Risk-Aware Submodular Optimization for Stochastic Travelling Salesperson Problem

Rishab Balasubramanian, Lifeng Zhou, Pratap Tokekar, and P.B. Sujit

Abstract— We introduce a risk-aware variant of the Traveling Salesperson Problem (TSP), where the robot tour cost and reward have to be optimized simultaneously, while being subjected to uncertainty in both. We study the case where the rewards and the costs exhibit diminishing marginal gains, i.e., are submodular. Since the costs and the rewards are stochastic, we seek to maximize a risk metric known as Conditional-Value-at-Risk (CVaR) of the submodular function. We propose a Risk-Aware Greedy Algorithm (`RAGA`) to find an approximate solution for this problem. The approximation algorithm runs in polynomial time and is within a constant factor of the optimal and an additive term that depends on the value of optimal solution. We use the submodular function's curvature to improve approximation results further and verify the algorithm's performance through simulations.

I. INTRODUCTION

Determining an optimal tour to visit all the locations in a given set while minimizing/maximizing a metric is the classical Travelling Salesperson Problem (TSP) that finds applications in robotics, logistics, etc. However, there are several applications where the environment is dynamic and uncertain, as a result of which classical approaches to solving the TSP are insufficient. Examples include determining routes to visit active volcanic regions (where the activity has temporal variability) for obtaining scientific information (as shown in Fig. 1); determining routes for a logistic delivery vehicle in dense urban regions with uncertain traffic, etc. In such scenarios, the risk due to uncertainty in the travel times and/or the rewards collected along the path needs to be considered while determining the tour.

Several approaches to stochastic TSP have been presented in the literature. In [1], a two-step process to convert the TSP to a multi-integer linear programming problem and then introduce a meta-heuristic based on the probabilistic hedging method proposed in [2] is carried out. Paulin [3] uses an extension of Stein's method for exchangeable pairs to approach the stochastic salesperson problem. In [4], [5] and [6], the authors develop a genetic algorithm to approach the uncertainty in TSP. In [7], a constant factor approximation algorithm is developed for a Dubin's vehicle visiting all the points, with cost minimization being the main objective. In [8], again an approximation algorithm is presented to minimize the time taken to visit the targets that are appearing stochastically in the environment. In all the above articles, the risk is not considered directly.

We argue that a different approach is necessary in many risk-sensitive applications. Specifically, instead of optimizing the expected cost, optimizing a risk-sensitive measure may be more appropriate. In this paper, we focus on this case and present a risk-aware TSP formulation. To do so, we develop an approximation to the stochastic TSP with the optimization objective represented as a submodular function. The resulting algorithm takes as input a risk tolerance parameter, α, and produces a tour that maximizes the expected behavior in the worst α percentile cases. Thus, the user can choose tours ranging from risk-neutral ($\alpha = 1$) to very conservative ($\alpha \approx 0$).

An important property of submodular functions are their diminishing marginal values. The use of submodular functions is wide-spread: from information gathering [9] and image segmentation [10] to document summarization [11]. Rockafellar and Uryasev [12] introduce a relationship between a submodular function and the Conditional-Value-at-Risk (CVaR). CVaR is a risk metric that is commonly employed in stochastic optimization in finance and stock portfolio optimization. Another popular measure of risk is the Value-at-Risk (VaR) [13], which is commonly used to formulate the chance-constrained optimization problems. In [14], and [15], the authors study the chance-constrained optimization problem while also considering risk in the multi-robot assignment, and then extend it to a knapsack formulation. In a comparison between the VaR and CVaR, Majumdar and Pavone [16] propose that the CVaR is a better measure of risk for robotics, especially when the risk can cause a huge loss. In [17] and [18], a greedy algorithm for maximizing the CVaR is proposed.

Building on the work by [17], [18], in this paper, we develop a polynomial-time algorithm for approximating a solution to the stochastic TSP. Our method differs from the previous approaches due to the presence of uncertainty in the tour cost, making traditional path-planning algorithms fail. The framework presented in [17], [18] is effective only for one-stage planning (selecting a path amongst a set of candidate paths). In this paper, we present a multi-stage

Rishab Balasubramanian is Research Associate at IISER Bhopal, Bhopal – 462066, India (email: `rishab.edu@gmail.com`).

Lifeng Zhou was with the Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA, USA when part of the work was completed. He is currently with the GRASP Laboratory, University of Pennsylvania, Philadelphia, PA, USA (email: `lfzhou@seas.upenn.edu`).

Pratap Tokekar is Assistant Professor in the Department of Computer Science at the University of Maryland, College Park, MD 20742, USA. (email: `tokekar@umd.edu`).

P.B. Sujit is Associate Professor in the Department of Electrical Engineering and Computer Science, IISER Bhopal, Bhopal – 462066, India (email: `sujit@iiserb.ac.in`).

(a) An image of an island, with the locations of active volcanoes highlighted in red

(b) Key sites of surveillance surrounding the volcanic sites

(c) Tour with low cost and low reward
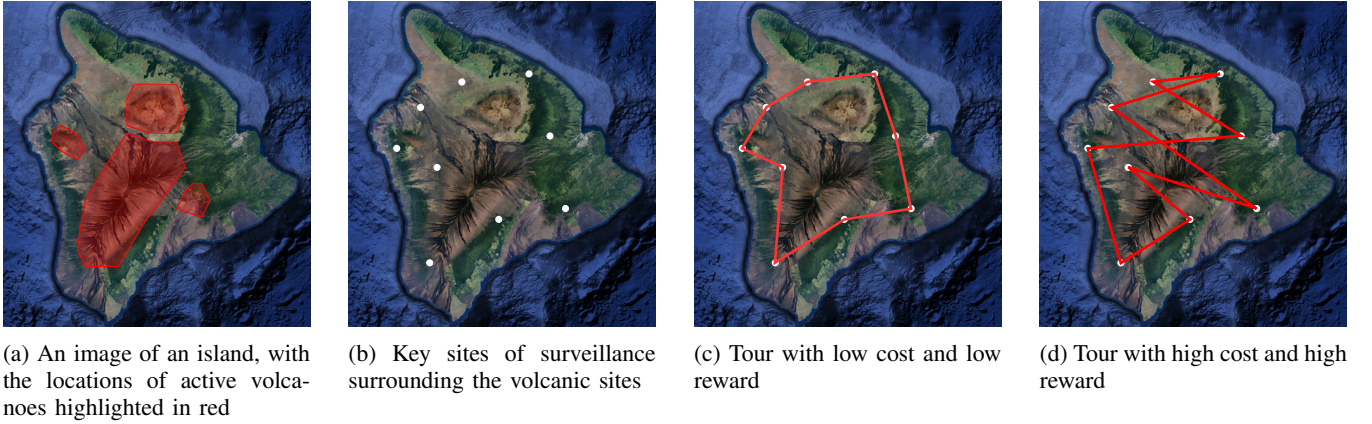
(d) Tour with high cost and high reward

Fig. 1: An example of risk-aware tour selection for volcano monitoring in an island. A low-risk, low-reward monitoring tour avoids the more interesting region in the middle which the high-risk, high-reward tour covers.

planner that finds a route taking the stochastic aspect into account. To achieve this, we propose an objective function that balances risk and reward for a tour and prove that this function is submodular. The method in [19] addresses the deterministic TSP with a reward-cost trade-off, while our work is focused on a stochastic version where the uncertainty in reward and cost is considered. The algorithm from [19] can be viewed as a special case of our algorithm, with $\alpha = 1$ and the subsequent risk ignored.

Contributions: The main contributions of this paper are:

- We present a risk-aware TSP with a stochastic objective that balances risk and reward for planing a TSP tour (Problem 1).
- We show the objective is submodular (Lemma 3) and propose a greedy algorithm (RAGA) to find tours that maximize the CVaR of a stochastic objective (Algorithm 1).
- We prove that the solution obtained by RAGA is within a constant approximation factor of the optimal and an additive term proportional to the optimal solution (Theorem 1) and prove that RAGA has a polynomial run-time (Theorem 2).
- We evaluate the performance of the algorithm through extensive simulations (Section V).

II. PRELIMINARIES

We first introduce the conventions and notations used in this paper. Calligraphic capital letters denote sets (e.g. $\mathcal{A}$). $2^{\mathcal{A}}$ denotes the power set of $\mathcal{A}$ and $|\mathcal{A}|$ represents its cardinality. Given a set $\mathcal{B}$, $\mathcal{A} \setminus \mathcal{B}$ denotes set difference. Let x be a random variable, then $\mathbb{E}[x]$ represents the expectation of the random variable x, and $\mathbb{P}[\cdot]$ denotes its probability.

A. Set and Function Properties

Optimization problems generally work over a set system $(\mathcal{X}, \mathcal{Y})$ where $\mathcal{X}$ is the base set and $\mathcal{Y} \subseteq 2^{\mathcal{X}}$. A reward/cost function $f : \mathcal{Y} \to \mathbb{R}$ is then either maximized or minimized.

Definition 1: (Monotonically Increasing): A set function $f : \mathcal{Y} \to \mathbb{R}$ is said to be monotonically increasing if and only if for any set $\mathcal{S}' \subseteq \mathcal{S} \in 2^{\mathcal{X}}$, $f(\mathcal{S}') < f(\mathcal{S})$.

Definition 2: (Submodularity): A function $f : 2^{\mathcal{X}} \to \mathbb{R}$ is submodular if and only if
$$f(\mathcal{S}) + f(\mathcal{T}) \geq f(\mathcal{S} \cup \mathcal{T}) + f(\mathcal{S} \cap \mathcal{T}), \ \forall \mathcal{S}, \mathcal{T} \in 2^{\mathcal{X}}.$$

Definition 3: (Matroid): An independence set system $(\mathcal{X}, \mathcal{Y})$ is called a matroid if for any sets $\mathcal{S}, \mathcal{P} \in 2^{\mathcal{X}}$ and $|\mathcal{P}| \leq |\mathcal{S}|$, it must hold that there exists an element $s \in \mathcal{S} \setminus \mathcal{P}$ such that $\mathcal{P} \cup \{s\} \in \mathcal{Y}$.

Definition 4: (Curvature): Curvature is used as a measure of the degree of submodularity of a function f. Consider the matroid pair $(\mathcal{X}, \mathcal{Y})$, and a function $f : 2^{\mathcal{X}} \to \mathbb{R}$, such that for any element $s \in \mathcal{X}, f(\{s\}) \neq 0$. The curvature $k, 0 \leq k \leq 1$ is then defined as:
$$k = 1 - \min_{s \in \mathcal{S}, \mathcal{S} \in \mathcal{Y}} \frac{f(\mathcal{S}) - f(\mathcal{S} \setminus \{s\})}{f(\{s\})}. \quad (1)$$

B. Travelling Salesperson Problem

Definition 5: (TSP): Given a complete graph $G(\mathcal{V}, \mathcal{E})$, the objective of the TSP is to find a minimum cost (maximum reward) Hamiltonian cycle.

In this paper, we consider the symmetric undirected TSP, where each edge has a reward and cost associated with it.

C. Measure of Risk

Let $f(\mathcal{S}, y)$ denote a utility function with solution set $\mathcal{S}$ and noise y. As a result of y, the value of $f(\mathcal{S}, y)$ is a random variable for every $\mathcal{S}$.

Definition 6: (Value at Risk): The Value at Risk (VaR) is defined as:
$$VaR_{\alpha}(\mathcal{S}) = \min_{\tau \in \mathbb{R}} \{\mathbb{P}[f(\mathcal{S}, y) \leq \tau] \geq \alpha\}, \ \alpha \in (0, 1], \quad (2)$$

where α is the user-defined risk-level. A higher value of α corresponds to the choice of a higher risk level.

Definition 7: (Conditional Value at Risk): The Conditional-Value-at-Risk (CVaR) is defined as
$$CVaR_{\alpha}(\mathcal{S}) = \mathbb{E}_y[f(\mathcal{S}, y) \mid f(\mathcal{S}, y) \leq VaR_{\alpha}(\mathcal{S})]. \quad (3)$$

Maximizing the value of $\mathrm{CVaR}_\alpha(\mathcal{S})$ is equivalent to maximizing the auxiliary function $H(\mathcal{S}, \tau)$ (Theorem 2, [12]):

$$H(\mathcal{S}, \tau) = \tau - \frac{1}{\alpha}\mathbb{E}_y[\,(\tau - f(\mathcal{S}, y))^+\,], \qquad (4)$$

where $[t]^+ = t, \ \forall t \geq 0$ and 0 when $t < 0$.

Lemma 1: (Lemma 1, [17]) If $f(\mathcal{S}, y)$ is monotone increasing, submodular and normalized in set $\mathcal{S}$ for any realization of y, then the auxiliary function $H(\mathcal{S}, \tau)$ is monotone increasing and submodular but not necessarily normalized [1] in set $\mathcal{S}$ for any given τ.

Lemma 2: (Lemma 2, [17]) The auxiliary function $H(\mathcal{S}, \tau)$ is concave in $\tau, \forall \mathcal{S}$.

III. PROBLEM FORMULATION

In this section, we first discuss a risk-aware TSP and then formulate the problem as a stochastic optimization problem by using CVaR.

A. Risk-Aware TSP

In order to motivate our formulation, consider the scenario of monitoring active volcanoes using a robot (say aerial robot) on an island as shown in Fig. 1a, where the red-colored patches represent active volcanic regions. The important sites ($\mathcal{V}$) that the robot needs to visit are shown in Fig. 1b. These sites are strategic positions from which it is possible to observe the volcanic situation from a safe distance. The robot travels between these monitoring sites and receives a reward based on the information gathered while traversing this tour. In this work, we do not constrain vehicle motion in terms of distance or time. While traveling directly above the volcano, the robot faces a higher chance of failure (due to volcanic activity) but can gather more information (a higher reward), while traveling along a shorter path (less cost). On the other hand, while traveling around the volcano, the robot has a lower risk but must travel a longer distance (more cost) while also receiving a lower reward. Fig. 1c and Fig. 1d show the paths that could be adopted based on the risk level specified for the robot. Our objective is to find a suitable tour for a single robot while considering the risk threshold and the trade-off between path cost and reward required.

The monitoring task is modeled as a risk-aware TSP on a graph $G(\mathcal{V}, \mathcal{E})$ of an environment E with $|\mathcal{V}|$ sites of interest. The notation $\mathcal{E}$ represents the set of edges connecting the vertices. A representative information density map M of the environment E is shown in Fig. 2. The robot has sensors with a sensing range R. As it travels along the tour, the robot receives a reward based on the amount of information it collects and a penalty proportional to the tour's cost. We assume that both the reward and cost are random variables, with the reward being positive, and the cost of every edge having an upper-bound of C. We use $r(\mathcal{S}, y_r)$ and $c(\mathcal{S}, y_c)$ to denote the reward and cost for a set of edges $\mathcal{S}$, where $r(\mathcal{S})$ and $c(\mathcal{S})$ are the sum of rewards of all points observed and the costs incurred when travelling along the edges in
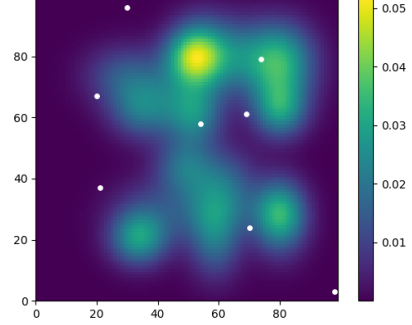
[1] The function $f(\mathcal{S}, y)$ is normalized in $\mathcal{S}$ if and only if $f(\emptyset, y) = 0$.

Fig. 2: Map M shows the information distribution in the environment E. The right bar shows the density degree of the information.

set $\mathcal{S}$ respectively, and y_r and y_c are the respective noises induced.

As we want to minimize cost and maximize reward simultaneously, our utility function $f(\mathcal{S}, y)$ is a combination of these two terms, with a weighting factor deciding the priority we place on the cost over the reward. We define $f(\mathcal{S}, y)$ as

$$f(\mathcal{S}, y) = (1 - \beta)\, r(\mathcal{S}, y_r) + \beta\,(|\mathcal{S}|C - c(\mathcal{S}, y_c)), \quad (5)$$

where $\beta \in [0, 1]$ is the weighting parameter. When $\beta = 0$, we ignore the cost incurred and consider the rewards received only. When $\beta = 1$, we ignore the rewards and are wary of only the cost penalized (classical TSP). Note that we directly incorporate the cost-reward trade-off into the utility function.

Let us define $r(\mathcal{S}, y_r)$ as $f_r(\mathcal{S}, y_r)$ and $|\mathcal{S}|C - c(\mathcal{S}, y_c)$ as $f_c(\mathcal{S}, y_c)$.

Lemma 3: The utility function $f(\mathcal{S}, y)$ is both submodular and monotone increasing in $\mathcal{S}$.

Proof: Let e be the edge to be added to the tour, where $e \in \mathcal{E} \setminus \mathcal{S}$ and $\mathcal{S}$ is set of edges selected so far. Let us consider the two parts of $f(\mathcal{S}, y)$ separately.

a) $f_r(\mathcal{S}, y)$: leftmargin=*

- As the rewards are sampled from a truncated Gaussian, with a lower bound of 0, they are thus always positive. Therefore, $f_r(\mathcal{S}, y)$ is always monotone increasing in $\mathcal{S}$,

$$f_r(\mathcal{S}) \leq f_r(\mathcal{S} \cup \{e\}). \qquad (6)$$

- While calculating the total reward, we add the rewards obtained from all *distinct* points observed while traversing this tour. Consider the current set of edges $\mathcal{S}$ and a new edge e. If the new edge e and the edges $\mathcal{S}$ have any overlap in the sensed regions, the total reward received from traversing $\mathcal{S}$ and e successively will be less than the sum of rewards of traversing $\mathcal{S}$ and e individually. Therefore,

$$f_r(\mathcal{S} \cup \{e\}) \leq f_r(\mathcal{S}) + f_r(\{e\}).$$

Also, since $e \in \mathcal{E} \setminus \mathcal{S}$, $f_r(\mathcal{S} \cap \{e\}) = 0$. Then we have,

$$f_r(\mathcal{S} \cup \{e\}) + f_r(\mathcal{S} \cap \{e\}) \leq f_r(\mathcal{S}) + f_r(\{e\}), \quad (7)$$

and thus $f_r(\mathcal{S}, y)$ is submodular in $\mathcal{S}$.

b) $f_c(\mathcal{S}, y)$:

- The cost of each edge is defined as a truncated Gaussian with a upper bound of M_c. For any set of edges in $\mathcal{S}$, the sum of costs will always be less than $|\mathcal{S}|M_c$, which means any sample (realization) of $f_c(\mathcal{S}, y)$ is positive. Therefore, $f_c(\mathcal{S}, y)$ is always monotone increasing in $\mathcal{S}$, i.e.,

$$f_c(\mathcal{S}) \leq f_c(\mathcal{S} \cup \{e\}). \qquad (8)$$

- Consider again, the current set of edges $\mathcal{S}$ and the new edge e. The total cost of traversing $\mathcal{S}$ and e is equal to the sum of costs of traversing $\mathcal{S}$ and e individually. Therefore,

$$f_c(\mathcal{S} \cup \{e\}) = f_c(\mathcal{S}) + f_c(\{e\}), \qquad (9)$$

and thus $f_c(\mathcal{S}, y)$ is modular in $\mathcal{S}$.

c) $f(\mathcal{S}, y)$: Since both $f_r(\mathcal{S}, y_r)$ and $f_c(\mathcal{S}, y_c)$ are monotone increasing in $\mathcal{S}$, $f(\mathcal{S}, y)$ as the summation of these two, is also monotone increasing in $\mathcal{S}$. Similarly, since $f_r(\mathcal{S}, y_r)$ is submodular in $\mathcal{S}$ and $f_c(\mathcal{S}, y_c)$ is modular in $\mathcal{S}$, $f(\mathcal{S}, y)$ as the summation of these two, is submodular in $\mathcal{S}$. □

B. Risk-Aware Submodular Maximization

Consider the set system $(\mathcal{E}, \mathcal{I})$, where $\mathcal{E}$ is the set of all edges in the graph $G(\mathcal{V}, \mathcal{E})$. If $\mathcal{A}_1, \mathcal{A}_2 \ldots \mathcal{A}_n$ each contains the edges forming n Hamiltonian tours, then $\mathcal{I} := 2^{\mathcal{A}_1} \cup 2^{\mathcal{A}_2} \ldots \cup 2^{\mathcal{A}_n}$. We define our *risk-aware TSP* by maximizing $\text{CVaR}_\alpha(\mathcal{S})$, where $\mathcal{S} \in \mathcal{I}$. We know that maximizing the $\text{CVaR}_\alpha(\mathcal{S})$ is equivalent to maximizing the auxiliary function $H(\mathcal{S}, \tau)$. Thus, we formally define the problem as:

Problem 1 (Risk-aware TSP):

$$\max_{\mathcal{S} \in \mathcal{I}, \ \tau \in [0, \Gamma]} \tau - \frac{1}{\alpha} \mathbb{E}_y[(\tau - f(\mathcal{S}, y))^+], \qquad (10)$$

where Γ is the upper bound on the value of τ.

IV. Algorithm And Analysis

In this section, we present a risk-aware greedy algorithm (RAGA) that extends the deterministic algorithm in [19] for solving Problem 1. We first explicitly introduce RAGA, then analyze its performance in terms of approximation bound and running efficiency.

A. Algorithm

RAGA has three main stages:
(i) Initialization (lines 2 - 3) The decision set $\mathcal{S}$ (Hamiltonian cycle) is initialized as an empty set. A degree vector D is set as $\mathbf{0}_{1 \times |\mathcal{V}|}$, which contains vertices' degrees in order. We use variables $\hat{H}_{\max} = 0$ and $\hat{H}_{\text{cur}} = 0$ store the maximal and the current values of $\hat{H}(\mathcal{S}, \tau)$ (line 2).
(ii) Search for valid tour for every τ (lines 4 - 20): For a specific value of τ, we continue to add edges greedily until we have a complete tour or the edge set $\mathcal{E}$ is empty. For every edge $e \in \mathcal{E} \setminus \mathcal{S}$, we calculate the marginal gain of

---

Algorithm 1: Risk-aware greedy algorithm (RAGA)

Input:
Graph $G(\mathcal{V}, \mathcal{E})$; Risk level $\alpha \in (0, 1]$; Weighing factor $\beta \in [0, 1]$; Upper bound $\Gamma \in \mathbb{R}^+$ on τ; Searching factor $\gamma \in (0, \Gamma)$; Oracle function $\mathbb{O}$ that approximates $H(\mathcal{S}, \tau)$ as $\hat{H}(\mathcal{S}, \tau)$.

Output:
A Hamiltonian tour S^G and its respective τ^G.

1 **for** $i = \{0, 1, 2, \ldots, \lceil \frac{\Gamma}{\gamma} \rceil\}$ **do**
2    $\mathcal{S} = \emptyset$; $D = \mathbf{0}_{1 \times |\mathcal{V}|}$; $\hat{H}_{\max} = 0$; $\hat{H}_{\text{cur}} = 0$
3    $\tau_i = i\gamma$
4   **while** $\mathcal{E} \neq \emptyset$ **and** $|\mathcal{S}| \leq |\mathcal{V}|$ **do**
5      $e^\star = \text{argmax}_{\forall e \in \mathcal{E} \setminus \mathcal{S}} \hat{H}(\mathcal{S} \cup \{e\}, \tau_i) - \hat{H}(\mathcal{S}, \tau_i)$
6      $(u, v) \leftarrow \mathcal{V}(e^\star)$
7   flag = False
8   **if** $D[u] < 2$ **and** $D[v] < 2$ **then**
9   check subtour existence in $\mathcal{S} \cup \{e^\star\}$ by running DFS
10   **if** *no subtour present* **then**
11   flag = True
12   **end**
13   **end**
14   **if** *flag == True* **then**
15        $\mathcal{S} \leftarrow \mathcal{S} \cup \{e^\star\}$
16        $D[u]+ = 1$; $D[v]+ = 1$
17        $\hat{H}_{\text{cur}} = \hat{H}(\mathcal{S} \cup \{e^\star\}, \tau_i)$
18   **end**
19      $\mathcal{E} \leftarrow \mathcal{E} \setminus \{e^\star\}$
20   **end**
21   **if** $\hat{H}_{cur} \geq \hat{H}_{max}$ **then**
22      $\hat{H}_{\max} = \hat{H}_{\text{cur}}$; $(\mathcal{S}^G, \tau^G) = (\mathcal{S}, \tau_i)$
23   **end**
24   **if** $\hat{H}_{cur} < 0$ **then**
25   **break**
26   **end**
27 **end**

---

the auxiliary function, $\hat{H}(\mathcal{S} \cup \{e\}) - \hat{H}(\mathcal{S})$ using the oracle function $\mathbb{O}$, and choose the edge e^* which maximizes the marginal gain (line 5). We then check if the selected edge e^* forms (or could form) a valid tour with the elements in $\mathcal{S}$ (i.e., if $\mathcal{S} \cup \{e^*\} \in \mathcal{I}$) (lines 8 - 13). Here, we first validate the degree constraint for each edge in $\mathcal{S} \cup \{e\}$, and then use the depth-first search (DFS) algorithm to iterate through the selected edges to check for subtours. If the new edge does not break the above constraints, we add the edge in $\mathcal{S}$ and update D and $\hat{H}_{cur}$ (lines 14 - 18). Finally, we remove e^* from $\mathcal{E}$ in line 19.
(iii) Selecting best tour set $(\mathcal{S}^G, \tau^G)$ (lines 21 - 23): For every tour we store the value of the auxiliary function $\hat{H}$ in $\hat{H}_{cur}$, and compare it to the best value $\hat{H}_{\max}$. We store the pair $(\mathcal{S}, \tau)$ whenever $\hat{H}_{cur} > \hat{H}_{\max}$, and update $\hat{H}_{\max}$.

Lines 24 - 26 show the condition of exiting the loop. As $H(\mathcal{S}, \tau)$ is concave in τ, and we start from a non-negative value (as seen in Fig. 6a), it is certain that if $H(\mathcal{S}, \tau)$
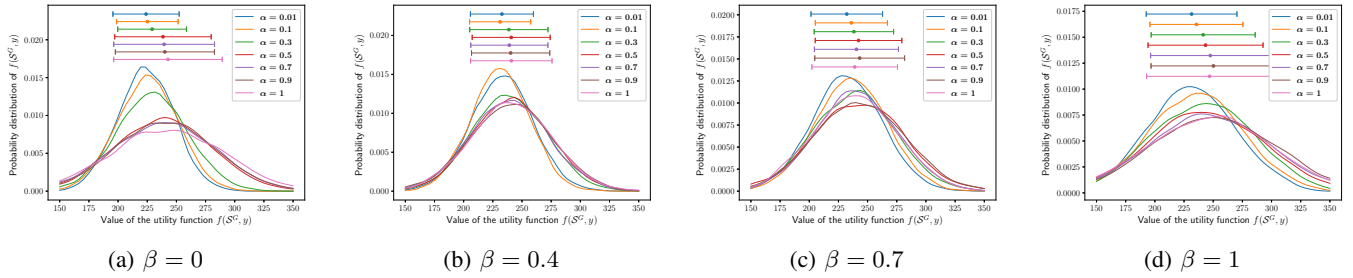
(a) $\beta = 0$ (b) $\beta = 0.4$ (c) $\beta = 0.7$ (d) $\beta = 1$

Fig. 3: Probability density function of $f(\mathcal{S}, y)$ at different values of α, keeping β fixed.

(a) $\alpha = 0.1$ (b) $\alpha = 0.4$ (c) $\alpha = 0.7$ (d) $\alpha = 1$

Fig. 4: Probability density function of $f(\mathcal{S}, y)$ at different values of β, keeping α fixed.

becomes negative, it will continue to further decrease. This improves the runtime of RAGA.

Designing the oracle function $\mathbb{O}$: We use a sampling based oracle function to approximate $H(\mathcal{S}, \tau)$ as $\hat{H}(\mathcal{S}, \tau)$. The authors of [20] have proved that if the number of samples $n_s = O(\frac{\Gamma^2}{\epsilon^2} \log \frac{1}{\delta})$, $\delta, \epsilon \in (0, 1)$, the approximation for the value of CVaR (or equivalently, the auxiliary function $\hat{H}(\mathcal{S}, \tau)$) gives an error less than ϵ with a probability of at least $1 - \delta$.

B. Performance Analysis

Theorem 1: Let $\mathcal{S}^G$ and τ^G be the tour and the searching scalar selected by RAGA and let $\mathcal{S}^\star$ and $\tau^\star$ be the tour and the searching scalar selected the optimal solution. Then,

$$H(\mathcal{S}^G, \tau^G) \geq \frac{1}{2+k}(H(\mathcal{S}^\star, \tau^\star) - \gamma) + \frac{1+k}{2+k}\Gamma(1 - \frac{1}{\alpha}) - \epsilon$$
(11)

with a probability of at least $1 - \delta$, when the number of samples $n_s = O(\frac{\Gamma^2}{\epsilon^2} \log \frac{1}{\delta})$, $\delta, \epsilon \in (0, 1)$. k is the curvature of $H(\mathcal{S}, \tau)$ with respect to $\mathcal{S}$.

Proof: Note that $H(\mathcal{S}, \tau)$ is monotone increasing, submodular, but not normalized (Lemma 1). [21, Theorem 6.1] and [19, Theorem 2.10] have shown that for a normalized submodular increasing function, the greedy algorithm gives an approximation of $\frac{1}{2+k}$ for maximizing it. Following this result, normalizing $H(\mathcal{S}, \tau)$ by $H(\mathcal{S}, \tau) - H(\emptyset, \tau)$, we get:

$$\frac{H(\mathcal{S}^G, \tau) - H(\emptyset, \tau)}{H(\mathcal{S}^\star, \tau) - H(\emptyset, \tau)} \geq \frac{1}{2+k}$$
(12)

where $H(\mathcal{S}^\star, \tau)$ is the optimal value of $H(\mathcal{S}, \tau)$ for any given value of τ. Then following the proof of Theorem 1 in [18], we have the bound approximation performance of RAGA in Equation (11). $\square$

Theorem 2: RAGA has a polynomial running time of $O(\lceil \frac{\Gamma}{\gamma} \rceil (|\mathcal{V}|^3 (|\mathcal{V}|n_p + n_s + 2\log|\mathcal{V}|+1)))$.

Proof: First, the outer *"for"* loop (lines 1-27) takes at most $\lceil \frac{\Gamma}{\gamma} \rceil$ time to search for τ. Second, the inner *"while"* loop (lines 4-20) needs to check for edges $e \in \mathcal{E}$ to find the tour $\mathcal{S}$, thus running at most $|\mathcal{E}|$ times.

Within the *"while"* loop, RAGA finds the edge $e^\star$ with maximal marginal gain in line 5. To compute the marginal gain for an edge e, RAGA calls the oracle function $\mathbb{O}$. As it estimates $\hat{H}(\mathcal{S}, \tau_i), \forall e \in \mathcal{E} \setminus \mathcal{S}$, $\mathbb{O}$ is called at most $|\mathcal{E}|$ times, during one iteration of the *"while"* loop in line 4. As RAGA needs to compute the marginal gains only when $\mathcal{S}$ is changed in line 15, and as $\mathcal{S}$ can have only $|\mathcal{V}|$ elements, it needs to perform a total of $|\mathcal{V}|$ recalculations. In addition, the edges need to be sorted every time a recalculation is performed, so that future calls to find the most beneficial element can be run in constant time without recomputing $\hat{H}(\mathcal{S}, \tau_i)$.

For any edge e, the oracle $\mathbb{O}$ must compute the new mean reward and cost of $\mathcal{S} \cup \{e\}$. As only the distinct points in the environment are considered while calculating reward, the oracle must maintain an array of points that were observed while traveling along $\mathcal{S}$. If the maximum points observed when traveling along any edge in $\mathcal{E}$ is n_p, then the maximum number of points observed while traversing $\mathcal{S}$ would be $|\mathcal{V}|n_p$. Afterward, the oracle takes n_s samples to compute $\hat{H}(\mathcal{S}, y)$. Thus the total runtime of the oracle is $|\mathcal{V}|n_p + n_s$. Therefore, the total time for computing gains and sorting would be $O(|\mathcal{V}|(|\mathcal{E}|(|\mathcal{V}|n_p + n_s) + |\mathcal{E}|\log|\mathcal{E}|))$.

Next, within the *"while"* loop, RAGA checks if a the selected edge e^* could form a valid tour with $\mathcal{S}$ (lines 8-13). In particular, it ensures the degree of the two vertices of edge $e^\star$ is less than two (line 8). If both vertices have a degree less than two, it runs DFS to traverse the elements in $\mathcal{S}$ to check for subtours (line 9), which takes at most $|\mathcal{V}|$

time. As this is performed for every edge e to be added to $\mathcal{S}$, in total, validating the selected edge runs in $O(|\mathcal{V}||\mathcal{E}|)$ time.

Assuming that all other commands take constant time, RAGA has a runtime of $O(|\mathcal{V}|(|\mathcal{E}|(|\mathcal{V}|n_p + n_s) + |\mathcal{E}|\log|\mathcal{E}|) + |\mathcal{V}||\mathcal{E}|)$ for one iteration of the *"for"* loop. Notably, for a complete graph, $|\mathcal{E}| = O(|\mathcal{V}|^2)$. Considering that the *"for"* loop has $\lceil \frac{\Gamma}{\gamma} \rceil$ iterations, the total runtime of RAGA becomes $O(\lceil \frac{\Gamma}{\gamma} \rceil(|\mathcal{V}|^3(|\mathcal{V}|n_p + n_s + 2\log|\mathcal{V}|+1)))$. $\qquad\square$

V. SIMULATION RESULTS

The performance of RAGA is evaluated through extensive simulations with various environment maps and varying numbers and locations of nodes.[2]

Simulation setup. We consider the scenario with the number of sites (nodes) $|\mathcal{V}| = 8$ in a 2D environment of size $100m \times 100m$ meters. The sensing radius of the robot is $R = 2$ meters. We assume both the reward and cost of an edge are modeled as a truncated Gaussian distribution, but RAGA can handle other distributions as well since it only requires samples of the distributions to approximate CVaR. Similarly, the noise terms y_r and y_c are assumed to be Gaussian. However, the terms are generic to accommodate any other distributions. Given the information density map M (Fig. 2), in which each point depicts the mean reward obtained from observing that position. The average reward for an edge $r(e)$ is the sum of rewards of all points on M sensed while traversing this edge. The mean cost $c(e)$ of an edge is proportional to its length. We assume the variances of the edge reward is proportional to its mean $r(e)$ and variance in the cost to be proportional to $C - c(e)$. The average rewards and costs for every edge are normalized and re-scaled to a maximum value of 10. We set the number of samples as $n_s = 250$.

Results. Fig. 3 shows the probability distributions of the utility function $f(\mathcal{S}, y)$ as a function of α, with a fixed β. From Fig. 3c, when the risk level α is small, a path with a lower mean utility value $f(\mathcal{S}, y)$ is chosen. This is because a lower α indicates a small risk level, and therefore a low-utility low-risk path is selected. As the value of α increases, we see that the paths selected are more rewarding but tend to have a higher variance. The case of $\alpha = 1$ is the risk-neutral scenario, where we disregard the variance in reward and cost but rather focus only on the mean values, which is the same as the deterministic settings in [19]. As expected, with $\alpha = 1$, we select tours with the highest risk. This gradation in tour selection thus illustrates RAGA's ability to select paths based on its evaluation of reward-risk trade-off.

Fig. 4 shows the probability distributions of the utility function $f(\mathcal{S}, y)$ as a function of β, when α is given. Let us use Fig. 4b as an example. We see that initially, for $\beta = 0$, RAGA selects a high-variance high-utility path. This is because, at $\beta = 0$, we only consider maximizing the reward and disregard the cost of the tour. As we increase β, we see that RAGA slowly shifts towards paths with lesser utility (lesser risk), with the most conservative paths chosen around
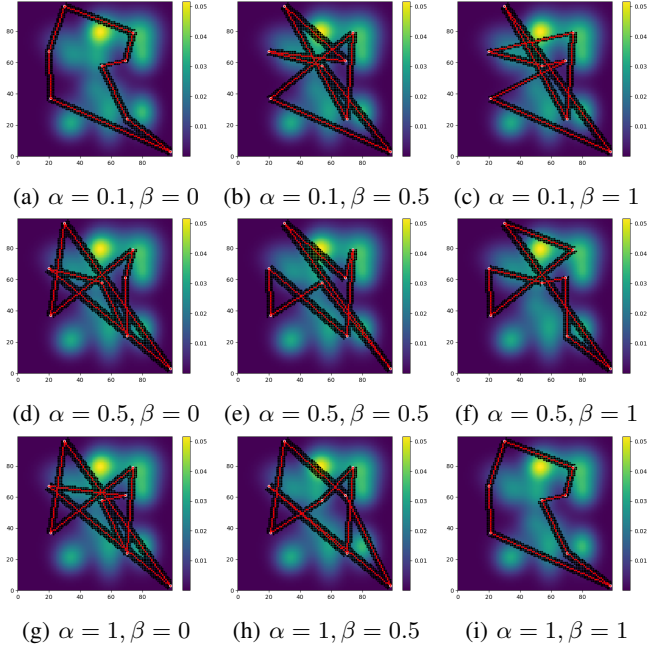

(a) $\alpha = 0.1, \beta = 0$ (b) $\alpha = 0.1, \beta = 0.5$ (c) $\alpha = 0.1, \beta = 1$

(d) $\alpha = 0.5, \beta = 0$ (e) $\alpha = 0.5, \beta = 0.5$ (f) $\alpha = 0.5, \beta = 1$

(g) $\alpha = 1, \beta = 0$ (h) $\alpha = 1, \beta = 0.5$ (i) $\alpha = 1, \beta = 1$

Fig. 5: The tours chosen for varying values of α and β.

$\beta = 0.2 \sim 0.4$. At this point, we wish to minimize cost while maximizing reward simultaneously, and therefore are more cautious of both terms. As β continues to increase further, we see that the tours with higher utilities are chosen, which have a higher variance. Finally, at $\beta = 1$, we consider only the cost minimization and select a tour with high-variance high-utility as the case of $\beta = 0$. Table I provides a quantitative representation of Fig. 3 and 4 by showing the variations in $f_r(\mathcal{S}, y)$ and $f_c(\mathcal{S}, y)$ with respect to α and β.

Fig. 5 shows the tours chosen by RAGA for different values of α and β. We can clearly see that the paths chosen in Fig. 5a and Fig. 5i are the same and Fig. 5c and Fig. 5g are similar. This is because we get a lower reward when we travel less and a higher reward when we travel more, thus highlighting our problem's true dual nature.

Fig. 6a shows the value of $H(\mathcal{S}, \tau)$ with respect to τ, and it can be seen that the auxiliary function is concave. Fig. 6b shows the computational time for varying number of vertices for $\alpha = 0.1, 0.5, 0.9$. We use $n_s = 250$, $\Gamma = 200$, and $\gamma = 1$. The results show that the running time increases with increase in α as for a larger α, reaching the maximum is slower. This is shown in Fig. 6a where the maximum for $H(\mathcal{S}, \tau)$ is reached at a larger value of τ when α is larger. Combining this, and lines 24 - 26 in RAGA, we can see that with a smaller value of α, RAGA stops quickly. As a result, the number of iteration over τ performed is lesser, and RAGA reaches a solution earlier than for a larger α. To show the scalability of RAGA, we test it on an environment of size $500m \times 500m$ consisting of 20 nodes along with the generated tour as shown in Fig. 6c. The parameters for the robot are set as $R = 1$, $\alpha = 0.1$ and $\beta = 0.8$.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we developed a risk-aware greedy algorithm (RAGA) for the TSP, while considering the risk-reward trade-

---

[2]RAGA code is available at https://github.com/rishabbala/Risk-Aware-TSP

β α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0.01	122 / 145	121 / 144	124 / 141	126 / 134	129 / 136	127 / 136	132 / 138	127 / 133	130 / 131	132 / 138	137 / 135
0.1	121 / 149	122 / 148	121 / 149	125 / 146	121 / 145	136 / 132	136 / 137	135 / 136	135 / 133	130 / 131	134 / 131
0.2	124 / 144	121 / 143	124 / 140	124 / 147	134 / 139	137 / 139	144 / 134	129 / 138	138 / 138	138 / 130	134 / 133
0.3	138 / 137	127 / 142	133 / 133	135 / 136	138 / 138	138 / 140	131 / 144	140 / 138	123 / 144	127 / 150	126 / 141
0.4	133 / 138	142 / 134	136 / 135	129 / 137	136 / 139	138 / 140	130 / 141	139 / 136	128 / 147	127 / 148	126 / 147
0.5	141 / 135	138 / 135	144 / 133	141 / 138	142 / 137	136 / 139	138 / 139	127 / 148	138 / 140	125 / 147	129 / 143
0.6	138 / 131	141 / 134	141 / 134	142 / 132	142 / 137	134 / 142	129 / 142	134 / 141	130 / 144	124 / 146	130 / 146
0.7	143 / 133	138 / 141	138 / 138	139 / 138	139 / 138	137 / 138	134 / 141	131 / 143	136 / 140	129 / 146	129 / 147
0.8	143 / 135	138 / 137	141 / 137	144 / 133	142 / 137	142 / 137	141 / 138	143 / 135	129 / 146	122 / 148	126 / 147
0.9	144 / 133	139 / 139	142 / 135	144 / 133	142 / 137	140 / 139	143 / 136	128 / 148	131 / 146	121 / 149	122 / 150
1	143 / 132	144 / 132	144 / 132	145 / 134	145 / 134	137 / 137	134 / 141	135 / 139	121 / 149	121 / 149	128 / 147

TABLE I: Variations of $f_r(\mathcal{S}, y)/f_c(\mathcal{S}, y)$ with α and β.

(a) Variation of $H(\mathcal{S}, \tau)$ with respect to τ

(b) Runtime performance of RAGA
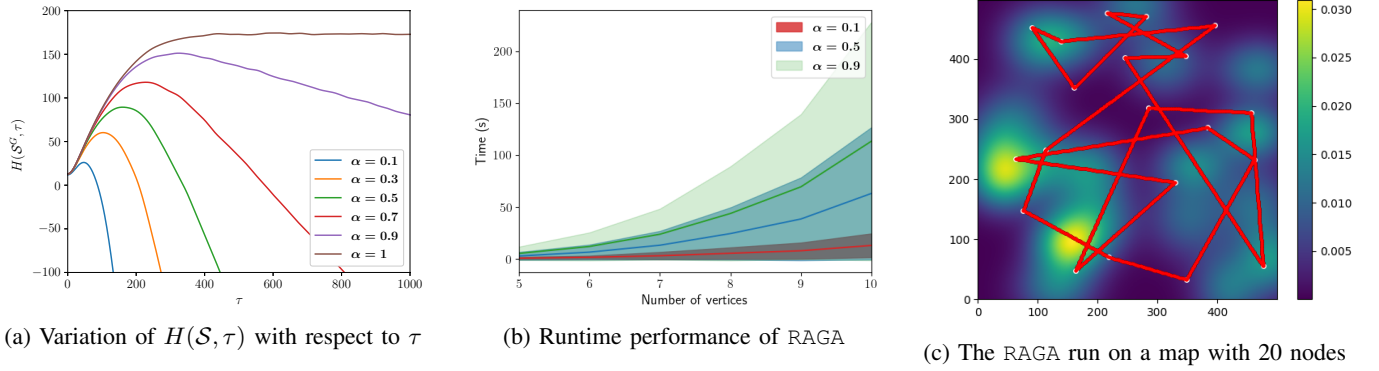
(c) The RAGA run on a map with 20 nodes

Fig. 6: Performance of RAGA.

off. We used a CVaR submodular maximization approach for selecting a solution set under matroidal constraints. We also analyzed the performance of RAGA and its running time. The results show RAGA's efficiency in optimizing both reward and cost simultaneously.

An ongoing work is to improve the running time of RAGA as a larger value of α has a higher running time. Another future avenue is to extend RAGA to address arbitrary distributions for the reward and cost, based on the real-world data. Further, the work can be extended for other types of combinatorial optimization problems, and for the cases where the risk associated with the graph can be learned, and the paths can be determined dynamically. Finally, we also plan to extend RAGA to the stochastic version of multi-robot TSP [22] where we plan tours for multiple robots with stochastic rewards and costs.

REFERENCES

[1] G. Perboli, L. Gobbato, and F. Maggioni, "A progressive hedging method for the multi-path travelling salesman problem with stochastic travel times," *IMA JMM*, vol. 28, no. 1, pp. 65–86, 2017.

[2] R. T. Rockafellar and R. J.-B. Wets, "Scenarios and policy aggregation in optimization under uncertainty," *Mathematics of operations research*, vol. 16, no. 1, pp. 119–147, 1991.

[3] D. Paulin *et al.*, "The convex distance inequality for dependent random variables, with applications to the stochastic travelling salesman and other problems," *Electronic Journal of Probability*, vol. 19, 2014.

[4] S. Maity, A. Roy, and M. Maiti, "A modified genetic algorithm for solving uncertain constrained solid travelling salesman problems," *Computers & Industrial Engineering*, vol. 83, pp. 273–296, 2015.

[5] ——, "An imprecise multi-objective genetic algorithm for uncertain constrained multi-objective solid travelling salesman problem," *Expert Systems With Applications*, vol. 46, pp. 196–223, 2016.

[6] A. Mukherjee, G. Panigrahi, S. Kar, and M. Maiti, "Constrained covering solid travelling salesman problems in uncertain environment," *JAIHC*, vol. 10, no. 1, pp. 125–141, 2019.

[7] S. Itani and M. A. Dahleh, "On the stochastic tsp for the dubins vehicle," in *ACC 2007*, pp. 443–448.

[8] A. Adler and S. Karaman, "The stochastic traveling salesman problem and orienteering for kinodynamic vehicles," in *IEEE ICRA*, 2016, pp. 2788–2795.

[9] A. Krause and C. Guestrin, "Submodularity and its applications in optimized information gathering," *ACM TIST*, vol. 2 (4), pp. 1–20, 2011.

[10] Y. Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary & region segmentation of objects in nd images," in *IEEE ICCV 2001*, vol. 1, 2001, pp. 105–112.

[11] H. Lin and J. Bilmes, "A class of submodular functions for document summarization," in *Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 2011, pp. 510–520.

[12] R. T. Rockafellar and S. Uryasev, "Optimization of conditional value at risk. journal of risk," *Volume*, vol. 2, 2000.

[13] J. Morgan, "Creditmetrics," *JP Morgan, New York*, 1997.

[14] F. Yang and N. Chakraborty, "Algorithm for optimal chance constrained linear assignment," in *IEEE ICRA*, 2017, pp. 801–808.

[15] ——, "Algorithm for optimal chance constrained knapsack problem with applications to multi-robot teaming," in *IEEE ICRA*. IEEE, 2018, pp. 1043–1049.

[16] A. Majumdar and M. Pavone, "How should a robot assess risk? towards an axiomatic theory of risk in robotics," in *Robotics Research*. Springer, 2020, pp. 75–84.

[17] L. Zhou and P. Tokekar, "An approximation algorithm for risk-averse submodular optimization," in *WAFR*. Springer, 2018, pp. 144–159.

[18] ——, "Risk-aware submodular optimization for multi-robot coordination," *arXiv preprint arXiv:2003.10492*, 2020.

[19] S. T. Jawaid and S. L. Smith, "The maximum traveling salesman problem with submodular rewards," in *ACC 2013*, pp. 3997–4002.

[20] N. Ohsaka and Y. Yoshida, "Portfolio optimization for influence spread," in *WWW Conference*, 2017, pp. 977–985.

[21] M. Conforti and G. Cornuéjols, "Submodular set functions, matroids and the greedy algorithm: tight worst-case bounds and some generalizations of the rado-edmonds theorem," *Discrete applied mathematics*, vol. 7, no. 3, pp. 251–274, 1984.

[22] G. N. Frederickson, M. S. Hecht, and C. E. Kim, "Approximation algorithms for some routing problems," in *IEEE Annual symposium on foundations of computer science*, 1976, pp. 216–227.