The Risk-Sensitive Coverage Problem: Multi-Robot Routing Under Uncertainty with Service Level and Survival Constraints

Stefan Jorgensen, Robert H. Chen, Mark B. Milam, and Marco Pavone

***Abstract*— Consider a scenario where robots traverse a graph, but crossing each edge bears a risk of failure. A team operator seeks a set of paths for the smallest team which guarantee the probabilities that at least one robot visits each node satisfies specified per-node visit thresholds, and the probabilities each robot reaches its destination satisfy a per-robot survival threshold. We present the Risk-Sensitive Coverage (RSC) problem formally as an instance of the submodular set cover problem and propose an efficient cost-benefit greedy algorithm for finding a feasible set of paths. We prove that the number of robots deployed by our algorithm is no more than $\frac{\lambda}{p_s}(1+\log(\lambda\Delta_K/p_s))$ times the smallest team, where Δ_K quantifies the relative benefit of the first and last paths, p_s is the per-robot survival probability threshold and $1/\lambda \leq 1$ is the approximation factor of an oracle routine for the well-known orienteering problem. We demonstrate the quality of our solutions by comparing to optimal solutions computed for special cases of the RSC and the efficiency of our approach by applying it to a search and rescue scenario where 225 sites must be visited, each with probability at least 0.95.**

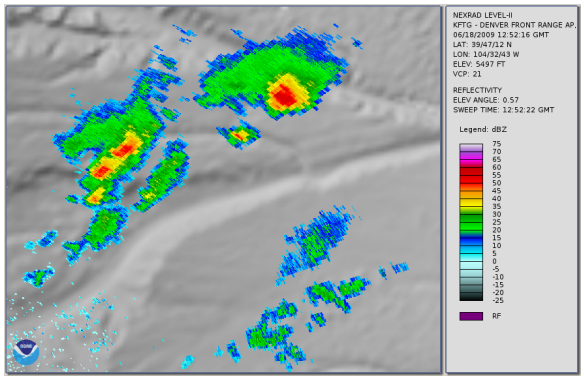


Fig. 1. Illustration of the "base reflectivity" of a storm, which can be used to infer the probability robots survive traversing between sites. Data from NOAA NEXRAD level II dataset, and visualization courtesy the Weather and Climate Toolkit [4].

I. Introduction

Consider a search and rescue mission where a team of robots searches for victims trapped in a storm, and the probability that a robot survives while traversing between two sites (corresponding to likely locations of victims) depends on the weather. The team operator seeks to find a set of paths for the smallest team which guarantees that sites are searched with given probability thresholds, and that the probabilities each robot returns safely satisfies a survival threshold.

We formalize the environment using a graph, where nodes correspond to sites, edges represent the ability to travel between sites, and edge weights correspond to the probability that a robot survives traveling among corresponding sites (see Figure 1). The challenge facing the operator can then be posed as a *set cover* problem, where it must choose the smallest set of paths which satisfy a coverage constraint (e.g. the search/visit probability thresholds). We call our formulation the Risk-Sensitive Coverage (RSC) problem, and while the set cover problem is well understood [1], [2], [3], the RSC problem is more challenging because 1) the ground set of paths is exponentially large in the number of nodes and edges, and 2) the notion of *risky traversal* (where a robot may fail while traversing an edge) precludes many existing approaches used for path planning because it introduces a complex interaction between the edges chosen and the probability a node is visited.

The RSC is the dual to the Team Surviving Orienteers (TSO) problem [5], which has the same risky traversal model but asks the operator to maximize the expected number of sites visited given a fixed team size. An efficient approximate algorithm for the TSO was designed by exploiting a diminishing returns property known as submodularity, which for set functions means that $f(A\cup B)+f(A\cap B) \leq f(A)+f(B)$. Our key insight in this paper is that the visit probability constraints can be represented using a submodular function, which allows us to precisely state the problem as a *submodular set cover* problem, and justifies using a cost-benefit greedy algorithm to find a solution. We develop a linearization procedure which allows us to efficiently find near-optimal paths and derive an approximation guarantee for the resulting algorithm. Although existing work considers the set cover problem with exponentially large ground sets [6] or submodular routing problems [5], [7] separately, our work is novel because we combine the two and give a computationally efficient algorithm with approximation guarantees.

Related work: Given a ground set of items, a cost function, and a submodular coverage function, the *submodular set cover* problem seeks a minimum cost subset which saturates the coverage function. The problem was posed and solved using a cost-benefit greedy algorithm by [1] which iteratively builds a solution by adding the item with the best ratio of benefit to cost. Several problem dependent $1+\log(\Delta)$ approximation guarantees (where Δ is a problem-dependent parameter) were given, which guarantee that the cost of the optimum and approximate solutions are nearly

Stefan Jorgensen is with the Department of Electrical Engineering, Stanford University, Stanford, California 94305. His work is supported by NSF grant DGE-114747 stefantj@stanford.edu Robert H. Chen and Mark B. Milam are with NG Next, Northrop Grumman, Redondo Beach, California 90278 {robert.chen,mark.milam}@ngc.com Marco Pavone is with the Department of Aeronautics & Astronautics, Stanford University, Stanford, California 94035 pavone@stanford.edu

the same. Matching hardness bounds were given by [2] and [1]. A more in-depth treatment of the submodular set cover and submodular knapsack problem is given by [3], which considers enhanced algorithms which exploit the curvature of the submodular coverage function in order to provide refined analysis. The approaches taken by [1], [3] assume that the ground set can be efficiently searched, but for our setting this is not the case. Exponentially large ground sets were considered by [6], who provide strong guarantees for the special case when the submodular function is integral and the ground set is the family of independent sets of a matroid (which extends linear independence to set functions). Their work does not generalize, because it relies on specific results for optimizing over independent sets of matroids and integral submodular set functions. Large ground sets were also considered by [7] from a submodular function maximization perspective, where the goal is to select edges which form a path (rather than to select paths which saturate a submodular function).

The *vehicle routing problem* [8] is a broad class of problems which consider finding routes for a team of vehicles which satisfy quality of service constraints (e.g. visiting customers within time windows, minimum emissions, minimum distance). A setting where travel time between nodes is stochastic and nodes must be visited within certain time windows is considered by [9]. Their approach seeks to ensure that the probability that each node is visited within its time window is above a given threshold, but their algorithm does not explicitly enforce this constraint and in practice many of their simulations do not achieve the desired guarantees. While this setting could be used to model the RSC by setting the deadlines to be the negative logarithm of the survival probability threshold and treating the negative logarithm of the survival probabilities as "time", our approach explicitly enforces the coverage constraint and provides approximation guarantees.

Coverage problems in robotics [10] seek to find a set of routes which guarantee some notion of coverage. Coverage problems are in general NP-hard, since they generalize the orienteering problem [11], which seeks a maximum weight path in a graph that satisfies a budget constraint. The persistent monitoring problem [12] seeks a set of cyclic routes which minimize the time between visits to a node. The authors provide an efficient linear program which can be solved efficiently and give theoretical guarantees concerning the robustness of their solution. Another broad category of coverage problems is the informative path planning problem (IPP), where the goal is to plan paths which gather as much information as possible about an environmental random variable. The IPP is related to our setting because information is also a submodular function, but typically IPP problems are stated as maximization problems rather than coverage problems. The literature on coverage problems is vast, however to our knowledge the risky traversal model, which is integral to this work, is not used by existing work. We note that although [13] proposes a similarly named problem, the notion of "risk" in their work is probability of detection, which does not interact with the coverage constraints in the same manner as in our setting.

Statement of contributions. The contribution of this paper is fourfold. First, we propose the Risk-Sensitive Coverage (RSC) problem and show that it is an instance of the submodular set cover problem. By considering risky traversal, we extend the state of the art for robot coverage problems, and by considering an exponentially large ground set we extend the state of the art in the submodular set cover problem. Second, we provide a linear relaxation which allows us to implement a cost-benefit greedy algorithm efficiently utilizing standard orienteering problem solvers. Third, we provide a bi-criteria approximation guarantee, which ensures that the size of the set output by our routine is close to the optimum for a closely related problem. Specifically, our result states that the solution with L robots is no more than $\frac{\lambda}{p_s}(1+\log(\lambda\Delta_L/p_s))$ times the optimum solution size for a problem satisfying at least L/K fraction of the constraints, where K is the size of the final output of our algorithm, p_s is the per-robot survival probability threshold, $1/\lambda \leq 1$ is the approximation guarantee for an oracle routine which solves the orienteering problem, and Δ_L is the ratio of incremental coverage gain from the first and Lth path planned. Fourth, we demonstrate the quality of the paths selected by our routine by comparing them with optimal paths computed for a pecial case of the RSC – our solution uses at most 33% more robots than the optimum. We then apply our routine to a search and rescue scenario with 225 nodes and visit probability thresholds of 0.95. Our routine finds a set of 36 paths which satisfy the constraints with approximation ratio 9. By relaxing the constraints slightly (80.7% satisfied) we find a set of 13 paths with approximation ratio 4.33.

Organization. In Section II we review key concepts and background information for the RSC. In Section III we state the RSC problem formally, give an example, and highlight variants of the problem. In Section IV we formalize the connection to the submodular set cover problem, describe an approximate greedy algorithm, its complexity, approximation guarantees and variants. We demonstrate the efficiency and applicability of our approach using numerical experiments in Section V, and draw conclusions in Section VI.

II. Background

In this section we review key material for our work and extend a well-known theorem in the combinatorial optimization literature to our setting.

A. Submodularity

Submodularity is the property of 'diminishing returns' for set functions. The following definitions are summarized from [14]. Given a set $\mathcal{X}$, its possible subsets are represented by $2^{\mathcal{X}}$. For two sets X and X', the set $X' \setminus X$ contains all elements in X' but not X. A set function $f : 2^{\mathcal{X}} \to \mathbb{R}$ is said to be *normalized* if $f(\emptyset) = 0$ and to be *monotone* if for every $X \subseteq X' \subseteq \mathcal{X}$, $f(X) \leq f(X')$. A set function $f : 2^{\mathcal{X}} \to \mathbb{R}$ is *submodular* if for every $X \subseteq X' \subset \mathcal{X}$, $x \in \mathcal{X} \setminus X'$, we have

$$f(X \cup \{x\}) - f(X) \geq f(X' \cup \{x\}) - f(X').$$

The quantity on the left hand side is the *discrete derivative* of f at X with respect to x, which we write as $\Delta f(x \mid X)$. If f_1 and f_2 are submodular set functions, then $g_1(X) := f_1(X) + f_2(X)$ and $g_2(X) = \min(f_1(X), f_2(X))$ are also submodular set functions.

B. The Approximate Cost-Benefit Greedy Algorithm

Given a positive cost function, $c : \mathcal{X} \to \mathbb{R}_+$ the cost of a set $X \subseteq \mathcal{X}$ is defined as $C(X) := \sum_{x \in X} c(x)$. Given a submodular function f, the *submodular set cover problem* entails finding a set with minimum cost such that the submodular function is saturated, that is $f(X) = f(\mathcal{X})$. Finding an optimal solution, X^*, to this problem is NP-hard for general submodular functions [1]. The *cost-benefit greedy algorithm* constructs a set $\bar{X}_K = \{\bar{x}_1, \ldots, \bar{x}_K\}$ by iteratively adding an element x which maximizes the ratio of the benefit (the discrete derivative of f at the partial set already selected), to the cost of the element $c(x)$. In other words the ℓth element satisfies:

$$\bar{x}_\ell \in \operatorname*{argmax}_{x \in \mathcal{X} \setminus \bar{X}_{\ell-1}} \frac{\Delta f(x \mid \bar{X}_{\ell-1})}{c(x)}.$$

We refer to the optimization problem above as 'the cost-benefit greedy sub-problem' at step ℓ. Suppose that after K iterations the coverage constraint is satisfied, i.e. $f(\bar{X}_K) = f(\mathcal{X})$. Theorem 1(ii) given by [1] states that if f is a monotone, normalized, non-negative, and submodular set function, then the cost of $\bar{X}_K$ is close to optimal,

$$C(\bar{X}_K) \leq \left(1 + \log\left(\frac{f(\bar{X}_1)c(\bar{x}_K)}{\Delta f(\bar{x}_K \mid \bar{X}_{K-1})c(\bar{x}_1)}\right)\right) C(X^*).$$

This is a powerful result with matching hardness bounds [2], [1], but if the set $\mathcal{X}$ is large we might only be able to *approximately* solve the cost-benefit greedy sub-problem. An α–approximate cost-benefit greedy algorithm constructs a solution $\hat{X}_K$ by iteratively adding elements which approximately maximize the ratio of benefit to cost. In particular for some fixed $\alpha \leq 1$, the ℓth element $\hat{x}_\ell$ satisfies:

$$\frac{\Delta f(\hat{x}_\ell \mid \hat{X}_{\ell-1})}{c(\hat{x}_\ell)} \geq \alpha \frac{\Delta f(x \mid \hat{X}_{\ell-1})}{c(x)} \qquad \forall x \in \mathcal{X} \setminus \hat{X}_{\ell-1}.$$

We extend Theorem 1(ii) given by [1] to the approximate greedy setting with a minor modification to their argument (propagating α throughout):

Theorem 1 (α–approximate cost-benefit greedy [1]): Let $c : \mathcal{X} \to \mathbb{R}_+$ be a positive cost function and f be a monotone, normalized, non-negative, and submodular function with discrete derivative Δf. Then for the output of any α–approximate cost-benefit greedy algorithm with K elements, $\hat{X}_K$, we have the following inequality:

$$C(\hat{X}_K) \leq \frac{1}{\alpha}\left(1 + \log\left(\frac{\frac{1}{\alpha} f(\hat{X}_1)c(\hat{x}_K)}{\Delta f(\hat{x}_K \mid \hat{X}_{K-1})c(\hat{x}_1)}\right)\right) C(X^*).$$

Proof: This result follows immediately from the proof given by [1]. When applying proposition 3 (at the top of page 389 in [1]), one must introduce the α approximation factor. Propagating the factor throughout the rest of the proof yields the result. We note that Theorem 1(i),(iii) can similarly be extended to the approximate setting, but as the results are nearly identical for our purposes, we do not state them. ■

This guarantee is strong if $\alpha \simeq 1$ (meaning that the sub-problem is solved near-optimally), and if $\Delta f(\hat{x}_K \mid \hat{X}_{K-1}) \gg 0$ (meaning that the last element has meaningful contribution to satisfying the coverage constraints). In the special case that the cost is uniform, e.g. $c(x) = c$ then the cost-benefit greedy algorithm is simply referred to as the greedy algorithm, since we only maximize the benefit.

C. Graphs

Let $\mathcal{G}(\mathcal{V}, \mathcal{E})$ denote an undirected graph, where $\mathcal{V}$ is the node set and $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$ is the edge set. Explicitly, an edge is a pair of nodes (i, j), and represents the ability to travel between nodes i and j. If the graph is directed, then the edge is an ordered pair of nodes, and represents the ability to travel from the *source node* i to the *sink node* j. A graph is called *simple* if there is only one edge which connects any given pair of nodes. A path is an ordered sequence of *unique* nodes such that there is an edge between adjacent nodes. For $n \geq 0$, we denote the nth node in path ρ by $\rho(n)$ and the number of edges in ρ by $|\rho|$. Note that $\rho(|\rho|)$ is the last node in path ρ.

III. Problem Statement

We use a similar setting and notation as the TSO problem presented in [5] which we repeat below to keep this paper self-contained. We then give the formal problem statement for the RSC problem, provide an example, and describe applications and variants of the RSC problem.

A. The RSC problem

Let $\mathcal{G}$ be a simple graph with $|\mathcal{V}| = V$ nodes. Edge weights $\omega : \mathcal{E} \to (0, 1]$ correspond to the probability of survival for traversing an edge. Time is discretized into iterations $n = 0, 1, \ldots, N$. At iteration n a robot following path ρ traverses edge $e^n_\rho = (\rho(n-1), \rho(n))$. Robots are indexed by variable k, and for each we define the independent Bernoulli random variables $s^k_n(\rho)$ which are 1 with probability $\omega(e^n_\rho)$ and 0 with probability $1 - \omega(e^n_\rho)$. If robot k follows path ρ, the random variables $a^k_n(\rho) := \prod_{i=1}^n s^k_i(\rho)$ can be interpreted as being 1 if the robot 'survived' all of the edges taken until iteration n and 0 if the robot 'fails' on or before iteration n.

Given a start node v_s, a terminal node v_t, visit probability thresholds $\{p_v(j)\}_{j=1}^V$ and survival probability p_s we must find a set of K paths $\{\rho_k\}_{k=1}^K$ (one for each of K robots) such that, for all k, the probability that $a^k_{|\rho_k|}(\rho_k) = 1$ is at least p_s, $\rho_k(0) = v_s$ and $\rho_k(|\rho_k|) = v_t$. The set of paths which satisfy these constraints is written as $\mathcal{X}(p_s, \omega)$. One can readily test whether $\mathcal{X}(p_s, \omega)$ is empty as follows: Set edge weights as $-\log(\omega(e))$, and for each node j, compute the shortest path from v_s to j, delete the edges and nodes (except node j) in that path, and compute the shortest path from j to v_t. If the sum of edge weights along both paths is less than $-\log(p_s)$ then the node is reachable, otherwise it is not. Using Dijkstra's algorithm this approach can determine whether $\mathcal{X}(p_s, \omega)$ is empty after

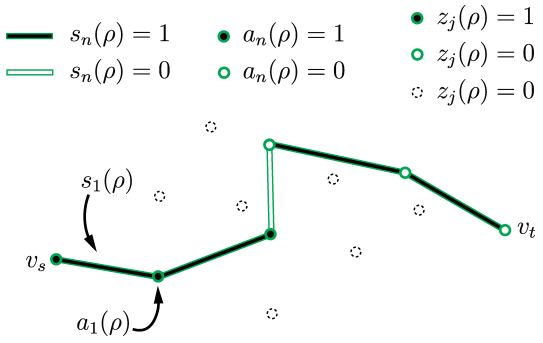


Fig. 2. Illustration of the notation used. Robot k plans to take path ρ, whose edges are represented by lines. The fill of the lines represent the value of $s_n^k(\rho)$. In this example $s_3^k(\rho) = 0$, which means that $a_3^k(\rho) = a_4^k(\rho) = a_5^k(\rho) = 0$. The variables $z_j^k(\rho)$ are zero if either the robot fails before reaching node j or if node j is not on the planned path.

$O(V^2 \log(V))$ computations. From here on we assume that $\mathcal{X}(p_s, \omega)$ is non-empty.

Define the indicator function $\mathbb{I}\{x\}$, which is 1 if x is true (or nonzero) and zero otherwise. Define the Bernoulli random variables for $j = 1, \ldots, V$:

$$z_j^k(\rho) := \sum_{n=1}^{|\rho|} a_n^k(\rho)\, \mathbb{I}\{\rho(n) = j\},$$

which are 1 if robot k following path ρ visits node j and 0 otherwise ($z_j^k(\rho)$ is binary because a path is defined as a unique set of nodes). Note that $z_j^k(\rho)$ is independent of $z_j^{k'}(\rho')$ for $k \neq k'$. The number of times that node j is visited by robots following the paths $\{\rho_k\}_{k=1}^K$ is given by $\sum_{k=1}^K z_j^k(\rho_k)$, and we write the probability that exactly m robots visit node j as $p_j(m, \{\rho_k\}_{k=1}^K)$. In this paper we are primarily interested in the probability that no robots visit node j, which has the simple expression:

$$p_j\left(0, \{\rho_k\}_{k=1}^K\right) = \prod_{k=1}^{K} (1 - \mathbb{E}[z_j^k(\rho_k)]).$$

Given visit probability thresholds $p_v(j) \in [0, 1)$, $j = 1, \ldots, V$ the Risk-Sensitive Coverage problem is formally stated as:

Risk-Sensitive Coverage problem: Given a graph $\mathcal{G}$, edge weights ω, survival probability threshold p_s and visit probability thresholds $\{p_v(j)\}_{j=1}^V$, minimize the team size while satisfying all constraints:

$$\begin{aligned}
\underset{K, \rho_1, \ldots, \rho_K}{\text{minimize}} \quad & K \\
\text{subject to} \quad & \mathbb{P}\left\{a_{|\rho_k|}^k(\rho_k) = 1\right\} \geq p_s && k = 1, \ldots, K \\
& \rho_k(0) = v_s && k = 1, \ldots, K \\
& \rho_k(|\rho_k|) = v_t && k = 1, \ldots, K \\
& 1 - p_j(0, \{\rho_k\}_{k=1}^K) \geq p_v(j) && j = 1, \ldots, V
\end{aligned}$$

The objective is to minimize the team size. The first set of constraints enforces the survival probability, the second and third sets of constraints enforce the initial and final node constraints. The last constraint is the coverage constraint and requires that each node j be visited by at least one agent with probability at least $p_v(j)$.

B. Example

An example of the RSC problem is given in Figure 3(a). There are four nodes and four edges. The survival threshold is $p_s = 0.8$ and the edge weights are all 0.9. There are two feasible paths, $\rho_1 = \{v_s, 1, v_t\}$, $\rho_2 = \{v_s, 2, v_t\}$, and the probability a robot reaches node v_t is 0.81 for either path. For visit probability thresholds $p_v(1) = p_v(2) = 0.9$, it is easy to verify that the optimal solution is $\{\rho_1, \rho_2\}$. For visit probability thresholds $p_v(1) = p_v(2) = 0.99$ the optimal solution is $\{\rho_1, \rho_1, \rho_2, \rho_2\}$.

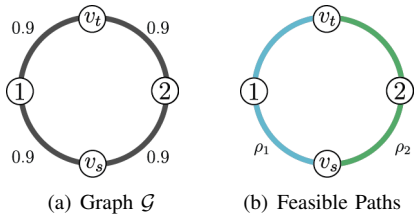
C. Variants

Heterogeneous teams. The RSC is easily extended to a scenario where there are M different types of robots available, each with different costs $c(m)$ and capabilities represented by feasible sets $\mathcal{X}_m$. The objective function is now the sum of the costs of the types of robots selected, and the constraints are now that each path be in the respective feasible set of the robot type. This can model a wide variety of scenarios, for example when some lower cost robots can be exposed to more risk than more expensive variants; or when each robot type can only visit a subset of the nodes in the graph, e.g. in underwater scenarios where nodes may correspond to different depths.

Edge variants. We can extend to an *edge* coverage scenario by re-defining all node variables as edge variables as described in [5]. This is useful for applications such as scientific missions where edges correspond to routes which the robots gather information along, and the objective is to find the smallest team which can complete the mission with the desired coverage.

Multiple visit coverage. We can also consider scenarios where the probability that $\ell_j \geq 1$ robots visit node j must exceed a specified threshold $p_v(j)$. In this setting, the coverage constraints become $1 - \sum_{m=0}^{\ell_j - 1} p_j(m, \{\rho_k\}_{k=1}^K) \geq p_v(j)$ for $j = 1, \ldots, V$.

Arbitrary terminal nodes. We can finally consider the problem where the robots can end at any subset of nodes by placing an edge with weight 1 (meaning survival is certain) between these nodes and node v_t, and removing all other edges which end at v_t. This would model a scenario where there are multiple depots where the robots may end at which are equally desirable.



(a) Graph $\mathcal{G}$ (b) Feasible Paths

Fig. 3. Example of the Risk-Sensitive Coverage problem. (a) The graph has four nodes and four edges. The probability of surviving a given edge is 0.9. (b) For survival probability threshold 0.8, there are two feasible paths from node v_s to v_t: $\rho_1 = \{v_s, 1, v_t\}$ and $\rho_2 = \{1, 2, v_t\}$.

IV. ALGORITHM

Our approach to solving the RSC problem is to exploit the submodularity of the coverage function using an α–approximate cost-benefit greedy algorithm, as described in Section II. In Section IV-A we show how to pose the RSC as a submodular set cover problem, and in Section IV-B we demonstrate properties of the cost-benefit greedy subproblem which lead to an efficient α–approximate cost-benefit greedy algorithm detailed in Section IV-C. We then derive a bi-criteria guarantee for our algorithm in Section IV-D, which ensures that the cost of our solution is close to the optimal cost for a similar problem. Such guarantees are more robust to problem specifications than those presented in [1], as we illustrate with an example. Finally, we discuss the complexity and extensions of our algorithm in Sections IV-E and IV-F, respectively.

A. RSC equivalence to submodular set cover

The RSC is equivalent to the submodular set cover problem with ground set $\mathcal{X}$, which contains finitely many copies of each path in $\mathcal{X}(p_s, \omega)$. A subset $X \subset \mathcal{X}$ automatically satisfies the first three sets of constraints for the RSC (survival probability, start and end nodes). Next we give a submodular function $g : 2^{\mathcal{X}} \to \mathbb{R}_+$ such that $g(X) = g(\mathcal{X})$ if and only if the visit probability constraints are satisfied.

Define the functions f_j for $j = 1, \ldots, V$ to be the minimum of the probability that node j is visited by robots following the paths in X and the visit probability threshold:

$$f_j(X) := \min\left\{1 - p_j(0, X), p_v(j)\right\}.$$

It was shown by [5] that $1 - p_j(0, X)$ is submodular hence f_j is also submodular. Now we define the coverage function,

$$g(X) = \sum_{j=1}^{V} f_j(X),$$

which is also a submodular function of the set of paths. If the constraints are satisfied by $X \subset \mathcal{X}$, then $f_j(X) = p_v(j) = f_j(\mathcal{X})$ which implies $g(\mathcal{X}) = g(X)$. If $g(X) = g(\mathcal{X})$ and there is some subset $Y \subseteq \mathcal{X}$ which satisfies the constraints, then $g(X) = \sum_{j=1}^{V} p_v(j)$, which implies that the set X satisfies the coverage constraints. Hence $g(X) = g(\mathcal{X})$ is equivalent to saying that the set X satisfies the coverage constraints, so the RSC problem is equivalent to

$$\begin{aligned} \underset{X \subseteq \mathcal{X}}{\text{minimize}} \quad & |X| \\ \text{subject to} \quad & g\left(X\right) = g(\mathcal{X}) \end{aligned}$$

which is a submodular set cover problem with domain $\mathcal{X}$, unit cost function $c(\cdot) = 1$, and coverage function g. Motivated by the strong performance of the cost-benefit greedy algorithm for the submodular set cover problem [1], we proceed to design one in the next section.

B. Solving the cost-benefit greedy subproblem

Given a previously selected set of paths $\hat{X}_{L-1}$, the cost-benefit greedy subproblem for the RSC problem requires us to find a path $\hat{\rho}_L$ which maximizes the discrete derivative of the coverage function at $\hat{X}_{L-1}$ with respect to $\hat{\rho}_L$ (since the cost is unit, we do not need to consider the ratio of benefit to cost).

By construction, the set $\mathcal{X}$ has (finitely) many copies of each path in $\mathcal{X}(p_s, \omega)$, so $\mathcal{X} \setminus \hat{X}_{L-1}$ always contains at least one copy of each feasible path. Since the discrete derivative of g is identical for each copy we can reduce the search domain to paths in $\mathcal{X}(p_s, \omega)$. This means the sub-problem involves finding a path in a graph which maximizes a submodular function and satisfies a budget constraint, which is an instance of the submodular orienteering problem. The submodular orienteering problem is not approximable within a constant factor for general submodular functions [15], so we design one specifically for the cost-benefit greedy sub-problem of the RSC using a similar approach as [5].

The key idea is to relax the problem of maximizing the discrete derivative by replacing the probability that robot L traversing path ρ visits node j with the maximum probability that any robot following a feasible path can visit node j, ζ_j:

$$\zeta_j := \max_{\rho \in \mathcal{X}(p_s, \omega)} \mathbb{E}[z_j^L(\rho)].$$

For a given graph this upper bound can be found easily by using Dijkstra's algorithm with log transformed edge weights $\omega_O(e) := -\log(\omega(e))$. Next we show how use ζ_j to replace the path *dependent* quantity Δg with a good path *independent* approximation, which allows us to use standard orienteering problem solvers.

The discrete derivative of g is the sum of the discrete derivatives of f_j, which are piecewise linear with behavior changing depending on two conditions. The first condition, $C_{1,j}(X)$, is whether sending robots on each of the paths in X satisfies the coverage constraint for node j. The second condition, $C_{2,j}(\phi, X)$, is whether sending robots down the paths in X *in addition* to a robot which visits node j with probability ϕ will satisfy the coverage constraint for node j. Formally,

$$\begin{aligned} C_{1,j}(X) &= \mathbb{I}\{1 - p_j(0, X) \geq p_v(j)\}, \\ C_{2,j}(\phi, X) &= \mathbb{I}\{1 - p_j(0, X)(1 - \phi) \geq p_v(j)\}. \end{aligned}$$

Note that $C_{1,j}(X) = 1$ implies that $C_{2,j}(\phi, X) = 1$ for any $\phi \in [0, 1]$. Define the function $\delta_j(\phi, X)$ as

$$\delta_j(\phi, X) = \begin{cases} 0, \text{ if } C_{1,j}(X) = 1 \\ \phi\, p_j(0, X), \text{ if } C_{1,j}(X) = 0,\ C_{2,j}(\phi, X) = 0 \\ p_v(j) - (1 - p_j(0, X)), \text{ otherwise.} \end{cases}$$

It is easy to verify that $\Delta f_j(\rho \mid X) = \delta_j(\mathbb{E}[z_j(\rho)], X)$, and if $p < p'$ then $\delta(p, X) \leq \delta(p', X)$. Let $\mathbb{I}_j(\rho)$ be equal to 1 if node j is in ρ and 0 otherwise. From the definition of δ_j and feasibility of ρ, we have the following inequalities:

$$\mathbb{I}_j(\rho) p_s \delta_j(\zeta_j, X) \leq \Delta f_j(\rho \mid X) \leq \mathbb{I}_j(\rho) \delta_j(\zeta_j, X).$$

This means that the path *independent* quantity $\delta_j(\zeta_j, X)$ is a good characterization of the path *dependent* quantity $\Delta f_j(\rho \mid X)$, especially when p_s is close to unity. We form our approximate greedy algorithm by using this path independent approximation, where we are looking to maximize the sum:

$$\Delta \bar{g}(\rho \mid \hat{X}_{L-1}) := \sum_{j=1}^{V} \mathbb{I}_j(\rho) \delta_j\left(\zeta_j, \hat{X}_{L-1}\right),$$

which represents an *optimistic* estimate of the discrete derivative of the coverage function at $\hat{X}_{L-1}$ with respect to ρ.

We can find the (approximately) best path by solving an orienteering problem on the graph $\mathcal{G}_O$, which has the same edges and nodes as $\mathcal{G}$ but has edge weights $\omega_O(e)$ and node rewards $\nu_L(j) = \delta_j(\zeta_j, \hat{X}_{L-1})$. Solving the orienteering problem on $\mathcal{G}_O$ with budget $-\log(p_s)$ will return a path that maximizes the sum of node rewards (which is $\Delta\bar{g}(\rho \mid X_{L-1})$), and satisfies $\sum_{e\in\rho} -\log(\omega(e)) \leq -\log(p_s)$, which is equivalent to $\mathbb{P}\{a^L_{|\rho|}(\rho) = 1\} \geq p_s$. Although solving the orienteering problem is NP-hard, several polynomial-time constant-factor approximation algorithms exist which guarantee that the returned objective is lower bounded by a factor of $1/\lambda \leq 1$ of the optimal objective. For undirected planar graphs [16] gives a guarantee $\lambda = (1+\epsilon)$, for undirected graphs [17] gives a guarantee $\lambda = (2+\epsilon)$, and for directed graphs [15] gives a guarantee in terms of the number of nodes. Using such an oracle, we have the following guarantee:

Lemma 1 (Single robot constant-factor guarantee): Suppose we are given a set of paths $X \subset \mathcal{X}$. Let `Orienteering` be a routine that solves the orienteering problem within constant-factor $1/\lambda$, that is for node weights $\nu(j) = \delta_j(\zeta_j, X)$, path $\hat{\rho}$ output by the routine, and any path $\rho \in \mathcal{X}(p_s, \omega)$,

$$\sum_{j=1}^{V} \mathbb{I}_j(\hat{\rho})\nu(j) \geq \frac{1}{\lambda}\sum_{j=1}^{V} \mathbb{I}_j(\rho)\nu(j).$$

Then for any $\rho \in \mathcal{X}(p_s, \omega)$ we have

$$\Delta g(\hat{\rho} \mid X) \geq \frac{p_s}{\lambda}\Delta g(\rho \mid X)$$

Proof: We have by the properties of δ_j and the `Orienteering` routine,

$$\Delta g(\rho \mid X) := \sum_{j=1}^{V} \Delta f_j(\rho \mid X) \leq \sum_{j=1}^{V} \mathbb{I}_j(\rho)\delta_j(\zeta_j, X)$$

$$\leq \lambda \sum_{j=1}^{V} \mathbb{I}_j(\hat{\rho})\delta_j(\zeta_j, X) \leq \frac{\lambda}{p_s}\sum_{j=1}^{V} \Delta f_j(\hat{\rho} \mid X) \leq \frac{\lambda}{p_s}\Delta g(\hat{\rho} \mid X)$$

■

Intuitively, this lemma follows because if p_s is close to unity, the probability that a robot *could* visit a node is not too far from the probability that a robot *actually* visits the node. Given this p_s/λ–approximate cost-benefit greedy algorithm, we can approximately solve the RSC problem.

C. Algorithm

Define the method `Dijkstra`($\mathcal{G}$, i, j), which returns the length of the shortest path from i to j on the edge weighted graph $\mathcal{G}$ using Dijkstra's algorithm. Given an edge weighted graph $\mathcal{G}$ and node rewards ν, the `Orienteering`$(\mathcal{G}, \nu)$ routine solves the orienteering problem (assuming $v_s = 1$, $v_t = V$ and budget $-\log(p_s)$) within factor $1/\lambda$, and returns the best path. Pseudocode for our algorithm is given in Figure 4. We begin by forming the graph $\mathcal{G}_O$ with log-transformed edge weights $\omega_O(e)$, and then use Dijkstra's algorithm to compute the maximum probability that a node can be reached. At each step, we solve the orienteering problem to greedily choose the path that maximizes $\Delta\bar{g}$. After selecting the path, we update the node rewards. The algorithm repeats unless all of the constraints have been satisfied, in which case it terminates.

procedure `CGreedySurvivors` $(\mathcal{G}, K)$
  Form $\mathcal{G}_O$ from $\mathcal{G}$, such that $v_s = 1$, $v_t = V$
  for $j = 1, \ldots, V$ **do**
    $\zeta_j \leftarrow \exp(-$`Dijkstra`$(\mathcal{G}_O, 1, j))$
    $\nu_1(j) \leftarrow \zeta_j$
  end for
  $\rho_1 \leftarrow$ `Orienteering`$(\mathcal{G}_O, \nu_1)$, $k \leftarrow 0$
  while $\max_j \nu_k(j) > 0$ **do**
    $\mathbb{E}[a^k_0(\rho_k)] \leftarrow 1$, $k \leftarrow k+1$
    for $n = 1, \ldots, |\rho_k|$ **do**
      $\mathbb{E}[a^k_n(\rho_k)] \leftarrow \mathbb{E}[a^k_{n-1}(\rho_k)]\omega(e^n_{\rho_k})$
      if $C_{1,j}(\{\hat{\rho}_\ell\}^k_{\ell=1})$ **then**
        $\nu_{k+1}(\rho_k(n)) = 0$
      else if $!C_{2,j}(\zeta_j, \{\hat{\rho}_\ell\}^k_{\ell=1})$ **then**
        $\nu_{k+1}(\rho_k(n)) \leftarrow (1 - \mathbb{E}[a^k_n(\rho_k)])\nu_k(\rho_k(n))$
      else
        $\nu_{k+1}(\rho_k(n)) \leftarrow p_v(j) - (1 - \nu_k(\rho_k(n))/\zeta_j)$
      end if
    end for
    $\rho_{k+1} \leftarrow$ `Orienteering`$(\mathcal{G}_O, \nu_{k+1})$
  end while
end procedure

Fig. 4. Approximate greedy algorithm for solving the RSC problem.

D. Approximation Guarantees

In this section we combine the results in this paper, [5], and [1] to give approximation guarantees for the `CGreedySurvivors` algorithm. The set cover problem is not approximable within a constant factor [2] and our guarantees are solution dependent, meaning that the approximation ratio depends on the specific problem posed and solution found by our algorithm. We give a *bi-criteria* guarantee, which ensures that the cost of our solution is no more than a factor $\gamma \geq 1$ greater than the minimum cost solution which satisfies fraction $\beta \leq 1$ of the constraints.

Theorem 2 (Bi-criteria approximation for the RSC): Given a RSC problem with coverage function g, let $\hat{X}_K$ be a solution found using an α–approximate cost-benefit greedy algorithm. For $L \leq K$, let $\hat{X}_L$ be the first L paths selected by the algorithm, and let X^*_L be a set with minimum cardinality which satisfies $g(X^*_L) \geq g(\hat{X}_L)$. Then the size of the set $\hat{X}_L$ is bounded above by

$$|\hat{X}_L| \leq \frac{1}{\alpha}\left(1 + \log\left(\frac{\frac{1}{\alpha}g(\hat{X}_1)}{g(\hat{X}_L) - g(\hat{X}_{L-1})}\right)\right)|X^*_L|,$$

and the set $\hat{X}_L$ satisfies

$$g(\hat{X}_L) \geq \frac{L}{K}g(\mathcal{X}).$$

Proof: For a given solution $\hat{X}_L$, define the functions

$$\hat{f}_j(X) = \min\{f_j(\hat{X}_L), f_j(X)\}, \qquad j = 1, \ldots, V,$$

that is the coverage constraint for node j is replaced by the smaller of $f_j(X)$ and the probability that robots following the paths $\hat{X}_L$ visit node j. Note that $\hat{f}_j(X) \leq f_j(X)$, and $\hat{f}_j(\hat{X}_\ell) = f_j(\hat{X}_\ell)$ for $\ell \leq L$. By definition of $\hat{X}_L$, we have for $\ell \leq L$

$$\sum_{j=1}^{V} \Delta\hat{f}_j(\hat{x}_\ell \mid \hat{X}_{\ell-1}) = \sum_{j=1}^{V} \Delta f_j(\hat{x}_\ell \mid \hat{X}_{\ell-1})$$
$$\geq \alpha \sum_{j=1}^{V} \Delta f_j(x \mid \hat{X}_{\ell-1}) \geq \alpha \sum_{j=1}^{V} \Delta\hat{f}_j(x \mid \hat{X}_{\ell-1}),$$

which implies that the set $\hat{X}_L$ is an α–approximate cost-benefit greedy solution to the RSC with normalized, monotone, submodular coverage function $\hat{g}(X) := \sum_{j=1}^{V} \hat{f}_j(X)$. Hence we can apply Theorem 1 to get the first result. The second result follow since $g(\hat{X}_K) = g(\mathcal{X})$ and g is submodular. ∎

The tightness of Theorem 2 for the case $L = K$ depends significantly on the given problem. Consider the example from Section III-B, with visit probability $p_v(1) = p_v(2) = 0.9$. Then the greedy algorithm will choose $\hat{X}_1 = \{\rho_1\}$, $\hat{X}_2 = \{\rho_1, \rho_2\}$. For this simple graph $\alpha = 1$ (since we can try all feasible paths), and so the guarantee states that $|\hat{X}_2| \leq (1 + \log(0.9/0.9))|X_2^*| = |X_2^*|$, that is the upper bound implies optimality. However if we change the visit probability threshold for node 1 to $p_v(1) = 0.9 + \epsilon$ for some small $\epsilon > 0$, then the greedy algorithm terminates after choosing $\hat{X}_3 = \{\rho_1, \rho_2, \rho_1\}$. The guarantee at $L = K$ becomes $|\hat{X}_3| \leq (1 + \log(1 + 0.9/\epsilon))|X_3^*|$, which can be arbitrarily loose as $\epsilon \to 0$. However using the bi-criteria guarantee for $L = 2$, we have $|\hat{X}_2| \leq (1 + \log(0.9/0.9))|X_2^*| = |X^*|$, and furthermore $g(X_2^*) \geq g(\mathcal{X}) - \epsilon$, meaning that the set $\hat{X}_2$ is an optimal solution to a problem which has nearly the same constraints.

E. Complexity

Let $\bar{p}_v = \max_j p_v(j) < 1$. Then we can upper bound the number of robots selected as $K \leq \bar{K} := V \frac{\log(1-\bar{p}_v)}{\log(1-p_s)}$, since the probability a robot visits any node is at least p_s and the visit probability threshold is at most $\bar{p}_v$. Suppose that the complexity of the `Orienteering` oracle is C_O. Then the complexity of our algorithm is $O(V^2 \log(V)) + O(V\bar{K}) + O(\bar{K}C_O)$. The first term is the complexity of calculating ζ_j for all nodes, the second term is the complexity of updating each weight $K \leq \bar{K}$ times, and the final term is the complexity of solving the $K \leq \bar{K}$ orienteering problems. If a suitable approximation algorithm is used for `Orienteering` (such as [17], [15], [16]), the procedure described above will have reasonable computation time.

F. Algorithm Variants

Heterogeneous teams. For the hetereogeneous team setting, we solve one cost-benefit greedy sub-problem per robot type at each step of the greedy routine, and select the path/type which has the best value to cost ratio. The complexity increases linearly in the number of robot types.

Edge variants. For the edge variant, we solve an *arc* orienteering problem [18] in place of the standard orienteering problem at each step of the cost-benefit greedy routine. The complexity remains the same, though the guarantees available for the arc orienteering problem are somewhat weaker than for the standard orienteering problem.

Multiple visits. For the multiple visit variant, we must track the probability that up to ℓ_j robots visit node j. Because this involves enumerating each of the K choose ℓ_j ways that ℓ_j robots can visit node j, this may require a large number of computations (see [5] for more details).

Arbitrary terminal nodes. Arbitrary terminal nodes are represented using the graph, and so the underlying algorithm remains the same. Appropriate modifications for the graph were discussed in Section III-C.

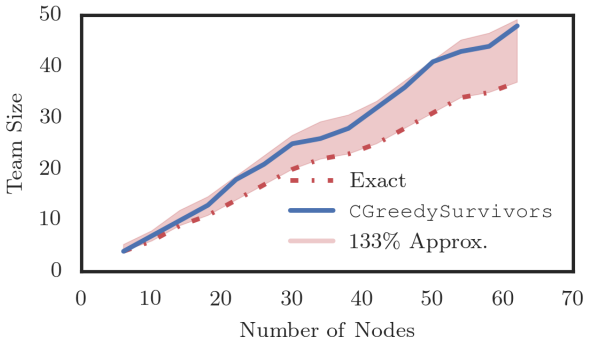


Fig. 5. Performance comparison over different graph sizes for a special case of the RSC.

V. Numerical Experiments

A. Comparison to optimal policies

The RSC problem is a mixed integer nonlinear programming (MINLP) problem because of the nonlinear coverage constraint. However, we can formulate a special case of the RSC problem as an integer linear programming (ILP) problem. The key assumption is that the edge weights ω are in the form of $P, P^2, \cdots$ where $0 \leq P \leq 1$ and the exponents are arbitrary positive integers. We can further simplify the ILP problem by assuming that the weights of all edges entering a node are the same, which reduces the number of decision variables and constraints considerably.

We used CPLEX and Gurobi to solve the ILP problem and the exact solutions served as the baseline for comparing with the output of the `CGreedySurvivors` routine. In Figure 5, for all of the cases tried (from 6 to 62 nodes), the output of our algorithm uses no more than 33% more robots than the optimal, and typically closer to 25%. This gives empirical justification for using the `CGreedySurvivors` routine as a high quality heuristic which significantly simplifies computational aspects of the problem while maintaining near-optimal solutions.

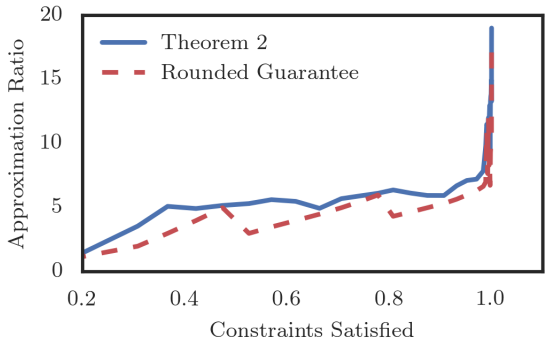


Fig. 6. Bi-criteria performance comparison for the search and rescue scenario with 225 nodes and $p_v(j) = 0.95$. The top curve (blue) shows the approximation guarantee from Theorem 2. The lower dashed curve (red) shows the guarantee while enforcing integrality and monotonicity on the lower bound on the optimal team size from Theorem 2.

B. Application to Search and Rescue

We demonstrate the applicability of our approach using the search and rescue scenario laid out in Section I, using a subset of the storm data shown in Figure 1. The risk posed by storms to robots was analyzed by [19] using a model very similar to ours, where the probability of survival is the product of the probability of surviving each edge in a path.

We place 225 sites uniformly and compute the edge weights by integrating the "base reflectivity" (the the amount of radar energy reflected by the weather system) across the straight-line connection between sites. We set $p_s = 0.8$ and $p_v(j) = 0.95$ for all nodes. Our routine finds a set of 36 paths which satisfy the coverage constraints exactly, but the guarantee from Theorem 2 gives an approximation ratio of 19.03. Using the bi-criteria guarantee and monotonicity of the lower bound (i.e. $|X_L^*| \leq |X_K^*|$) we know that $|X_K^*| \geq 4$, which means the approximation ratio is at most 9. We can find tighter guarantees by relaxing the constraints: the first 27 robots satisfy 99.7% of the constraints with approximation ratio 6.75, and the first 13 robots satisfy 80.7% of the constraints with approximation ratio 4.33. Figure 6 shows the fraction of constraints satisfied versus the approximation guarantee from Theorem 2 and a refined analysis which accounts for the integrality and monotonicity of the optimal solution. Hence, by slightly relaxing the constraint satisfaction one obtains reasonably tight approximation ratios.

VI. Conclusion

We consider the *Risk-Sensitive Coverage* problem, where we seek the smallest set of routes which satisfy visit probability thresholds and survival probability thresholds. We demonstrate that the RSC is an instance of the submodular set cover problem with a very large ground set. We give an approximate cost-benefit greedy routine for constructing a feasible solution to the RSC, and then provide a bi-criteria approximation guarantee which ensures that the solution returned is close to an optimal solution of a similar problem. We then compare the solutions returned by our routine to an exact solution for a special case of the RSC, and demonstrate the applicability of the approach on simulated data which represents a search and rescue setting during a severe storm.

There are several directions for future work. First, considering arbitrary cost functions would allow for significantly more general applications such as minimizing the expected losses, rather than the team size. The challenge here is to find an approximate cost-benefit greedy sub-routine – while one can use our algorithm to derive such a guarantee, it scales as $\frac{1}{1-p_s} \gg 1$. A second direction is to generalize the survival model, such as allowing for multiple 'hits' or more general distributions. Finally we are interested in on-line extensions of this problem, where routes are re-planned periodically.

References

[1] L. A. Wolsey, "An analysis of the greedy algorithm for the submodular set covering problem," *Combinatorica*, vol. 2, no. 4, pp. 385–393, 1982.

[2] U. Feige, "A threshold of ln n for approximating set cover," *Journal of the Association for Computing Machinery*, vol. 45, no. 4, pp. 634–652, 1998.

[3] R. K. Iyer and J. A. Bilmes, "Submodular optimization with submodular cover and submodular knapsack constraints," in *Advances in Neural Information Processing Systems*, 2013.

[4] NOAA National Weather Service Radar Operations Center, "NOAA next generation radar (NEXRAD) level II base data," 1991.

[5] S. Jorgensen, R. H. Chen, M. B. Milam, and M. Pavone, "The team surviving orienteers problem: Routing robots in uncertain environments with survival constraints," in *IEEE Int. Conf. on Robotic Computing*, 2017, to appear.

[6] L. Gargano and M. Hammar, "A note on submodular set cover on matroids," *Discrete Mathematics*, vol. 309, no. 18, pp. 5739–5744, 2009.

[7] H. Zhang and Y. Vorobeychik, "Submodular optimization with routing constraints," in *Proc. AAAI Conf. on Artificial Intelligence*, 2016.

[8] H. N. Psaraftis, M. Wen, and C. A. Kontovas, "Dynamic vehicle routing problems: Three decades and counting," *Networks*, vol. 67, no. 1, pp. 3–31, 2016.

[9] J. F. Ehmke, A. M. Campbell, and T. L. Urban, "Ensuring service levels in routing problems with time windows and stochastic travel times," *European Journal of Operational Research*, vol. 240, no. 2, pp. 539–550, 2015.

[10] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1258 – 1276, 2013.

[11] A. Gunawan, H. C. Lau, and P. Vansteenwegen, "Orienteering problem: A survey of recent variants, solution approaches and applications," *European Journal of Operational Research*, vol. 255, no. 2, pp. 315 – 332, 2016.

[12] S. L. Smith, M. Schwager, and D. Rus, "Persistent robotic tasks: Monitoring and sweeping in changing environments," *IEEE Transactions on Robotics*, vol. 28, no. 2, pp. 410–426, 2012.

[13] A. Wallar, E. Plaku, and D. A. Sofge, "Reactive motion planning for unmanned aerial surveillance of risk-sensitive areas," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 3, pp. 969–980, 2015.

[14] A. Krause and D. Golovin, "Submodular function maximization," *Tractability: Practical Approaches to Hard Problems*, vol. 3, no. 19, p. 8, 2012.

[15] C. Chekuri and M. Pál, "A recursive greedy algorithm for walks in directed graphs," in *IEEE Symp. on Foundations of Computer Science*, 2005.

[16] K. Chen and S. Har-Peled, "The orienteering problem in the plane revisited," in *ACM Symp. on Computational Geometry*, 2006.

[17] C. Chekuri, N. Korula, and M. Pál, "Improved algorithms for orienteering and related problems," *ACM Transactions on Algorithms*, vol. 8, no. 3, p. 23, 2012.

[18] D. Gavalas, C. Konstantopoulos, K. Mastakas, G. Pantziou, and N. Vathis, "Approximation algorithms for the arc orienteering problem," *Information Processing Letters*, vol. 115, no. 2, pp. 313–315, 2015.

[19] B. Zhang, L. Tang, and M. Roemer, "Probabilistic planning and risk evaluation based on ensemble weather forecasting," *IEEE Transactions on Automation Science and Engineering*, vol. PP, no. 99, pp. 1–11, 2017.