) represents the min-max objective function of maximum distance traveled by any salesmen. Equations (11) - (14) represent the same constraints as (2) - (5). Equation (15) ensures if salesman *k* enters node *j* then salesman *k* must exit node *j*. Equations (16) - (19) represent the same constraints as (6) - (9).

C. *Robust Min-Sum and Min-Max Formulations*

The objective of the classical min-sum or min-max mTSP can be modified to add failure-robustness to the formulation. Given a set of initial routes for the *m* UAVs, let *s* be a failure scenario that denotes the failure of a specific UAV at one of the locations that it will visit. Let *S* be the set of all possible scenarios, including the no-failure scenario. At the time of one UAV's failure, each surviving UAV is either at a node or along an edge. Since these positions are needed to solve the second stage replanning problem, a timing model is needed to provide the UAV positions with respect to the failed UAV's final position. The timing model utilizes the initial plan ($\boldsymbol{x_1}$) and the failure scenario *s* to determine the current positions of the remaining $m - 1$ UAVs and the set of nodes left to be visited. The UAVs positions and leftover nodes are used to formulate the second stage replanning problem, which is an MDmTSP in which each UAV's position is a depot and the leftover nodes are the cities. The classical integer programming formulation for the MDmTSP is very similar to that of the mTSP [13].

Using the definitions of each stage, for robust min-sum optimization the objective (1) becomes:

$$\min_{\boldsymbol{x_1}} \max_{s \in S} \{ f_1(\boldsymbol{x_1}, s) + f_2(\boldsymbol{x_1}, s) \} \tag{20}$$

where $f_1(\boldsymbol{x_1}, s)$ is the objective value of the first stage mTSP solution up to the point of the vehicle failure in scenario *s*, the uncertain scenario which specifies the UAV that fails and its location when it fails. This function computes the total sum of the costs of edges traveled by the UAVs up to the point of failure. Let the variable t_i denote the time at which a UAV leaves location *i*. The variable a_i is the amount of time a UAV spends at location *i* which is comprised of a fixed search time spent at each location plus a variable amount of idle time. The variable t_F denotes the time at which the failure occurs at location *h*. $f_1(\boldsymbol{x_1}, s)$ can be defined as follows:

$$f_1(\boldsymbol{x_1}, s) = \sum_{i,j} c_{ij} x_{ij} + \sum_{l,c} v(t_F - t_l) x_{lc} \tag{21}$$

$$i, j : \; t_j - a_j < t_F \; ; \qquad l, c : t_F \geq t_l \text{ and } t_F \leq t_c - a_c$$

$$t_F = t_h - a_h \tag{22}$$

Equation (21) represents the total distance traveled by all UAVs up to the point of failure which can be split into two summations. The first summation is for edges completely traversed at the time of failure and the second summation is for edges that are partially traversed which occurs when UAVs are still along edges at the time of failure. Equation (22) sets the time passed as the time upon exiting failure location h minus the time spent at location h.

$f_2(\boldsymbol{x_1}, s)$ is the objective value of the second stage MDmTSP solution that computes the cost of the approximate MDmTSP solution after the UAV failure in scenario s. This function computes the total sum of the costs of edges traveled by all vehicles after the point of failure. Let the variable y_{ijk} denote the approximate solution to the MDmTSP replanning problem which equals 1 if the edge from node i to node j is included in the solution of UAV k, otherwise $y_{ijk} = 0$. $APPROX$ is a function that takes in the initial plan $\boldsymbol{x_1}$ and the scenario s, and outputs the approximate solution to the constructed MDmTSP instance. $f_2(\boldsymbol{x_1}, s)$ can be defined as follows:

$$f_2(\boldsymbol{x_1}, s) = \sum_i \sum_j \sum_k c_{ij} y_{ijk} \tag{23}$$

$$\{y_{ijk}\} = APPROX(\boldsymbol{x_1}, s) \tag{24}$$

Equation (23) represents the total distance traveled by all remaining UAVs after the point of failure which is a summation over all edges used in the replanning solution. Equation (24) sets y_{ijk} as the replanning solution outputted from the approximation method.

For the robust min-max formulation, the objective in Equation (10) becomes:

$$\min_{\boldsymbol{x_1}} \max_{s \in S} \max_{d \in D} \{g_1(\boldsymbol{x_1}, s, d) + g_2(\boldsymbol{x_1}, s, d)\} \tag{25}$$

For UAV d in the set of UAVs D, $g_1(\boldsymbol{x_1}, s, d)$ is the total distance that UAV d travels from time 0 until the vehicle failure in scenario s at time t_F.

$$g_1(\boldsymbol{x_1}, s, d) = \sum_{i,j} c_{ij} x_{ijd} + \sum_{l,c} v(t_F - t_l) x_{lcd} \tag{26}$$

$$i, j : \; t_j - a_j < t_F \; ; \qquad l, c : t_F \geq t_l \text{ and } t_F \leq t_c - a_c$$

$$t_F = t_h - a_h \tag{27}$$

Equation (26) is the same as (21) except the summation is only over UAV d's edges. Equation (27) is the same as (22).

For UAV d, $g_2(\boldsymbol{x_1}, s, d)$ is the total distance that UAV d travels after the UAV failure in scenario s.

$$g_2(\boldsymbol{x_1}, s, d) = \sum_i \sum_j c_{ij} y_{ijd} \tag{28}$$

$$\{y_{ijk}\} = APPROX(\boldsymbol{x_1}, s) \tag{29}$$

Equation (28) represents the distance traveled by UAV d after the point of failure which is a summation over all edges used by UAV d in the replanning solution. Equation (29) sets y_{ijk} as the replanning solution outputted from the approximation method.

V. Solution Approach

To solve the multi-UAV failure-robust route planning problem, we modified the GA presented by Kirk [14] and used it to generate robust solutions to the min-sum and min-max variants of the mTSP. The GA initializes a population of randomly generated solutions to the mTSP, and each solution's cost is evaluated based on Equation (20) or (25) depending on the objective function considered. The initial population also includes the non-robust optimal min-sum solution found by the Gurobi Optimizer [15]. Each solution is represented as a sequence of locations, a sequence of breakpoints that split the sequence of locations into distinct routes, and a sequence of idle times, one for each location. After evaluating the solutions in the population, the GA mutates some of the best solutions to form a new population of children. The GA uses eight mutation operators: flip, swap, slide, breakpoint modification, flip and breakpoint modification, swap and breakpoint modification, slide and breakpoint modification, and idle time modification. Flip reverses a subsequence of the sequence of locations. Swap takes two locations in the sequence and switches them. Slide shifts a subsequence of the sequence of locations. Breakpoint modification replaces the current set of breakpoints with a randomly generated new one. Idle time modification replaces the current set of idle times with a randomly generated new one.

The GA repeats the process of generating a new population for a specified number of iterations (or generations) and returns the lowest cost solution found. The cost evaluation procedure generates every possible failure scenario for a solution and evaluates the cost of the solution for every scenario. The pseudocode for this procedure is given by the function COST_EVAL in Algorithm 1 below. The method for generating the set of scenarios from a solution is defined as follows: for each route in the solution, if the route contains more than one location to visit, then every location in that route other than the last is included in the set of scenarios. The last location does not need to be considered because if the UAV were to fail there then the initial plan is still feasible and optimal. For this method, the number of scenarios will always equal $n - m$ including the scenario where no UAV fails, if the problem is constrained such that every UAV visits at least one location. In the unconstrained case where not all UAVs need to be used in a solution, the number of scenarios is bounded below by $n - m$ and above by $n - 1$ including the scenario where no UAV fails.

For each scenario generated, the COST_EVAL procedure creates an instance of the MDmTSP with a set of m-1 depots ($\boldsymbol{D}$) and set of locations ($\boldsymbol{q}$) that need to be visited (Algorithm 1, line 5). Each depot represents the current position of each surviving UAV, and the cities are the locations that have yet to be visited at the time of failure. This procedure also computes $f_1(\boldsymbol{x_1}, s)$ or $g_1(\boldsymbol{x_1}, s, d)$ for each UAV depending on the objective considered.

After creating the MDmTSP instance, the MST_APPROX procedure uses a Minimum Spanning Tree (MST) approximation method [16] to rapidly generate a new solution in which the remaining locations are visited and the UAVs return to the initial depot. The pseudocode for this procedure is given by MST_APPROX in Algorithm 2. This procedure creates a complete graph with zero-cost edges between every pair of depots. The MST that is generated includes m - 2 edges connecting the depots. The MST_APPROX procedure removes these edges to form a forest of trees, each rooted at a surviving UAV. The procedure then uses a preorder traversal on each tree and appends the original depot to that sequence to generate a route for each UAV. For the min-sum objective, the COST_EVAL procedure adds the cost of the new route to the initial cost and computes the total cost in this scenario. For the min-max objective, the procedure determines the cost for each UAV, and Line 7 (Algorithm 1) uses the greatest value of $g_1(\boldsymbol{x_1}, s, d) + g_2(\boldsymbol{x_1}, s, d)$ as the cost of the solution for this scenario. It repeats this for every scenario to determine the worst-case cost.

Algorithm 1: Pseudocode for evaluating the cost of the initial plan.

```
1:  function COST_EVAL($\boldsymbol{x_1}$, ini_dpt)
2:  scenarios ← scenarios_gen($\boldsymbol{x_1}$)
3:  worst_cost ← 0
4:  for each scenario $s$ in scenarios
5:     {$\boldsymbol{D}$, $\boldsymbol{q}$, ini_cst} ← mdmtsp_gen($\boldsymbol{x_1}$, $s$)
6:     routes ← MST_APPROX($\boldsymbol{D}$, $\boldsymbol{q}$, ini_dpt)
7:     worst_cost ← max(worst_cost, ini_cst + cost(routes))
8:  end for
9:  return worst_cost
10: end function
```

Algorithm 2: Pseudocode for generating a replanning solution using the MST approximation.

```
1:  function MST_APPROX($\boldsymbol{D}$, $\boldsymbol{q}$, ini_dpt)
2:  nodes ← $\boldsymbol{D} \cup \boldsymbol{q}$
3:  edges ← {}
4:  for each node $i$ in nodes
5:    for each node $j \neq i$ in nodes
6:      if is_depot($i$) and is_depot($j$)
7:        cost ← 0
8:      else
9:        cost ← dist($i$, $j$)
10:     end if
11:     edges ← edges ∪ {$i$, $j$, cost}
12:   end for
13: end for
14: G ← graph(nodes, edges)
15: mst ← PRIMS(G)
16: forest ← remove_depot_edges(mst)
17: routes ← {}
18: for each tree $t$ in forest
19:   route ← preorder_traversal($t$)
20:   route ← route.append(ini_dpt)
21:   routes ← routes ∪ route
22: end for
23: return routes
24: end function
```

VI. Experimental Setup

The approach described has been tested on 11 x 5, 26 x 8, 41 x 12, and 101 x 20 (n x m) randomly generated problem instances and three TSPLib [17] instances: eil51, eil76, eil101 with the number of UAVs being: m = 7, 12, and 17, respectively. The four randomly generated instances were created by selecting locations from a uniform distribution over a two-dimensional region [0, 100] x [0, 100]. The min-max objective was tested more extensively than min-sum because preliminary testing with the min-sum objective showed little to no improvement in robustness relative to the non-robust optimal solution in all instances. Evidence of this can be seen in Fig. 2 which depicts the solution found in the 26 x 8 min-sum instance. The progression of best worst-case cost over time (Fig. 2(b)) shows the insignificant improvement in worst-case cost of the "robust" solution over the initial non-robust optimal solution. Therefore, only min-max computational results are reported. Each problem instance was tested in constrained and unconstrained cases, where constrained means every UAV must be used in a solution. The GA was run for 1000 iterations with a population of 90 individuals. The fixed task time was set to 10 seconds, and velocity was set to 2 units per second. The variable idle times were limited to the range from 0 to 10 seconds. The GA was run for only three trials for each case reported due to time restrictions.

To evaluate the quality of the solutions that the GA found, we also constructed solutions using one-shot heuristics including a Time-Oriented Nearest Neighbors (TONN) construction heuristic based on [18] and the MST approximation described previously. We also generated solutions by solving the classical non-robust MILP using Gurobi (with a time limit of 1000 seconds).

We also ran the GA with a reduced scenario set to test the tradeoff between solution quality and computational effort. This reduced scenario set was comprised of the no-failure scenario and scenarios where the failed location is the first in a UAV's route. Preliminary testing indicated that those scenarios comprised the majority of worst-case scenarios for the min-max cases. The cardinality of the reduced scenario set was in the range from 2 to m+1. Experiments were run in MATLAB® R2018a on an Intel® Xeon® CPU E3-1245 v5 @ 3.50 GHz with 16.00 GB of RAM. For each trial of the GA, the solution history was saved to analyze the convergence properties of the method for every case. The best solution history refers to the progression of the best worst-case cost over time.

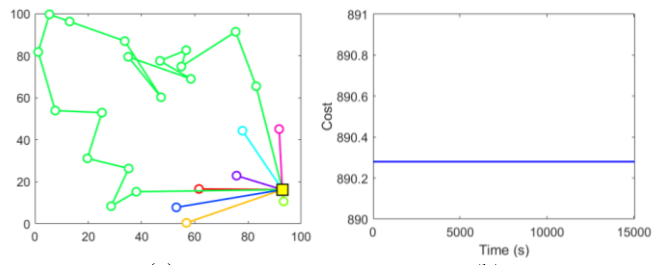
(a) (b)

Figure 2. Sample (a) solution and (b) progression of best worst-case cost over time found by the GA (full) in the 26 x 8 min-sum instance.

VII. Results

Fig. 3 shows an example of how replanning responds to failure for a less robust solution and a more robust solution for the 26 x 8 instance with the min-max objective function. The vehicle routes are shown with distinct colors. Fig. 3(a) and 3(b) depict failure scenarios for the two solutions; in each one, the vehicle failure is marked by the red "x". Fig. 3(c) shows the recovery plan for the less robust solution in Fig. 3(a); Fig. 3(d) shows the recovery plan for the more robust solution in Fig. 3(b). The locations of the surviving vehicles at the time of failure are indicated by black squares. In the recovery plan of the more robust solution (Fig. 3(d)), the gold, red, and green vehicles work together to cover the remaining locations that need to be visited after the failure. However, in the recovery plan of the less robust solution (Fig. 3(c)), the purple vehicle is forced to cover these remaining locations on its own. While the less robust and more robust solution have similar min-max costs without failure (241.8 and 250.7 units, respectively), the worst-case cost of the less robust solution is significantly higher than that of the more robust solution (563.5 and 260.9 units, respectively).

Table I summarizes the worst-case costs of the final solutions found by five different methods: (1) the constrained GA with the full scenario set, (2) the constrained GA with the reduced scenario set, (3) the TONN heuristic, (4) the MST approximation method, and (5) optimistic planning. The optimistic planning procedure uses the Gurobi optimization solver to find a solution to the classical problem (Equations 10-19) without considering the possibility of failure. Reported GA costs were averaged over three trials. Averaged over the seven instances, relative to the optimistic solutions, the solutions found by the GA with the full scenario set reduced the worst-case cost by 43%.

Fig. 4 shows an overlay of sample solution history plots, each describing results from three trials for the three TSPLib instances. Fig. 4 shows that for smaller instances, high-quality solutions were found more quickly.

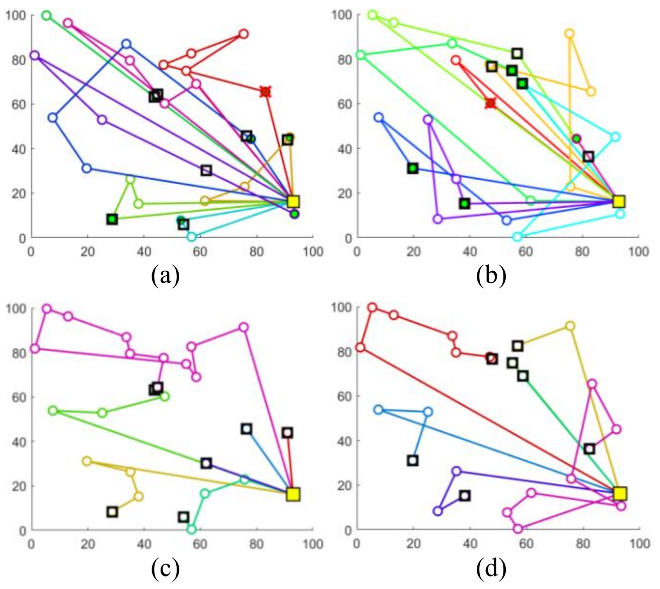
(a) (b) (c) (d)

Figure 3. Comparison of original and recovery routes for a failure scenario using the (a) Gurobi and (b) GA (full) solutions for the 26 x 8 min-max instance. The recovery plans for the Gurobi and GA solutions are shown in (c) and (d), respectively. The yellow square is the depot and circles are the locations of interest. Each color indicates the route of a different UAV. Black squares indicate locations of surviving UAVs at the time of failure. For (a) and (b), green-filled circles are visited locations at the time of failure. The red "x" indicates the failure scenario.

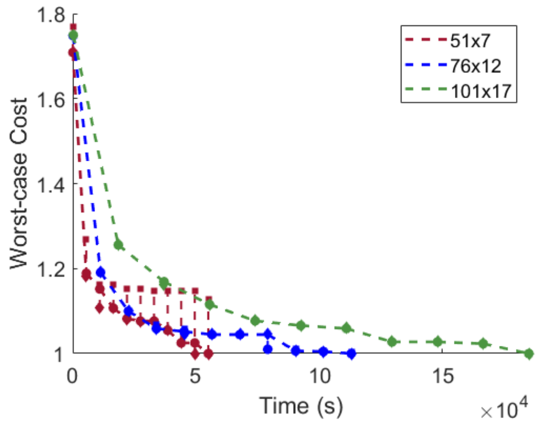

Figure 4. Constrained min-max GA (full) solution histories for TSPLib instances. Worst-case cost is scaled by best cost found in each instance. Each marker shape (diamond, circle, and square) corresponds to each trial. The dashed line is the median over three trials with vertical intervals created using minimum and maximum of the trials at each point.

Table I. Comparison of min-max worst-case costs for all methods averaged over three trials. GA and optimistic planning solutions reported were found under constrained conditions. The boldface value is the best worst-case cost (in units) for each instance.

Instance	Optimistic	GA (Full)	GA (Red.)	TONN	MST
11 x 5	193.3	**137.2**	171.7	337.9	330.3
26 x 8	563.5	**260.9**	306.9	569.7	562.8
41 x 12	623.0	**258.7**	365.4	593.0	646.8
51 x 7	289.1	**232.1**	398.3	410.1	602.8
76 x 12	624.2	**249.2**	331.9	618.5	748.1
101 x 17	401.4	**237.2**	350.5	614.3	901.2
101 x 20	501.7	**291.5**	356.9	528.9	1036.4

VIII. Discussion

The GA with the full scenario set was able to converge to solutions with lower worst-case cost on all instances when compared with the solutions found by other methods. As mentioned, little improvement was seen in the worst-case cost relative to the non-robust optimal solution for all instances with the min-sum objective. Min-sum solution paths tended to contain intraroute intersections which is typically considered non-optimal in terms of non-robust cost. However, such intersections yielded slight improvement in worst-case cost in min-sum cases. The constrained GA almost always found lower cost solutions than the unconstrained with the min-max objective. This occurred because using more vehicles typically reduces the greatest distance traveled. For the min-sum objective, however, it is always better to use a single vehicle to achieve the lowest total distance.

For the min-max objective, among all instances tested, significant improvement was seen in the worst-case cost between the solutions generated by optimistic planning and the solutions generated by the GA, which considered the worst-case cost. It is likely that the poor worst-case costs of the optimistic solutions stem from the MST method of replanning which is poorly suited for the min-max objective. This can be seen in Fig. 3(c) where the purple vehicle is assigned more locations than other remaining UAVs. To accurately determine the effects of the replanning method, similar studies utilizing different replanning methods are needed. However, it is also important to consider that replanning methods need to be sufficiently fast to run in real-time, so such inefficiencies may be unavoidable.

More variation was seen in the medium instances (51 x 7, 41 x 12) as seen in Fig. 4. This may be attributed to the layout of the locations and depot in these instances. Additional trials are needed to fully understand the variation of the GA in different instances.

The high computational cost of the approach can be seen in the solution histories. As the problem size grows, the convergence rate tends to slow and the time to finish 1000 iterations increases significantly. The number of scenarios considered increases with the problem size. Each additional scenario requires constructing a replanning solution using the approximate MST method, which also increases in complexity with the problem size. This motivates using a reduced scenario set as described previously. As seen in Table I, using the reduced scenario set yielded solutions with a worse solution cost than those generated using the full scenario set, but these solutions were better than the solutions generated by the TONN, MST, and optimistic planning procedures (other than the 51 x 7 instance). Using the reduced scenario set required less computational effort than using the full scenario set but more computational effort than the construction heuristics (TONN and MST). For example, in one trial of the 101 x 17 constrained min-max instance, the computation time of the GA using the full set of scenarios was on the order of 10^5 seconds while, with the reduced scenario set, it was 10^4 seconds. The time order of magnitude for the construction heuristics was 10^0 seconds. Thus, these results show that there is a tradeoff between the robustness of a solution and the computational effort needed to find it.

IX. Conclusion

This study considered a new multi-UAV route planning problem with a goal of planning robust routes for UAVs that can fail. The cost function in the problem is the cost of the initial plan up to the failure plus the cost of the recovery plan generated to complete the mission after the failure. In this way, the paper contributes to the problem of generating initial route plans that are robust against vehicle failure. Robustness provides guarantees on worst-case performance without the need of knowing probability distributions for uncertain parameters. This paper formulated the route planning problem and presented a GA that can find solutions. We tested the approach on problem instances of different sizes and used the results to evaluate the method's strengths and weaknesses. The results showed the method can find high quality solutions, although it requires more computational effort than running construction heuristics or solving an integer programming problem to find a solution without considering uncertainty. This motivates the desire for more efficient methods with less sacrifice in solution quality. Using a reduced scenario set required less computational effort but yielded solutions that were not as robust.

Future work should consider alternatives to the proposed reduced scenario set such as iteratively building up the scenario set. Other heuristics that can better exploit the structure of the robust planning problem should be explored using the insights gained from this study. In order to understand the optimality gap of solutions found by the GA and other heuristics, an MILP should be formulated to exactly solve the robust problem. The possibility of multiple UAV failures should also be considered in future studies. Considering multiple failures significantly increases the complexity of this already complex problem, therefore, simplifying assumptions or heuristics are needed. Solutions found by considering a single UAV failure may perform well even if multiple failures did occur. One possible approach is to solve the one-failure problem every time a vehicle fails so that the recovery plan is robust against the next failure. Such solutions should be evaluated in multi-failure scenarios to analyze how robust they are to multiple failures.

Acknowledgment

The work presented in this paper was supported in part by a contract from Toyon Research Corporation as part of a U.S. Navy STTR project and a contract from the Air Force Research Lab (AFRL), contract number FA87501820114. This support is gratefully acknowledged.

References

[1] U. Ritzinger, J. Puchinger, and R. F. Hartl, "A survey on dynamic and stochastic vehicle routing problems," *International Journal of Production Research*, vol. 54, no. 1, pp. 215–231, 2016.

[2] K. Sundar, S. Venkatachalam, and S. G. Manyam, "Path planning for multiple heterogeneous Unmanned Vehicles with uncertain service times," in *2017 International Conference on Unmanned Aircraft Systems (ICUAS),* pp. 480–487, 2017.

[3] T. Dulai, g. Werner-Stark, and K. M. Hangos, "Algorithm for directing cooperative vehicles of a vehicle routing problem for improving fault-tolerance," *Optimization and Engineering*, vol. 19, p. 239, June 2018.

[4] T. Bektas, "The multiple traveling salesman problem: an overview of formulations and solution procedures," *Omega*, vol. 34, no. 3, pp. 209–219, 2006.

[5] A. Singh, "A review on algorithms used to solve multiple travelling salesman problem," *International Research Journal of Engineering and Technology (IRJET)*, 2016.

[6] S. N. Kumar and R. Panneerselvam, "A survey on the vehicle routing problem and its variants," *Intelligent Information Management*, vol. 4, no. 03, p. 66, 2012.

[7] I. Sungur, F. Ordonez, and M. Dessouky, "A robust optimization approach for the capacitated vehicle routing problem with demand uncertainty," *IIE Transactions*, vol. 40, no. 5, pp. 509–523, 2008.

[8] J. Han, C. Lee, and S. Park, "A Robust Scenario Approach for the Vehicle Routing Problem with Uncertain Travel Times," *Transportation Science*, vol. 48, no. 3, p. 373-390, 2014.

[9] D. Habib, H. Jamal, and S. A. Khan, "Employing multiple unmanned aerial vehicles for co-operative path planning," *International Journal of Advanced Robotic Systems*, vol. 10, no. 5, p. 235, 2013.

[10] G. Giger, M. Kandemir, and J. Dzielski, "Reliable mission execution using unreliable UUVs," *AUVSIs Unmanned Systems North America*, 2008.

[11] C. E. Miller, A. W. Tucker, and R. A. Zemlin, "Integer programming formulation of traveling salesman problems," *Journal of the ACM (JACM)*, vol. 7, no. 4, pp. 326–329, 1960.

[12] R. Necula, M. Breaban, and M. Raschip, "Tackling the bi-criteria facet of multiple traveling salesman problem with ant colony systems," in *2015 IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 873–880, Nov 2015.

[13] I. Kara and T. Bektas, "Integer linear programming formulations of multiple salesman problems and its variations," *European Journal of Operational Research*, vol. 174, no. 3, pp. 1449–1458, 2006.

[14] J. Kirk, "Multiple Traveling Salesman Problem – Genetic Algorithm," *Mathworks.com*, May. 6, 2014. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/19049-multiple-traveling-salesmen-problem-genetic-algorithm. [Accessed: Aug. 23, 2018].

[15] "Gurobi Optimizer Reference Manual", *Gurobi Optimization*, 2018, [Online]. Available: http://www.gurobi.com/documentation/8.0/refman.pdf. [Accessed: Sep. 11, 2018].

[16] S. Rathinam, R. Sengupta, and S. Darbha, "A resource allocation algorithm for multivehicle systems with nonholonomic constraints," *IEEE Transactions on Automation Science and Engineering*, vol. 4, no. 1, pp. 98–104, 2007.

[17] "MP-TESTDATA - The TSPLIB Symmetric Traveling Salesman Problem Instances." [Online]. Available: http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsp/index.html. [Accessed: Aug. 23, 2018].

[18] M. M. Solomon, "Algorithms for the vehicle routing and scheduling problems with time window constraints," *Operations Research*, vol. 35, no. 2, pp. 254–265, 1987.