The Team Surviving Orienteers Problem: Routing Robots in Uncertain Environments with Survival Constraints

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

Abstract—**We study the following multi-robot coordination problem: given a graph, where each edge is weighted by the probability of surviving while traversing it, find a set of paths for K robots that maximizes the expected number of nodes collectively visited, subject to constraints on the probabilities that each robot survives to its destination. We call this the Team Surviving Orienteers (TSO) problem, which is motivated by scenarios where a team of robots must traverse a dangerous environment, such as aid delivery in disaster or war zones. We present the TSO problem formally along with several variants, which represent "survivability-aware" counterparts for a wide range of multi-robot coordination problems such as vehicle routing, patrolling, and informative path planning. We propose an approximate greedy approach for selecting paths, and prove that the value of its output is within a factor $1 - e^{-p_s/\lambda}$ of the optimum where 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. Our approach has linear time complexity in the team size and polynomial complexity in the graph size. Using numerical simulations, we verify that our approach works well in practice and that it scales to problems with hundreds of nodes and tens of robots.**

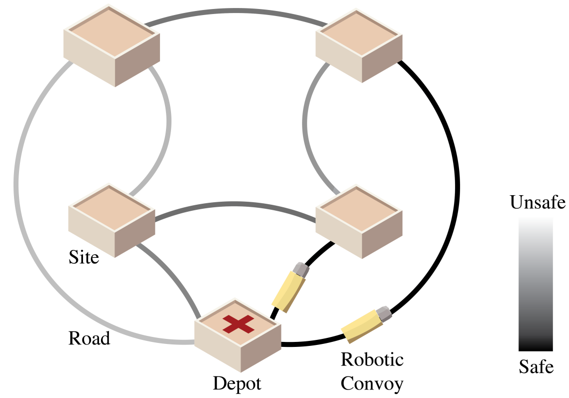


Fig. 1. Illustration of the TSO problem applied to an aid delivery scenario. The objective is to maximize the expected number of sites visited by at least one robotic convoy. Travel between sites is risky (as emphasized by the gray color scale for each edge), and paths must be planned to ensure that the return probability for each vehicle is above a survival threshold.

I. Introduction

Consider the problem of delivering humanitarian aid in a disaster or war zone with a team of robots. There are a number of sites which need the resources, but traveling among these sites is dangerous. While the aid agency wants to deliver aid to every city, it also seeks to limit the number of assets that are lost. We formalize this problem as a generalization of the orienteering problem [1], whereby one seeks to visit as many nodes in a graph as possible given a budget constraint and travel costs. In the aid delivery case, the travel costs are the probability that a robotic aid vehicle is lost while traveling between sites, and the goal is to maximize the *expected* number of sites visited by the vehicles, while keeping the return *probability* for each vehicle above a specified survival threshold (i.e., while fulfilling a chance constraint for the survival of each vehicle). We call this problem formulation the "team surviving orienteers" (TSO) problem, illustrated in Figure 1. The TSO problem is distinct from previous work because of its notion of *risky traversal:* when a robot traverses an edge, there is a probability that it is lost and does not visit any other nodes. This creates a complex, history-dependent coupling between the edges chosen and the distribution of nodes visited, which precludes the application of existing approaches available for the traditional orienteering problem.

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 Aerospace Systems, 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 objective of this paper is to devise a constant-factor approximation algorithm for the TSO problem. Our key technical insight is that the expected number of nodes visited satisfies a diminishing returns property known as submodularity, which for set functions means that $f(A \cup B) + f(A \cap B) \leq f(A) + f(B)$. We develop a *linearization procedure* for the problem, which leads to a greedy algorithm that enjoys a constant-factor approximation guarantee. We emphasize that while a number of works have considered orienteering problems with submodular objectives [2], [3], [4] or chance constraints [5], [6] separately, the combination of the two makes the TSO problem novel, as detailed next.

Related work. The orienteering problem (OP) has been extensively studied [7], [8] and is known to be NP-hard. Over the past decade a number of constant-factor approximation algorithms have been developed for special cases of the problem [9]. Below we highlight several variants which share either similar objectives or constraints as the TSO problem.

The *submodular orienteering problem* considers finding a single path which maximizes a submodular reward function of the nodes visited. The recursive greedy algorithm proposed in [2] yields a solution in quasi-polynomial time with reward lower bounded as $\Omega(\text{OPT}/\log(\text{OPT}))$, where OPT is the optimum value. More recently, [4] develops a (polynomial time) generalized cost-benefit algorithm, useful when searching the feasible set is NP-hard (such as longest path problems). The authors show that the output of their algorithm is $\Omega(\frac{1}{2}(1-1/e)\text{OPT}^*)$, where OPT^* is the optimum for a relaxed problem. In our context, OPT^* roughly corresponds to the maximum expected number of nodes visited with survival probability threshold $\sqrt{p_s}$, which may be significantly different from the actual optimum. Our work considers a specific submodular function, however we incorporate risky traversal, give stronger guarantees, and discuss an extension to general submodular functions. In the *orienteering problem with stochastic travel times* proposed by [3], travel times are stochastic and reward is accumulated at a node only if it is visited before a deadline. This setting could be used to solve the single robot special case of the TSO problem by using a logarithmic transformation on the survival probabilities, but [3] does not provide any polynomial time guarantees. In the *risk-sensitive orienteering problem* [6], the goal is to maximize the sum of rewards (which is history-independent) subject to a constraint on the probability that the path cost is large. The TSO problem unifies the models of the risk-sensitive and stochastic travel time variants by considering both a history-dependent objective (expected number of nodes visited) and a chance constraint on the total cost. Furthermore, we provide a constant-factor guarantee for the *team* version of this problem.

A second closely-related area of research is represented by the vehicle routing problem (VRP) [10], [11], which is a family of problems focused on finding a set of paths that maximize quality of service subject to budget or time constraints. The *probabilistic VRP* (PVRP) considers stochastic edge costs with chance constraints on the path costs – similar to the risk-averse orienteering and the TSO problem constraints. The authors of [12] pose the simultaneous location-routing problem, where both paths and depot locations are selected to minimize path costs subject to a probabilistic connectivity constraint, which specifies the average case risk rather than individual risks. More general settings were considered in [13], which considers several distribution families (such as the exponential and normal distributions), and [14], which considers non-linear risk constraints. In contrast to the TSO problem, the PVRP requires *every* node to be visited and seeks to minimize the travel cost. In the TSO problem, we require every path to be safe and maximize the expected number of nodes visited.

A third related branch of literature is the informative path planning problem (IPP), which seeks to find a set of K paths for mobile robotic sensors in order to maximize the information gained about an environment. One of the earliest IPP approaches [15] extends the recursive greedy algorithm of [2] using a spatial decomposition to generate paths for multiple robots. They use submodularity of information gain to provide performance guarantees. Sampling-based approaches to IPP were proposed by [16], which come with asymptotic guarantees on optimality. The structure of the IPP is very similar to that of the TSO problem since it is a multi-robot path planning problem with a submodular objective function which is non-linear and history dependent. However the IPP problem does not capture the notion of risky traversal which is essential to the TSO problem. Our general approach is inspired by works such as [17], which iteratively assigns paths to each robot, but for the TSO problem we are able to further exploit the problem structure to derive constant-factor guarantees for our polynomial time algorithm.

Statement of Contributions. The contribution of this paper is fourfold. First, we propose a generalization of the orienteering problem, referred to as the TSO problem. By considering a multi-robot (team) setting, we extend the state of the art for the submodular orienteering problem, and by maximizing the expected number of nodes visited at least once, we extend the state of the art in the probabilistic vehicle routing literature. From a practical standpoint, as discussed in Section III, the TSO problem represents a "survivability-aware" counterpart for a wide range of multi-robot coordination problems such as vehicle routing, patrolling, and informative path planning. Second, we establish that the objective function of the TSO problem is submodular, provide a linear relaxation of the single robot TSO problem (which can be solved as a standard orienteering problem), and show that the solution to the relaxed problem provides a close approximation of the optimal solution of the single robot TSO problem. Third, we propose an approximate greedy algorithm which has polynomial complexity in the number of nodes and linear complexity in the team size, and prove that the value of the output of our algorithm is $\Omega((1-e^{-p_s/\lambda})\mathrm{OPT})$, where OPT is the optimum value, p_s is the per-robot survival probability threshold, and $1/\lambda \leq 1$ is the approximation factor of an oracle routine for the solution to the orienteering problem (we note that, in practice, p_s is usually close to unity). Finally, we demonstrate the effectiveness of our algorithm for large problems using simulations by solving a problem with 900 nodes and 25 robots.

Organization. In Section II we review key background information. In Section III we state the TSO problem formally, give an example, and describe several variants and applications. In Section IV we show that the objective function is submodular and describe the linear relaxation technique. We then outline a greedy solution approach for the TSO problem, give approximation guarantees, characterize the algorithm's complexity, and give extensions of the algorithm for variants of the TSO problem. In Section V we verify the performance bounds and demonstrate the scalability of our approach. Finally, we outline future work 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 [18]. 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 $\Delta f(x \mid X)$.

B. The Approximate Greedy Algorithm

A typical submodular maximization problem entails finding a set $X \subseteq \mathcal{X}$ with cardinality K that maximizes f. Finding an optimal solution, X^*, is NP-hard for general submodular functions [18]. The *greedy algorithm* constructs a set $\bar{X}_K = \{x_1, \ldots, x_K\}$ by iteratively adding an element x which maximizes the discrete derivative of f at the partial set already selected. In other words the ℓth element satisfies:

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

We refer to the optimization problem above as 'the greedy sub-problem' at step ℓ. A well-known theorem proven by [19] states that if f is a monotone, normalized, non-negative, and submodular function, then $f(\bar{X}_K) \geq (1 - \frac{1}{e}) f(X^*)$. This is a powerful result, but if the set $\mathcal{X}$ is large we might only be able to approximately solve the greedy subproblem. An α–approximate greedy algorithm constructs the set $\hat{X}_K$ by iteratively adding elements which *approximately* maximize the discrete derivative of f at the partial set already selected. In particular for some fixed $\alpha \leq 1$, the ℓth element $\hat{x}_\ell$ satisfies:

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

We provide a guarantee for the α–approximate greedy algorithm analogous to the guarantee for the greedy algorithm, thereby extending Theorem 4.2 of [19]:

Theorem 1 (α–approximate greedy guarantee): Let f be a monotone, normalized, non-negative, and submodular function with discrete derivative Δf. Then for the output of any α–approximate greedy algorithm with L elements, $\hat{X}_L$, we have the following inequality:

$$f(\hat{X}_L) \geq \left(1 - e^{-\alpha L/K}\right) \max_{X \subseteq \mathcal{X} : |X| = K} f(X).$$

Proof: The case where $L = K$ is a special case of Theorem 1 from [20]. To generalize to $L > K$ we extend the proof for the greedy algorithm in [18]. Let $X^* \in 2^{\mathcal{X}}$ be a set which maximizes $f(X)$ subject to the cardinality constraint $|X| = K$. For $\ell < L$, we have:

$$\begin{aligned} f(X^*) &\leq f(X^* \cup \hat{X}_\ell) \\ &= f(\hat{X}_\ell) + \sum_{k=1}^{K} \Delta f(x_k^* \mid \hat{X}_\ell \cup \{x_1^*, \ldots, x_{k-1}^*\}) \\ &\leq f(\hat{X}_\ell) + \sum_{k=1}^{K} \Delta f(x_k^* \mid \hat{X}_\ell) \\ &\leq f(\hat{X}_\ell) + \frac{1}{\alpha} \sum_{k=1}^{K} \Delta f(\hat{x}_{\ell+1} \mid \hat{X}_\ell) \\ &\leq f(\hat{X}_\ell) + \frac{K}{\alpha} (f(\hat{X}_{\ell+1}) - f(\hat{X}_\ell)). \end{aligned}$$

The first line follows from the monotonicity of f, the second is a telescoping sum, and the third follows from the submodularity of f. The fourth line is due to the α–approximate greedy construction of $\hat{X}_L$, and the last is because $|X^*| = K$. Now define $\delta_\ell = f(X^*) - f(\hat{X}_\ell)$. We can re-arrange the inequality above to yield:

$$\delta_{\ell+1} \leq \left(1 - \frac{\alpha}{K}\right) \delta_\ell \leq \left(1 - \frac{\alpha}{K}\right)^{\ell+1} \delta_0.$$

Since f is non-negative, $\delta_0 \leq f(X^*)$ and using the inequality $1 - x \leq e^{-x}$ we get

$$\delta_L \leq \left(1 - \frac{\alpha}{K}\right)^L \delta_0 \leq \left(e^{-\alpha L/K}\right) f(X^*).$$

Now substituting $\delta_L = f(X^*) - f(\hat{X}_L)$ and rearranging:

$$f(\hat{X}_L) \geq \left(1 - e^{-\alpha L/K}\right) f(X^*).$$

■

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

In this section we give the formal problem statement for the TSO problem, work out an example problem, and describe applications and variants of the problem.

A. Formal Problem Description

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 $e_\rho^n = (\rho(n-1), \rho(n))$. Robots are indexed by variable k, and for each we define the independent Bernoulli random variables $s_n^k(\rho)$ which are 1 with probability $\omega(e_\rho^n)$ and 0 with probability $1 - \omega(e_\rho^n)$. If robot k follows path ρ, the random variables $a_n^k(\rho) := \prod_{i=1}^{n} s_i^k(\rho)$ 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, and survival probability p_s we must find $K \geq 1$ paths $\{\rho_k\}_{k=1}^K$ (one for each of K robots) such that, for all k, the probability that $a_{|\rho_k|}^k(\rho_k) = 1$ 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 prove

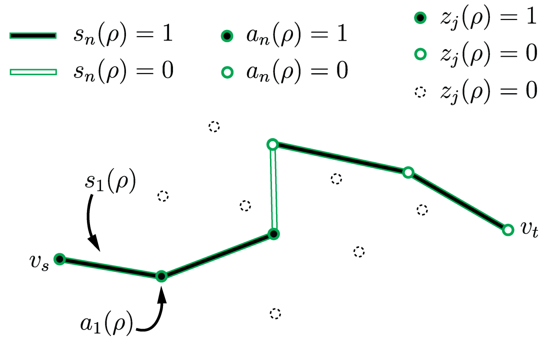
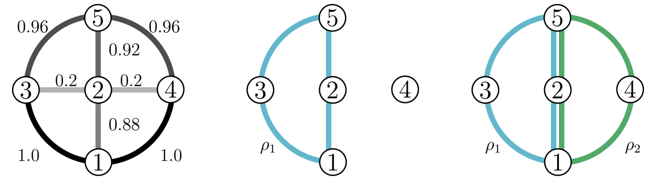


Fig. 2. Illustration of the notation used. A robot plans to take path ρ, whose edges are represented by lines. The fill of the lines represent the value of $s_n(\rho)$. In this example $s_3(\rho) = 0$, which means that $a_3(\rho) = a_4(\rho) = a_5(\rho) = 0$. The variables $z_j(\rho)$ are zero if either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



(a) Graph $\mathcal{G}$ (b) Path for one robot (c) Paths for two robots

Fig. 3. (a) Example of a TSO problem. Robots start at the bottom (node 1) and darker lines correspond to safer paths. (b) A single robot can only visit four nodes safely. (c) Two robots can visit all nodes safely. It is easy to see that adding more robots yields diminishing returns.

whether $\mathcal{X}(p_s, \omega)$ is empty after $O(V^2 \log(V))$ 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 $j = 1, \dots, V$:

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

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:

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

Let $d_j > 0$ be the reward accumulated for visiting node j at least once. Then the TSO problem is formally stated as:

> **Team Surviving Orienteers (TSO) Problem:** Given a graph $\mathcal{G}$, edge weights ω, survival probability threshold p_s and team size K, maximize the weighted expected number of nodes visited by at least one robot:

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

The objective represents the cumulative expected reward obtained by the K robots. The first set of constraints enforces the survival probability, the second and third sets of constraints enforce the initial and final node constraints. In particular, the survival probability threshold p_s serves two purposes: first, it requires that, on average, $p_s K$ robots will reach node v_t safely, and second, it enforces that risk is distributed fairly (e.g., no robot fails with too high a probability).

The TSO problem can be viewed as a set maximization problem with a cardinality constraint, where the domain of optimization is the set $\mathcal{X}$ containing K copies of *each* path in $\mathcal{X}(p_s, \omega)$. Crucially, if the objective function is a submodular function, then Theorem 1 guarantees that the greedily selected set of paths will achieve reward close to the optimum – a central result for this paper. We show that the objective is indeed submodular in Section IV-A, but first we provide illustrative examples of the TSO problem.

B. Example

An example of the TSO problem is given in Figure 3(a). There are five nodes, and edge weights are shown next to their respective edges. Two robots start at node 1, and must end at virtual node $1'$ (which is a copy of node 1) with probability at least $p_s = 0.75$. Path $\rho_1 = \{1, 3, 5, 2, 1'\}$ is shown in Figure 3(b), and path $\rho_2 = \{1, 4, 5, 2, 1'\}$ is shown alongside ρ_1 in Figure 3(c). Robot 1 visits node 3 with probability 1.0 and node 5 with probability 0.96. Robot 2 also visits node 5 with probability 0.96 and so the probability at least one robot visits node 5 is $1 - p_5(0, \{\rho_1, \rho_2\}) = 0.9984$. The probability that robot 1 returns safely is $\mathbb{E}\left[a_4^1(\rho_1)\right] = 0.794$. The expected number of nodes visited by the first robot following ρ_1 is 3.88, and for two robots following ρ_1 and ρ_2 it is 4.905. Since there are only five nodes, it is clear that adding more robots must yield diminishing returns.

C. Variants and Applications

Edge rewards and patrolling: Our formulation can easily be extended to a scenario where the goal is to maximize the weighted expected number of *edges* visited by at least one robot. Define $z_{i,j}^k(\rho)$ to indicate whether robot k following path ρ takes edge (i, j), and for $(i, j) \in \mathcal{E}$ define $p_{i,j}(m, \{\rho_k\}_{k=1}^K)$ as before with z_j^k replaced by $z_{i,j}^k$ (if $(i, j) \notin \mathcal{E}$, then set $p_{i,j}(0, \cdot) = 1$). The objective function for this problem is $\sum_{i=1}^V \sum_{j=1}^V d_{i,j} \left(1 - p_{i,j}(0, \{\rho_k\}_{k=1}^K)\right)$. This variant could be used to model a patrolling problem, where the goal is to inspect the maximum number of roads subject to the survival probability threshold. Such problems also occur when planning scientific missions (e.g., on Mars), where the objective is to execute the most important traversals.

Multiple visits and IPP: We consider rewards for multiple visits as follows. For $m \leq K$, let $d_j^{(m)}$ be the marginal benefit of the mth visit. The reward function is now $\sum_{m=1}^{K}\sum_{j=1}^{V} d_j^{(m)} p_j\left(m, \{\rho_k\}_{k=1}^{K}\right)$. In order for our solution approach and guarantees to apply, we require that $d_j^{(m)}$ be a non-increasing function of m (this ensures submodularity). We can build an approximation for any submodular function of the node visits by assigning $d_j^{(m)}$ to be the incremental gain for visiting node j the mth time. A concrete example of this formulation is informative path planning where the goal is to maximize the reduction in entropy of the posterior distribution of node variables $\{Y_j\}_{j=1}^{V}$, and $d_j^{(m)}$ represents the reduction in entropy of the posterior distribution of Y_j by taking the mth measurement.

IV. Approximate solution approach

Our approach to solving the TSO problem is to exploit submodularity of the objective function using an α–approximate greedy algorithm (as defined in Section II-B). Accordingly, in Section IV-A we show that the objective function of TSO problem is submodular. In Section IV-B we present a linearization of the greedy sub-problem, which in the context of the TSO problem entails finding a path which maximizes the discrete derivative of the expected number of nodes visited, at the partial set already constructed. We use this linearization to find a polynomial time (p_s/λ)–approximate greedy algorithm. Leveraging this result, we describe our `GreedySurvivors` algorithm for the TSO problem in Section IV-C, discuss its approximation guarantee in Section IV-D, and characterize its computational complexity in Section IV-E. Finally, in Section IV-F we discuss algorithm modifications for variants of the TSO problem.

A. Submodularity of the Objective Function

For the TSO problem, a straightforward calculation gives the discrete derivative of the objective function as

$$\Delta J(\rho \mid \{\rho_k\}_{k=1}^{\ell}) = \sum_{j=1}^{V} \mathbb{E}[z_j^{\ell+1}(\rho)] d_j p_j(0, \{\rho_k\}_{k=1}^{\ell}).$$

The value placed on each node is the product of the probability that the robot visits the node, the importance of the node, and the probability the node has not been visited by any of the ℓ robots following paths $\{\rho_k\}_{k=1}^{\ell}$.

Lemma 1 (Objective is submodular): The objective function for the TSO problem is normalized, non-negative, monotone and submodular.

Proof: The sum over an empty set is zero which immediately implies that the objective function is normalized. Because $d_j > 0$ and $\mathbb{E}[z_j^k(\cdot)] \in [0,1]$, the discrete derivative is everywhere non-negative and so the objective function is both non-negative and monotone. Now consider $X \subseteq X' \subset \mathcal{X}$ and $\rho \in \mathcal{X} \setminus X'$. Since $X \subseteq X'$ and $\mathbb{E}[z_j^k(\cdot)] \in [0,1]$, and for $L > |X'|$ the index of a robot not assigned a path in X',

$$\Delta J(\rho \mid X') = \sum_{j=1}^{V} \mathbb{E}[z_j^L(\rho)] d_j \prod_{\rho_k \in X'} (1 - \mathbb{E}[z_j^k(\rho_k)])$$
$$\leq \sum_{j=1}^{V} \mathbb{E}[z_j^L(\rho)] d_j \prod_{\rho_k \in X} (1 - \mathbb{E}[z_j^k(\rho_k)]) = \Delta J(\rho \mid X).$$

Therefore the objective function is submodular. ■

Intuitively, this statement follows from the fact that the marginal gain of adding one more robot is proportional to the probability that nodes have not yet been visited, which is a decreasing function of the number of robots already selected.

B. Linear Relaxation for Greedy Sub-problem

Given a previously selected set of paths, $X_{L-1} = \{\rho_\ell\}_{\ell=1}^{L-1}$, the greedy sub-problem for the TSO problem at step L requires us to find a path ρ_L from the set $\mathcal{X} \setminus X_{L-1}$ which maximizes the discrete derivative of the objective function at X_{L-1} with respect to ρ_L. Note that because we define $\mathcal{X}$ to have as many copies of each path as the maximum number of robots we plan for, the set $\mathcal{X} \setminus \{\rho_\ell\}_{\ell=1}^{L-1}$ always contains at least one copy of each path in $\mathcal{X}(p_s, \omega)$. Since the discrete derivative of the objective function at X_{L-1} with respect to any of the copies of a path $\rho \in \mathcal{X}(p_s, \omega)$ is the same, we can solve the greedy sub-problem by only considering elements in the set $\mathcal{X}(p_s, \omega)$.

Even with this simplification, the greedy sub-problem is very difficult for the TSO problem: it requires finding a path which maximizes submodular node rewards subject to a budget constraint (this is the submodular orienteering problem). No polynomial time constant-factor approximation algorithm is known for general submodular orienteering problems [2], and so in this section we design one specifically for the greedy sub-problem of the TSO problem.

We relax the problem of maximizing the discrete derivative by replacing the probability that robot L traversing path ρ visits node j, $\mathbb{E}[z_j^L(\rho)]$, 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 $\omega_O(e) := -\log(\omega(e))$. Let $\mathbb{I}_j(\rho)$ be equal to 1 if node j is in ρ and 0 otherwise. In the relaxed problem we are looking to maximize the sum:

$$\Delta \bar{J}(\rho \mid X_{L-1}) := \sum_{j=1}^{V} \mathbb{I}_j(\rho) \zeta_j d_j p_j\left(0, X_{L-1}\right),$$

which represents an *optimistic* estimate of the discrete derivative of the objective function at 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 $\nu_L(j) = \zeta_j d_j p_j(0, X_{L-1})$. 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 $\Delta \bar{J}(\rho \mid X_{L-1})$), 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 [21] gives a guarantee $\lambda = (1+\epsilon)$, for undirected graphs [9] gives a guarantee $\lambda = (2+\epsilon)$, and for directed graphs [2] gives a guarantee in terms of the number of nodes. Using such an oracle, we have the following guarantee:

Lemma 2 (Single robot constant-factor guarantee): Let `Orienteering` be a routine that solves the orienteering problem within constant-factor $1/\lambda$, that is for $c_j > 0$ and node weights $\nu(j) = c_j \zeta_j$, 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 $c_j > 0$ and any $\rho \in \mathcal{X}(p_s, \omega)$, the cumulative rewards for a robot following path $\hat{\rho}$ satisfies

$$\sum_{j=1}^{V} c_j \mathbb{E}[z_j(\hat{\rho})] \geq \frac{p_s}{\lambda} \sum_{j=1}^{V} c_j \mathbb{E}[z_j(\rho)].$$

Proof: By definition of ζ_j and the `Orienteering` routine, we have:

$$\sum_{j=1}^{V} c_j \mathbb{E}[z_j(\rho)] \leq \sum_{j=1}^{V} \mathbb{I}_j(\rho)\zeta_j c_j \leq \lambda \sum_{j=1}^{V} \mathbb{I}_j(\hat{\rho})\zeta_j c_j.$$

Path $\hat{\rho}$ is feasible, so $\mathbb{I}_j(\hat{\rho}) p_s \zeta_j \leq \mathbb{I}_j(\hat{\rho}) p_s \leq \mathbb{E}[z_j(\hat{\rho})]$, which combined with the equation above completes the proof. ■

This is a remarkable statement because it guarantees that, if we solve the orienteering problem near-optimally, choose $c_j = d_j p_j(0, X_{L-1})$ and p_s is not too small, the solution to the linear relaxation will give nearly the same result as the optimal solution to the greedy sub-problem at step L for the TSO problem. The intuition is that for p_s close to unity no feasible path can be very risky and so the probability that a robot *actually* reaches a node will not be too far from the maximum probability that it *could* reach the node.

C. Greedy Approximation for the TSO Problem

Using this relaxation with $c_j = d_j p_j(0, X_{L-1})$ we have an p_s/λ–approximate algorithm for the greedy sub-problem at step L. This gives us a $(1 - e^{-p_s/\lambda})$–approximate greedy algorithm for the TSO problem, as detailed next.

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. For each robot $k = 1, \ldots, K$, we solve the orienteering problem to greedily choose the path that maximizes $\Delta\bar{J}$.

procedure GREEDYSURVIVORS$(\mathcal{G}, K)$
  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`$(\mathcal{G}_O, 1, j))$
    $\nu_1(j) \leftarrow \zeta_j d_j$
  end for
  $\rho_1 \leftarrow$ `Orienteering`$(\mathcal{G}_O, \nu_1)$
  for $k = 1, \ldots, K-1$ **do**
    $\mathbb{E}[a^k_0(\rho_k)] \leftarrow 1$
    for $n = 1, \ldots, |\rho_k|$ **do**
      $\mathbb{E}[a^k_n(\rho_k)] \leftarrow \mathbb{E}[a^k_{n-1}(\rho_k)]\omega(e^n_{\rho_k})$
      $\nu_{k+1}(\rho_k(n)) \leftarrow (1 - \mathbb{E}[a^k_n(\rho_k)])\nu_k(\rho_k(n))$
    end for
    $\rho_{k+1} \leftarrow$ `Orienteering`$(\mathcal{G}_O, \nu_{k+1})$
  end for
end procedure

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

D. Approximation Guarantees

In this section we combine the results from Section II-B and IV-B to prove that the output of the `GreedySurvivors` algorithm is close to the optimal solution to the TSO problem. Specifically, we compare a team with $L \geq K$ robots using greedily selected paths to a team with K optimally selected paths, because this gives us a way to compute tighter bounds on the performance of our algorithm.

Theorem 2 (Multi-robot constant-factor guarantee): Let $1/\lambda$ be the constant-factor guarantee for the `Orienteering` routine as in Lemma 1, and assign robot ℓ the path $\hat{\rho}_\ell$ output by the orienteering routine given graph $\mathcal{G}_\mathrm{O}$ with node weights

$$\nu_\ell(j) = \zeta_j d_j p_j\left(0, \{\hat{\rho}_k\}_{k=1}^{\ell-1}\right).$$

Let $X^*_K = \{\rho^*_k\}_{k=1}^K$ be an optimal solution to the TSO with K robots. Then the weighted expected number of nodes visited by a team of $L \geq K$ robots following the paths $\hat{X}_L = \{\hat{\rho}_\ell\}_{\ell=1}^L$ is at least a fraction $\gamma = 1 - \exp\left(-p_s L/\lambda K\right)$ of the optimum:

$$\sum_{j=1}^{V} d_j \left(1 - p_j\left(0, \hat{X}_L\right)\right) \geq \gamma \sum_{j=1}^{V} d_j \left(1 - p_j\left(0, X^*_K\right)\right).$$

Proof: As discussed at the beginning of Section IV-B, it suffices to solve the greedy sub-problem only considering elements in $\mathcal{X}(p_s, \omega)$. Using Lemma 2 with c_j chosen appropriately for the objective function, we have a constant-factor guarantee $\alpha = p_s/\lambda$ for the problem of finding the path from $\mathcal{X}(p_s, \omega)$ that maximizes the discrete derivative of our objective function. Now applying Theorem 1 to our objective function (which by Lemma 1 is normalized, non-negative, monotone and submodular) we have the desired result. ■

In many scenarios of interest p_s is quite close to 1, since robots are typically valuable or difficult to replace. For $L = K$ this theorem gives an $1 - e^{-p_s/\lambda}$ guarantee for the output of our algorithm. This bound holds for any team size, and guarantees that the output of the (polynomial time) linearized greedy algorithm will have a similar reward to the output of the (exponential time) optimal algorithm.

Taking $L > K$ gives a practical way of testing how much more efficient the allocation for K robots could be. For example, if $L\frac{p_s}{\lambda} = 6K$ we have a $(1 - 1/e^6) \simeq 0.997$ factor approximation for the optimal value achieved by K robots. We use this approach to generate tight upper bounds for our experimental results. Note that as $L \to \infty$, the output of our algorithm has at least the same value as the optimum, which emphasizes the importance of guarantees for *small* teams.

E. Computational Complexity

Suppose that the complexity of the `Orienteering` oracle is C_O. Then the complexity of our algorithm is $O(V^2 \log(V)) + O(KV^2) + O(KC_O)$. The first term is the complexity of running Dijkstra's to calculate ζ_j for all nodes, the second term is the complexity of updating the V weights K times (each update costs at most $|\rho_k| \leq V$ flops), and the final term is the complexity of solving the K orienteering problems. For many approximation algorithms $C_O = V^{O(1/\epsilon)}$, and so the complexity is dominated by KC_O. If a suitable approximation algorithm is used for `Orienteering` (such as [2], [9], [21]), the procedure described above will have reasonable computation time even for large team sizes.

F. Algorithm Variants

We can solve the variants from Section III-C using minor modifications to the `GreedySurvivors` routine.

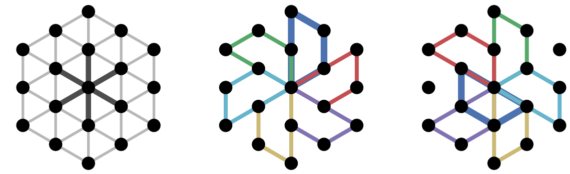
Edge Rewards and Patrolling: Define $\zeta_{i,j} = \zeta_i \omega(i,j)$, which is the largest probability that edge (i,j) is successfully traversed. The linearized greedy algorithm will still have constant-factor guarantee $\alpha = p_s/\lambda$, but now requires solving an *arc* orienteering problem. Constant-factor approximations for the arc orienteering problem can be found using algorithms for the orienteering problem as demonstrated in [22]: for an undirected graph $\lambda = 6+\epsilon+o(1)$ in polynomial time $V^{O(1/\epsilon)}$.

Multiple Visits and IPP: The only modification for the multiple visits variant is to linearize the greedy sub-problem by choosing $c_j = \sum_{m=1}^{K} d_j^{(m)} p_j(m-1, \hat{X}_\ell)$. It is important to note that the complexity results change unfavorably in the multi-visit case. Computing $p_j(m, \hat{X}_K)$ requires evaluating the K choose m visit events. If the number of profitable visits is at most $M < K$ then the number of visit events is a polynomial function of the team size (bounded by $K^M/M!$), but if $K \leq M$ then there are 2^K visit events which must be evaluated.

V. Numerical Experiments

A. Verification of Bounds

We consider a TSO problem on the graph shown in Figure 5(a): the central starting node has 'safe' transitions to six nodes, which have 'unsafe' transitions to the remaining twelve nodes. Due to the symmetry of the problem we can quickly compute an optimal policy for a team of six robots, which is shown in Figure 5(b). The output of the greedy algorithm is shown in Figure 5(c). The `GreedySurvivors` solution comes close to the optimal, although the initial path planned (shown by the thick dark blue line) does not anticipate its impact on later paths. The expected number of nodes visited by robots following optimal paths, greedy paths, and the upper bound are shown in Figure 6. Note that the upper bound is close to the optimal, even for small teams, and that the `GreedySurvivors` performance is nearly optimal.



(a) Graph $\mathcal{G}$ (b) Optimal X_6^* (c) Greedy $\bar{X}_6$

Fig. 5. (a) Example of a team surviving orienteers problem with depot in the center. Thick edges correspond to survival probability 0.98, light edges have survival probability 0.91. (b) Optimal paths for survival threshold $p_s = 0.70$ and $K = 6$. (c) Greedy paths for the same problem.

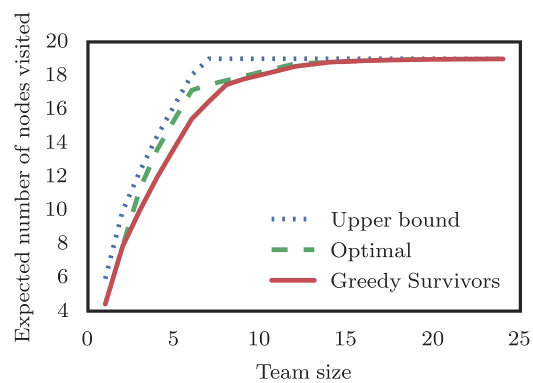


Fig. 6. Performance comparison for the example in Figure 5(a). The optimal value is shown in green and the GreedySurvivors value is shown in red. The upper bound on the optimum from Theorem 2 is shown by the dotted line.

B. Empirical Approximation Factor

We compare our algorithm's performance against an upper bound on the optimal value. We use an exact solver for the orienteering problem (using the Gurobi MIP solver), and generate instances on a graph with $V = 65$ nodes and uniformly distributed edge weights in the interval $[0.3, 1)$. The upper bound used for comparison is the smallest of 1) the number of nodes which can be reached within the budget, 2) the constant-factor guarantee times our approximate solution, and 3) the guarantee from solving the problem with an oversized team (from Theorem 2). The average performance (relative to the upper bound) along with the total range of results are shown in Figure 7, with the function $1-e^{-p_s/\lambda}$ drawn as a dashed line. As shown, the approximation factor converges to the optimal as the team size grows. The dip around $p_s = 0.85$ is due to looseness in the bound and the fact that the optimum is not yet reached by the greedy routine.

C. Large Scale Performance

We demonstrate the run-time of `GreedySurvivors` for large-scale problems by planning $K = 25$ paths for complete

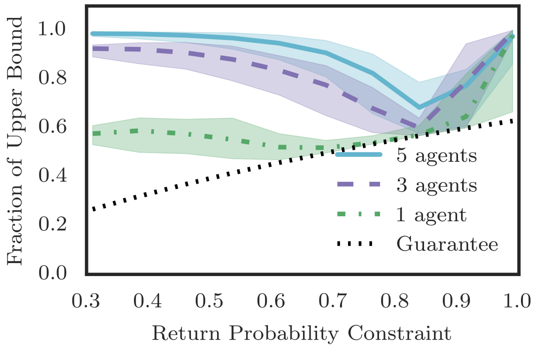


Fig. 7. Ratio of actual result to upper bound for a 65 node complete graph. The team size ranges from 1 (at the bottom) to 5 (at the top), and in all cases a significant fraction of the possible reward is accumulated even for small p_s.

graphs of various sizes. We use two `Orienteering` routines: the mixed integer formulation from [23] with Gurobi's MIP solver, and an adapted version of the open source heuristic developed by the authors of [24]. We use a heuristic approach because in practice it performs better than a polynomial time approximation algorithm. For the cases where we have comparison data (up to $V = 100$ nodes) the heuristic achieves an average of 0.982 the reward of the MIP algorithm. Even very large problems, e.g. 25 robots on a 900 node graph, can be solved in approximately an hour with the heuristic on a machine that has a 3GHz i7 processor using 8 cores and 64GB of RAM.

VI. Conclusion

In this paper we formulate the *Team Surviving Orienteers problem*, where we are asked to plan a set of paths that maximizes the expected number of nodes visited while guaranteeing that each individual robot survives with probability at least p_s. What sets this problem apart from previous work is the notion of risky traversal, where a robot might not complete its planned path. This creates a complex, history-dependent coupling between the edges chosen and the distribution of nodes visited, which precludes the application of existing approaches available for the traditional orienteering problem. We develop the `GreedySurvivors` algorithm which has polynomial time complexity and a constant-factor guarantee, demonstrate the effectiveness of our algorithm in numerical simulations, and discuss extensions to several variants of the TSO problem.

There are numerous directions for future work: First, an on-line version of this algorithm would react to knowledge of robot failure and re-plan the paths without exposing the surviving robots to more risk. Second, considering non-homogeneous teams would expand the many practical applications of the TSO problem. Third, extending the analysis to *walks* on a graph (where a robot can re-visit nodes) would allow for a broader set of solutions and may yield better performance. Finally, we are interested in using some of the concepts from [3] to consider more general probability models for the TSO.

Acknowledgements

The authors would like to thank Federico Rossi and Edward Schmerling for their insights which led to tighter analysis.

References

[1] B. L. Golden, L. Levy, and R. Vohra, "The orienteering problem," *Naval Research Logistics*, vol. 34, no. 3, pp. 307–318, 1987.

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

[3] A. M. Campbell, M. Gendreau, and B. W. Thomas, "The orienteering problem with stochastic travel and service times," *Annals of Operations Research*, vol. 186, no. 1, pp. 61–81, 2011.

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

[5] A. Gupta, R. Krishnaswamy, V. Nagarajan, and R. Ravi, "Approximation algorithms for stochastic orienteering," in *ACM-SIAM Symp. on Discrete Algorithms*, 2012.

[6] P. Varakantham and A. Kumar, "Optimization approaches for solving chance constrained stochastic orienteering problems," in *International Conference on Algorithmic Decision Theory*. Springer, 2013.

[7] P. Vansteenwegen, W. Souffriau, and D. Van Oudheusden, "The orienteering problem: A survey," *European Journal of Operational Research*, vol. 209, no. 1, pp. 1–10, 2011.

[8] 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.

[9] 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.

[10] V. Pillac, M. Gendreau, C. Guéret, and A. L. Medaglia, "A review of dynamic vehicle routing problems," *European Journal of Operational Research*, vol. 225, no. 1, pp. 1–11, 2013.

[11] 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.

[12] G. Laporte, F. Louveaux, and H. Mercure, "Models and exact solutions for a class of stochastic location-routing problems," *European Journal of Operational Research*, vol. 39, no. 1, pp. 71–78, 1989.

[13] B. L. Golden and J. R. Yee, "A framework for probabilistic vehicle routing," *AIIE Transactions*, vol. 11, no. 2, pp. 109–112, 1979.

[14] W. R. Stewart and B. L. Golden, "Stochastic vehicle routing: A comprehensive approach," *European Journal of Operational Research*, vol. 14, no. 4, pp. 371–385, 1983.

[15] A. Singh, A. Krause, C. Guestrin, and W. J. Kaiser, "Efficient informative sensing using multiple robots," *Journal of Artificial Intelligence Research*, vol. 34, pp. 707–755, 2009.

[16] G. A. Hollinger and G. S. Sukhatme, "Sampling-based motion planning for robotic information gathering." in *Robotics: Science and Systems*, 2013.

[17] N. Atanasov, J. Le Ny, K. Daniilidis, and G. J. Pappas, "Decentralized active information acquisition: theory and application to multi-robot slam," in *Proc. IEEE Conf. on Robotics and Automation*, 2015.

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

[19] G. Nemhauser, L. Wolsey, and M. Fisher, "An analysis of approximations for maximizing submodular set functions–I," *Mathematical Programming*, vol. 14, no. 1, pp. 265–294, 1978.

[20] K. Wei, R. Iyer, and J. Bilmes, "Fast multi-stage submodular maximization," in *International Conference on Machine Learning*, 2014, pp. 1494–1502.

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

[22] 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.

[23] İ. Kara, P. S. Biçakci, and T. Derya, "New formulations for the orienteering problem," *Procedia Economics and Finance*, vol. 39, pp. 849 – 854, 2016.

[24] P. Vansteenwegen, W. Souffriau, G. V. Berghe, and D. Van Oudheusden, "Iterated local search for the team orienteering problem with time windows," *Computers & Operations Research*, vol. 36, no. 12, pp. 3281–3290, 2009.