Unmanned Aerial Vehicle Routing Problem for Patrolling Missions - A Progressive Hedging Algorithm

Sudarshan Rajan[a], Kaarthik Sundar[b], Natarajan Gautam[a]

[a] Texas A& M University, College Station, TX
[b] Information Systems & Modeling, Los Alamos National Laboratory, Los Alamos, NM

Abstract

The paper presents a two-stage stochastic program to model a routing problem involving an Unmanned Aerial Vehicle (UAV) in the context of patrolling missions. In particular, given a set of targets and a set of satellite targets corresponding to each target, the first stage decisions involve finding the sequence in which the vehicle has to visit the set of targets. Upon reaching each target, the UAV collects information and if the operator of the UAV deems that the information collected is not of sufficient fidelity, then the UAV has to visit all the satellite targets corresponding to that target to collect additional information before proceeding to visit the next target. The problem is solved using a progressive hedging algorithm and extensive computational results corroborating the effectiveness of the proposed model and the solution methodology is presented.

Keywords: two-stage stochastic program, progressive-hedging, integer programming, routing, unmanned aerial vehicles

1. Introduction

The use of small Unmanned Aerial Vehicles (UAVs) has seen a tremendous increase in the past decade for both civilian applications [1, 2] viz., precision agriculture [3, 4], forest fire monitoring and management [5], ocean bathymetry [6] etc. and military applications [7] viz., monitoring, surveillance, border patrol, intelligence, and reconnaissance (see [8, 9] and references therein). The primary reason for this increase is attributed to the low fixed and operational costs, ease of use, payload capacity, and ability to fly low-altitude missions. There is extensive work in the literature [10] that examines specific applications and develops algorithms ranging from higher level path planning to lower level control algorithms and the integration of the aforementioned applications. Higher level path planning algorithms are offline algorithms that are used to obtain paths for the vehicles a few hours prior to the start of the mission; in the optimization literature, this corresponds to vehicle routing problems (VRP) for UAVs. The lower level control algorithms are online algorithms that perform real-time trajectory adjustments and vehicle speed changes based on the environment at play. Most of the literature in small UAVs have had an application-specific focus because each application tends to impose its own unique set of constraints that need to be handled separately. One application of small UAVs that has received little attention in literature, in terms of higher level path planning, is that of patrolling [11]. Patrolling applications are essentially data collection missions and despite the term patrolling having a military application connotation, similar application exists even in a civilian context like precision agriculture, crop health monitoring [3]. Such missions are typically associated with uncertainty in the information collected from pre-specified targets. It also entails re-routing to visit additional nearby locations near the pre-specified targets to improve the fidelity of the information gathered. Hence, higher level planning algorithms that take into account this information uncertainty and plan vehicle routes are needed for such applications. There have been several attempts in the last few years to build platforms for operating small UAVs in data gathering missions pertaining to precision agriculture [3], mapping disaster recovery [12], etc. To make things concrete, consider the application of precision farming where the UAV visits a particular target to obtain images of the crops. Suppose the images indicate that the crops at that location are damaged or have an unusual pattern in leaf color, then the mission control would want the vehicle to inspect additional nearby locations to gather more information to come up with robust conclusions on the crop health. If nothing unusual is observed then the vehicle is expected to continue its mission. In this article, we address the problem of higher level planning that can occur very frequently in almost all data gathering applications that is informally defined as follows:

Given a set of targets, a source, a destination, and a UAV, the objective of the problem is to find a path for the UAV that starts at the source, terminates at the destination, and visits each target exactly once, such that the following conditions are satisfied (i) if the information collected at a particular target (uncertain) is not sufficient, the UAV has to visit one [13] or more satellite targets corresponding to the visited target, before proceeding towards the next target in the path and, (ii) the sum of the cost of the path and the expected additional travel cost to visit the satellite targets is a minimum.

We shall refer to the above problem as the single vehicle data gathering problem (SVDGP). The uncertainty in the SVDGP is associated with the information collected at a particular target. This uncertainty is modeled using a probability distribution and we do not make assumption on the exact form of the distribution. Rather, we assume that the distribution permits sampling. This makes the problem setting very general and increases its applicability to a very large pool of civilian and military applications detailed in the previous paragraphs. We formulate the SVDGP as a two-stage decision making problem where the first stage ("here-and-now") decisions are made before of the mission under the face of uncertainty and the second stage, recourse decisions are made once the mission starts and uncertainty is revealed. Despite the fact that a multi-stage setting would provide more modeling power to such data-gathering missions, we avoid this setting as it comes at a cost of computational burden and complexity of presentation. Nevertheless, we note that the formulation and the algorithms presented in this paper can be extended to a multi-stage setting.

1.1. Related Work

The literature contains many variants of UAV routing problems, and algorithms that can obtain optimal solutions and heuristics have been extensively studied for these variants. For ease of exposition, we will analyze the work done in the literature using the following two categories: (1) deterministic UAV routing approaches and (2) stochastic approaches.

As far as deterministic approaches are concerned, plenty of optimization models and algorithms to compute optimal solutions, and fast heuristics to compute good feasible solutions have received extensive attention over the past decade. As mentioned in the introduction, most of the literature concerning deterministic approaches small UAV routing have had an application-specific focus because each application tends to impose its own unique set of constraints that need to be handled separately. In most of the approaches, the concerned routing problems are modeled as variants of the single traveling salesman problem (TSP), multiple TSP, or some variants of the Vehicle Routing Problems (VRPs) with additional constraints to model the specific mission at hand; for instance see [14, 15, 16]. An interested reader is referred to [17] for an extensive survey on optimization approaches for deterministic drone routing problems. To the best of our knowledge, the only work in the literature that addresses the specific application of patrolling is that of [11] where the authors propose an integer programming formulation to model the patrolling problem as a deterministic multiple TSP and present heuristics to solve the same.

When compared to the deterministic approaches, literature that focuses on modeling uncertainty in the missions and developing algorithms to solve stochastic variants of the problems concerning UAVs are scarce and very recent. Two-stage stochastic programming formulations have been explored in the context of single and multiple drone delivery problems [18, 19, 20, 21]. To the best of our knowledge, apart from our preliminary conference article that introduces a simpler variant of the SVDGP with only one satellite location per target [13], there is no work in the literature that explores such formulations and algorithms in the context of data gathering missions. In [13], we formulate the SVDGP with only one satellite location per target [13] to make the second stage optimization problem a linear program; this in turn enables a simple solution algorithm to compute the optimal two-stage solution. In this paper, we explore a general and more practical setting where there are multiple satellite locations per target that need to be visited to obtain high fidelity information in the target under consideration. The formulation, as presented in the subsequent sections, contains binary variables in both the first as well as the second stage and we present the progressive hedging algorithm [22] to solve this problem. The rest of the article is organized as follows: in Sec. 2, we present the formal problem statement after introducing the necessary notations, in Sec. 3, we mathematically formulate the SVDGP as a two-stage stochastic program and present the algorithm to solve the same in Sec. 4; finally in Sec. 5, we present the computational results followed by conclusions and future research directions in Sec. 6.

2. Problem Statement

Before we present the formal problem statement, we introduce some notations that will be used throughout the rest of the article. We are given a set of n targets $\hat{T} = \{t_1, \dots, t_n\}$, a source s, and a destination d. For ease for exposition, we will assume that $s = d = t_0$ and remark that the formulations and algorithms can be extended easily to the case where they are distinct. We also let $T = \hat{T} \cup \{t_0\}$. As detailed in the previous section, the decision making process is two-staged. The first-stage decision for the mission is to compute a path for the UAV that starts and ends at t_0 and visits each target in the set $\hat{T}$ and collects information. The information can be anything and usually depends on the application at hand. Also associated with each target $k \in T$ is a set of peripheral locations S_k whose purpose is as follows: suppose the UAV visits the target k and collects some information, then depending on the fidelity of this information, it may have to visits some additional set of locations (peripheral or satellite location set S_k) to either validate the collected information or to gather more information to aid in the decision making process. The decision on whether the UAV needs to visit these peripheral targets are recourse decisions and are taken after the realization of uncertainty is revealed. For the purpose of this article, we assume $|S_k|$, the cardinality of S_k for each $k = 1, \dots, n$, is m, where $m \geqslant 2$ and $S_0 = \emptyset$. We shall, from here on, refer to the set $V = T \bigcup S_1 \bigcup \cdots \bigcup S_n$ as the set of vertices. Given these notations, the SVDGP is formulated on a graph $G = (V, E)$ where E denotes the edge set; we delegate the definition of the edge set E to the Sec. 3. We note that G can be a directed or an undirected graph and the formulation and the algorithms presented in this article are applicable to both cases. To keep the exposition fairly general, we will assume that the graph is directed and that the cost of traveling between any pair of targets is asymmetric i.e., given $(i, j) \in V$, $c_{ij} \neq c_{ji}$. The cost of traversing an edge $(i, j) \in E$ can be anything ranging from distance to travel time. We also assume that this cost of traversal between any pair of vertices is known a-priori or can be pre-computed. Since the focus of the article is to develop a model to account for uncertainty in a systematic way, in the next section, we detail the uncertainty model associated with the information collected at each target.

2.1. Uncertainty Modeling

The uncertainty in the problem arises from the fidelity of information collected at any target $i \in T$. Based on the information collected at target i, the UAV may or may not have to visit the peripheral locations in the set S_i before proceeding to visit the next target in the route. Hence, we associate with each target $i \in T$, a Bernoulli random variable with probability p_i which denotes the probability that the information collected at target i is not of sufficient fidelity. For the target t_0, we assume that $p_i = 0$, where $i \in \{t_0\}$. For ease of exposition, we assume that the random variables corresponding to any pair of targets, is independent of each other. However, this can be relaxed or changed and the uncertainty can also be modeled using a Markov chain. The uncertainty in this information collected over all the targets is modeled as a binary scenario vector of size $|T|$ where the component corresponding to target $i \in T$ takes a value 1 with a probability p_i. These realizations of uncertainty are captured in a countable set of scenarios Ω, in which each element $\omega \in \Omega$ occurs with probability p_ω. Each element $\omega \in \Omega$ is a $|T|$-dimensional binary vector where each element is 1 with a probability p_i. We let ω_i denote the i^{th} component of ω. In this particular case, given that the probabilities p_i and p_j associated with any pair of targets (i, j) are independent, the probability of occurrence of the scenario ω is given by

$$p_\omega = \prod_{i=0}^{n} \{\mathbf{1}(\omega_i = 1) \cdot p_i + \mathbf{1}(\omega_i = 0) \cdot (1 - p_i)\} \tag{1}$$

where, the function $\mathbf{1}(\cdot)$ is the indicator function.

If we let $\mathcal{F}$ denote the set of all feasible first-stage paths for the UAV, the first-stage cost is the cost of the path and for any feasible path in $\mathcal{F}$, the second-stage decision is a set of recourse decisions which is a set of routes through the peripheral targets based on the fidelity of information collected at the targets. The second-stage decisions are scenario-dependent and the second-stage cost is given by the expected additional cost of traversal for the UAV to visit the peripheral targets. The goal of the SVDGP is to find a path that minimizes the sum of the first and second-stage costs.

3. Mathematical Formulation

We remark that when there is no uncertainty associated with the problem and when the information gathered at every target $i \in T$ is already of sufficient fidelity, then no peripheral targets need to be visited by the vehicle and the first-stage problem reduces to a asymmetric TSP on the set T. For the SVDGP, if the information associated with a target $i \in T$ is not of sufficient fidelity, the UAV needs to visits the peripheral targets in the set S_i before proceeding towards the next target. To formulate this problem as a two-stage stochastic program, we first define two edge sets E^1 and E^2 for the first and the second-stages respectively. The edge set E^1 includes edges that are permissible for the vehicle in the first-stage i.e., any edge between every pair of vertices in the set T. The second-stage edge set E^2 includes the edges that the vehicle is permissible to traverse in the second stage i.e., apart from containing all the edges in the set E^1, it includes the edges from every target $i \in T$ to every peripheral target in the set S_i, from every peripheral target in the set $\cup_i S_i$ to every target in the set T and for every $i \in T$, between any pair of peripheral targets in the set S_i. For any edge in the set $(i,j) \in E^1 \cup E^2$, c_{ij} denotes the cost of traversal of that edge. Also, given a subset of vertices $\hat{V}$, we define $\delta^1_+(\hat{V}) = \{(i,j) \in E^1 : j \notin \hat{V} \text{ and } i \in \hat{V}\}$ and $\delta^2_+(\hat{V}) = \{(i,j) \in E^2 : j \notin \hat{V} \text{ and } i \in \hat{V}\}$ as the set of outgoing edges in the first and second stage edge set, respectively. Similarly, given $\hat{V} \subset V$, we let $\delta^1_-(\hat{V}) = \{(j,i) \in E^1 : j \notin \hat{V} \text{ and } i \in \hat{V}\}$ and $\delta^2_-(\hat{V}) = \{(j,i) \in E^2 : j \notin \hat{V} \text{ and } i \in \hat{V}\}$ as the set of incoming edge. Furthermore, when $\hat{V} = \{i\}$ i.e., a singleton, we simply write $\delta^1_+(i)$ instead of to $\delta^1_+(\{i\})$. Finally, given two disjoint subsets of vertices $V_1, V_2 \subset V$, we define $\gamma(V_1 \to V_2) = \{(i,j) \in E^2 : i \in V_1, j \in V_2\}$.

3.1. Objective function

Given the above notations, we introduce binary first-stage decision variables x_{ij} for each $(i,j) \in E^1$, denoting the presence of the edge (i,j) in the first-stage solution and another set of binary second-stage decision variables y_{ij} for each $(i,j) \in E^2$ denoting the presence of the edge (i,j) in the second stage. We let $\boldsymbol{x}$ and $\boldsymbol{y}$ denote the vector of first and second stage decision variables, respectively. Finally, given a subset of edges $\hat{E}^1 \subset E^1$ and $\hat{E}^2 \subset E^2$, we let $x(\hat{E}^1)$ and $y(\hat{E}^2)$ denote the sums $\sum_{(i,j)\in\hat{E}^1} x_{ij}$ and $\sum_{(i,j)\in\hat{E}^2} y_{ij}$, respectively. Now, the objective for the two-stage stochastic programming formulation for the SVDGP is given by:

$$\min \; C, \text{ where } C \triangleq \sum_{(i,j)\in E^1} c_{ij}x_{ij} + \mathbb{E}_\Omega\left[\beta(\boldsymbol{x},\omega)\right] \quad (2)$$
$$= \sum_{(i,j)\in E^1} c_{ij}x_{ij} + \sum_{\omega\in\Omega} p_\omega \beta(\boldsymbol{x},\omega)$$

Here, $\beta(\boldsymbol{x},\omega)$ is the additional traversal cost required to visit the peripheral targets given a scenario $\omega \in \Omega$.

3.2. First-stage constraints

Since the first-stage solution is a feasible tour through the set of targets in T, the constraints in this stage are the TSP constraints which are given below:

$$x(\delta^1_+(i)) = 1 \quad \forall i \in T, \quad (3a)$$
$$x(\delta^1_-(i)) = 1 \quad \forall i \in T, \quad (3b)$$
$$x(\delta^1_+(S)) \geqslant 1 \quad \forall S \subset T, \; |S| \geqslant 2, \text{ and} \quad (3c)$$
$$x_{ij} \in \{0,1\} \quad \forall (i,j) \in E^1. \quad (3d)$$

Here, Eqs. (3a) and (3b) are the in-degree and out-degree constraints that enforce exactly one edge to enter and leave each target in the set T. Eq. (3c) ensures that there are no sub-tours in the solution, and finally Eq. (3d) enforces the binary restrictions on the decision variables x_{ij}.

3.3. Second-stage formulation

The second-stage model for a fixed first-stage solution $\boldsymbol{x}$ and the realization of uncertainty $\omega \in \Omega$ is as follows:

$$\beta(\boldsymbol{x},\omega) = \min \sum_{(i,j)\in E^2} c_{ij}y_{ij} - \sum_{(i,j)\in E^1} c_{ij}y_{ij}(1-\omega_i) \quad (4a)$$

subject to:

$$y(\delta^2_+(i)) = 1 \quad \forall i \in T, \quad (4b)$$
$$y(\delta^2_-(i)) = 1 \quad \forall i \in T, \quad (4c)$$
$$y(\delta^2_+(j)) = \omega_i \quad \forall j \in S_i, i \in T, \quad (4d)$$
$$y(\delta^2_-(j)) = \omega_i \quad \forall j \in S_i, i \in T \quad (4e)$$
$$y_{ij} = x_{ij}(1-\omega_i) \quad \forall (i,j) \in E^1 \quad (4f)$$
$$y(\gamma(S_i \to \{j\})) = x_{ij}\omega_i \quad \forall i,j \in T, \quad (4g)$$
$$y(\delta^2_+(S)) \geqslant 1 \quad \forall S \subset V, \; |S| \geqslant 2, S \cap T \neq \emptyset \text{ and} \quad (4h)$$
$$y_{ij} \in \{0,1\} \quad \forall (i,j) \in E^2. \quad (4i)$$

The objective in Eq. (4a) minimizes the additional cost of visiting the peripheral targets if information collected at a particular target is not of sufficient fidelity. The first summation is the total cost of the route including the additional peripheral target visits and the second term is the cost of the edges that are present both in the first and the second stage solutions. The Eqs. (4b) and (4c) are the in-degree and out-degree constraints for the targets in the set T. The Eqs. (4d) and (4e) ensure that if the information collected at any target $i \in T$ is not of sufficient fidelity i.e., $\omega_i = 1$, then the peripheral targets in the set S_i have to be visited by the vehicle. The Eq. (4f) ensures that if an edge $(i, j) \in E^1$ is traversed by the vehicle in the first stage and if $\omega_i = 0$, then this edge is necessarily used in the second stage by setting $y_{ij} = 1$ and if an edge $(i, j) \in E^1$ is not used by the vehicle in the first stage, then it is also not used in the second stage. Eq. (4g) ensures that given a target $i \in T$ with $\omega_i = 1$ and given that $j \in T$ is the target that vehicle visits immediately after i in the first stage solution, the vehicle has to collect additional information from all the peripheral targets in the set S_i before visiting the next target j. Together, Eqs. (4f) and (4g) also ensure that the sequence in which the vehicle visits the targets in the first-stage solution given by $\boldsymbol{x}$ remains unchanged in the second-stage. The Eq. (4h) eliminate sub-tours in the second-stage solution and finally, the Eq. (4i) impose binary restrictions on the decision variables y_{ij}.

4. Solution Methodology

This section details the description of our approach to solve the two-stage mathematical formulation of the SVDGP presented in Sec. 3. The formulation, as presented, contains binary decision variables both in the first and the second stages. We leverage the Progressive hedging (PH) algorithm proposed by Rockafellar and Wets [23] to solve the SVDGP; PH is also referred to as scenario decomposition in the literature [22] since it decomposes stochastic programs by scenarios i.e., samples of the realization of the uncertainty in the problem. PH is a well-known algorithm for solving multistage stochastic convex optimization problems [24]. In fact, PH possesses strong theoretical convergence properties when all decision variables in the convex multistage stochastic program are continuous. In the presence of discrete variables, a wealth of recent theoretical and empirical research [25, 26, 27] has shown that the PH algorithm can prove to be a very robust heuristic to solve stochastic programs, specifically the case of pure binary programs in both stages of the formulation even in the case that the number of scenarios is prohibitively large. These algorithmic features of PH make its application ideal for the the SVDGP. To that end, the subsequent sections detail the techniques involved in the PH approach and present findings that shows its impact as an effective heuristic to solve the practically relevant

SVDGP. In the forthcoming paragraphs, we present an overview of the PH algorithm specifically for the SVDGP.

4.1. Algorithm Overview

Before, we present the overview of the algorithm, we restate the two-stage stochastic program for the SVDGP in a concise manner as follows:

$$\min \quad \sum_{(i,j)\in E^1} c_{ij}x_{ij} + \sum_{\omega\in\Omega} p_\omega \beta(\boldsymbol{x},\omega) \tag{5a}$$

$$\text{subject to:} \quad \boldsymbol{x} \in \mathcal{Q} \tag{5b}$$

where, $\mathcal{Q} = \{\boldsymbol{x} : \boldsymbol{x} \text{ satisfies Eq. (3)}\}$. The concise formulation, as presented in Eq. (5), is the well-known extensive form of the two-stage stochastic program [28] in Sec. 3. The PH is a scenario-decomposition algorithm that uses a separate set of first-stage decision variables for each scenario to perform parallel solves i.e., for each $\omega \in \Omega$, it introduces decision variables $\boldsymbol{x}_\omega$ and implicitly enforces the non-anticipativity constraints ($\boldsymbol{x} = \boldsymbol{x}_\omega \ \ \forall \omega \in \Omega$) via penalization; these constraints avoid allowing the first-stage decision vector $\boldsymbol{x}$ to depend on the scenario. The basic PH algorithm takes as input two parameters (i) a penalty factor, $\rho > 0$ and (ii) a termination threshold, ε. Given ρ and ε, the pseudo-code for the PH algorithm is as follows:

In Step 6 of Algorithm 1, the function $\langle \cdot, \cdot \rangle$ denotes the dot product of the two vectors. Furthermore, we notice that the computations in Steps 2 and 6 involve solving multiple mixed-integer linear programs, one for each scenario in the set Ω and that the solves are completely parallelizable. The quadratic penalty term (or the proximal term [22]) penalizes the non-anticipativity constraints and ensures that it is satisfied as the algorithm terminates. For every iteration k, the term ϵ^k is also refered to as the "residual". We used a constant value of ρ equal to 0.50 and a value of epsilon was set to 1×10^{-5} for all the computational experiments. These values were chosen by trial-and-error. No further experiments were done to compute an effective ρ value or establish termination criterion as a way to speed up the algorithm once preliminary results showed that the algorithm ran and converged successfully with the chosen values of ρ and ϵ for our particular problem. The following sections show the results obtained by adopting this approach to our two stage formulation of the problem.

5. Computational Results

The PH algorithm for the SVDGP was implemented using the C++ programming language, and CPLEX 12.8 was used as the underlying solver to solve the multiple MILPs that occur in each iteration of the PH algorithm. All the computational experiments were performed using an 2.9 GHz Intel Core

Algorithm 1 Progressive Hedging: A pseudo-code

Initialization:

1: $k \leftarrow 0$ ▷ iteration count

2: For every $\omega \in \Omega$, $\boldsymbol{x}_\omega^k \leftarrow \operatorname{argmin}_{\boldsymbol{x} \in \mathcal{Q}} \sum_{(i,j) \in E^1} c_{ij} x_{ij} + \beta(\boldsymbol{x}, \omega)$

3: $\boldsymbol{x}^k \leftarrow \sum_{\omega \in \Omega} p_\omega \boldsymbol{x}_\omega^k$

4: For every $\omega \in \Omega$, $\boldsymbol{w}_\omega^k \triangleq \rho \cdot (\boldsymbol{x}_\omega^k - \boldsymbol{x}_\omega)$ ▷ $\boldsymbol{w}_\omega^k$ is a weight vector

Iteration update:

5: $k \leftarrow k + 1$

Decomposition:

6: For every $\omega \in \Omega$,

$$\boldsymbol{x}_\omega^k \leftarrow \operatorname*{argmin}_{\boldsymbol{x} \in \mathcal{Q}} \left(\sum_{(i,j) \in E^1} c_{ij} x_{ij} + \langle \boldsymbol{w}_\omega^{k-1}, \boldsymbol{x} \rangle + \frac{\rho}{2} \left\| \boldsymbol{x} - \boldsymbol{x}^{k-1} \right\|^2 + \beta(\boldsymbol{x}, \omega) \right)$$

Aggregation:

7: $\boldsymbol{x}^k \leftarrow \sum_{\omega \in \Omega} p_\omega \boldsymbol{x}_\omega^k$

Weight update:

8: For every $\omega \in \Omega$, $\boldsymbol{w}_\omega^k \triangleq \boldsymbol{w}_\omega^{k-1} + \rho \cdot (\boldsymbol{x}_\omega^k - \boldsymbol{x}_\omega)$

Termination criterion check:

9: $\epsilon^k \triangleq \sum_{\omega \in \Omega} p_\omega \left\| \boldsymbol{x}_\omega^k - \boldsymbol{x}^k \right\|$

10: **if** $\epsilon^k > \varepsilon$ **then**

11: Go to Step 5

12: **else**

13: Terminate with $\boldsymbol{x}^k$ as the first-stage solution

14: **end if**

i7 processor with 16 GB RAM. The performance of the algorithm was tested on randomly generated test instances. The instances generation procedure is detailed in the subsequent section.

5.1. Instance Generation

All the targets, the satellite locations, the source, and the destination were all generated on a 100×100 grid. The source and the destination vertices for all the instances were located at $(5, 5)$ and $(95, 95)$, respectively. The number of targets n were varied from 10 to 40 in steps of 5. For each target, the locations of its corresponding satellites were also randomly generated within a maximum pre-specified radius, R, from the target location; the value of R was chosen from the set $\{5, 10\}$ units. The number of satellite locations per target, m was chosen from the set $\{3, 5, 7, 9\}$. Now, as for the vehicle itself, we assume that the vehicle is a fixed-wing drone with a minimum turn-radius of 5 units. For this fixed-wing drone, the cost of traversal between any pair of targets is assumed to be the shortest the path taken by the vehicle to go from one target to the other. This path length is in-turn a function of the heading angle of the drone at each target. To that end, we generate random heading angles for every target and its corresponding satellite location and then compute the length using the well-known result by Dubins [29]. We remark that the shortest path computed using the result in [29] is asymmetric. Given this instance generation procedure, the total number of instances was 56 and all computational experiments were performed on this set of 56 instances. A subset of these instances are used for each of the computational experiments and the subset of chosen instances for each experiment is presented in the respective sections.

5.2. PH Algorithm Parameters and Scenario Generation

The PH algorithm has two main input parameters, ρ and ϵ, whose values are set to 0.5 and 1×10^{-5}, respectively. The uncertain scenarios for each run of the problem are generated using a vector of Bernoulli random variable, one for each target. Across a majority of the experiments in the subsequent sections, the probability that the information collected at a particular target is of sufficient fidelity is assumed to be 0.5 for every target. For a few other computation experiments other probability values are used and these values are presented when the corresponding experiment is described. For the source and the destination, we set this probability to 0 and required number of scenarios are generated from this vector of Bernoulli random variables. The number of scenarios was varied from a minimum value of 10 to a maximum value of 200 across all experiments. Each randomly generated scenario, ω is a binary vector indicating if the information collected at a particular target is of sufficient fidelity or not. In the subsequent paragraphs, we present the results for the various computational experiments perform to evaluate the PH algorithm on the SVDGP. Before we present the results obtained using the PH algorithm, we remark that when the full two-stage stochastic formulation in Sec. 3 was provided to CPLEX with a time-limit of three hours, all problem instances with number of targets greater than 50 and number of scenarios greater than 20 timed-out. Hence, we do not present any results that show that CPLEX was not able to solve the full problem as stated in Sec. 3.

Effect of increasing the number of scenarios. Here, we present the first set of results to demonstrate the scalability of the PH algorithm for the SVDGP with increasing number of scenarios. To that end, for this experiment, the instances with value of $n \in \{10, 15, 20, 25, 30, 35, 40\}$, $m = 5$, $R = 5$ are chosen. For each of these instances, the number of scenarios are varied in the set $|\Omega| \in \{10, 25, 50, 100, 200\}$. The results for this set of experiments is shown in Tables 1 and 2. It is observed from the tables that the PH algorithm is effectively able to solve all the instances with up to 100 scenarios within two hours of computation time.

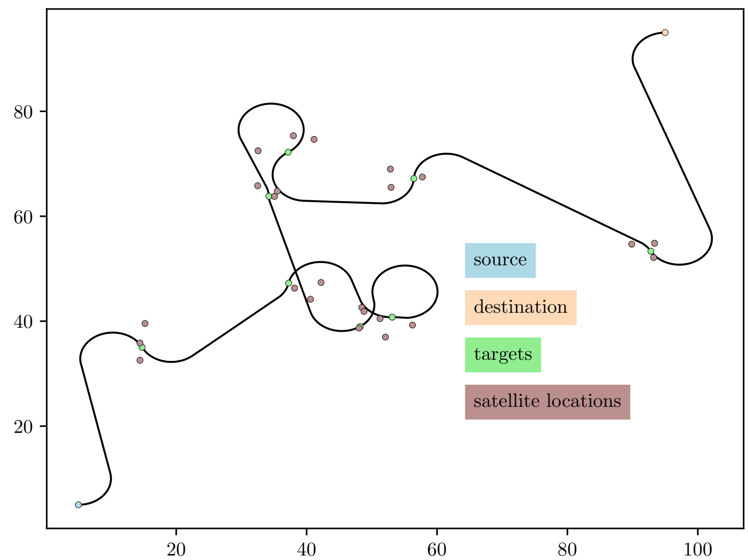
$\|\Omega\|$	$n = 10$	$n = 15$	$n = 20$	$n = 25$	$n = 30$	$n = 35$	$n = 40$
10	0.24	1.05	4.06	281.30	60.40	13.99	351.94
25	0.59	2.71	12.99	564.80	134.28	47.31	1084.65
50	0.95	4.85	21.86	1766.95	262.78	95.25	3121.86
100	2.02	9.94	46.57	5025.31	598.94	191.47	7256.49
200	4.28	19.07	57.45	9304.92	1133.70	336.65	16315.99

Table 1: Computation time (in seconds) taken by the PH algorithm for increasing number of targets and number of scenarios.

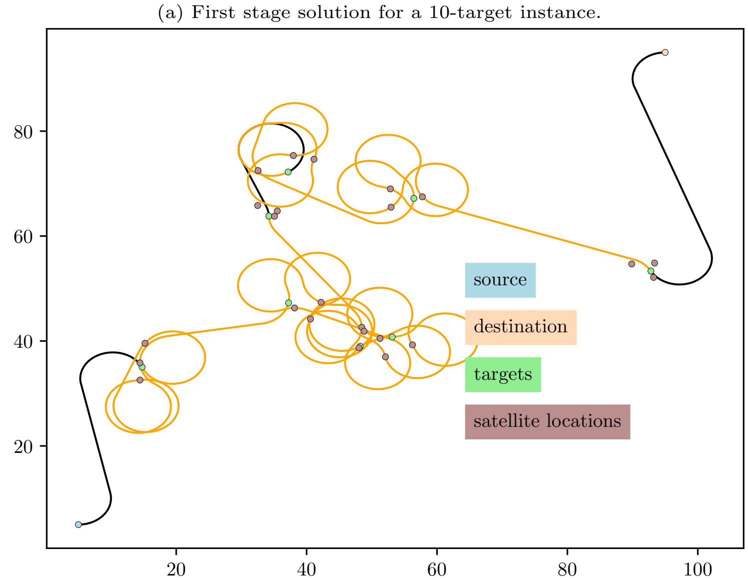
$\|\Omega\|$	$n = 10$	$n = 15$	$n = 20$	$n = 25$	$n = 30$	$n = 35$	$n = 40$
10	1	1	1	16	9	1	14
25	1	1	1	14	10	1	15
50	1	1	2	26	11	1	22
100	1	1	1	34	12	1	23
200	1	1	1	30	12	1	25

Table 2: Number of iterations taken to obtain convergence of the PH algorithm for increasing number of targets and number of scenarios.

The Fig. 2 shows the feasible first stage (Fig. 1a) and a second stage solution (Fig. 1b) for one particular realization of the random variable for a



(a) First stage solution for a 10-target instance.



(b) The solution to the two-stage problem for one particular realization of the uncertainty. The black lines are the path that is same as the first stage solution and the orange routes correspond to the additional visits to the satellite locations corresponding to the targets where the fidelity of information collected was not sufficient.

Figure 1: The solution provided by the PH algorithm for a 10-target instance.

10-target instance. Notice that the paths taken by the vehicle are not straight lines because we assume that the vehicle is a fixed-wing aircraft with a minimum turn-radius of 5 units.

Effect of increasing the number of satellite locations per target. The second set of results are aimed at demonstrating the scalability of the algorithm with increasing number of satellite locations per target. To that end, for this experiment, the instances with value of $n \in \{10, 15, 20, 25, 30, 35, 40\}$, $m = \{3, 5, 7, 9\}$, $R = 5$, and $|\Omega| = 100$ (generated using a probability of 0.5) are chosen and the results are reported in Tables 3 and 4. Though the expectation is for the computation time and the number of iterations to increase with increasing value of m, it is not really the case for this problem, as observed from Tables 3 and 4. Furthermore, we also observe that for the 40-target instances, the computation time is much greater than two hours.

m	$n = 10$	$n = 15$	$n = 20$	$n = 25$	$n = 30$	$n = 35$	$n = 40$
3	1.68	6.34	21.90	185.49	92.15	66.25	2777.89
5	2.02	9.94	46.57	5025.31	598.94	191.47	7256.49
7	5.82	33.55	63.68	162.63	1605.04	6082.88	9404.69
9	22.38	57.53	870.38	1422.58	412.64	7152.56	2325.88

Table 3: Computation time (in seconds) taken by the PH algorithm for increasing number of targets and number of satellite locations per target.

m	$n = 10$	$n = 15$	$n = 20$	$n = 25$	$n = 30$	$n = 35$	$n = 40$
3	1	1	1	11	1	1	20
5	1	1	2	34	12	1	23
7	1	1	1	1	7	19	20
9	1	1	6	6	1	8	1

Table 4: Number of iterations taken to obtain convergence of the PH algorithm for increasing number of targets and number of satellite locations per target.

Effect of increasing the pre-specifed radius for the satellite locations. This set of results presents the variation in the computation time and the number of iterations for the PH algorithm to converge when the pre-specified radius in which the satellite locations are present is varied in the set $\{5, 10\}$. The results present the change in computation time and number of iterations for $n = \{10, 15, 20, 25, 30, 35, 40\}$ and when $m = 5$, $|\Omega| = 100$ (generated using a probability of 0.5). The Tables 5 and 6 present the computation times and the number of iterations for the above instances, respectively. Again, no clear trend exists between the value of R and the computation time or the number of iterations taken for the PH algorithm to converge; nevertheless, the trivial trend that can be observed from the two tables is that greater the number of iterations the greater the computation time.

R	$n = 10$	$n = 15$	$n = 20$	$n = 25$	$n = 30$	$n = 35$	$n = 40$
5	2.023	9.943	46.569	5025.308	598.943	191.472	7256.488
10	45.824	762.224	23.985	98.602	736.891	1284.960	2022.578

Table 5: Computation time (in seconds) taken by the PH algorithm for values of R.

R	$n = 10$	$n = 15$	$n = 20$	$n = 25$	$n = 30$	$n = 35$	$n = 40$
5	1	1	2	34	12	1	23
10	13	37	1	1	10	12	7

Table 6: Number of iterations taken by the PH algorithm for varying values of R.

Effect of changing the probabilities in the scenario generation. This set of results is aimed at examining the effect of changing the probabilities that the information collected at any target is not of sufficient fidelity. We choose a specific instance with $n = 20$, $m = 5$, and $R = 5$ units. There is no specific reason for this choice and the results for all the other instances followed the same trend. To that end, the probabilities that we choose to generate 100 scenarios for this particular instance is given by $\{0.0, 0.2, 0.5, 0.8, 1.0\}$. We remark that when the probability is 0.0, it basically means that the information collected at all the targets is of sufficient fidelity and this reduces the SVDGP to computing a traveling salesman tour through the set of targets. On the other hand, if all the probabilities take a value 1.0, then the information collected at every target is not of sufficient fidelity and the SVDGP reduces to a TSP the set of targets and their satellite locations with additional sequencing constraints i.e., the satellite locations have to be visited immediately after their respective target visits. The Table 7 presents both the computation time and the number of iterations for this experiment. The computation time for the case when the probability value is 0.0 is the least as solving a TSP with 20 targets is substantially fast. On the other end of the spectrum, the computation time for case with a probability value of 1.0 is the maximum as it reduces to the problem of solving a TSP with 120 vertices and additional sequencing constraints. Also, for these two cases, the recourse action is obtained directly by solving the TSP or the TSP with additional sequencing constraints and hence, the number of iterations for the PH algorithm to converge for these two cases will always be equal to one.

p	time (seconds)	# iter
0	4.88	1
0.2	39.62	5
0.5	45.79	2
0.8	89.38	1
1	269.26	1

Table 7: Computation time and number of iterations for different values of probabilities. Here p represents the probability that the information collected at the target is not of sufficient fidelity for any target, and # iter is the number of iterations taken by the PH algorithm to converge.

PH algorithm residual statistics. This set of results shows the changes in the residual of the PH algorithm (i.e., ϵ^k in Algorithm 1) for varying number of scenarios. We remark that this residual function is not a strictly decreasing function. For this experiment, the instance with $n = 40$, $m = 5$, and $R = 5$ was chosen. As far as the scenarios are concerned, all of them were created using a probability value of 0.5. The Fig. 2 shows the residual values for varying number of scenarios. All the results presented thus far illustrate the effectiveness of the PH algorithm in computing heuristic solutions to the SVDGP with uncertainty in the fidelity of information collected.

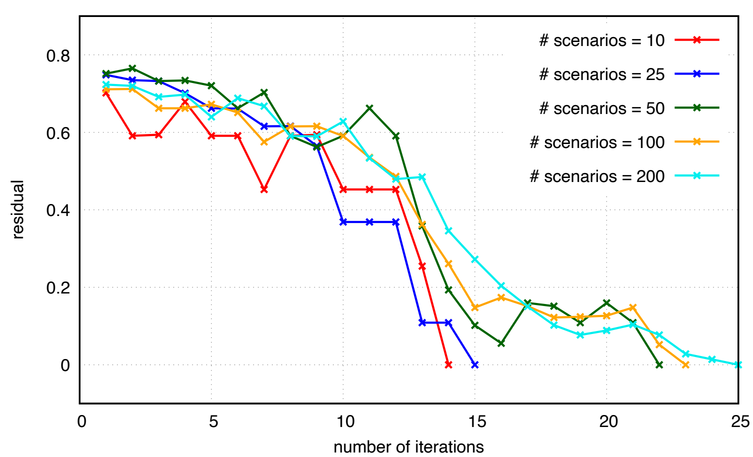


Figure 2: Residual value progress for the PH algorithm. The value of (n, m, R) for the instance for this set of runs is given by $(40, 5, 5)$, respectively.

6. Conclusion

This article introduces a single vehicle routing problem that arises in the context of UAV routing and other vehicle path planning applications. It presents a two-stage stochastic programming formulation and a corresponding algorithm for the problem where there might be a need to visit additional peripheral locations to collect information of good fidelity for the purposes of any particular mission to account for the uncertainty of the information collected at any particular target. This problem can be extended to a multi-stage setting using the same framework. This is the first attempt to solve such a sequential stochastic programming problem in this context and there is scope to work on this further to develop and refine distributed computing algorithms to solve it for a multi-stage setting while reducing the computational times. In addition, this could be easily extended to the case of a multi-vehicle setting as well. Finally, the use of Markov Decision Processes could also be investigated in developing powerful heuristics for this problem.

References

[1] J. Curry, J. Maslanik, G. Holland, J. Pinto, Applications of Aerosondes in the Arctic, Bulletin of the American Meteorological Society 85 (2004) 1855–1862.

[2] C. Corrigan, G. Roberts, M. Ramana, D. Kim, V. Ramanathan, Capturing vertical profiles of aerosols and black carbon over the Indian Ocean using autonomous unmanned aerial vehicles, Atmospheric Chemistry and Physics Discussions 7 (2007) 11429–11463.

[3] P. Tokekar, J. Vander Hook, D. Mulla, V. Isler, Sensor planning for a symbiotic UAV and UGV system for precision agriculture, IEEE Transactions on Robotics 32 (2016) 1498–1511.

[4] J. F. Shanahan, J. S. Schepers, D. D. Francis, G. E. Varvel, W. W. Wilhelm, J. M. Tringe, M. R. Schlemmer, D. J. Major, Use of remote-sensing imagery to estimate corn grain yield, Agronomy Journal 93 (2001) 583–589.

[5] D. W. Casbeer, R. W. Beard, T. W. McLain, S.-M. Li, R. K. Mehra, Forest fire monitoring with multiple small UAVs, in: Proceedings of the 2005, American Control Conference, 2005., IEEE, 2005, pp. 3530–3535.

[6] H. Ferreira, C. Almeida, A. Martins, J. Almeida, N. Dias, A. Dias, E. Silva, Autonomous bathymetry for risk assessment with ROAZ robotic surface vehicle, in: Oceans 2009-Europe, IEEE, 2009, pp. 1–6.

[7] T. Zajkowski, S. Dunagan, J. Eilers, Small UAS communications mission, 2006. URL: https://www.fs.fed.us/eng/rsac/RS2006/presentations/zajkowski2.pdf.

[8] S. G. Manyam, K. Sundar, D. W. Casbeer, Cooperative routing for an air-ground vehicle team–exact algorithm, transformation method, and heuristics, IEEE Transactions on Automation Science and Engineering (2019).

[9] K. Sundar, S. Venkatachalam, S. G. Manyam, Path planning for multiple heterogeneous unmanned vehicles with uncertain service times, in: Unmanned Aircraft Systems (ICUAS), 2017 International Conference on, IEEE, 2017, pp. 480–487.

[10] T. Shima, S. Rasmussen, UAV cooperative decision and control: challenges and practical approaches, SIAM, 2009.

[11] Y. Liu, Z. Liu, J. Shi, G. Wu, C. Chen, Optimization of Base Location and Patrol Routes for Unmanned Aerial Vehicles in Border Intelligence, Surveillance, and Reconnaissance, Journal of Advanced Transportation 2019 (2019).

[12] S. M. Adams, C. J. Friedland, A survey of unmanned aerial vehicle (UAV) usage for imagery collection in disaster research and management, in: 9th International Workshop on Remote Sensing for Disaster Response, 2011, p. 8.

[13] S. Rajan, K. Sundar, N. Gautam, Routing problems for reconnaissance patrolling missions, in: 2019 International Conference on Unmanned Aircraft Systems (ICUAS), IEEE, 2019, pp. 213–220.

[14] K. Sundar, S. Rathinam, Algorithms for routing an unmanned aerial vehicle in the presence of refueling depots, IEEE Transactions on Automation Science and Engineering 11 (2013) 287–294.

[15] K. Sundar, S. Rathinam, Algorithms for heterogeneous, multiple depot, multiple unmanned vehicle path planning problems, Journal of Intelligent & Robotic Systems 88 (2017) 513–526.

[16] D. Levy, K. Sundar, S. Rathinam, Heuristics for routing heterogeneous unmanned vehicles with fuel constraints, Mathematical Problems in Engineering 2014 (2014).

[17] A. Otto, N. Agatz, J. Campbell, B. Golden, E. Pesch, Optimization approaches for civil applications of unmanned aerial vehicles (uavs) or aerial drones: A survey, Networks 72 (2018) 411–458.

[18] S. M. Shavarani, M. G. Nejad, F. Rismanchian, G. Izbirak, Application of hierarchical facility location problem for optimization of a drone delivery system: a case study of amazon prime air in the city of san francisco, The International Journal of Advanced Manufacturing Technology 95 (2018) 3141–3153.

[19] M. Torabbeigi, G. J. Lim, S. J. Kim, Drone delivery schedule optimization considering the reliability of drones, in: 2018 International Conference on Unmanned Aircraft Systems (ICUAS), IEEE, 2018, pp. 1048–1053.

[20] S. Venkatachalam, K. Sundar, S. Rathinam, A two-stage approach for routing multiple unmanned aerial vehicles with stochastic fuel consumption, Sensors 18 (2018) 3756.

[21] M. Liu, X. Liu, M. Zhu, F. Zheng, Stochastic drone fleet deployment and planning problem considering multiple-type delivery service, Sustainability 11 (2019) 3871.

[22] J.-P. Watson, D. L. Woodruff, Progressive hedging innovations for a class of stochastic mixed-integer resource allocation problems, Computational Management Science 8 (2011) 355–370.

[23] R. T. Rockafellar, R. J.-B. Wets, Scenarios and policy aggregation in optimization under uncertainty, Mathematics of operations research 16 (1991) 119–147.

[24] S. Atakan, S. Sen, A progressive hedging based branch-and-bound algorithm for mixed-integer stochastic programs, Computational Management Science 15 (2018) 501–540.

[25] Y. Fan, C. Liu, Solving stochastic transportation network protection problems using the progressive hedging-based method, Networks and Spatial Economics 10 (2010) 193–208.

[26] O. Listes, R. Dekker, A scenario aggregation–based approach for determining a robust airline fleet composition for dynamic capacity allocation, Transportation Science 39 (2005) 367–382.

[27] A. Løkketangen, D. L. Woodruff, Progressive hedging and tabu search applied to mixed integer (0, 1) multistage stochastic programming, Journal of Heuristics 2 (1996) 111–128.

[28] S. W. Wallace, W. T. Ziemba, Applications of stochastic programming, SIAM, 2005.

[29] L. E. Dubins, On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents, American Journal of mathematics 79 (1957) 497–516.