Proceedings of Machine Learning Research vol 242:1464–1476, 2024

Multi-Agent Coverage Control with Transient Behavior Consideration

Runyu Zhang* RUNYUZHANG@FAS.HARVARD.EDU
Haitong Ma* HAITONGMA@G.HARVARD.EDU
Na Li NALI@SEAS.HARVARD.EDU
School of Engineering and Applied Science, Harvard University†

Abstract

This paper studies the multi-agent coverage control (MAC) problem where agents must dynamically learn an unknown density function while performing coverage tasks. Unlike many current theoretical frameworks that concentrate solely on the regret occurring at specific targeted sensory locations, our approach additionally considers the regret caused by transient behavior – the path from one location and another. We propose the multi-agent coverage control with the doubling trick (MAC-DT) algorithm and demonstrate that it achieves (approximated) regret of $\widetilde{O}(\sqrt{T})$ even when accounting for the transient behavior. Our result is also supported by numerical experiments, showcasing that the proposed algorithm manages to match or even outperform the baseline algorithms in simulation environments. We also show how our algorithm can be modified to handle safety constraints and further implement the algorithm on a real-robotic testbed.

Keywords: Multi-agent coverage control, Gaussian Processes, Bayesian optimization, no-regret learning

1. Introduction

In multi-agent coverage control (MAC) problems, there is a group of agents collectively tasked with efficiently exploring and covering an environment typically characterized by certain density functions. MAC problems find a lot of applications such as sensor networks (Krause et al., 2006), search and rescue (Wasim et al., 2020), underwater exploration (Karapetyan et al., 2018) and habitat monitoring (Mainwaring et al., 2002) etc. In the extensive studies of the MAC problem, classical approaches to coverage control (Cortes et al., 2004; Cortés and Bullo, 2005; Cortes et al., 2005; Lekien and Leonard, 2009; Hussein and Stipanovic, 2007; Bullo et al., 2012; Durham et al., 2011) generally assume a priori knowledge of the density function and employ Lloyd's algorithm (Lloyd, 1982) to guarantee the convergence of agents to a local minimum of the coverage cost.

Recent investigations have expanded this paradigm to accommodate scenarios where the density function is unknown. In such cases, agents must simultaneously conduct coverage tasks and learn the density function dynamically. A prevalent approach involves modeling the density function as a Gaussian process (GP) and employing non-parametric learning through sensor measurements. In order to achieve good performance, agents need to balance exploration and exploitation, collecting informative samples to learn the density function (exploration) while concurrently converging to optimal coverage locations (exploitation). In particular, a specific subset of existing research focuses on the development of practical, adaptive, and distributed algorithms for coverage control utilizing the GP model (Luo and Sycara, 2018; Luo et al., 2019; Choi et al., 2008; Kemna et al., 2017; Nakamura et al., 2022). Despite their practical relevance, these endeavors lack a rigorous theoretical analysis for performance guarantees. An alternative strand of research, represented by (Carron et al., 2015; Todescato et al., 2017), offers asymptotic guarantees of convergence to near-local optimal solutions, yet the convergence rate is not studied. Recent contributions (Prajapat et al., 2022; Benevento et al., 2020; Santos et al., 2021) have sought to address this void by proposing algorithms with convergence rate guarantees, particularly in the context of regret. It is noteworthy, however, that most papers only consider regret accumulated at the targeted sensory locations that agents travel between while neglecting regrets along the path where agents travel from one location to the next location. For applications that involve a physical moving sensor that can not move to a new point instantaneously or quickly, regret along the paths may be substantial. This observation motivates us to design an algorithm for the MAC problem with rigorous regret guarantees that explicitly account for the transient behavior.

* The first two authors contributed equally to this work.
† This work was funded by NSF AI institute: 2112085, NSF ASCENT: 2328241, NSF CNS: 2003111.

Our contribution: In this paper, we study the MAC problem under an unknown density function. We present a novel algorithm, multi-agent coverage control with doubling trick (MAC-DT), demonstrating an $\widetilde{O}(\sqrt{T})$ regret, even when accounting for transient behavior. The algorithm leverages the Upper Confidence Bound (UCB) technique, wherein, during each episode, it computes the UCB of the reward map and assigns a specific sensory location to each agent. Subsequently, each agent plans its path toward the designated location. The termination condition for each episode is determined by the 'doubling trick' (detailed explanation in Algorithm 1). Our research is closely aligned with the works of (Prajapat et al., 2022) and (Wei et al., 2021). Our proposed algorithm bears resemblance to the MACOPT algorithm introduced by (Prajapat et al., 2022), employing the UCB of the Gaussian Process to navigate the trade-off between exploration and exploitation. However, Prajapat et al. (2022) do not consider transient behavior, while our algorithm incorporates this aspect by carefully designing episodes using the doubling trick. Though Wei et al. (2021) addresses regret in the presence of transient behavior, due to differences in problem settings and algorithmic design, they achieve a slightly inferior regret rate of $\widetilde{O}(T^{2/3})$. Further, the regret defined in (Wei et al., 2021) is with respect to a local-optimal solution, where our (approximated-) regret is defined by the global-optimal solution. Our results are also supported by numerical studies, suggesting that the MAC-DT algorithm can match or even outperform the aforementioned baseline algorithms. Further, our algorithm can naturally be combined with safety considerations and operate in settings with obstacles or safety constraints. Lastly, we also validate the algorithm on a physical robotic testbed with three quadrotors covering an area.

Due to space limit, we defer some of the auxiliary proofs and numerical details into the online version of the paper (Zhang et al., 2024).

Other related works In the setting where the density function is known, beyond Lloyd's algorithm-based approaches, there are alternative methods leveraging submodularity (Krause and Guestrin, 2011; Nemhauser et al., 1978; Feige, 1998) to address the MAC problem (Ramaswamy and Marden, 2016; Sun et al., 2017). In scenarios where the density function is unknown, various papers (Schwager et al., 2009, 2015) explore parametric estimation as an alternative to GP modeling. These algorithms model the function as a linear combination of basis functions, aiming to learn the weights associated with each basis function. Furthermore, alternative strategies (Davison et al., 2014; Choi and Horowitz, 2010) take a distinctive route by not involving the identification of the unknown probability density function. Instead, they solely rely on random samples from the environment to determine the agents' coverage locations. Apart from traditional algorithmic approaches to coverage control, recent studies have also delved into the application of reinforcement learning techniques (Faryadi and Mohammadpour Velni, 2021; Din et al., 2022; Battocletti et al., 2021), which incorporates the power of learning and adaptation into the field.

2. Problem Setup and Preliminaries

2.1. Multi-agent Coverage Control

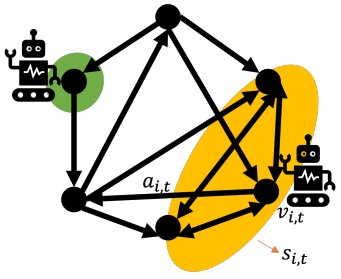

Figure 1: Multi-agent Coverage Control.

Consider the MAC problem on a connected directed graph $G = \{V, E\}$, where V are the vertices of the graph and E are the edges. The density function/reward map is given by $w : V \to \mathbb{R}^+$, where each vertex $v \in V$ is associated with a reward $w(v) \geq 0$. For notational simplicity we also use $w(S)$ to denote the entrywise function evaluation on the grid subset $S \subseteq V$. We use the notation $E(v) \subseteq E$ to denote the set of edges whose source vertex is v. It is also assumed that the graph is connected and with diameter D. There are N agents/robots located on the vertices of the graph and at each time step t agent i chooses an action $a_{i,t}$ from the edge set $E(v_{i,t})$, where $v_{i,t}$ is agent i's location at time t, and then transit to the end vertex of $a_{i,t}$. We assume that when located at vertex v, agent i can cover a certain surrounding area of its current location, which is denoted as $s_i(v) \subseteq V$. For example, $s_i(v)$ can be just the vertex v that agent i is at, or a κ-hop neighborhood area of v. Also note that agents may have different covering ability, that is, two different agent i, j may have different $s_i(v)$ and $s_j(v)$ even at the same location. For simplicity, we denote $s_{i,t} := s_i(v_{i,t})$. We also use $s_t := \cup_{i=1}^N s_{i,t}$ to denote the total area covered by all agents. It is assumed that at every location v each agent's covering area is smaller than size K, i.e. $|s_i(v)| \leq K$ for

all $v \in V$. At each time step, each agent could select one vertex $v_{i,t}^{\text{eval}} \in s_{i,t}$ and observe a noisy reward value which is a random variable $Y(v_{i,t}^{\text{eval}}) := w(v_{i,t}^{\text{eval}}) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2)$ is some Gaussian noise. We also denote $s_t^{\text{eval}} := \cup_{i=1}^N v_{i,t}^{\text{eval}}$. For vertex v, $n_v(t)$ denotes the number of times that vertex v has been sampled (i.e. chosen as the evaluation point) until time t, i.e. $n_v(t) = \sum_{\tau=1}^t \mathbf{1}\{v \in s_\tau^{\text{eval}}\}$.

At time t, the total covered reward is given by $\|w(s_t)\|_1 := \sum_{v \in s_t} w(v)$, and the objective is to cover as much reward as possible. The optimal coverage value is denoted as $s^\star := \operatorname{argmax}_{s=\cup_{i=1}^N s_i(v_i), v_i \in V} \|w(s)\|_1$. For an algorithm $\texttt{Alg}$ that plans the path $v_{i,t}(\texttt{Alg})$ of the agents, we define the regret of the algorithm as

$$R(\texttt{Alg}, T, w) := T\|w(s^\star)\|_1 - \sum_{t=1}^T \|w(s_t(\texttt{Alg}))\|_1. \tag{1}$$

We also define the α-approximated regret as

$$R^\alpha(\texttt{Alg}, T, w) := \alpha T\|w(s^\star)\|_1 - \sum_{t=1}^T \|w(s_t(\texttt{Alg}))\|_1. \tag{2}$$

Compared with the regret definition in (Prajapat et al., 2022) which considers regret accumulated only at the targeted sensory locations, we account for the regret associated with the path from one target location to the next. In many energy-constrained exploration tasks, the regret during the transient phase is important and should not be neglected. For instance, Mars exploration rovers aim to maximize coverage within a limited life-cycle traveling distance; agricultural spray drones must optimize coverage in unsprayed areas within constrained flight time. In both cases, regret during the transient phase demands careful consideration in the algorithmic design.

2.2. Gaussian Processes

Some assumptions on the reward map w are required to guarantee no-regret. Here we assume that the reward distribution w is sampled from a Gaussian Process (GP) $\mathcal{GP}(\mu, \kappa)$,[1] which is specified by a mean function $\mu : V \to \mathbb{R}$ and a covariance function $\kappa : V \times V \to \mathbb{R}$. It represents a collection of dependent random variables, one for each $v \in V$, every finite subset $s \subseteq V$ of which is multivariate Gaussian distribution with mean $\mu(s)$ and variance $\kappa(s, s)$. Here for subsets $s, s' \subseteq V$, the term $\kappa(s, s')$ represents a matrix of size $|s| \times |s'|$, where the ij-th entry $[\kappa(s, s')]_{i,j} = \kappa(s_i, s'_j)$ (s_i is the i-th entry of s). For noisy samples $Y(s) = w(s) + \epsilon$ at vertices $s = \{v_1, v_2, \ldots, v_k\}$ with i.i.d. Gaussian noice $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$ (without causing notational confusion we also represent $Y(s)$ as a $|s|$ dimensional vector where $[Y(s)]_i = Y(s_i)$), the posterior distribution over the reward map w is a GP again, i.e. $w|\{s, Y(s)\} \sim \mathcal{GP}(\mu', \kappa')$, with mean μ' and covariance κ' given by:

$$\mu'(v) = \mu(v) + \kappa(v, s)(\kappa(s, s) + \sigma^2 I)^{-1} Y(s), \quad \kappa'(u, v) = \kappa(u, v) - \kappa(u, s)(\kappa(s, s) + \sigma^2 I)^{-1}\kappa(s, v) \tag{3}$$

A more comprehensive discussion of topics related to GP can be found in (Williams and Rasmussen, 2006).

3. Algorithm Design

We present a detailed description of the multi-agent coverage control with the doubling trick (MAC-DT, Algorithm 1), which can be decomposed into the following steps:

Upper Confidence Bound Construction. Similar to the standard UCB algorithm for Bayesian optimization with GP (c.f. (Srinivas et al., 2009)), we keep track of a posterior estimate of the GP and use it to construct a UCB for the reward map w at each episode. At the start of each episode e, the mean function $\mu^{(e-1)}$ and the covariance $\kappa^{(e-1)}$ is updated based on the posterior estimation, and the UCB for the reward map is calculated by $w_{\text{UCB}}^{(e)} := \mu^{(e-1)} + \beta^{(e)}\sigma^{(e-1)}$, where $\sigma^{(e-1)}$ is the standard deviation function, i.e. $\sigma^{(e-1)}(v) := \sqrt{\kappa^{(e-1)}(v, v)}$ and $\beta^{(e)}$ is some pre-specified constant.

1. Our assumption $w \geq 0$ does not conflict with the Gaussian Process (GP) assumption. This is because we have the flexibility to shift the function sampled from the GP, ensuring non-negativity while preserving the same regret.

Destination Selection with `Oracle` In each episode e, given the UCB reward map $w_{\text{UCB}}^{(e)}$, the algorithm computes the destination of each agent using an oracle algorithm `Oracle` which takes a given reward map as its input and outputs the destination of each agent $\{v_{1,dest}, v_{2,dest}, \dots, v_{N,dest}\} = \texttt{Oracle}(w_{\text{UCB}}^{(e)})$. The most ideal `Oracle` would solve the optimal max-coverage problem of the given reward map w, i.e. $\texttt{Oracle}(w) = \text{argmax}_{s=\cup_{i=1}^N s_i(v_i), v_i \in V} \|w(s)\|_1$. However, the maximum coverage problem can be NP-hard (Feige, 1998). Consequently, practical solutions frequently resort to approximation algorithms, such as greedy algorithms, to provide an approximate solution. For this consideration, we introduce the α-approximated oracle:

Definition 1 (α-approximated oracle) *An oracle algorithm `Oracle` is called an α-approximated oracle if for every reward map w, the output of the oracle $\{v_{1,dest}, v_{2,dest}, \dots, v_{N,dest}\} = \texttt{Oracle}(w)$ satisfies:*

$$\|w(s_{dest})\|_1 \geq \alpha \|w(s^\star)\|_1, \text{ where } s_{dest} = \cup_{i=1}^N s_i(v_{i,dest}), \quad s^\star = \text{argmax}_{s=\cup_{i=1}^N s_i(v_i), v_i \in V} \|w(s)\|_1$$

Example 1 (The greedy oracle) *It can be shown that the greedy algorithm oracle is a $(1-\frac{1}{e})$-approximated oracle (Nemhauser et al., 1978; Krause and Golovin, 2014; Prajapat et al., 2022). The `Oracle` is defined as*

$$v_{1,dest} = \text{argmax}_v \, w(s_1(v)), \quad v_{i,dest} = \text{argmax}_v \, w(s_i(v) \backslash \cup_{j=1}^{i-1} s_j(v_{j,dest})), i \geq 2,$$

where $A \backslash B$ denotes the relative complement of B in A.

Path Planning and Sample Selection Given the destination output by the `Oracle`, each agent plans its path on the graph to reach its destination. Along the path, at each time step t agents selects its evaluation point $v_{i,t}^{\text{eval}}$ such that it is the most uncertain point within its covering region $s_{i,t}$, i.e. $v_{i,t}^{\text{eval}} = \text{argmax}_{v \in s_{i,t}} \sigma^{(e-1)}(v)$.

The Doubling Trick As stated in Line 6 of the algorithm, the termination condition is given by the 'doubling trick', i.e., at least one of the vertex's samples doubles. If an agent reaches its destination before the episode terminates, it stays at the destination and keeps collecting samples until the end of the episode. If the termination condition is triggered before the agent reaches its target, it promptly terminates its current path and transitions to the next episode.It is worth noting that the doubling trick serves as a major difference between our algorithm and MACOPT in (Prajapat et al., 2022), which plays an important role in proving sublinear regret considering transient behavior. Similar techniques have also been used for regret analysis in different settings (Zhang et al., 2023; Auer et al., 2008). Numerical simulations also suggest that the doubling trick can improve and stabilize the coverage performance, especially in the case where the observation is noisy and high reward locations are relatively scattered.

Remark 1 *It's essential to note that MAC-DT maintains a partial decentralization approach. Specifically, it requires a centralized coordinator responsible for storing and broadcasting critical global information, such as the UCB of the reward map $w_{\text{UCB}}^{(e)}$ and $n_v(t)$'s, to each individual agent.*

4. Main Result

We first define the following variable which takes the same definition as in (Srinivas et al., 2009):

$$\gamma_N := \max_{s \subseteq V, |s|=N} I(Y(s); w), \tag{4}$$

where $I(X;Y)$ denotes the mutual information (c.f. (MacKay, 2003)) between random variables X and Y. This quantity is a frequently employed term in the context of proving regret bounds for Bayesian type algorithms based on GPs (e.g. (Srinivas et al., 2009; Prajapat et al., 2022)). Now we state the main theorem:

Theorem 1 *For Algorithm 1 with `Oracle` as an α-approximated oracle, by setting $\beta^{(e)} = \sqrt{2\log(|V|\pi^2 e^2/6\delta)}$, with probability at least $1-\delta$, the α-approximated regret of Algorithm 1 can be bounded by*

$$R^\alpha(\texttt{Alg}, T, w) \leq \underbrace{8\sigma\sqrt{\gamma_{2NT} K|V| T \log(2|V|\pi^2 T^2/3\delta)}}_{\text{Part I}}$$
$$\underbrace{+nDK|V| \max_v(\mu(v) + \beta^{(1)}\sqrt{\kappa(v,v)})(\log T + 2) + 2|V|\sqrt{2NK \log(2|V|\pi^2 T^2/3\delta) \max_v \kappa(v,v)}}_{\text{Part II}}$$

where the term γ_{2NT} is defined as in (4).

Algorithm 1 Multi-agent Coverage Control with the Doubling Trick (MAC-DT)

Require: An Oracle algorithm `Oracle` that calculates agents' destination given a reward map. We also denote $\mu^{(0)} = \mu, \kappa^{(0)} = \kappa, \sigma^{(0)}(v) = \sqrt{\kappa(v,v)}$. We use t_e to represent the total timestep when episode e starts.

1: **for** episode $e = 1, 2, ...$ **do**

2: Calculate the upper confidence bound $w_{\text{UCB}}^{(e)} = \mu^{(e-1)} + \beta^{(e)}\sigma^{(e-1)}$

3: Set the destination using the oracle $\{v_{1,dest}, v_{2,dest}, \ldots, v_{N,dest}\} = \texttt{Oracle}(w_{\text{UCB}}^{(e)})$.

4: Path Planning: Compute the shortest path on the graph G from current location v_{i,t_e} to $v_{i,dest}$.

5: Collect samples: Agents follow their planned paths and at time step t, each agent i sets the evaluation point as $v_{i,t}^{\text{eval}} = \text{argmax}_{v \in s_{i,t}} \sigma^{(e-1)}(v)$, and collect sample $Y(v_{i,t}^{\text{eval}})$. When an agent reaches its destination, it stays at the destination and keeps collecting samples until the episode terminates.

6: Episode termination criteria-the doubling trick: episode terminates at a minimum t such that there exists $v \in V$, such that $n_v(t) \geq \max\{2n_v(t_e - 1), 1\}$, i.e. when at least one of the vertex's samples doubles.

7: Update the mean value $\mu^{(e)}$ and $\kappa^{(e)}$ covariance of GP according to (3) such that $w|D \sim \mathcal{GP}(\mu^{(e)}, \kappa^{(e)})$, where $D = \{s_{t_e}^{\text{eval}}, Y(s_{t_e}^{\text{eval}}), s_{t_e-1}^{\text{eval}}, Y(s_{t_e-1}^{\text{eval}}), \ldots, s_1^{\text{eval}}, Y(s_1^{\text{eval}})\}$. Set $\sigma^{(e)}(v) = \sqrt{\kappa^{(e)}(v,v)}$

8: **end for**

Remark 2 *The regret consists of a term (Part II) that scales with $O(\log(T))$ and a term that scales with $\widetilde{O}(\gamma_{2NT}K|V|T)$ (Part I). We mainly focus our discussion on Part I. Note that the term γ_{2NT} captures the largest possible mutual information between the samples with size $2NT$ and the true reward map. This quantity generally grows sublinearly with T for commonly used kernels (Srinivas et al., 2009), e.g. for the squared-exponential kernel in the 2-dimension case, $\gamma_{2NT} \sim O((\log(2NT))^3)$, which leads to a final regret of order $\widetilde{O}(\sqrt{T})$. We would also like to compare our result with the regret bound in (Prajapat et al., 2022), where they bound the regret without accounting for the transition behavior as $\widetilde{O}(\gamma_{NT}NK|V|T)$. Note that our Part I managed to remove the dependency on $\sqrt{N}$. This major difference arises from a more careful analysis on the bound of the maximum eigenvalue of the covariance matrix (see Remark 3 in the Appendix). It is worth noting that the application of our derived bound to the analysis in (Prajapat et al., 2022) has the potential to enhance their regret bound by eliminating the dependency on $\sqrt{N}$ as well.*

4.1. Proof Sketches

This section provides a brief proof sketch for Theorem 1, which can be decomposed into three major steps. The first step, regret decomposition, breaks down the regret into two terms, namely the 'destination switch' and the 'price of optimism'. Then the second and third steps bound these two terms respectively.

Regret Decomposition We define the clean event to be $\mu^{(e-1)} - \beta^{(e)}\sigma^{(e-1)} \leq w \leq= \mu^{(e-1)} + \beta^{(e)}\sigma^{(e-1)} = w_{\text{UCB}}^{(e)}$ (for all $v \in V$), i.e., the true reward map lies within the confidence bound created by $\mu^{(e-1)} \pm \beta^{(e)}\sigma^{(e-1)}$. By carefully selecting the parameters $\beta^{(e)}$, it can be demonstrated that the algorithm will consistently fall within the clean event with a high probability. Thus, for the proof sketch, we will focus solely on the clean event. For simplicity, we also first consider T where T is the last time step of episode E, i.e., $T = t_{E+1} - 1$.

$$
\begin{aligned}
&R^{\alpha}(\texttt{Alg}, T, w) = \alpha T\|w(s^\star)\|_1 - \mathbb{E}\sum_{t=1}^{T}\|w(s_t))\|_1 = \mathbb{E}\sum_{e=1}^{E}\sum_{t=t_e}^{t=t_{e+1}-1}[\alpha\|w(s^\star)\|_1 - \|w(s_t))\|_1] \\
&= \mathbb{E}\sum_{e=1}^{E}\sum_{t=t_e}^{t=t_{e+1}-1}[\alpha\|w(s^\star)\|_1 - \|w_{\text{UCB}}^{(e)}(s_t)\|_1] + \mathbb{E}\sum_{e=1}^{E}\sum_{t=t_e}^{t=t_{e+1}-1}[\|w_{\text{UCB}}^{(e)}(s_t)\|_1 - \|w(s_t)\|_1] \\
&\leq \mathbb{E}\sum_{e=1}^{E}\sum_{t=t_e}^{t=t_{e+1}-1}[\alpha\|w(s^\star)\|_1 - \|w_{\text{UCB}}^{(e)}(s_t)\|_1] + \mathbb{E}\sum_{e=1}^{E}\sum_{t=t_e}^{t=t_{e+1}-1}[\|w_{\text{UCB}}^{(e)}(s_t)\|_1 - \|[\mu^{(e-1)} - \beta^{(e)}\sigma^{(e-1)}](s_t)\|_1] \\
&= \underbrace{\mathbb{E}\sum_{e=1}^{E}\sum_{t=t_e}^{t=t_{e+1}-1}[\alpha\|w(s^\star)\|_1 - \|w_{\text{UCB}}^{(e)}(s_t)\|_1]}_{\text{Destination Switch}} + \underbrace{\mathbb{E}\sum_{e=1}^{E} 2\beta^{(e)}\sum_{t=t_e}^{t=t_{e+1}-1}\|\sigma^{(e-1)}(s_t)\|_1}_{\text{Price of Optimism}}. \qquad (5)
\end{aligned}
$$

Here the term 'destination switch' measures the regret from visiting sub-optimal nodes during transit to the destination in each episode. The term 'price of optimism' captures the regret from using the UCB as a surrogate for the actual rewards. We now bound each term separately.

Bound the destination switch The bound for the destination switch is relatively easy and straight forward:

Lemma 2 $\underbrace{\mathbb{E}\sum_{e=1}^{E}\sum_{t=t_e}^{t=t_{e+1}-1}[\alpha\|w(s^\star)\|_1-\|w_{\text{UCB}}^{(e)}(s_t)\|_1]}_{\text{Destination Switch}}\leq\alpha nDK|V|\max_v(\mu(v)+\beta^{(1)}\sqrt{\kappa(v,v)})(\log(t_{E+1}-1)+1).$

Proof *When agents haven't reached the destinations,* $\alpha\|w(s^\star)\|_1-\|w_{\text{UCB}}^{(e)}(s_t)\|_1\leq\alpha\|w(s^\star)\|_1\leq NK\max_v w(v)$. *Since* $w(v)\leq\mu^{(0)}(v)+\beta^{(1)}\sigma^{(0)}(v)\leq\max_v(\mu(v)+\beta^{(1)}\sqrt{\kappa(v,v)})\Rightarrow\alpha\|w(s^\star)\|_1-\|w_{\text{UCB}}^{(e)}(s_t)\|_1\leq NK\max_v(\mu(v)+\beta^{(1)}\sqrt{\kappa(v,v)})$. *When the agents have reached the destination, we have that* $s_t=s_{dest}=\texttt{Oracle}(w_{\text{UCB}}^{(e)})$, *and thus* $\|w_{\text{UCB}}^{(e)}(s_t)\|_1\geq\alpha\max_s\|w_{\text{UCB}}^{(e)}(s)\|_1\geq\alpha\|w(s^\star)\|_1$. *Since the diameter of graph* G *is* D, *it takes the agents at least* D *steps to reach the destination, thus for each episode, we have that* $\sum_{t=t_e}^{t=t_{e+1}-1}[\alpha\|w(s^\star)\|_1-\|w_{\text{UCB}}^{(e)}(s_t)\|_1]\leq nDK\max_v(\mu(v)+\beta^{(1)}\sqrt{\kappa(v,v)})$. *Then from Lemma 12 in Appendix D in (Zhang et al., 2024), the number of episode* E *can be bounded by* $E\leq|V|\log(t_{E+1}-1)+1$, *the destination switch can then be bounded by* $\mathbb{E}\alpha nDKE\leq\alpha nDK|V|\max_v(\mu(v)+\beta^{(1)}\sqrt{\kappa(v,v)})(\log(t_{E+1}-1)+1)$. ■

Bound the price of optimism The bound for price of optimism is technically more involved, thus we defer the full proof to Appendix A. The key step is to bound $\|\sigma^{(e-1)}(s_t^{\text{eval}})\|_2^2$ using the mutual information of the function evaluations and the true reward map, i.e., $I(Y(s_{1:t_{E+1}-1}^{\text{eval}});w)$ as stated in the following lemma.

Lemma 3 *(Informal, formal statement see Lemma 8)* $\sum_{e=1}^{E}\sum_{t=t_e}^{t_{e+1}-1}\|\sigma^{(e-1)}(s_t^{\text{eval}})\|_2^2\leq\frac{2\sigma^2}{\log 2}I(Y(s_{1:t_{E+1}-1}^{\text{eval}});w)$.

Then using Cauchy schwartz inequality we can get the bound:

Lemma 4 $\underbrace{\mathbb{E}\sum_{e=1}^{E}2\beta^{(e)}\sum_{t=t_e}^{t=t_{e+1}-1}\|\sigma^{(e-1)}(s_t)\|_1}_{\text{Price of Optimism}}\leq 4\sigma\beta^{(E)}\sqrt{K|V|(t_{E+1}-1)\gamma_{N(t_{E+1}-1)}}+2\beta^{(E)}|V|\sqrt{NK\max_v\kappa(v,v)}.$

5. Experimental Results

This section evaluates our algorithm on multiple numerical simulation tasks and a physical multi-robot coverage task. We also present a variant of Algorithm 1 for the multi-agent coverage with safety considerations.

5.1. Numerical Simulations

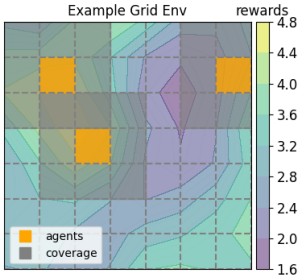

Figure 2: Grids environment setup.

Environment setup. We discretize the environments into grids where each agent can cover its 1-hop neighborhood as shown in Figure 2. We evaluate our algorithm, `MAC-DT`, with different reward maps, kernel hyperparameters, and observation noises at different scales. Figure 3 shows the results with noise variance 0.1, the first three figures show results with three agents in 8×8 grids with different reward maps, and the last two show 6 and 10 agents in 10×10 grids. The reward maps (listed in titles in Figure 3) are denoted as w_{Normal}, w_{Uniform}, and w_{Sparse}. The first two mean rewards are sampled from Gaussian or uniform distributions, and the last one means only a small number of grids have reward 1 and others all have reward 0. We use Gaussian kernels for the GPs. The complete results with 6 and 10 agents, more noise levels, and different kernel hyperparameters can be found in E.2 in Zhang et al. (2024).

We compare our algorithm with two baselines. (i) Shortest path planning with MacOpt (Prajapat et al., 2022) named as `MacOpt-SP`. The only difference between `MacOpt-SP` and proposed algorithm is the episode termination criteria in line 6 in Algorithm 1. Instead of the doubling trick, `MacOpt-SP` terminates episodes when all the agents reach their destinations. (ii) A modification of Voronoi partition coverage control from Wei et al. (2021) named `Voronoi`. Modifications can be found in Appendix E.2 in (Zhang et al., 2024). Results in Figure 3 show that the regret curves of the `MAC-DT` stay level after a few iterations under all rewards and kernel settings, which means the proposed algorithm quickly finds the no-regret maximal coverage. `MacOpt-SP` is not efficient since its regret increases faster than MAC-DT, which shows the doubling trick greatly improves the performance. The regret of the Voronoi partition quickly increases in the initial stages since it has an initial sampling stage to reduce the uncertainty globally, which is inefficient when considering the transient behaviors.

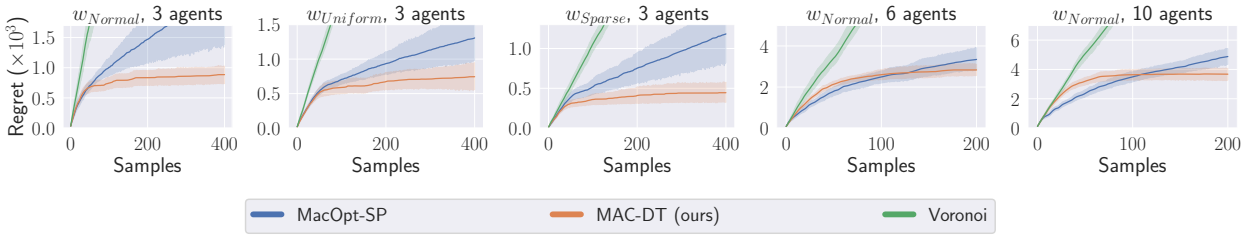

Figure 3: Regret $R(\texttt{Alg}, T, w)$ with respect to T with different algorithm $\texttt{Alg}$ rewards w. Lines and shaded regions are mean and confidential intervals over 10 randomly generated reward maps.

5.2. Safety Considerations

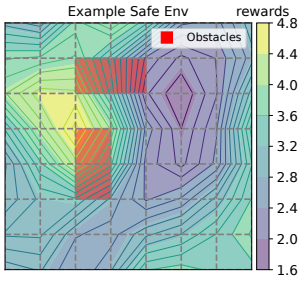

Figure 4: Example safe exploration environment.

Safety considerations are common in real-world multi-robot coverage tasks. For example, there are usually obstacles in the environment (like grids painted red in Figure 4) that agents should avoid. We show that our methods could be combined with safety considerations in this section. We define the set of safe nodes $V_{\text{Safe}} = \{v \in V | g(s) \geq 0\}$ by a safety function $g : V \to \mathbb{R}$. The agents should travel within the subgraph containing only nodes in V_{Safe}. We assume there is also uncertainty in the safety function g and agents need to learn from the noisy samples. Similar to the rewards $w(\cdot)$, we maintain a GP for safety function g whose posterior mean and variance at episode e are denoted as $\mu_g^{(e)}(\cdot)$ and $\sigma_g^{(e)}(\cdot)$.

We made two modifications to Algorithm 1 to consider safety. (i) Changing the oracle and path planning to ensure safety. We use a safe oracle to set the destinations in Line 3 following the SafeMac algorithm in Prajapat et al. (2022) that ensures the destinations are safe. The oracle also maintains an estimation of the safe set at episode e denoted by $\hat{V}_{\text{Safe}}^{(e)}$ and learns to gradually expand it. Then in Line 4, the shortest path planning is restricted within the estimated safe set $\hat{V}_{\text{Safe}}^{(e)}$. (ii) Assigning edge weights during shortest path planning in Line 4 to encourage safe set expansion. The edges are weighted by the mean value of posterior safety value $\mu_g^{(e)}(\cdot)$. In this way, the agents are prone to travel and collect samples on less safe nodes. These less safe nodes usually lie on the boundaries of the estimated safe set so the agents have more useful samples to expand the safe sets. We name the safety algorithm as `MAC-DT-SafelyExplore`. The algorithm block and details about the safe oracle can be found in Appendix E.3.1 in Zhang et al. (2024).

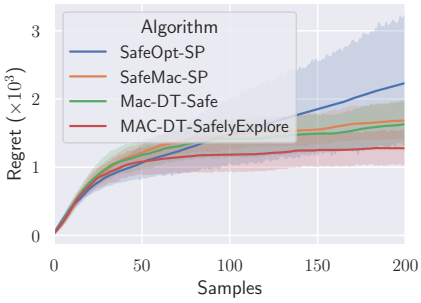

Figure 5: Regret of safe planning algorithm.

We compare the `MAC-DT-SafelyExplore` with three baselines: (a) the ablation study of weighted shortest path planning, which means only adding the safe oracle and restricted path planning to Algorithm 1. The algorithm is named as `Mac-DT-Safe`; (b) Adding shortest path planning for transient behavior considerations to SafeOpt in Sui et al. (2015) and SafeMaC in Prajapat et al. (2022), named `SafeOpt-SP` and `SafeMac-SP`. Both two algorithms did not consider transient behaviors originally.

The regret is shown in Figure 5. Results show that `MAC-DT-SafelyExplore` outperforms all baselines and achieves no-regret coverage quickly after about 50 samples. All the baselines cannot achieve no-regret coverage within 200 samples. The comparison with `Mac-DT-Safe` shows that the proposed weighted path planning is effective in encouraging safe set expansion. The results also suggest that it is important to consider the transient phase if the problem has safety considerations.

5.3. Real World Experiments

We set up a physical multi-drone coverage testbed as shown in Figure 6. We fetched the real-time rain data from the OpenWeatherMap and used drones to cover the area with the heaviest rains. We discretize the region into a 13×10 grid. Then we project the real-world weather data on the ground in our lab, where each 17cm square grid represents 1 km^2 in the real world. We use three Crazyflie 2.1 and each can cover the 1-hop neighborhood in the grids. The planning (including collision avoidance) and control are computed onboard the Crazyflie while receiving virtual source signals. The terminal condition is achieving optimal coverage. The experiment video is shown in https://youtu.be/ImgSS5QyhT0. The three drones find the optimal coverage with only 19 steps and 57 environment samples in total.

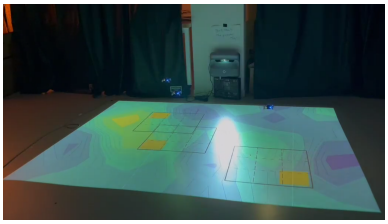

Figure 6: Real-world experiment setup. The projected contour plot indicates the node rewards, and the sensory range and sensory points.

6. Conclusion and Future Works

In this paper, we propose the MAC-DT algorithm to solve the MAC problem, which achieves (approximated) regret of $\widetilde{O}(\sqrt{T})$ even when accounting for the transient behavior. Our results are also supported by numerical studies in multiple settings, showcasing the algorithm's effectiveness and its adeptness in accommodating safety constraints. Our result admittedly has its limitations. First of all, MAC-DT is not fully decentralized. Secondly, although the simulation results demonstrate that our methods could naturally be combined with safety considerations, our current theoretical analysis doesn't consider safety issues. To address these shortcomings, we envision our future work focusing on the design of decentralized algorithms that not only maintain efficiency but also incorporate robust safety guarantees.

Appendix

Notations: We first define some notations that will be useful for the proof. Recall the definition of $n_v(t)$ which denotes the number of times that vertex v has been sampled until time t. We define $n_v^{(e)} := n_v(t_{e+1}-1)$, i.e., $n_v^{(e)}$ is the number of times that grid u is being sampled/covered from $t=1$ to the end of episode e. Further, we denote $c_v^{(e)} := n_v^{(e)} - n_v^{(e-1)}$ as the number of times that grid v is being sampled/covered within episode e.

We define the following matrix $K_s^{(e)} := \kappa^{(e)}(s,s)$, where $s \subseteq V$ is a subset of V. Note that for s we allow repetition, i.e. $v \in V$ can appear multiple times in s. Additionally, we use $s_{t:\tau} \subseteq V$ to denote the covering profile from time t to τ (allows repetition).

Appendix A. Proof of Lemma 3 and Lemma 4

We first state some important lemmas that will be used in the proofs.

Lemma 5 *(Quantify the information gain)* $I(Y(s^{\text{eval}}_{1:t_{e+1}-1}); w) = \sum_{e'=1}^{e} \frac{1}{2}\log\left(\det\left(I + \sigma^{-2}K^{(e'-1)}_{s^{\text{eval}}_{t_{e'}:t_{e'+1}-1}}\right)\right)$

Proof *We use $H(X)$ to denote the entropy of a random variable X. From the definition of mutual information and properties of entropy function (c.f. (MacKay, 2003)), we have*

$$\begin{aligned}
&I(Y(s^{\text{eval}}_{1:t_{e+1}-1}); w) = H(Y(s^{\text{eval}}_{1:t_{e+1}-1})) - H(Y(s^{\text{eval}}_{1:t_{e+1}-1})|w) \\
&= H(Y(s^{\text{eval}}_{1:t_e-1})) + H(Y(s^{\text{eval}}_{t_e:t_{e+1}-1})|Y(s^{\text{eval}}_{1:t_e-1})) - \left(H(Y(s^{\text{eval}}_{1:t_e-1})|w) + H(Y(s^{\text{eval}}_{t_e:t_{e+1}-1})|w)\right) \\
&= I(Y(s^{\text{eval}}_{1:t_e-1})|w) + H(Y(s^{\text{eval}}_{t_e:t_{e+1}-1})|Y(s^{\text{eval}}_{1:t_e-1})) - H(Y(s^{\text{eval}}_{t_e:t_{e+1}-1})|w)
\end{aligned}$$

Further,

$$\begin{aligned}
H(Y(s^{\text{eval}}_{t_e:t_{e+1}-1})|Y(s^{\text{eval}}_{1:t_e-1})) &= \frac{1}{2}\log\left(\det\left(2\pi e\left(\sigma^2 I + K^{(e-1)}_{s^{\text{eval}}_{t_e:t_{e+1}-1}}\right)\right)\right) \\
&= \frac{1}{2}\log\left(2\pi e(\sigma)^{2\sum_{t=t_e}^{t_{e+1}-1}|s^{\text{eval}}_t|}\right) + \frac{1}{2}\log\left(\det\left(I + \sigma^{-2}K^{(e-1)}_{s^{\text{eval}}_{t_e:t_{e+1}-1}}\right)\right)
\end{aligned}$$

Additionally, given that $H(Y(s^{\text{eval}}_{t_e:t_{e+1}-1})|w) = \frac{1}{2}\log\left(2\pi e(\sigma)^2 \sum_{t=t_e}^{t_{e+1}-1} |s_t^{\text{eval}}|\right)$, *we have*

$$I(Y(s^{\text{eval}}_{1:t_{e+1}-1}); w) = I(Y(s^{\text{eval}}_{1:t_e-1})|w) + \frac{1}{2}\log\left(\det\left(I + \sigma^{-2}K^{(e-1)}_{s^{\text{eval}}_{t_e:t_{e+1}-1}}\right)\right).$$

Applying this equality iteratively completes the proof. ■

Lemma 6 *If for all* $v \in s_{t_{e+1}-1}$, $n_v^{(e-1)} \geq 1$, *then* $\lambda_{\max}\left(K^{(e-1)}_{s_{t_e:t_{e+1}-1}}\right) \leq \sigma^2$.

otherwise, there exists at least one $v \in s_{t_{e+1}-1}$, *such that* $n_v^{(e-1)} = 0$, *then* $\lambda_{\max}\left(K^{(e-1)}_{s_{t_e:t_{e+1}-2}}\right) \leq \sigma^2$

Remark 3 *Lemma 6 serves as one of the technical novelty of our work. Notably, compared with (Prajapat et al., 2022) where they directly bound the maximum value of the covariance matrix by the trace of the matrix, i.e.,* $\lambda_{\max}\left(K^{(e-1)}_{s_{t_e:t_{e+1}-1}}\right) \leq \text{trace}\left(K^{(e-1)}_{s_{t_e:t_{e+1}-1}}\right) \sim O(N)$, *we bound the maximum eigenvalue in a more careful manner and thus eliminates the dependency on the number of agent* N, *thereby removing the* $\sqrt{N}$ *dependency in our regret bound. The proof is technically involved and is deferred to Appendix C in (Zhang et al., 2024).*

Lemma 7 *If* $\forall v \in s_{t_{e+1}-1}$, $n_v^{(e-1)} \geq 1$, *then* $\frac{\sigma^2}{\log 2}\log\left(\det\left(I+\sigma^{-2}K^{(e-1)}_{s^{\text{eval}}_{t_e:t_{e+1}-1}}\right)\right) \geq \sum_{t=t_e}^{t_{e+1}-1} \|\sigma^{(e-1)}(s_t^{\text{eval}})\|_2^2$,
otherwise, $\frac{\sigma^2}{\log 2}\log\left(\det\left(I+\sigma^{-2}K^{(e-1)}_{s^{\text{eval}}_{t_e:t_{e+1}-1}}\right)\right) \geq \sum_{t=t_e}^{t_{e+1}-2} \|\sigma^{(e-1)}(s_t^{\text{eval}})\|_2^2$.

Proof *For the first case, from Lemma 6 we have that* $\lambda \leq \sigma^2$ *for all* $\lambda \in \lambda\left(K^{(e-1)}_{s^{\text{eval}}_{t_e:t_{e+1}-1}}\right)$, *then*

$$\log\left(\det\left(I+\sigma^{-2}K^{(e-1)}_{s^{\text{eval}}_{t_e:t_{e+1}-1}}\right)\right) = \sum_{\lambda\in\lambda\left(K^{(e-1)}_{s^{\text{eval}}_{t_e:t_{e+1}-1}}\right)} \log\left(\det\left(I+\sigma^{-2}\lambda\right)\right)$$

$$\geq \sigma^{-2}\log 2 \sum_{e=1}^{E}\sum_{\lambda\in\lambda\left(K^{(e-1)}_{s^{\text{eval}}_{t_e:t_{e+1}-1}}\right)} \lambda \;\; (\lambda \leq \sigma^2 \text{ by Lemma 6 and that } \log(1+x) \geq (\log 2)x, 0 \leq x \leq 1)$$

$$= \sigma^{-2}\log 2 \sum_{e=1}^{E} \text{trace}\left(K^{(e-1)}_{s^{\text{eval}}_{t_e:t_{e+1}-1}}\right) = \sigma^{-2}\log 2 \sum_{e=1}^{E}\sum_{t=t_e}^{t_{e+1}-1} \|\sigma^{(e-1)}(s_t^{\text{eval}})\|_2^2.$$

For the second case, i.e., there exists at least one $v \in s_{t_{e+1}-1}$, *such that* $n_v^{(e-1)} = 0$, *we first argue that for all* $v \in s_{t_e:t_{e+1}-2}$, $n_v^{(e-1)} \geq 1$. *This can be easily verified by contradiction. If there exists at least one* $v \in s_{t_e:t_{e+1}-2}$ *such that* $n_v = 0$, *then given the doubling trick in the algorithm, the episode should terminate at the time step when* v *is sampled, which contradicts the fact that it terminates at* $t_{e+1}-1$. *Then from Lemma 6 we have that* $\lambda \leq \sigma^2$ *for all* $\lambda \in \lambda\left(K^{(e-1)}_{s^{\text{eval}}_{t_e:t_{e+1}-2}}\right)$, *and similarly,* $\log\left(\det\left(I+\sigma^{-2}K^{(e-1)}_{s^{\text{eval}}_{t_e:t_{e+1}-2}}\right)\right) \geq \sigma^{-2}\log 2\sum_{e=1}^{E}\sum_{t=t_e}^{t_{e+1}-2}\|\sigma^{(e-1)}(s_t^{\text{eval}})\|_2^2$.
Since $K^{(e-1)}_{s^{\text{eval}}_{t_e:t_{e+1}-2}}$ *is a principal submatrix of* $K^{(e-1)}_{s^{\text{eval}}_{t_e:t_{e+1}-1}}$, *from Lemma 14 in (Zhang et al., 2024), we have that*

$$\log\left(\det\left(I+\sigma^{-2}K^{(e-1)}_{s^{\text{eval}}_{t_e:t_{e+1}-1}}\right)\right) \geq \log\left(\det\left(I+\sigma^{-2}K^{(e-1)}_{s^{\text{eval}}_{t_e:t_{e+1}-2}}\right)\right) \geq \sigma^{-2}\log 2\sum_{t=t_e}^{t_{e+1}-2}\|\sigma^{(e-1)}(s_t^{\text{eval}})\|_2^2.$$ ■

We are now ready to state and prove the formal version of Lemma 3.

Lemma 8 (Formal version of Lemma 3)

$$\sum_{e=1}^{E}\left(\sum_{t=t_e}^{t=t_{e+1}-2}\|\sigma^{(e-1)}(s_t^{\text{eval}})\|_2^2 + \mathbf{1}\{\forall v \in s_{t_{e+1}-1}, n_v^{(e-1)} \geq 1\}\|\sigma^{(e-1)}(s_t^{\text{eval}})\|_2^2\right) \leq \frac{2\sigma^2}{\log 2} I(Y(s^{\text{eval}}_{1:t_{E+1}-1}); w)$$

Proof [Proof of Lemma 8] From Lemma 5 and 7 we have

$$I(Y(s^{\text{eval}}_{1:t_{E+1}-1}); w) = \sum_{e=1}^{E} \frac{1}{2} \log\left(\det\left(I + \sigma^{-2} K^{(e-1)}_{s^{\text{eval}}_{t_e:t_{e+1}-1}}\right)\right)$$
$$\geq \frac{\sigma^{-2}\log 2}{2} \sum_{e=1}^{E}\left(\sum_{t=t_e}^{t=t_{e+1}-2} \|\sigma^{(e-1)}(s^{\text{eval}}_t)\|_2^2 + \mathbf{1}\{\forall v \in s_{t_{e+1}-1}, n_v^{(e-1)} \geq 1\}\|\sigma^{(e-1)}(s^{\text{eval}}_t)\|_2^2\right).$$

■

Proof [Proof of Lemma 4] From Cauchy Schwarz inequality and that $\beta^{(e+1)} \geq \beta^{(e)}$,

$$\left(\mathbb{E}\sum_{e=1}^{E} 2\beta^{(e)} \sum_{t=t_e}^{t=t_{e+1}-1} \|\sigma^{(e-1)}(s_t)\|_1\right)^2 \leq \mathbb{E}4(\beta^{(E)})^2 \left(\sum_{e=1}^{E}\sum_{t=t_e}^{t=t_{e+1}-1}|s_t|\right)\left(\sum_{e=1}^{E}\sum_{t=t_e}^{t=t_{e+1}-1}\|\sigma^{(e-1)}(s_t)\|_2^2\right)$$
$$\leq \mathbb{E}4(\beta^{(E)})^2|V|(t_{E+1}-1)\left(K\underbrace{\sum_{e=1}^{E}\sum_{t=t_e}^{t=t_{e+1}-1}\|\sigma^{(e-1)}(s^{\text{eval}}_t)\|_2^2}_{\text{Part A}}\right)$$

Since
$$\text{Part A} = \sum_{e=1}^{E}\left(\sum_{t=t_e}^{t=t_{e+1}-2}\|\sigma^{(e-1)}(s^{\text{eval}}_t)\|_2^2 + \mathbf{1}\{\forall v \in s_{t_{e+1}-1}, n_v^{(e-1)} \geq 1\}\|\sigma^{(e-1)}(s^{\text{eval}}_t)\|_2^2\right)$$
$$+ \sum_{e=1}^{E}\left(\mathbf{1}\{\exists v \in s^{\text{eval}}_{t_{e+1}-1}, n_v^{(e-1)} = 0\}\|\sigma^{(e-1)}(s^{\text{eval}}_t)\|_2^2\right) \leq \frac{2\sigma^2}{\log 2} I(Y(s^{\text{eval}}_{1:t_{E+1}-1}); w) + N|V|\max_{v\in V}\kappa(v,v)$$

where the last inequality comes from Lemma 8 and the fact that $\sigma^{(0)}(v) = \max_{v\in V}\sqrt{\kappa(v,v)}$. Thus

$$\mathbb{E}\sum_{e=1}^{E} 2\beta^{(e)}\sum_{t=t_e}^{t=t_{e+1}-1}\|\sigma^{(e-1)}(s_t)\|_1 \leq 2\beta^{(E)}\sqrt{K|V|(t_{E+1}-1)}\sqrt{\frac{2\sigma^2}{\log 2} I(Y(s^{\text{eval}}_{1:t_{E+1}-1}); w) + N|V|\max_v \kappa(v,v)}$$
$$\leq 4\sigma\beta^{(E)}\sqrt{K|V|(t_{E+1}-1)\gamma_{N(t_{E+1}-1)}} + 2\beta^{(E)}|V|\sqrt{nK\max_v\kappa(v,v)}$$

■

Appendix B. Proof of Theorem 1

Before proving the theorem, we first state the following lemma that suggests that with probability at least $1-\delta$, the algorithm will operate on the 'clean events':

Lemma 9 (Lemma 5.1 in (Srinivas et al., 2009)) *By setting* $\beta^{(e)} = \sqrt{2\log(|V|\pi^2 e^2/6\delta)}$, *then* $\left|w(v) - \mu^{(e-1)}(v)\right| \leq \beta^{(e)}\sigma^{(e-1)}(v), \ \ \forall v \in V, \forall e = 1, 2, \ldots,$ *holds with probability at least* $1-\delta$.

Proof [Proof of Theorem 1] We set E to be such that $t_E < T \leq t_{E+1} - 1$. From Lemma 9 and (5), we have that by setting $\beta^{(e)} = \sqrt{2\log(|V|\pi^2 e^2/6\delta)}$, then with probability $1-\delta$

$$R^{\alpha}(\texttt{Alg}, T, w) \leq \mathbb{E}_{t_E < T \leq t_{E+1}-1} R^{\alpha}(\texttt{Alg}, t_{E+1}-1, w)$$
$$\leq \underbrace{\mathbb{E}\sum_{e=1}^{E}\sum_{t=t_e}^{t=t_{e+1}-1}[\alpha\|w(s^\star)\|_1 - \|w^{(e)}_{\text{UCB}}(s_t)\|_1]}_{\text{Destination Switch}} + \underbrace{\mathbb{E}\sum_{e=1}^{E}\sum_{t=t_e}^{t=t_{e+1}-1} 2\beta^{(e)}\|\sigma^{(e-1)}(s_t)\|_1]}_{\text{Price of Optimism}}$$
$$\overset{\text{(Lemma 2 and 4)}}{\leq} \mathbb{E}nDK|V|\max_v(\mu(v) + \beta^{(1)}\sqrt{\kappa(v,v)})(\log(t_{E+1}-1)+1)$$
$$+ 4\sigma\beta^{(E)}\sqrt{K|V|(t_{E+1}-1)\gamma_{N(t_{E+1}-1)}} + 2\beta^{(E)}|V|\sqrt{nK\max_v\kappa(v,v)}$$

From the doubling trick, we know that $t_{E+1} - 1 \leq 2T$. Thus

$$R^{\alpha}(\texttt{Alg}, T, w) \leq \mathbb{E}nDK|V|\max_v(\mu(v) + \beta^{(1)}\sqrt{\kappa(v,v)})(\log(2T)+1)$$
$$+ 4\sigma\beta^{(E)}\sqrt{2K|V|T\gamma_{2nT}} + 2\beta^{(E)}|V|\sqrt{nK\max_v\kappa(v,v)}$$
$$= 8\sigma\sqrt{\gamma_{2nT}K|V|T\log(2|V|\pi^2T^2/3\delta)}$$
$$+ nDK|V|\max_v(\mu(v) + \beta^{(1)}\sqrt{\kappa(v,v)})(\log T + 2) + 2|V|\sqrt{2nK\log(2|V|\pi^2T^2/3\delta)\max_v\kappa(v,v)},$$

which completes the proof. ■

References

Peter Auer, Thomas Jaksch, and Ronald Ortner. Near-optimal regret bounds for reinforcement learning. *Advances in neural information processing systems*, 21, 2008.

Gianpietro Battocletti, Riccardo Urban, Simone Godio, and Giorgio Guglieri. Rl-based path planning for autonomous aerial vehicles in unknown environments. In *AIAA AVIATION 2021 FORUM*, page 3016, 2021.

Alessia Benevento, María Santos, Giuseppe Notarstefano, Kamran Paynabar, Matthieu Bloch, and Magnus Egerstedt. Multi-robot coordination for estimation and coverage of unknown spatial fields. In *2020 ieee international conference on robotics and automation (icra)*, pages 7740–7746. IEEE, 2020.

Francesco Bullo, Ruggero Carli, and Paolo Frasca. Gossip coverage control for robotic networks: Dynamical systems on the space of partitions. *SIAM Journal on Control and Optimization*, 50(1):419–447, 2012.

Andrea Carron, Marco Todescato, Ruggero Carli, Luca Schenato, and Gianluigi Pillonetto. Multi-agents adaptive estimation and coverage control using gaussian regression. In *2015 European Control Conference (ECC)*, pages 2490–2495. IEEE, 2015.

Jongeun Choi and Roberto Horowitz. Learning coverage control of mobile sensing agents in one-dimensional stochastic environments. *IEEE Transactions on Automatic Control*, 55(3):804–809, 2010.

Jongeun Choi, Joonho Lee, and Songhwai Oh. Swarm intelligence for achieving the global maximum using spatio-temporal gaussian processes. In *2008 American Control Conference*, pages 135–140. IEEE, 2008.

Jorge Cortés and Francesco Bullo. Coordination and geometric optimization via distributed dynamical systems. *SIAM journal on control and optimization*, 44(5):1543–1574, 2005.

Jorge Cortes, Sonia Martinez, Timur Karatas, and Francesco Bullo. Coverage control for mobile sensing networks. *IEEE Transactions on robotics and Automation*, 20(2):243–255, 2004.

Jorge Cortes, Sonia Martinez, and Francesco Bullo. Spatially-distributed coverage optimization and control with limited-range interactions. *ESAIM: Control, Optimisation and Calculus of Variations*, 11(4):691–719, 2005.

Peter Davison, Naomi Ehrich Leonard, Alex Olshevsky, and Michael Schwemmer. Nonuniform line coverage from noisy scalar measurements. *IEEE Transactions on Automatic Control*, 60(7):1975–1980, 2014.

Ahmad Din, Muhammed Yousoof Ismail, Babar Shah, Mohammad Babar, Farman Ali, and Siddique Ullah Baig. A deep reinforcement learning-based multi-agent area coverage control for smart agriculture. *Computers and Electrical Engineering*, 101:108089, 2022. ISSN 0045-7906. doi: https://doi.org/10.1016/j.compeleceng.2022.108089. URL https://www.sciencedirect.com/science/article/pii/S0045790622003445.

Joseph W Durham, Ruggero Carli, Paolo Frasca, and Francesco Bullo. Discrete partitioning and coverage control for gossiping robots. *IEEE Transactions on Robotics*, 28(2):364–378, 2011.

Saba Faryadi and Javad Mohammadpour Velni. A reinforcement learning-based approach for modeling and coverage of an unknown field using a team of autonomous ground vehicles. *International journal of intelligent systems*, 36(2):1069–1084, 2021.

Uriel Feige. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.

Islam I Hussein and Dusan M Stipanovic. Effective coverage control for mobile sensor networks with guaranteed collision avoidance. *IEEE Transactions on Control Systems Technology*, 15(4):642–657, 2007.

Nare Karapetyan, Jason Moulton, Jeremy S Lewis, Alberto Quattrini Li, Jason M O'Kane, and Ioannis Rekleitis. Multi-robot dubins coverage with autonomous surface vehicles. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2373–2379. IEEE, 2018.

Stephanie Kemna, John G Rogers, Carlos Nieto-Granda, Stuart Young, and Gaurav S Sukhatme. Multi-robot coordination through dynamic voronoi partitioning for informative adaptive sampling in communication-constrained environments. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2124–2130. IEEE, 2017.

A. Krause, C. Guestrin, A. Gupta, and J. Kleinberg. Near-optimal sensor placements: maximizing information while minimizing communication cost. In *2006 5th International Conference on Information Processing in Sensor Networks*, pages 2–10, 2006. doi: 10.1145/1127777.1127782.

Andreas Krause and Daniel Golovin. Submodular function maximization. *Tractability*, 3(71-104):3, 2014.

Andreas Krause and Carlos Guestrin. Submodularity and its applications in optimized information gathering. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(4):1–20, 2011.

Francois Lekien and Naomi Ehrich Leonard. Nonuniform coverage and cartograms. *SIAM Journal on Control and Optimization*, 48(1):351–372, 2009.

Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

Wenhao Luo and Katia Sycara. Adaptive sampling and online learning in multi-robot sensor coverage with mixture of gaussian processes. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 6359–6364. IEEE, 2018.

Wenhao Luo, Changjoo Nam, George Kantor, and Katia Sycara. Distributed environmental modeling and adaptive sampling for multi-robot sensor coverage. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 1488–1496, 2019.

David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

Alan Mainwaring, David Culler, Joseph Polastre, Robert Szewczyk, and John Anderson. Wireless sensor networks for habitat monitoring. In *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pages 88–97, 2002.

Kensuke Nakamura, María Santos, and Naomi Ehrich Leonard. Decentralized learning with limited communications for multi-robot coverage of unknown spatial fields. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9980–9986. IEEE, 2022.

George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14:265–294, 1978.

Manish Prajapat, Matteo Turchetta, Melanie Zeilinger, and Andreas Krause. Near-optimal multi-agent learning for safe coverage control. *Advances in Neural Information Processing Systems*, 35:14998–15012, 2022.

Vinod Ramaswamy and Jason R Marden. A sensor coverage game with improved efficiency guarantees. In *2016 American Control Conference (ACC)*, pages 6399–6404. IEEE, 2016.

Maria Santos, Udari Madhushani, Alessia Benevento, and Naomi Ehrich Leonard. Multi-robot learning and coverage of unknown spatial fields. In *2021 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, pages 137–145. IEEE, 2021.

Mac Schwager, Daniela Rus, and Jean-Jacques Slotine. Decentralized, adaptive coverage control for networked robots. *The International Journal of Robotics Research*, 28(3):357–375, 2009.

Mac Schwager, Michael P Vitus, Samantha Powers, Daniela Rus, and Claire J Tomlin. Robust adaptive coverage control for robotic sensor networks. *IEEE Transactions on Control of Network Systems*, 4(3):462–476, 2015.

Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.

Yanan Sui, Alkis Gotovos, Joel Burdick, and Andreas Krause. Safe exploration for optimization with gaussian processes. In *International conference on machine learning*, pages 997–1005. PMLR, 2015.

Xinmiao Sun, Christos G Cassandras, and Xiangyu Meng. A submodularity-based approach for multi-agent optimal coverage problems. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 4082–4087. IEEE, 2017.

Marco Todescato, Andrea Carron, Ruggero Carli, Gianluigi Pillonetto, and Luca Schenato. Multi-robots gaussian estimation and coverage control: From client–server to peer-to-peer architectures. *Automatica*, 80: 284–294, 2017.

Shiraz Wasim, Zendai Kashino, Goldie Nejat, and Beno Benhabib. Directional-sensor network deployment planning for mobile-target search. *Robotics*, 9(4), 2020. ISSN 2218-6581. doi: 10.3390/robotics9040082. URL https://www.mdpi.com/2218-6581/9/4/82.

Lai Wei, Andrew McDonald, and Vaibhav Srivastava. Multi-robot gaussian process estimation and coverage: Deterministic sequencing algorithm and regret analysis. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9080–9085. IEEE, 2021.

Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

Runyu Zhang, Haitong Ma, and Na Li. Multi-agent coverage control with transient behavior consideration. *arXiv preprint arXiv:2404.05995*, 2024.

Tianpeng Zhang, Kasper Johansson, and Na Li. Multi-armed bandit learning on a graph. In *2023 57th Annual Conference on Information Sciences and Systems (CISS)*, pages 1–6. IEEE, 2023.