Communication- and Computation-Efficient Distributed Submodular Optimization in Robot Mesh Networks

Zirui Xu, *Graduate Student Member, IEEE*, Sandilya Sai Garimella, *Graduate Student Member, IEEE*, and Vasileios Tzoumas, *Senior Member, IEEE*

***Abstract*—In this article, we provide a communication- and computation-efficient method for distributed submodular optimization in robot mesh networks. Submodularity is a property of diminishing returns that arises in active information gathering such as mapping, surveillance, and target tracking. Our method, resource-aware distributed greedy (RAG), introduces a new distributed optimization paradigm that enables scalable and near-optimal action coordination. To this end, RAG requires each robot to make decisions based only on information received from and about their neighbors. In contrast, the current paradigms allow the relay of information about all robots across the network. As a result, RAG's decision-time scales linearly with the network size, while state-of-the-art near-optimal submodular optimization algorithms scale cubically. We also characterize how the designed mesh-network topology affects RAG's approximation performance. Our analysis implies that sparser networks favor scalability without proportionally compromising approximation performance: while RAG's decision-time scales linearly with network size, the gain in approximation performance scales sublinearly. We demonstrate RAG's performance in simulated scenarios of area detection with up to 45 robots, simulating realistic robot-to-robot (r2r) communication speeds such as the 0.25 Mb/s speed of the Digi XBee 3 Zigbee 3.0. In the simulations, RAG enables real-time planning, up to three orders of magnitude faster than competitive near-optimal algorithms, while also achieving superior mean coverage performance. To enable the simulations, we extend the high-fidelity and photo-realistic simulator AirSim by integrating a scalable collaborative autonomy pipeline to tens of robots and simulating r2r communication delays.**

***Index Terms*—Active information gathering, approximation algorithms, multirobot mesh networks, robot-to-robot (r2r) communication, submodular optimization.**

Received 29 January 2025; accepted 14 April 2025. Date of publication 6 May 2025; date of current version 30 May 2025. This work was supported in part by National Science Foundation CAREER 2337412. This article was recommended for publication by Associate Editor H. G. de Marina and Editor M. Schwager upon evaluation of the reviewers' comments. *(Zirui Xu and Sandilya Sai Garimella contributed equally to this work.)*

Zirui Xu and Vasileios Tzoumas are with the Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI 48109 USA (e-mail: ziruixu@umich.edu; vtzoumas@umich.edu).

Sandilya Sai Garimella was with the Department of Robotics, University of Michigan, Ann Arbor, MI 48109 USA. He is now with the Institute for Robotics and Intelligent Machines, Georgia Institute of Technology, Atlanta, GA 30332 USA (e-mail: sgarimella34@gatech.edu).

Code is available online at: https://github.com/UM-iRaL/Resource-Aware-Coordination-AirSim

Digital Object Identifier 10.1109/TRO.2025.3567540

I. Introduction

In THE future, numerous distributed robots will be coordinating actions via *robot-to-robot* (r2r) communication to execute information-gathering tasks, such as collaborative mapping [1], surveillance [2], and target tracking [3]. Such tasks require efficiency (scalability) and effectiveness (optimality), especially in crowded scenarios where the robots are many and operate close to each other; see, for example, the semantic-driven task of road detection with 45 aerial robots in Fig. 1.

But achieving scalability and optimality is hard. The first reason is that such tasks take the form of distributed submodular optimization, an NP-hard combinatorial optimization problem [4]. Therefore, such tasks require increased computations and communications to be solved optimally. Submodularity is a property of diminishing returns, and is encountered across robotics [5] as well as machine learning [6] and control [7]. It captures the intuition that when any two robots are nearby and collect same information, e.g., track the same targets, then one of the robots is redundant (it would be more effective if the robots were collecting different information). On top of the above reason, another fundamental reason is the communication and computation limitations of the robots versus the r2r messaging requirements by the tasks. For example, in active distributed simultaneous localization and mapping (SLAM), the inter-robot communication messages can grow to a few MBs within a few hundred meters of explored environment [8] (1 MB $=$ 8 Mb). But the wireless r2r communication speeds range from 0.25 Mb/s, e.g., achieved by the Digi XBee 3 Zigbee 3.0 antenna module, to 100 Mb/s, e.g., achieved by the Silvus Tech SL5200 antenna module. Therefore, transmitting even a single message can take seconds.

The current approaches are either near-optimal but not scalable, or real time but offer no performance guarantees. The near-optimal methods [9], [10], [11] that are being used for active information gathering with multiple robots [1], [2], [12], [13], [14], [15], [16], [17], [18], [19], [20], scale cubically with the network size, resulting in impractical running times in mesh networks with tens of robots (up to tens of minutes per action coordination step in the presence of real-world communication delays). The state-of-the-art multirobot motion planning methods are real time [21], [22], [23], [24] but they are suitable in spatial exploration where robots are expected to operate far from

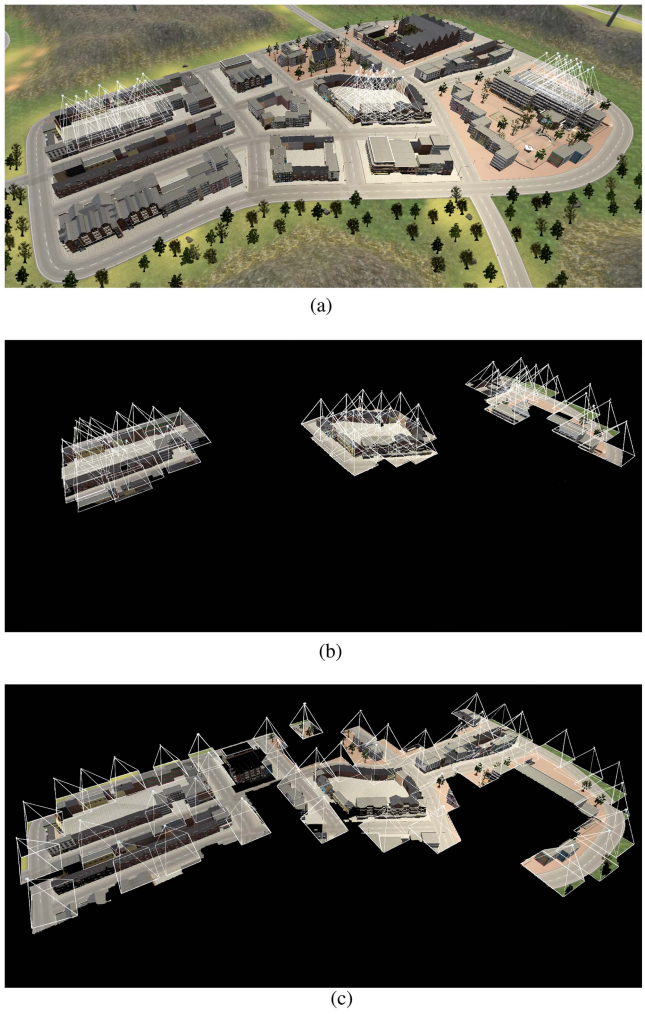


Fig. 1. Scenario of collaborative road detection with 45 aerial robots. The drones are deployed in an unknown environment and tasked to collaboratively detect the roads and visually cover them. Particularly, the drones aim to collectively maximize the total covered road area in their top-view FOV. At each action coordination step, each drone chooses a few other drones to coordinate with, subject to its onboard communication and computation bandwidth constraints. The full collaborative autonomy pipeline is depicted in Fig. 2. (a)–(c) Task progress across time. (a) Drones start from their initial deployment points. (b) Progress after 3 replanning steps. (c) Drones have achieved maximal road coverage.

each other. Then, the submodular structure can be ignored and opportunistic coordination is sufficient. We discuss the related work in extension in Section II.

In this article, instead, we focus on crowded scenarios where robots operate close to one another, and their objective is not merely exploration; it is discovery and/or monitoring of specific information, e.g., landmarks, mobile targets, areas, and objects with specific semantics [1], [5], [26] (see Fig. 1). Then, the submodular structure is present. Thus, to jointly maximize the submodular objective, the robots must intelligently coordinate actions to enable both scalability and near-optimality.

Contributions: We provide a communication- and computation-efficient method for distributed submodular optimization in robot mesh networks. The method, *resource-Aware distributed greedy* (RAG), enables both scalable and near-optimal coordination over robot mesh networks despite real-world communication and computational delays. In contrast, the current approaches are either near-optimal but not scalable or are real-time but offer no performance guarantees. RAG applies to any distributed submodular optimization task. In this article, we evaluate RAG in a semantics-driven area exploration task of road detection and coverage with tens of robots (see Fig. 1), via high-fidelity simulations. In more detail, our technical and simulator contributions are as follows.

We characterize RAG's time complexity and approximation performance as a function of the state of the robot mesh network such as the robots' communication neighborhoods, task objective function, and expected delays due to the robots' computation and communication capabilities.

1) *Time Complexity:* RAG's decision-time scales linearly with the network size. Instead, the competitive near-optimal submodular optimization algorithms scale cubically with the network size. To this end, RAG introduces a resource-minimal distributed optimization paradigm that requires each robot to make decisions based only on locally observed information and information received from and about their neighbors only. In contrast, the current submodular optimization paradigms [9], [10], [11], widely applied in robotics and control [1], [2], [12], [13], [14], [15], [16], [17], [18], [19], [20] that instead require the relay of information about all robots across the network.
2) *Approximation Performance:* We present suboptimality bounds for RAG. The bounds are useful as follows: 1) They capture the approximation performance of RAG as a function of each agent's local mesh-network topology, quantifying how denser mesh networks (more coordination) improve the approximation performance. 2) The bounds inform how to design each agent's communication neighborhood to balance the tradeoff between decision speed and optimality. Particularly, we prove that the approximation performance gain grows sublinearly with the agents' communication neighborhood size. Therefore, in combination with the fact that RAG's decision time scales linearly with the network size, the bounds imply that sparser neighborhoods would favor decision speed without proportionally compromising approximation performance. The appropriate level of neighborhood sparsity that balances the tradeoff for the task at hand can be found via experimentation in high-fidelity simulations, as we illustrate in our simulations.

To perform high-fidelity evaluations, we enhance the physics-based and photo-realistic simulator AirSim [25] to enable scalable simulation of robot mesh networks, including simulation of r2r communication delays (see Fig. 3). To this end, we first implement multithreading for sensor data processing in ROS wrappers, resulting in a tenfold improvement in data frequency across multiple sensing modalities, including RGB, depth, segmentation, inertial measurement unit (IMU), GPS, and barometer. Second, we develop an r2r coordination module that models realistic communication delays and ranges between drones, with signal attenuation and dynamic neighborhood

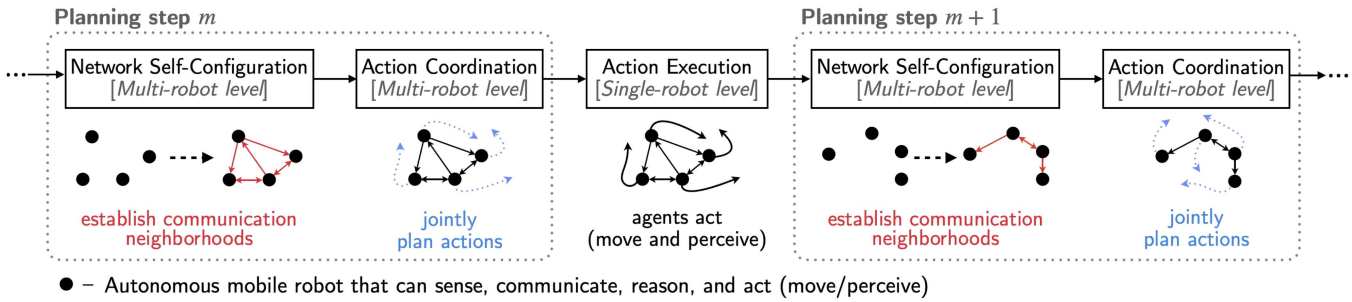


Fig. 2. Pipeline of collaborative mobile autonomy. The robots sequentially perform over a mesh network. (a) *Network self-configuration step*: Given the observed environment and state of the robots, the robots decide with which other robots to communicate with, subject to their onboard bandwidth constraints; and (b) *Action coordination step*: The robots jointly plan actions—how to move and sense the environment—upon coordinating over the established communication network. (c) *Action execution step*: The robots execute their selected actions and perceive the environment. In this article, our focus is on action coordination over tasks that take the form of submodular optimization. We present a communication- and computation-efficient distributed algorithm that scales linearly with the network size, in contrast to the cubic time complexity of the competitive near-optimal algorithms. Along with our algorithmic contribution, we provide a rigorous analysis of how each agent's communication neighborhood affects the near-optimality of the optimization. The analysis implies that establishing sparser neighborhoods favors scalability without proportionally compromising approximation performance.

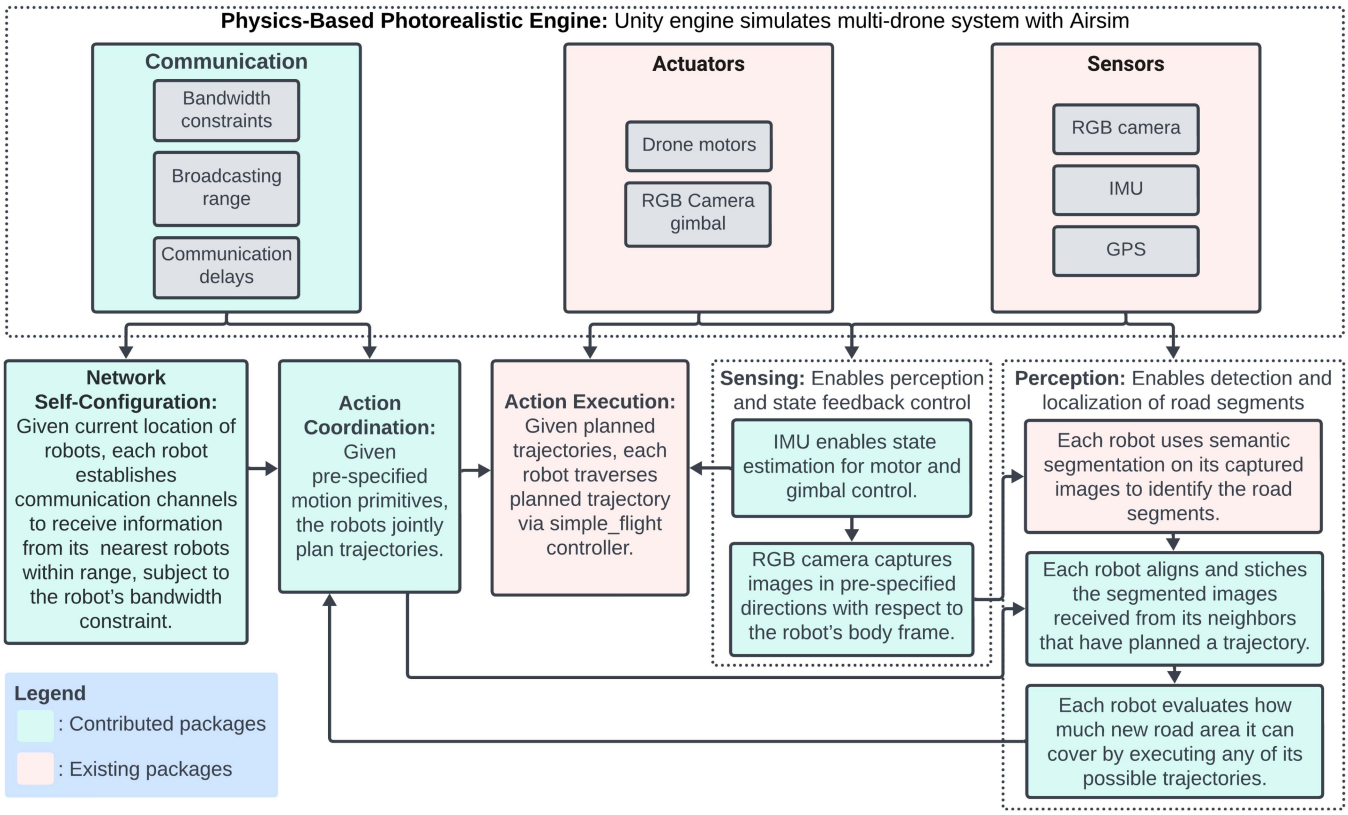


Fig. 3. Simulator pipeline. We provide a high-fidelity simulator that extends AirSim [25] to the multirobot setting by integrating the autonomy pipeline of Fig. 2 and simulating r2r communication delays among other communication constraints. The illustrated pipeline is customized for the action coordination algorithm provided herein. The pipeline can be modified to other algorithms as desired.

relationships managed through the ROS master. Finally, we implemented automatons for capturing segmentation images via rotating gimbals and calculating marginal information gain by stitching semantic data from multiple drones. The implementation uses ROS1, thus inherently incorporating its transport layer delays and message passing overhead, as shown in Figs. 7–10.

Our code is available at https://github.com/UM-iRaL/Resource-Aware-Coordination-AirSim.

Evaluations: We evaluate RAG in the provided simulator across four scenarios of road detection and coverage (see Fig. 1). The scenarios include two team sizes, 15 and 45 robots, and two communication rates, 0.25 Mb/s, e.g., achieved by the Digi XBee

3 Zigbee 3.0 antenna module, and 100 Mb/s, e.g., achieved by the Silvus Tech SL5200 antenna module.

RAG achieves real-time planning, up to three orders of magnitude faster than competitive near-optimal submodular optimization algorithms; and, when the robots maintain a neighborhood size ≥ 2, RAG also achieves superior mean coverage. The experiments also demonstrate RAG scales linearly with the network size, as predicted by our theoretical analysis: from the 15-robot to the 45-robot case, RAG scales linearly, being at most $3\times$ slower compared to the 15-robot case. In contrast, the compared near-optimal algorithms scale cubically, being up to $60\times$ slower compared to the 15-robot case.

The rest of this article is organized as follows. In Section II, we present background on distributed submodular coordination and related work. In Section III, we define the problem of *resource-aware distributed submodular optimization*. Then, we present our algorithm in Section IV-A. The theoretical analysis of the algorithm's approximation guarantees and decision time are presented in Sections V and VI, respectively. Section VII presents the simulator and the evaluations. Finally, Section VIII concludes this article. All proofs are presented in the Appendix.

Comparison with preliminary work [27] and [28]: This article extends our preliminary work [27] to include new theory, the AirSim extension, the evaluations, and proofs of all claims. Particularly, this article provides novel performance guarantees, including an a posteriori bound with network-design implications, and an extension of the a priori bound in [27] to functions that are not submodular. Also, this article bounds the decision time of the algorithm, characterizing for the first time its communication and computation complexity as a function of the network size and the time to perform computations and r2r communications. This article introduces the AirSim-based simulator and the evaluations on the simulator. Instead, Xu and Tzoumas [27] employed MATLAB simulations only without simulating r2r communication delays. All proofs were omitted in [27], and here are presented for all original and new results.

We also compare this article with our preliminary work [28]. First of all, Xu and Tzoumas [28] introduced a more complex problem formulation that requires the mesh-network topology to be co-optimized with the robots' actions. The solution provided therein requires robots at fixed locations, instead of mobile sensors, which are the focus herein. Finally, the results in [28] hold in probability and are based on regret optimization that requires quadratic time to converge, whereas the results herein are deterministic and are based on discrete optimization that requires linear time to converge.

II. Related Work

We discuss related work as follows:

1) near-optimal but not necessarily real-time distributed submodular optimization;
2) rapid but not necessarily near-optimal distributed multirobot motion planning;
3) the tradeoff of decision speed versus optimality;
4) high-fidelity communication simulators.

A. Near-Optimal Distributed Submodular Optimization

The current approximation paradigms for submodular maximization problem [9], [10], [11] have been widely used in active information gathering with multiple robots [1], [2], [12], [13], [14], [15], [16], [17], [18], [19], [20] (see also the survey [5]). However, they may not scale. The reason is that their communication and computation complexities are superlinear in the number of robots, particularly, quadratic or cubic (see Table I). Instead, we provide an algorithm with linear communication and computation complexity.

Quadratic or higher communication and/or computation complexity can become prohibitive in large-scale networks due to real-world communication and computation delays. To illustrate the point, we give a toy example upon noting that communication delays are introduced by the limited r2r communication speeds. For example, state-of-the-art r2r communication speeds range from less than 1 Mb/s up to 100 Mb/s [29]. Computational delays are caused by the time required to perform function evaluation. This time depends on the processing required by the task at hand, e.g., image segmentation for the said task of road coverage (see Fig. 1). Assume now that the total delay per communication and computation is on average 1 ms. Then, for cubic decision-time complexity and 100 robots ($|\mathcal{N}| = 100$), the total time delay is at the order $100^3 \cdot 1\,\text{ms} \simeq 17\,\text{min}$.

The quadratic or higher complexity of [9], [10], [11] over r2r networks is due to their coordination protocols: they instruct the robots to retain and relay information about all or most other robots in the network. Particularly, the authors in [9] and [10] required *iterative decision-making via consensus* where at each iteration each robot needs to retain and transmit estimates of all robots' actions [9], [10]. Fisher et al. [11] required *sequential decision-making* where all currently finalized actions need to be relayed to all robots in the network that have not finalized actions yet.

In more detail, the multirobot coordination algorithm in [30], inspired by [9], [10], [11], achieves the best possible approximation bound for submodular maximization, namely, $1 - 1/e$. However, it may require tens of minutes to terminate in simulated tasks of 10 robots even with no simulated communication delays [27]. The reason is that it requires a near-cubic number of iterations in the number of robots to converge (see Table I). Similarly, for the *sequential greedy* (SG) algorithm [11], also known as *coordinate descent*, which is the gold standard in robotics and control for submodular task optimization [1], [2], [12], [13], [14], [15], [16], [17], [18], [19], [20], although it sacrifices some approximation performance to enable faster decision speed—achieving the bound $1/2$ instead of the bound $1 - 1/e$—has communication complexity that is cubic in the number of robots [28, Appendix II]. The reason is that it requires inter-robot messages that carry information about all the robots and a quadratic number of communication rounds over directed networks [31, Proposition 2].

a) Real-time multirobot motion planning: Current motion planning methods also employ approaches that do not need to account for the submodular structure of the multirobot task: the robots are expected to operate far from one another, collecting

TABLE I
RAG VERSUS STATE-OF-THE-ART DISTRIBUTED SUBMODULAR MAXIMIZATION ALGORITHMS

	Method	Trade-off of Decision Time and Suboptimality Guarantee			Communication Network Topology
		Decision Time: Computation	Decision Time: Communication	Suboptimality Guarantee	
Continuous	Robey et al. [30]	$\Omega(\tau_f \lvert\mathcal{N}\rvert^{2.5} / \epsilon)$	$\Omega(\tau_\# \lvert\mathcal{V}_\mathcal{N}\rvert \lvert\mathcal{N}\rvert^{2.5} / \epsilon)$	$(1-1/e)\,\mathsf{OPT} - \epsilon$	connected, undirected
	Rezazadeh and Kia [52]	$\Omega(\tau_f \lvert\mathcal{N}\rvert^2 \operatorname{diam}(\mathcal{G}) / \epsilon)$	$\Omega(\tau_\# \lvert\mathcal{V}_\mathcal{N}\rvert \lvert\mathcal{N}\rvert^2 \operatorname{diam}(\mathcal{G}) / \epsilon)$	$(1-1/e-\epsilon)\,\mathsf{OPT}$	connected, undirected
	Du et al. [53]	$\sim \Theta(\tau_f \lvert\mathcal{N}\rvert^2)$	$\sim \Theta(\tau_c \lvert\mathcal{N}\rvert^2)$	$(1/2-\epsilon)\,\mathsf{OPT}$	connected, undirected
Discrete	Liu et al. [48]	$O(\tau_f \lvert\mathcal{V}_i\rvert \lvert\mathcal{N}\rvert^2)$	$O((\tau_c+\tau_\#) \lvert\mathcal{N}\rvert^2 \operatorname{diam}(\mathcal{G}))$	$1/2\,\mathsf{OPT}$	connected, directed
	Konda et al. [31]	$\tau_f \lvert\mathcal{V}_i\rvert \lvert\mathcal{N}\rvert$	$O(\tau_c \lvert\mathcal{N}\rvert^2)$	$1/2\,\mathsf{OPT}$	connected, undirected
			$O(\tau_c \lvert\mathcal{N}\rvert^3)$		strongly connected, directed
	Corah and Michael [44]	$O(\tau_f \lvert\mathcal{V}_i\rvert / \epsilon)$, $\leq \tau_f \lvert\mathcal{V}_i\rvert \lvert\mathcal{N}\rvert$	$O(\tau_c / \epsilon^2)$, $\leq \frac{1}{2}\tau_c (\lvert\mathcal{N}\rvert-1)^2$	$1/2\,(\mathsf{OPT} - \epsilon)$	complete
	Gharesifard and Smith [37]	$\leq \tau_f \lvert\mathcal{V}_i\rvert \lvert\mathcal{N}\rvert$	$O(\tau_c \lvert\mathcal{N}\rvert^2)$	$\eta\,\mathsf{OPT},\ \eta \in \left[\frac{1}{\lvert\mathcal{N}\rvert-\omega(\mathcal{G}_{\text{info}})+2}, \frac{\chi(\mathcal{G}_{\text{info}})}{\lvert\mathcal{N}\rvert}\right]$	possibly disconnected, directed
	Grimsman et al. [38]	$\leq \tau_f \lvert\mathcal{V}_i\rvert \lvert\mathcal{N}\rvert$	$O(\tau_c \lvert\mathcal{N}\rvert^2)$	$\psi\,\mathsf{OPT},\ \psi \in \left[\frac{1}{\alpha^\star(\mathcal{G}_{\text{info}})+1}, \frac{1}{\alpha^\star(\mathcal{G}_{\text{info}})}\right]$	
	RAG (this paper)	$\leq \tau_f \max_{i\in\mathcal{N}} (\lvert\mathcal{V}_i\rvert \lvert\mathcal{N}_i\rvert)$	$\leq (\tau_c+\tau_\#)(\lvert\mathcal{N}\rvert-1)$	$1/2\left[\mathsf{OPT} - \sum_{i\in\mathcal{N}} \mathsf{coin}_{f,i}(\mathcal{N}_i)\right]$, $\xi\,\mathsf{OPT},\ \xi \in \left[1-\kappa_f, \frac{1}{1+\kappa_f}\right]$	possibly disconnected, directed

The state of the art is divided into algorithms that optimize (i) in the continuous domain, employing a continuous representation of f [54], and (ii) in the discrete domain. For each algorithm, we present its decision time, split into computation and communication times (see Section VI for definitions of τ_f, τ_c, and $\tau_\#$), and suboptimality guarantee. We also specify the communication network topology that is required by each algorithm. The best performances for each metric are colored in blue. We assume for simplicity that $\lvert\mathcal{V}_i\rvert = \lvert\mathcal{V}_j\rvert, \forall i,j \in \mathcal{N}$.

decoupled information. Then, the submodular objective simplifies to a distributed, modular objective: it can be expressed as the sum of decoupled functions, one for each robot, each computable by the robot based on its action only. Therefore, these methods often allocate robots to unexplored areas and instruct the robots to explore them independently. Opportunistic communication is sufficient, when the robots happen to be nearby. For example, in collaborative exploration, the robots coordinate actions to move to the closest frontiers using shared map information in [32]. The authors in [21], [33], [34], and [35] used an auction-based mechanism for task allocation among robots. The auction is also leveraged for high-probability communication maintenance in [35]. In [22], robots are assigned to different frontiers based on potential functions for distributed mapping and exploration. The robots set periodic meeting destinations in [23] to meet and coordinate actions during distributed exploration. Klodt and Willert [36] introduced a pair-wise coordination protocol where all pairs of neighboring robots sequentially coordinate trajectories to optimize their exploration task allocation. A similar pair-wise protocol is proposed in [24] where each robot coordinates trajectories with the neighbor that it has longest not communicated with.

In contrast, this article focuses on crowded scenarios where robots operate close to one another, and their objective is not merely exploration; it is discovery and/or monitoring of specific information, e.g., landmarks, mobile targets, and areas or objects with specific semantics [1], [5], [26]. Thus, to jointly maximize the submodular objective, the robots must coordinate actions [1]: the submodular objective cannot be decomposed as the sum of locally computable objectives, one for each robot, where the actions of each robot are unaffected by the actions of other robots. In addition, while the discussed task allocation works are application-specific, e.g., exploration [23], [24], our algorithmic results are general-purpose and apply to any distributed submodular optimization setting in robotics, control, machine learning, and beyond.

b) Tradeoff of decision speed versus optimality: To enable rapid distributed submodular optimization, we need to curtail the information explosion in robot mesh networks. Thereby, we need to limit what and how much information can travel across the network. However, the consequence of imposing such information limitations is suboptimal coordinated actions.

Current works have captured the suboptimality cost due to such limited information access for the case of the SG algorithm [18], [37], [38]. The provided characterizations are task-agnostic, holding true for the worst case over all possible submodular functions and action sets. In contrast, we capture the suboptimality cost as a function of the task objective f at hand, the robots' coordinated actions, and the current mesh network topology. Thus, our characterizations, being task specific, (i) can be tighter, and (ii) can be used by the robots to design communication neighborhoods that tune the tradeoff of decision speed and near-optimality, subject to their communication bandwidth constraints.

c) Communication simulators: Papers focus on high-fidelity simulations of wireless communications, including the simulation of protocols and communication limitations such as communication delays and package dropouts. Lizzio et al.[39] simulated communication delays, and demonstrates its influence in consensus-based applications. The authors in [40], [41], and [42] simulate communication channels and protocols, analyzing how distance and line-of-sight conditions impact communication delays. Selden et al. [43] simulated realistic propagation models and scheduling functions for mobile radio-frequency

communications. We instead focus on enabling large-scale photo-realistic simulations with tens of robots, analyzing how communication delays impact active information gathering with robot mesh networks.

III. Resource-Aware Distributed Submodular Optimization

We define the problem of *resource-aware distributed submodular optimization* (Problem 1) with regard to the action coordination module in Fig. 2. To this end, we use the following notation:

1) $\mathcal{N}$ is the set of robots;
2) $\mathcal{E}$ is the set of communication channels among the robots;
3) $f(a \,|\, \mathcal{A}) \triangleq f(\mathcal{A} \cup \{a\}) - f(\mathcal{A})$ is the marginal gain due to adding a to $\mathcal{A}$, given a set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$.

We also use the following framework. For easiness of illustration, we present the framework focusing on multirobot information-gathering tasks such as SLAM, target tracking, surveillance [1], [15], [17]. *Nonetheless, Problem 1 applies to any distributed optimization problem with a submodular objective function.*

Robot dynamics: We assume mobile robots that act as mobile sensors. Their motion dynamics may take the form

$$x_{i,t} = f_i(x_{i,t-1}, u_{i,t-1}) + n_{i,t}, \quad t = 1, 2, \ldots, \tag{1}$$

where $x_{i,t} \in \mathbb{R}^{n_{x_{i,t}}}$ denotes the state of robot i at time step t, $u_{i,t} \in \mathcal{U}_{i,t}$ denotes the control input applied to robot i, where $\mathcal{U}_{i,t}$ is a finite set of admissible control inputs [1], and $n_{i,t}$ is process noise, e.g., Gaussian noise.

Target dynamics: The robots use their sensors to measure and infer information about the environment. For example, in SLAM and target tracking, at each time step t, the robots aim to estimate a target state vector y_t that encapsulates the landmark locations or the mobile targets' to be localized [1]. Generally, y_t evolves with motion dynamics of the form

$$y_t = \phi(y_{t-1}) + w_t, \quad t = 1, 2, \ldots, \tag{2}$$

where w_t denotes process noise, e.g., Gaussian noise.

Sensor model: For each robot i, we assume sensor model

$$z_{i,t} = g_i(x_{i,t}, y_t) + v_{i,t}(x_{i,t}), \quad t = 1, 2, \ldots, \tag{3}$$

where $z_{i,t}$ denotes the measurement taken by robot i at time t, and $v_{i,t}$ is noise that possibly depends on the robots' and targets' state. For example, $v_{i,t}$ may be Gaussian noise with mean $\mu_{v_{i,t}}(x_{i,t}, y_t)$, and covariance $\Sigma_{v_{i,t}}(x_{i,t}, y_t)$.

Communication neighborhood: Before each action coordination step (see Fig. 2), given the observed environment and states of the robots, the robots decide with which others to establish communication, subject to their onboard bandwidth constraints. Specifically, we assume that each robot i can receive information from up to α_i other robots due to onboard bandwidth constraints ($|\mathcal{N}_i| \leq \alpha_i$).

When a communication channel is established from robot j to robot i, i.e., $(j \to i) \in \mathcal{E}$, then robot i can receive, store, and process information from robot j. The set of all robots that robot i receives information from is denoted by $\mathcal{N}_i$. We refer to $\mathcal{N}_i$ as robot is *neighborhood*.

Communication network: The resulting communication network can be directed and even disconnected. When the network is fully connected (all robots receive information from all others), we call it *fully centralized*. In contrast, when the network is fully disconnected (all robots receive no information from other robots), we call it *fully decentralized*.

We assume communication to be synchronous.

Communication data rate: All communication channels $(j \to i) \in \mathcal{E}$ have finite *data rates* (communication speeds). In the simulations, we assume two values for the data rates: 0.25 Mb/s (simulating the Digi XBee 3 Zigbee 3.0 antenna), and 100.0 Mb/s (simulating the Silvus SL5200 antenna). Due to the finite data rates, the decision time of action coordination depends on both (i) the *number of communication rounds* and (ii) the *size of transmitted messages* it requires for the robots to find a joint plan. We assume that all communication channels have the same data rate in this article.

Objective function: At each action coordination step, the robots choose actions to maximize an objective function f that captures the current multirobot task. For example, in SLAM and target tracking, at each time step t, f can be [1]

$$f(\{a_i\}_{i \in \mathcal{N}}) = \sum_{\tau=t+1}^{t+T} \mathbf{h}(y_\tau | \{z_{i,\, t+1:\tau}(a_i)\}_{i \in \mathcal{N}}) \tag{4}$$

per the robot, sensing, and target models in (1), (3), and (2)

$$a_i \triangleq \{u_{i,t+1}, \ldots, u_{i,t+T}\}$$

$z_{i,t+1:\tau}(a_i) \triangleq \{z_{i,t+1}(a_i), \ldots, z_{i,\tau}(a_i)\}$ denotes the measurements induced by a_i up until τ, T is an action-planning look-ahead horizon, and $\mathbf{h}(\cdot|\cdot)$ is an information metric such as the conditional entropy [1] or the mutual information [15]. When $\mathbf{h}$ is conditional entropy and the process and sensor noises in (1), (3), and (2) are uncorrelated Gaussian, then (4) is computable a priori given any action set $\{a_i\}_{i \in \mathcal{N}}$: $\mathbf{h}$ is then equal to the $\log\det$ volume of the Kalman filtering uncertainty over the horizon T and is, thus, independent of the measurements' realization [1]. Conditional entropy, mutual information, and, more broadly, covering functions [1], [15], [30], [44], [45] are used to model active information-gathering tasks such as target tracking, SLAM, and surveillance [1], [15], [17]. These functions capture how much information is observed given the actions of all robots. They belong to a broader class of functions that satisfy the properties below (Definitions 1 and 2).

Definition 1 (Normalized and Non-Decreasing Submodular Set Function [11]): A set function $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is *normalized and nondecreasing submodular* if and only if:

1) $f(\emptyset) = 0$;
2) $f(\mathcal{A}) \leq f(\mathcal{B})$, for any $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}$;
3) $f(s \,|\, \mathcal{A}) \geq f(s \,|\, \mathcal{B})$, for any $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}$ and $s \in \mathcal{V}$.

Normalization ($f(\emptyset) = 0$) holds without loss of generality. In contrast, monotonicity and submodularity are intrinsic to the function. Intuitively, if $f(\mathcal{A})$ captures the area *covered* by a set $\mathcal{A}$ of activated cameras, then the more sensors are activated ($\mathcal{A} \subseteq \mathcal{B}$), the more area is covered ($f(\mathcal{A}) \leq f(\mathcal{B})$); this is the

nondecreasing property. Also, the marginal gain of covered area caused by activating a camera s *drops* ($f(s\,|\,\mathcal{A}) \geq f(s\,|\,\mathcal{B})$) when *more* cameras are already activated ($\mathcal{A} \subseteq \mathcal{B}$); this is the submodularity property.

Definition 2 (Second-order Submodular Set Function [46], [47]): $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is *second-order submodular* if and only if

$$f(s\,|\,\mathcal{C}) - f(s\,|\,\mathcal{A}\cup\mathcal{C}) \geq f(s\,|\,\mathcal{B}\cup\mathcal{C}) - f(s\,|\,\mathcal{A}\cup\mathcal{B}\cup\mathcal{C}) \tag{5}$$

for any *disjoint* $\mathcal{A},\mathcal{B},\mathcal{C} \subseteq \mathcal{V}$ ($\mathcal{A}\cap\mathcal{B}\cap\mathcal{C} = \emptyset$) and $s \in \mathcal{V}$.

The second-order submodularity is another intrinsic property of the function. Intuitively, if $f(\mathcal{A})$ captures the area *covered* by a set $\mathcal{A}$ of cameras, then *the marginal gain of marginal gains* drops when more cameras are already activated.

Problem 1 (Resource-Aware Distributed Submodular Optimization): Consider a normalized and nondecreasing submodular function $f : 2^{\prod_{i\in\mathcal{N}}\mathcal{V}_i} \mapsto \mathbb{R}$ that captures the optimization task at hand at the current action coordination step. Each robot $i \in \mathcal{N}$ selects an action $a_i \in \mathcal{V}_i$, *using only information from and about its neighbors* $\mathcal{N}_i$, such that the actions $\{a_i\}_{i\in\mathcal{N}}$ jointly solve the optimization problem:[1]

$$\max_{a_i\in\mathcal{V}_i,\forall i\in\mathcal{N}} f(\{a_i\}_{i\in\mathcal{N}}). \tag{6}$$

Remark 1 (Generality): Problem 1 applies to any distributed optimization problem in control, robotics, machine learning, and beyond where the objective function is nondecreasing and submodular, including the information-gathering tasks captured via (1) to (4).[2]

Problem 1 is resource-aware in that it requires each robot to coordinate actions with its neighbors only, receiving information only about them instead of more robots in the network. The reason is to curtail the explosion of information passing across the network and, thus, to enable rapid coordination. This is in contrast to standard distributed methods that allow information about the whole network to travel to all other robots via information passing (multihop communication) [31], [44], [48]. Multihop communication does not reduce the amount of information flowing in the network compared to centralized methods, introducing impractical communication delays, as we will demonstrate in the simulations.

IV. RAG Algorithm

We present the RAG algorithm. Examples of how the algorithm works are given in Fig. 4. Therein, we also compare RAG to the SG algorithm [11]. SG is the "gold standard" in submodular maximization. SG is presented in (see Section IV-B).

A. RAG *Algorithm*

The pseudocode of RAG, as it is used onboard an robot i, is presented in Algorithm 1. The purpose of each iteration of RAG, namely, of each "while loop" (lines 2–15), is to enable robot i to decide whether to select an action over its neighbors at this iteration or to pass because a neighbor has an action with a higher marginal gain. If passing, then the robot must wait for a future iteration to select an action. In more detail, at each "while loop."

1) Robot i finds an action a_i with the highest marginal gain g_i given the actions selected by neighbors $\mathcal{I}_i \subseteq \mathcal{N}_i$ so far (lines 3–4).
2) Robot i receives the respective highest marginal gain g_j of all neighbors j that have not selected an action yet, namely, of all $j \in \mathcal{N}_i \setminus \mathcal{I}_i$ (line 5).
3) Robot i compares g_i with all g_js (line 6).
4) If $g_i > g_j, \forall j \in \mathcal{N}_i \setminus \mathcal{I}_i$, then robot i selects a_i, i.e., $a_i^{\mathsf{RAG}} \leftarrow a_i$, broadcast a_i^{RAG}, and RAG terminates onboard robot i (lines 6–8 and 2, respectively).
5) Otherwise, robot i passes (line 9), and receives the actions selected at this iteration by its neighbors with the highest marginal gain among their respective neighbors, if any (line 11)—the set of these neighbors is denoted as $\mathcal{S}_i^{\mathsf{new}}$ (line 10). Particularly, $\mathcal{S}_i^{\mathsf{new}}$ may be empty if no neighbor can select an action per their onboard iteration of RAG.

Algorithm 1: Resource-Aware Distributed Greedy (RAG).

Input: robot i's actions $\mathcal{V}_i$; neighborhood $\mathcal{N}_i$; non-decreasing set function $f : 2^{\mathcal{V}_\mathcal{N}} \mapsto \mathbb{R}$;
Output: robot i's action a_i^{RAG}.

1: $\mathcal{I}_i \leftarrow \emptyset$; $\mathcal{A}_i \leftarrow \emptyset$; $a_i^{\mathsf{RAG}} \leftarrow \emptyset$; // $\mathcal{I}_i$ is the robots in $\mathcal{N}_i$ that have selected their actions; $\mathcal{A}_i$ stores $\mathcal{I}_i$'s selected actions; a_i^{RAG} is robot i's selected action
2: **while** $a_i^{\mathsf{RAG}} = \emptyset$ **do**
3: $a_i \leftarrow \arg\max_{a\in\mathcal{V}_i} f(\mathcal{A}_i \cup \{a\}) - f(\mathcal{A}_i)$;
4: $g_i \leftarrow f(\mathcal{A}_i \cup \{a_i\}) - f(\mathcal{A}_i)$;
5: **receive** $\{g_j\}_{j\in\mathcal{N}_i\setminus\mathcal{I}_i}$;
6: **if** $i = \arg\max_{j\in\mathcal{N}_i\cup\{i\}\setminus\mathcal{I}_i} g_j$ **then**
7: $a_i^{\mathsf{RAG}} \leftarrow a_i$; // i has the best action candidate across $\mathcal{N}_i \cup \{i\} \setminus \mathcal{I}_i$ and it selects this action candidate
8: **broadcast** a_i^{RAG} **to** out-neighbors; // out-neighbors can be different from $\mathcal{N}_i$
9: **else**
10: **denote** as $\mathcal{S}_i^{\mathsf{new}} \subseteq \mathcal{N}_i \setminus \mathcal{I}_i$ the set of neighbor(s) that just selected action(s) in this iteration; // $\mathcal{S}_i^{\mathsf{new}}$ may be empty as we explain in Section IV-A
11: **receive** a_j^{RAG} **from** each robot $j \in \mathcal{S}_i^{\mathsf{new}}$;
12: $\mathcal{I}_i \leftarrow \mathcal{I}_i \cup \mathcal{S}_i^{\mathsf{new}}$;
13: $\mathcal{A}_i \leftarrow \mathcal{A}_i \cup \{a_j^{\mathsf{RAG}}\}_{j\in\mathcal{S}_i^{\mathsf{new}}}$;
14: **end if**
15: **end while**
16: **return** a_i^{RAG}.

Remark 2 (Directed, Possibly Disconnected Communication Topology): RAG is valid for directed and even disconnected communication topologies. For example, RAG can be applied to a robot i that is completely disconnected from the network.

B. *Comparison to the* SG *Algorithm*

RAG is compared with SG [11] in Fig. 4. We rigorously present SG next, and provide a qualitative comparison with RAG. The

[1] We extend our framework to any normalized, nondecreasing, and merely submodular or approximately submodular function f in the Appendix.

[2] We refer the reader to the papers [1], [15] for implementation details on rapidly computing single-robot actions $a_i \triangleq \{u_{i,t+1},\ldots,u_{i,t+T}\}$ that optimize f per (1) to (4), given actions for the remaining robots.

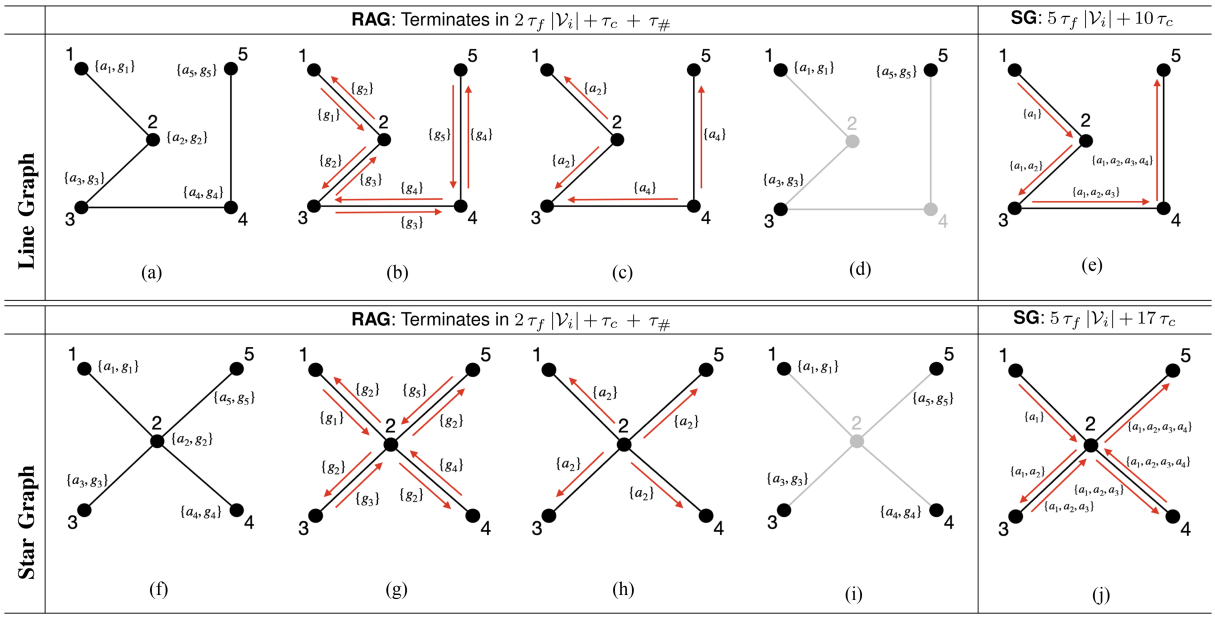


Fig. 4. RAG versus SG [11]. The two algorithms are compared in their execution steps and in the time they need to terminate. We present the case of five agents in two scenarios with the communication networks being (i) an undirected line graph (top row), and (ii) an undirected star graph (bottom row). Nodes 1 to 5 are the agents 1 to 5; black lines are undirected communication links; $\{a_i, g_i\}$ denotes agent is newly updated action candidate a_i with marginal gain g_i; $\{a_i\}$ and $\{g_i\}$ alongside red arrows are the actions and marginal gain values being transmitted between two agents; the transparent nodes are the agents that have selected their actions and, thus, already ended running RAG; and the transparent edges are the communication channels that have become "disappeared" since at least one end of them has finished RAG. The implementation of RAG [(a)–(d) and (f)–(i)] results in shorter decision time than SG [(e) and (j)] in both scenarios with the line and star graphs, respectively. (a) First iteration of RAG starts. Each agent i simultaneously finds its action candidate a_i with the largest marginal gain g_i from all available actions $\mathcal{V}_i$. The operation takes $\tau_f\,|\mathcal{V}_i|$. (b) Each agent i simultaneously receives g_j from each neighbor $j \in \mathcal{N}_i$. This takes $\tau_\#$. Then, it compares g_i with them. We assume $g_2 = \max(g_1, g_2, g_3)$ and $g_4 = \max(g_3, g_4, g_5)$. (c) Thus, agents 2 and 4 get to select actions. Agents 1, 3, 5 simultaneously receive the actions selected by their neighbors. This takes τ_c. The first iteration ends. Only 1, 3, 5 continue. (d) Second iteration starts. Given the received actions, agents 1, 3, 5 simultaneously select actions. This requires $\tau_f\,|\mathcal{V}_i|$. Then, all agents have selected actions, and RAG terminates. (e) Sequentially, from $i = 1$ to 5, agent i receives $\{a_1, \ldots, a_{i-1}\}$, in $\tau_c\,(i-1)$ time, then selects a_i, which takes $\tau_f\,|\mathcal{V}_i|$ time, and then transmits $\{a_1, \ldots, a_i\}$ to agent $i+1$. (f) First iteration of RAG starts. Each agent i simultaneously finds its action candidate a_i with the largest marginal gain g_i from all available actions $\mathcal{V}_i$. The operation takes $\tau_f\,|\mathcal{V}_i|$. (g) Each agent i simultaneously receives g_j from each neighbor $j \in \mathcal{N}_i$. This takes $\tau_\#$. Then, it compares g_i with them. We assume $g_2 = \max(g_1, g_2, g_3, g_4, g_5)$. (h) Thus, agent 2 gets to select an action. Agents 1, 3, 4, 5 simultaneously receive the action selected by their neighbor. This takes τ_c. The first iteration ends. Only 1, 3, 4, 5 continue. (i) Second iteration starts. Given the received action, agents 1, 3, 4, 5 simultaneously select actions. This requires $\tau_f\,|\mathcal{V}_i|$. Then, all agents have selected actions, and RAG terminates. (j) From $i = 1$ to 5, agent i receives $\{a_1, \ldots, a_{i-1}\}$, possibly via r_i relay nodes (takes $\tau_c\,(r_i+1)\,(i-1)$), then selects a_i (takes $\tau_f\,|\mathcal{V}_i|$), and then transmits $\{a_1, \ldots, a_i\}$ to agent $i+1$.

rigorous comparison of runtime and approximation performance is postponed to Sections V and VI, where, e.g., we prove that RAG's runtime scales linearly with the number of the robots whereas SG's scales cubically.

SG instructs the robots to sequentially select actions such that the ith robot in the sequence selects

$$a_i^{\mathsf{SG}} \in \max_{a \in \mathcal{V}_i} f(a \mid \{a_1^{\mathsf{SG}}, \ldots, a_{i-1}^{\mathsf{SG}}\}) \tag{7}$$

i.e., a_i^{SG} maximizes the marginal gain over the actions that have been selected by the $i-1$ previous robots in the sequence. In contrast, RAG enables the robots to select actions in parallel, and even if not all their neighbors have selected an action.

The above action-selection features of RAG—parallelization and action selection before all neighbors have chosen an action—can further speed up the algorithm's termination. Also, they enable RAG to work on arbitrary communication topologies. In contrast, SG requires a line path connecting all robots in the action-selection sequence. If such a path does not exist (see the star graph example in Fig. 4), the ith robot in the action-selection sequence cannot communicate directly with the $(i+1)$th robot. Then, SG requires extra communication rounds for message relaying, further delaying its termination. Specifically, given the limited communication speed of r2r communication channels [49], the termination of SG is delayed due to both the increased number of communication rounds, and the communication delay incurred from relaying the actions of multiple robots—the robots in the sequence that have chosen an action so far—across the network.

V. Approximation Guarantees: Centralization Versus Decentralization Perspective

We present a priori and posteriori suboptimality bounds for RAG (Theorems 1 and 2). To this end, we first introduce the notion of *centralization of information* (coin) to quantify the a priori bound (see Section V-A). Then, we present the a priori bound (see Section V-B) and the a posteriori bound (see Section V-C). Finally, we compare the a priori bound with the a priori bounds in the state of the art (see Section V-D).

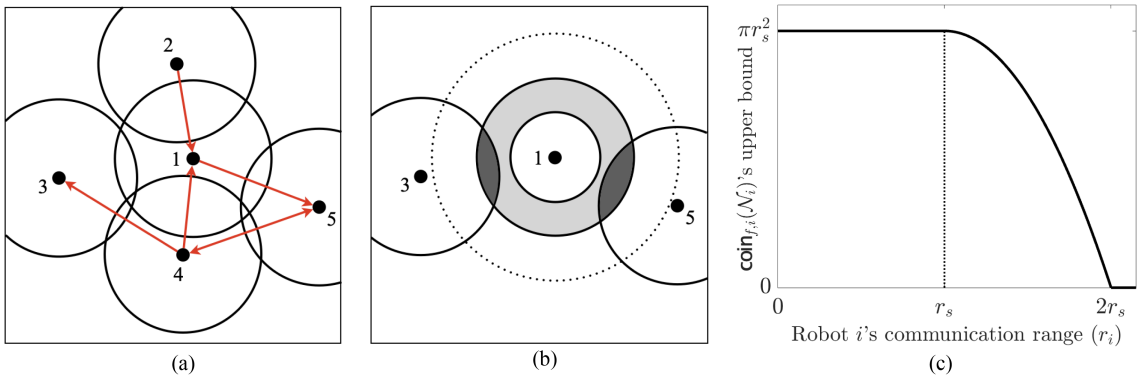


Fig. 5. Multirobot network for area coverage. (a) Scenario with 5 robots, their communication network, and a limited square area that the robots are tasked to cover. (b) Illustration of robot 1's non-neighbors $\mathcal{N}_1^c$, $\mathsf{coin}_{f,1}(\mathcal{N}_1)$, and worst-case $\mathsf{coin}_{f,1}$ $\mathsf{coin}_{f,1}(\mathcal{N}_1)$. (c) Computable upper bound of $\mathsf{coin}_{f,i}$ as a function of robot is communication range per the analysis in Example 1. (a) *Setup.* The robots are tasked to maximize the area covered among the available square area by picking locations to stay at. Each robot (dot) has an FOV (circle), and established communication channels with its neighbors (red arrows). For example, for robot 1, its neighbors are the robots in $\mathcal{N}_1 = \{2, 4\}$, thus, its non-neighbors are those in $\mathcal{N}_1^c = \{3, 5\}$. (b) Robot 1's non-neighbors $\mathcal{N}_1^c$, $\mathsf{coin}_{f,1}(\mathcal{N}_1)$, and worst-case $\mathsf{coin}_{f,1}$. The non-neighbors $\mathcal{N}_1^c = \{3, 5\}$ are the robots that robot 1 does not communicate range, e.g., because of limited bandwidth or because they are outside its communication range (dashed circle). $\mathsf{coin}_{f,1}(\mathcal{N}_1)$ is depicted by the dark gray area, and its upper bound the light gray ring area (Example 1). (c) Computable upper bound of $\mathsf{coin}_{f,i}$ as a function of robot is communication range. r_s is the robots' sensing radius (Example 1).

A. Centralization of Information

We introduce the notion of coin. We use the notion to quantify the a priori suboptimality cost due to decentralization. coin measures how the agents' actions can overlap due to not coordinating with their non-neighbors. We also relate coin to curvature [50] and pair-wise consistency [44] (Remarks 3 and 4), and show that coin is a less conservative measure of action overlap. We use the following notation and definition.

1) $\mathcal{N}_i^c \triangleq \mathcal{N} \setminus \{\mathcal{N}_i \cup \{i\}\}$ is the set of robot is non-neighbors, i.e., the robots beyond is neighborhood [see Fig. 5(b)].

Definition 3 (Curvature [50]): Consider a function $f : 2^{\mathcal{V}_\mathcal{N}} \to \mathbb{R}$ that is nondecreasing and submodular. Without loss of generality, we assume that for any action $a \in \{\mathcal{V}_i\}_{i\in\mathcal{N}}$, it holds that $f(a) \neq 0$. The curvature of f is defined as

$$\kappa_f \triangleq 1 - \min_{\mathcal{A}\in\mathcal{V}_\mathcal{N}} \min_{a\in\mathcal{A}} \frac{f(\mathcal{A}) - f(\mathcal{A}\setminus\{a\})}{f(a)}. \tag{8}$$

κ_f measures how much an agent's action a can overlap with other agents' actions. Particularly, $\kappa_f \in [0, 1]$, and if $\kappa_f = 0$, then $f(\mathcal{A}) - f(\mathcal{A}\setminus\{a\}) = f(a)$, for all $a \in \mathcal{A}$, i.e., no agent's action overlaps with any other agent's actions. In contrast, if $\kappa_f = 1$, then there exists an action $a \in \mathcal{A}$ such that $f(\mathcal{A}) = f(\mathcal{A}\setminus\{a\})$, i.e., action a has no contribution to $f(\mathcal{A})$ in the presence of all other agents.

*Definition 4 (*coin*):* Consider a function $f : 2^{\mathcal{V}_\mathcal{N}} \mapsto \mathbb{R}$ and a communication network $\{\mathcal{N}_i\}_{i\in\mathcal{N}}$ where each agent $i \in \mathcal{N}$ has an selected an action a_i. Then, agent is coin is defined as

$$\mathsf{coin}_{f,i}(\mathcal{N}_i) \triangleq f(a_i) - f(a_i \,|\, \{a_j\}_{j\in\mathcal{N}_i^c}). \tag{9}$$

$\mathsf{coin}_{f,i}$ measures how much a_i can overlap with the actions of agent is non-neighbors. In the best case where a_i does not overlap at all, i.e., $f(a_i \,|\, \{a_j\}_{j\in\mathcal{N}_i^c}) = f(a_i)$, then $\mathsf{coin}_{f,i} = 0$. In the worst case instead where a_i is fully overlapped, i.e., $f(a_i \,|\, \{a_j\}_{j\in\mathcal{N}_i^c}) = 0$, then $\mathsf{coin}_{f,i} = f(a_i)$.

From an information-theoretic perspective, $\mathsf{coin}_{f,i}$ measures how much the information collected by a_i overlaps with the information collected by $\{a_j\}_{j\in\mathcal{N}_i^c}$. Rigorously, if f is an entropy metric, then $\mathsf{coin}_{f,i}$ is mutual information [51]. Thus, $\mathsf{coin}_{f,i} = 0$ if and only if the information collected by a_i is decentralized from (independent of) the information collected by $\{a_j\}_{j\in\mathcal{N}_i^c}$. In this sense, coin_f captures the decentralization of information across the network.

Remark 3 (Relation to Curvature [50]): coin_f is a less conservative measure of action overlap compared to κ_f. κ_f measures the overlap of an agent's action with the actions of all other agents, whereas coin_f measures the overlap of an agent's action with the actions of its non-neighbors only. Particularly, we prove that, for all $i \in \mathcal{N}$, $\mathsf{coin}_{f,i}/f(a_i) \leq \kappa_f$ (see Proposition 1, which is presented later on in this section).

Remark 4 (Relation to Pairwise Redundancy [44]): coin_f generalizes the notion of *pairwise redundancy* to capture the action overlap among multiple agents instead of a pair. Specifically, given any two agents i and j, their *pair-wise consistency* is defined as $w_{ij} \triangleq \max_{s_i\in\mathcal{V}_i} \max_{s_j\in\mathcal{V}_j, j\in\mathcal{N}_i} [f(s_i) - f(s_i \,|\, s_j)]$. In contrast, $\mathsf{coin}_{f,i}$ captures the action overlap between an agent i and its *non*-neighbors, capturing that way the decentralization of information across the network.

By measuring how much agent is action overlaps with the actions of its non-neighbors, $\mathsf{coin}_{f,i}$ equivalently captures agent is suboptimality cost due to not coordinating with its non-neighbors. We, thus, expect that the more neighbors agent i has the smaller is $\mathsf{coin}_{f,i}$. Indeed, the following result holds.

Proposition 1 (Monotonicity): For any $i \in \mathcal{N}$, $\mathsf{coin}_{f,i}(\mathcal{N}_i)$ is non-increasing in $\mathcal{N}_i$. Its least and maximum values, attained for

$\mathcal{N}_i = \mathcal{N} \setminus \{i\}$ and $\mathcal{N}_i = \emptyset$, respectively, are as follows:

$$\underbrace{0 = \mathsf{coin}_{f,i}(\mathcal{N} \setminus \{i\})}_{\text{full centralization}} \leq \mathsf{coin}_{f,i}(\mathcal{N}_i) \leq \underbrace{\mathsf{coin}_{f,i}(\emptyset) = \kappa_f\, f(a_i)}_{\text{full decentralization}}. \tag{10}$$

The sum of all $\{\mathsf{coin}_{f,i}\}_{i\in\mathcal{N}}$ will be used in the next section to characterize the global suboptimality cost due to decentralization. Given this characterization, we may want to enable the agents to pick their neighborhoods to minimize coin subject to their communication-bandwidth constraints. coin_f can be uncomputable a priori since agent i will not have access to the actions of its non-neighbors. Notwithstanding, finding a computable upper bound for $\mathsf{coin}_{f,i}$ may be easy, as we demonstrate in the following example.

Example 1 (Computable Upper Bound: Example of Area Coverage): Consider an area coverage task where each robot carries a camera with a circular field-of-view (FOV) of radius r_s [see Fig. 5(a)]. Consider that each robot i has fixed its neighborhood $\mathcal{N}_i$ by picking a communication range r_i. Then, $\mathsf{coin}_{f,i}$ is equal to the overlap of the FOVs of robot i and its non-neighbors. Since the number of robot is non-neighbors may be unknown, an upper bound to $\mathsf{coin}_{f,i}$ is the gray ring area in Fig. 5(b), obtained assuming an infinite amount of non-neighbors around robot i, located just outside the boundary of is communication range. Specifically

$$\mathsf{coin}_{f,i} \leq \max(0,\ \pi[r_s^2 - (r_i - r_s)^2]). \tag{11}$$

The bound as a function of the communication range r_i is plotted in Fig. 5(c). It tends to zero for increasing r_i, as expected. When the distance of agent i for its nearest non-neighbor is larger than $2r_s$, then the FOVs of agent i and its non-neighbors cannot overlap, thus, $\mathsf{coin}_{f,i} = 0$.

B. *Priori Suboptimality Bound of* RAG

We present the a priori suboptimality bound of RAG. by bounding coin_f with a computable bound as a function of the agents' neighborhoods, we enable the agents to optimize their neighborhoods to maximize the suboptimality bound of RAG subject to their communication-bandwidth constraints.

We focus the presentation on nondecreasing and doubly submodular functions, for sake of simplicity. In Appendix I (Corollary 4), we generalize the results to functions that are nondecreasing and submodular or approximately submodular.

We use the following notation:

1) $\mathcal{A}^{\mathsf{OPT}} \in \arg\max_{a_i\in\mathcal{V}_i,\forall i\in\mathcal{N}} f(\{a_i\}_{i\in\mathcal{N}})$, i.e., $\mathcal{A}^{\mathsf{OPT}}$ is an optimal solution to Problem 1;
2) $\mathcal{A}^{\mathsf{RAG}} \triangleq \{a_i^{\mathsf{RAG}}\}_{i\in\mathcal{N}}$ is RAG's output for robots $\mathcal{N}$.

Theorem 1 (A Priori Suboptimality Bound): Given a communication topology $\{\mathcal{N}_i\}_{i\in\mathcal{N}}$, RAG guarantees

$$f(\mathcal{A}^{\mathsf{RAG}}) \geq \frac{1}{1+\kappa_f}\left[f(\mathcal{A}^{\mathsf{OPT}}) - \kappa_f \sum_{i\in\mathcal{N}} \mathsf{coin}_{f,i}(\mathcal{N}_i)\right]. \tag{12}$$

Theorem 1 captures the intuition that when the agents coordinate with fewer other agents, then the approximation performance will deteriorate. This intuition is made rigorous by applying Proposition 1 to (12) along the spectrum from fully centralized to fully decentralized networks.

1) If $\mathcal{G}$ is *fully centralized* (all agents communicate with all), then the approximation bound in (12) becomes

$$f(\mathcal{A}^{\mathsf{RAG}}) \geq \frac{1}{1+\kappa_f} f(\mathcal{A}^{\mathsf{OPT}}) \tag{13}$$

i.e., RAG is near-optimal, matching the approximation ratio $1/(1+\kappa_f)$ of the seminal SG algorithm [50]. The bound $1/(1+\kappa_f)$ is near-optimal since the best possible bound for the optimization problem in (6) is $1-\kappa_f/e$ [10].

2) If $\mathcal{G}$ is *in between* fully centralized and fully decentralized, then the approximation bound in (12) captures as is the cost of decentralization. It does so through coin_f, which measures how the agents' actions overlap due to not coordinating with all others. Specifically, as the network becomes less and less centralized (the agents have less neighbors), then suboptimality bound in (12) deteriorates since, for all $i \in \mathcal{N}$, $\mathsf{coin}_{f,i}(\mathcal{N}_i)$ increases when the neighborhood $\mathcal{N}_i$ becomes smaller (Proposition 1).

3) If $\mathcal{G}$ is *fully decentralized* (all agents isolated), then the approximation bound in (12) becomes

$$f(\mathcal{A}^{\mathsf{RAG}}) \geq \frac{1}{1+\kappa_f}\left[f(\mathcal{A}^{\mathsf{OPT}}) - \kappa_f \sum_{i\in\mathcal{N}} \mathsf{coin}_{f,i}(\emptyset)\right] \tag{14}$$

$$\in \left[1-\kappa_f,\ \frac{1}{1+\kappa_f}\right] f(\mathcal{A}^{\mathsf{OPT}}). \tag{15}$$

Equation (14) captures the intuition that when the agents' actions do not overlap, then no communication still leads to near-optimal performance. For example, per the area coverage Example 1, when the agents are sufficiently far away such that their FOVs cannot overlap upon executing their actions, then $\mathsf{coin}_{f,i}(\emptyset) = 0$ for all $i \in \mathcal{N}$. Particularly, then the bound in (14) becomes $1/(1+\kappa_f)$, matching the fully centralized performance.

In the worst case, the bound in (12) takes the value $1-\kappa_f$, and becomes zero when the actions of all agents fully overlap with each other ($\kappa_f = 1$). This is inevitable since all agents ignore all others and, thus, cannot coordinate actions to reduce the overlap.

The tightness of the bound will be analyzed in future work.

Corollary 1 (A Priori Bound with Approximate Greedy Selection): If Algorithm 1's line 3 can only perform *approximate* greedy selection such that $f(a_i) \geq \eta\, f(a_i^{\mathsf{OPT}} \mid \mathcal{A}_i), 0 < \eta \leq 1$, then RAG guarantees

$$f(\mathcal{A}^{\mathsf{RAG}}) \geq \frac{\eta}{1+\eta\kappa_f}\left[f(\mathcal{A}^{\mathsf{OPT}}) - \left(\frac{1}{\eta} - 1 + \kappa_f\right)\sum_{i\in\mathcal{N}} \mathsf{coin}_{f,i}(\mathcal{N}_i)\right]. \tag{16}$$

Also, it holds that

$$\frac{f(\mathcal{A}^{\text{RAG}})}{f(\mathcal{A}^{\text{OPT}})} \geq \begin{cases} \frac{\eta}{1+\eta\kappa_f}, & \mathcal{G} \text{ is fully centralized} \\ \eta(1-\kappa_f), & \mathcal{G} \text{ is not fully centralized.} \end{cases} \quad (17)$$

RAG still has near-optimal performance guarantees even with approximate greedy selection [13], [15]. When $\eta = 1$, Corollary 1 reduces to Theorem 1.

C. A Posteriori Suboptimality Bound of RAG

We present the a posteriori approximation bound of RAG (see Theorem 2). We recall the notation:

1) $\mathcal{I}_i$ is robot is neighbors that select actions prior to i during the execution of RAG.

Theorem 2 (A Posteriori Suboptimality Bound): Given the actions $\{a_i^{\text{RAG}}\}_{i\in\mathcal{N}}$ selected by the agents, RAG guarantees:[3]

$$f(\mathcal{A}^{\text{RAG}}) \geq f(\mathcal{A}^{\text{OPT}}) - \kappa_f \sum_{i\in\mathcal{N}} f(a_i^{\text{RAG}} \mid \mathcal{A}_{\mathcal{I}_i}^{\text{RAG}}). \quad (18)$$

Theorem 2 captures the suboptimality cost due to decentralization, similarly to Theorem 1. In contrast to Theorem 1, Theorem 2 captures the decentralization cost as a function of the action overlap between agent i and its neighbors, instead of its non-neighbors. As such, Theorem 2 captures the intuition that the larger agent is neighborhood is, the better the suboptimality guarantee can be since then agent i would have the chance to coordinate actions with more agents.

Proposition 2 (Approximate Submodularity of A Posteriori Bound): The right-hand side of (18) is nondecreasing and approximate submodular as a function of $\{\mathcal{I}_i\}_{i\in\mathcal{N}}$.

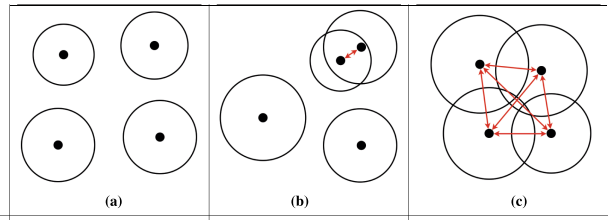
The proposition implies that although the approximation performance will improve if the agents have more neighbors, the gained improvement diminishes.

Remark 5 (Trade-Off between Decision Speed and Optimality): For larger neighborhoods, the marginal increase in the approximation guarantee is negatively outweighed by a greater increase in decision time. The reason is that the gain in the approximation guarantee diminishes as the neighborhoods become larger (see Theorem 2) while the decision time may increase linearly for RAG and superlinearly for the state of the art, as we present in Section VI.

The appropriate size of neighborhoods that balances the tradeoff for the task at hand can be found via experimentation in high-fidelity simulations, as we illustrate in the experiments.

D. Comparison to the State of the Art

We summarize the approximation guarantees of the state of the art and RAG in Table I. We observe the tradeoff of decision time and optimality: the algorithms with the best suboptimality guarantees—first six rows of the table, achieving the near-optimal $1/2$ or $1-1/e$ [10]—can exhibit also the worst decision times, one to two orders of the network size higher than that of RAG. Among the remaining algorithms—three last rows of the table— RAG is the only algorithm that provides task-aware (f-based) performance guarantees and that quantifies the suboptimality guarantee as a function of each agent's local communication network. Instead, the guarantees of [37], [38] are task-agnostic and can scale inversely proportional to the number of agents even when RAG can still guarantee the near-optimal $1/2$ (see Fig. 6).



	(a)	(b)	(c)
[37]	$1/4$	$[1/4, 1/2]$	$1/2$
[38]	$1/4$	$[1/4, 1/3]$	$1/2$
RAG	1	$1/(1+\kappa_f) \geq 1/2$	$1/(1+\kappa_f) \geq 1/2$

Fig. 6. Comparison of Suboptimality Guarantees of [37], [38] and RAG: Example of Area Coverage. We make the comparison over an area coverage task with multiple drones, as in Fig. 5(a), where the drones need to each select a short trajectory to maximize the total covered area at the next time step. The stars are the drones' positions, the circles are their FOVs, and the red lines denote undirected communication links. The best performances for each metric are in blue. (a) Drones are fully decentralized (no communication). The bounds by [37], [38] are both $1/4$. For $|\mathcal{N}|$ drones, instead of 4, the bounds would be $O(1/|\mathcal{N}|)$. Instead, RAG guarantees $1/(1+\kappa_f) = 1$ since the drones are far enough for their FOV to overlap, $\text{coin}_{f,i} = 0, \forall i$, and $\kappa_f = 0$. (b) Drones are partially decentralized such that only the physically close drones communicate. For this communication graph, that we assume to be the same as the information-action access graph, $\omega = 2, \chi = 2$, and $\alpha^\star = 3$. Therefore, Gharesifard and Smith [37] give $[1/4, 1/2]$ and Grimsman et al. [38] give $[1/4, 1/3]$. RAG still gives $1/(1+\kappa_f) \geq 1/2$ since non-neighbors are far away. (c) In the fully centralized case, [37], [38] give $1/2$ and RAG gives $1/(1+\kappa_f)$. In all, RAG may provide tighter bounds than [37], [38] since it considers the actual function f using coin_f and κ_f, whereas the authors in [37] and [38] are agnostic to f. (a) fully decentralized. (b) partially decentralized. (c) fully centralized.

To discuss the suboptimality guarantees of [37], [38] in more detail, we present their decision-making rule. Specifically, the authors in [37] and [38] used the following distributed submodular maximization (DSM) rule, introduced in [37]

$$a_i^{\text{DSM}} \in \max_{a\in\mathcal{V}_i} f\left(a \mid \{a_j^{\text{DSM}}\}_{j\in\mathcal{N}_i^{\text{in}}}\right) \quad (19)$$

where $\mathcal{N}_i^{\text{in}} \subseteq [i-1]$. Equation (19) generalizes SG's rule in (7) to the setting where agent i has access only to the actions selected by the agents in $\mathcal{N}_i^{\text{in}} \subseteq [i-1]$, instead of all agents that have selected an action before agent i. The information-access structure prescribed by the rule in (19) can be represented as a DAG $\mathcal{G}_{\text{info}}$, where agent is neighbors in $\mathcal{G}_{\text{info}}$ are the set $\mathcal{N}_i^{\text{in}}$ of agents.[4] Due to the limited information access, the suboptimality guarantees of [37], [38] take the form presented in Table I, where $\omega(\mathcal{G}_{\text{info}})$ is the clique number of $\mathcal{G}_{\text{info}}$, $\chi(\mathcal{G}_{\text{info}})$ is the chromatic number, and $\alpha^\star(\mathcal{G}_{\text{info}})$ is the fractional independence number [55].

[3]Theorem 2 holds true for f nondecreasing and submodular and not necessarily second-order submodular.

[4]The graph $\mathcal{G}_{\text{info}}$ is in general different from the communication graph $\mathcal{G}$. When an agent i and an agent j do not communicate (they are not neighbors in the communication graph $\mathcal{G}$) but agent $j \in \mathcal{N}_i^{\text{in}}$, then agent js action needs to be relayed to agent i via other agents in $\mathcal{G}$ that form a connected communication path in $\mathcal{G}$ between agent i and agent j.

VI. Decision Time Analysis

We bound the time it takes for RAG to terminate. RAG's decision time scales linearly with the size of the network, up to two orders of the network size faster than the state of the art. We summarize the decision time of the state of the art and of RAG in Table I, where we use the notation:

1) τ_f is the time required for one evaluation of f;
2) τ_c is the time for transmitting an action through a communication channel $(i \to j) \in \mathcal{E}$;
3) $\tau_\#$ is the time for transmitting a real number through a communication channel $(i \to j) \in \mathcal{E}$; evidently, $\tau_\# \ll \tau_f$ and $\tau_\# \ll \tau_c$;
4) $\mathsf{diam}(\mathcal{G})$ is the diameter of a graph $\mathcal{G}$, i.e., the longest shortest path among any pair of nodes in $\mathcal{G}$ [56].

We base our analysis on the observation that the decision time of any distributed algorithm depends on the algorithm's.

1) *Computational complexity*, namely, the number of function evaluations required till termination (ignoring addition and multiplications as negligible in comparison).
2) *Communication complexity*, namely, the number of communication rounds needed till termination, accounting for the length of the communication messages per each round.

A. Decision Time of RAG

We first analyze the computational and communication complexities of RAG and then present its decision time.

Proposition 3 (Computational Complexity): RAG requires each agent i to perform at most $|\mathcal{V}_i||\mathcal{N}_i|$ function evaluations.

Proof: For each agent i, $|\mathcal{A}_i|$ increases by at least one with each "while loop" iteration of RAG. At each such iteration, agent i needs to perform $|\mathcal{V}_i|$ function evaluations to evaluate its marginal gain of all $v \in \mathcal{V}_i$ (lines 3–4). Since $|\mathcal{A}_i| \leq |\mathcal{N}_i|$, agent i will perform at most $|\mathcal{V}_i||\mathcal{N}_i|$ function evaluations. □

Proposition 4 (Communication Complexity): RAG requires at most $|\mathcal{N}| - 1$ communication rounds where a real number is transmitted, and at most $|\mathcal{N}| - 1$ communication rounds where an action is transmitted.

Proof: The number of "while loop" iterations of RAG is at most $|\mathcal{N}| - 1$ because at each iteration at least one agent will select an action. Besides, each "while loop" iteration includes two communication rounds: one for transmitting a marginal gain value (line 5), and one for transmitting an action (lines 8 and 11). Hence, Proposition 4 holds. □

Theorem 3 (Decision Time of: RAG) RAG terminates in at most $(\tau_c + \tau_\#)(|\mathcal{N}| - 1) + \tau_f \max_{i \in \mathcal{N}} (|\mathcal{V}_i||\mathcal{N}_i|)$ time.

Proof: Theorem 3 holds from Propositions 3 and 4. □

B. Comparison to the State of the Art

We summarize the decision times of the state of the art and RAG in Table I. RAG scales linearly with the number of robots, $|\mathcal{N}|$, whereas the state of the art scales at least quadratically with $|\mathcal{N}|$. RAG has computational time that is linear in $|\mathcal{N}_i|$, independent of $|\mathcal{N}|$, and communication time linear in $|\mathcal{N}|$.

In Table I, we assume for simplicity that $|\mathcal{V}_i| = |\mathcal{V}_j|, \forall i, j \in \mathcal{N}$. We divide the state of the art into algorithms that solve Problem 1 either in the continuous domain via employing the multilinear extension [54] of the set function f [30], [52], [53], or in the discrete domain [31], [37], [38], [44], [48]:[5,6,7]

a) Computation time: RAG requires $\tau_f |\mathcal{N}_i| |\mathcal{V}_i|$ computation time. The method in [31] requires computation time $\tau_f \sum_{i \in \mathcal{N}} |\mathcal{V}_i| = \tau_f |\mathcal{V}_i| |\mathcal{N}|$ since each agent i needs to perform $|\mathcal{V}_i|$ computations and the agents perform the computations sequentially. Given a prespecified information access prescribed by directed acyclic graph (DAG) $\mathcal{G}_{\text{info}}$, the methods in [37] and [38] also instruct the agents to select actions sequentially leading to a computation time at most $\tau_f |\mathcal{V}_i| |\mathcal{N}|$. This time excludes the time needed to find the $\mathcal{G}_{\text{info}}$ given an arbitrary communication graph $\mathcal{G}$. The method in [44] enables parallelized computation among agents by ignoring certain edges of an initially complete $\mathcal{G}$, resulting in a computation time of $O(\tau_f |\mathcal{V}_i| / \epsilon) \leq \tau_f |\mathcal{V}_i| |\mathcal{N}|$. Compared to [37], [38], the method in [44] also provides a distributed way to find which edges to ignore such that the suboptimality guarantee is optimized, a process that requires an additional computation time of $O(\tau_f |\mathcal{V}_i|^2 (|\mathcal{N}| - 1))$. The remaining algorithms require longer computation times, proportional to $|\mathcal{N}|^2$ or more.

b) Communication time: RAG requires at most $(\tau_c + \tau_\#)(|\mathcal{N}| - 1)$ communication time. In [37] and [38], the agents need to communicate over $\mathcal{G}$ per the prespecified information-access DAG $\mathcal{G}_{\text{info}}$ per the rule in (19) [37, Remark 3.2]. In Appendix D, we identify a worst case where this rule results in $O(\tau_c |\mathcal{N}|^2)$ communication time. This happens when each agent $i - 1$ and agent i in the decision sequence do not communicate directly, thus, the information of agent $i - 1$ needs to be relayed to agent i via other agents that form a connected communication path between the two. The method in [31], which introduces a DFS procedure to determine the best agents' ordering to run SG [11] over arbitrary (strongly) connected networks (instead of just line graphs), requires a worst-case communication time of $O(\tau_c |\mathcal{N}|^3)$ for directed networks, and $O(\tau_c |\mathcal{N}|^2)$ for undirected networks [28, Appendix II]. For some ϵ, the method in [44] may require less communication time for running the algorithm per se than other methods, but an additional communication time of $O(\tau_c |\mathcal{V}_i|)$ is needed to distributively find a DAG that optimizes the algorithm's approximation performance. The remaining algorithms require communication times proportional to $|\mathcal{N}|^2$ or more.

VII. Evaluation in Road Detection and Coverage

We evaluate RAG in the provided simulator across four scenarios of road detection and coverage. The scenarios span two team sizes (15 and 45 robots), and two communication rates (0.25 Mb/s), and 100 Mb/s)—up to three orders of magnitude

[5]The continuous-domain algorithms employ consensus-based techniques [30], [52], or algorithmic game theory [53], and need to compute the multilinear extension's gradients via sampling.

[6]The decision times of the continuous-domain algorithms depend on additional problem-dependent parameters (such as Lipschitz constants, the diameter of the domain set of the multilinear extension, and a bound on the gradient of the multilinear extension), which we make implicit in Table I via the O, Ω, and Θ notations.

[7]The computational and communication times reported for [53] are based on the numerical evaluations therein since a theoretical quantification is not included in [53] and appears nontrivial to derive one as a function of $\mathcal{N}$, ϵ, or the other problem parameters.

faster than competitive near-optimal submodular optimization algorithms—and, when the robots maintain a neighborhood size ≥ 2, superior mean coverage. The experiments also demonstrate RAG scales linearly with the network size, as predicted by our theoretical analysis: from the 15-robot to the 45-robot case, RAG scales linearly, being at most 3x slower compared to the 15-robot case. In contrast, the compared near-optimal algorithms scale cubically, being up to 60x slower compared to the 15-robot case.

Our code is available at https://github.com/UM-iRaL/Resource-Aware-Coordination-AirSim.

Common Simulation Setup across Simulated Scenarios: We first present the task of road detection (see Fig. 1), then the compared algorithms and the simulation pipeline (see Fig. 3).

a) Road detection and coverage task: Multiple aerial robots with onboard cameras are deployed in an unknown urban environment and tasked to maximize the total new road area detected after each action coordination step (see Fig. 1). The environment is unknown to the robots (no map is available a priori), necessitating the robots to use their onboard cameras to perform exploration for road detection.

The robots are deployed close to each other relative to the size of their FOVs and, as a result, their FOVs may often overlap (see Fig. 1). *Therefore, coordination becomes necessary for the robots to spread in the environment such that they detect different road segments and maximize the total new road area detected after each action coordination step.*

To perform the task, given the currently visible environment, the robots coordinate how to move per the collaborative autonomy pipeline in Fig. 2. Particularly, the task takes the form of the optimization problem in Problem 1 where f denotes the number of road pixels captured by all robots' collective FOV after they traverse their agreed trajectories $\{a_i\}_{i\in\mathcal{N}}$—$f$ is nondecreasing, submodular, and second-order submodular [44]—and $\mathcal{V}_i$ denotes robot is available trajectories at the current coordination round. $\mathcal{V}_i$ defines the possible directions that the robot can move in, and the speed the robot can move with. For simplicity, we assume that every robot can move in any of the 8 cardinal directions—N E S W NE SE NW SW—relative to its body frame, for 10 m at 3 m/s.

Without loss of generality, the deployed robots are assumed to have the same onboard sensing and communication capabilities: All robots are equipped with the following:

1) an IMU;
2) a GPS signal receiver;
3) a downward-facing camera mounted on a gimbal that enables the camera to point to any of the 8 cardinal directions relative to the robot's body frame;
4) a communication module—either a Digi XBee 3 Zigbee 3.0 (0.25 Mb/s) or Silvus Tech SL5200 (100 Mb/s)—for inter-robot communication.

Each robot can establish a few communication channels with robots within 100 m range, subject to bandwidth constraints.

b) Compared algorithms: Across various bandwidth constraints for the robots, we compare RAG with two competitive near-optimal algorithms: the SG algorithm [11], also known as coordinate descent [1], and its state-of-the-art depth-first-search variant (DFS-SG) [31]. The algorithms are commonly used for information gathering with multiple robots (e.g., see the papers [1], [15], [17] and the survey [5]. In more detail, the setup is as follows.

We test RAG for different bandwidth constraints varying from 0 up to 7.[8] In each case, the same bandwidth constraint applies to all robots. Each such version of RAG is denoted by RAG-$k$nn, where $k = 0, \ldots, 7$. For each k, the communication network is heuristically determined by having each robot select k nearest other robots as neighbors subject to the 100 m communication range. If fewer than k others are within the communication range, then all are selected as neighbors.

The SG algorithm requires the robots to be arranged on a line graph that defines the order in which the robots select actions per (7) and enables the information relay from robots that have already selected actions to the robot currently selecting an action. To ensure the existence of a line graph in our simulations of SG, we randomly generate one, adjusting the robots' communication ranges to infinity.

The DFS-SG algorithm enables SG to be applied to networks that are not necessarily a line graph, but the networks still need to be strongly connected. To this end, we construct strongly connected graphs by first randomly constructing line graphs as for SG, then randomly adding a few undirected edges to the line graphs, particularly, 30 edges for the 15-robot case and 90 edges for the 45-robot case. At each coordination step, DFS-SG randomly picks the first robot to select an action, and the order of all other robots is determined by a distributed method based on DFS. The resulting decision sequence may involve relay robots that transmit information between robots that are not directly connected and, thus, DFS-SG generally requires longer decision times than SG.

c) Simulation pipeline: The simulation pipeline consists of the following modules (see Fig. 3).

1) *Network self-configuration:* This module applies only to RAG since only RAG enables the network to self-configure itself across the planning steps subject to the robots' bandwidth constraints and the robots' relative locations. Particularly, as described also in the above paragraph for RAG-$k$nn, at the beginning of each planning step, each robot selects the k nearest other robots within its communication range. This neighborhood selection scheme is justified by Example 1.
2) *Perception:* The process is different for each algorithm since each algorithm requires the robots to process information received from different sets of robots. For RAG, the robots need the following three operations to detect the nearby environment and evaluate where would be best to move.

[8] Although neighborhood sizes larger than 7 are possible in the simulator, we limit them to at most size 7 to achieve a reasonable duration for completing each Monte Carlo trial. Particularly, the neighborhoods' scalability to $k > 7$ is constrained by the msgpack-rpc protocol implementation over TCP/IP via rpclib. The current use of a single TCP port for multidrone API calls creates a communication bottleneck, increasing client response times. For example, our experiments with 45 robots, which include 30 trials that simulate the robots' evolution for 500 s for 9 different algorithms (RAG k-nn for $k = 1, \ldots, 7$, SG, and DFS-SG), require 3 days for each algorithm to be tested.

i) At the beginning of each planning step, each robot hovers at a constant height (30 m) and rotates its camera to take a picture in each of the 8 cardinal directions relative to its body frame. The captured FOV in each direction is of size $34.6 \times 26.0\,\mathrm{m}^2$. Then, the robot uses semantic segmentation[9] to detect the road segments in each of the 8 captured images. The resulting segmented image has size 25 KB with label information after compression.

ii) Each robot, upon receiving segmented images and relative poses from neighbors that have committed actions in specific directions, stitches these images together. That way, the robot reconstructs the collective FOV of its neighbors that have committed an action. This reconstruction will be used next by the robot to select in which direction to move to cover the most new road area. To this end.

iii) The robot stitches its 8 captured images, respectively, with the previous reconstruction and counts the extra road pixels that can be covered in each of the corresponding 8 cardinal directions.

For SG and DFS-SG, the operations above are similar, with the modification being that instead of just the neighbors' segmented and stitched images, each robot leverages all previous robots' stitched images in (ii) and (iii).

3) *Action coordination:* The process differs across algorithms: For RAG, given their neighbors' already committed actions, the robots that are still deciding will first each pick a trajectory whose FOV gives the most amount of newly covered road area. Then, they will compare this amount with one another. Those who win their neighbors will commit to their picked trajectories and share the corresponding FOV and camera pose with neighbors. Otherwise, they will receive the FOVs and camera poses selected by newly committed robots and repeat the process above. For SG, the robots make decisions sequentially in a line graph. Each robot, upon receiving the stitched FOVs of the trajectories selected by all predecessors, will first choose the trajectory whose FOV offers the most newly covered area based on previous robots' selections. It will then align and stitch this FOV with the received ones, and finally send the newly stitched FOVs to the next robot. The coordination process for DFS-SG is similar to SG, with the difference being that DFS-SG operates over a strongly connected network, given the robots' locations, bandwidth, and communication range, instead of a line graph. Thus, during DFS-SG, the ith robot in the decision sequence may need to transmit FOVs to the $(i+1)$th robot in the decision sequence via relay robots.

4) *Control:* The robots simultaneously traverse the trajectories after coordination using "simple_flight," a flight controller in AirSim. "simple_flight" uses cascade PID controllers to drive the aerial robots to move in the selected direction with speed 3.0 m/s for 10.0 m.

[9]We simulate semantic segmentation using the in-built ground truth result of AirSim. The process runs at 20 Hz, ensuring segmentation does not create a bottleneck for the replanning frequency of RAG.

A. Evaluations With 15 Robots

We present the results for the 15-robot scenario. Each compared algorithm was tested in 50 trials, each lasting 300 s. The simulation setup is as follows.

Simulation setup: Across the Monte Carlo tests, the robots are initialized near one another such that their FOVs largely overlap (e.g., see the leftmost group of 15 robots in Fig. 1). Therefore, coordination becomes necessary for the robots to spread in the environment such that they detect different road segments and, thus, maximize the total new road area detected after each action coordination step.

We ran the simulations on a 32-core CPU with 2 Nvidia RTX 4090 24 GB GPUs and 128 GB RAM on Ubuntu 18.04.

Results: The results are summarized in Figs. 7 (100 Mb/s) and 8 (0.25 Mb/s). RAG-$k$nn achieves real-time planning for all k, and superior mean coverage for $k \geq 2$.

1) Road Coverage Over Time: Figs. 7(a) and 8(a) present the total area of the road covered over time for each algorithm. RAG-$k$nn rapidly achieves superior mean performance for $k \geq 2$, and maintains comparable performance for $k = 1$. The lower performance for $k = 0$ is expected, since then no robot ever coordinates with any other.

2) Tradeoff Between Decision Time Versus Road Coverage: Figs. 7(b) and 8(b) present the average action coordination time versus the peak coverage of the road area over the duration of the task [the peak values in Figs. 7(a) and 8(a)]. For larger k, the marginal increase in total coverage is negatively outweighed by a greater increase in decision time. The observation demonstrates the tradeoff between decision speed versus optimality predicted in Section V (Remark 5).

B. Evaluations With 45 Drones

We demonstrate the scalability of our algorithm from 15 to 45 robots. Each compared algorithm was tested in 30 trials, each lasting 500 s. The experiments demonstrate that RAG scales linearly, being at most 3 times slower compared to the 15-robot case. In contrast, the compared near-optimal algorithms scale cubically, as predicted by our theoretical analysis, being $> 27 = 3^3$ times slower compared to the 15-robot case: e.g., SG's decision time increases from around 2 to 60 s (100 Mb/s case), a 30x increase, and from around 14 to 850 s (0.25 Mb/s case), a 60x increase.

Simulation setup: Across the Monte Carlo tests, the robots are divided into three groups, with the robots within each group being deployed near to one another such that their FOVs largely overlap (see Fig. 1). In such a setting, RAG automatically enables parallelized decision-making, achieving scalability and similar coverage as for the 15-robot setting, in contrast to the state-of-the-art near-optimal algorithms.

We ran the simulations on a remote server with 80 CPU cores (4x 2.4 GHz Intel Xeon Gold 6148), 360 GB RAM, and 4 NVIDIA Tesla V100 16 GB GPU.

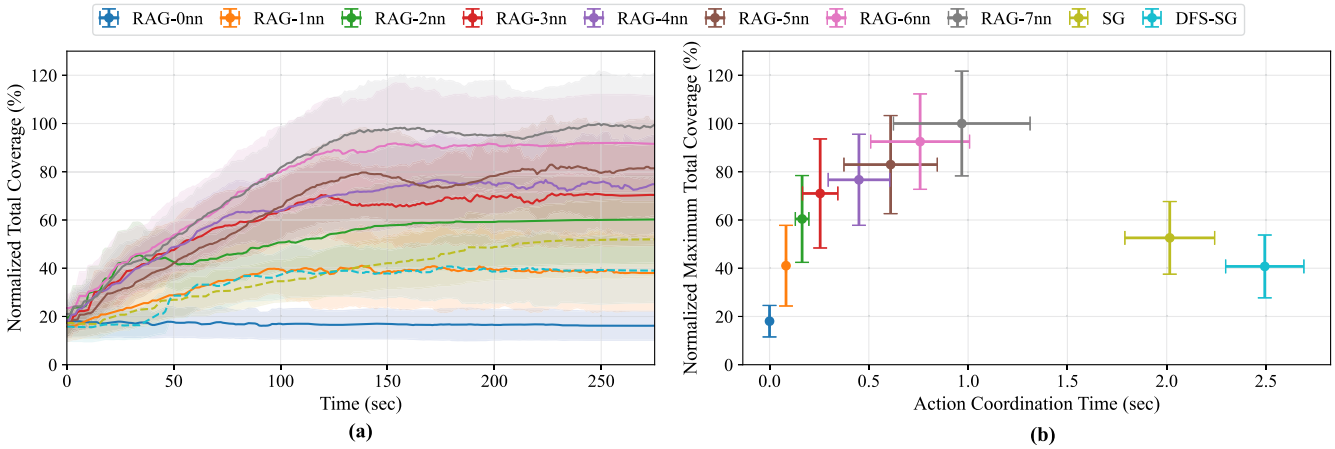


Fig. 7. Comparative Analysis of Coverage Performance with 15 robots for 100 Mb/s data rate. The reported times include ROS1 delays of up to 0.18 s per action coordination step. Particularly, in Figs. 7–10, all coverage data are normalized by the mean results of the corresponding RAG-7nn instances, and the shared area and cross-hairs represent 1 standard deviation for y-axis in (a) and both axes in (b). (a) Total road-area coverage versus time. (b) Maximum total coverage versus action coordination time.

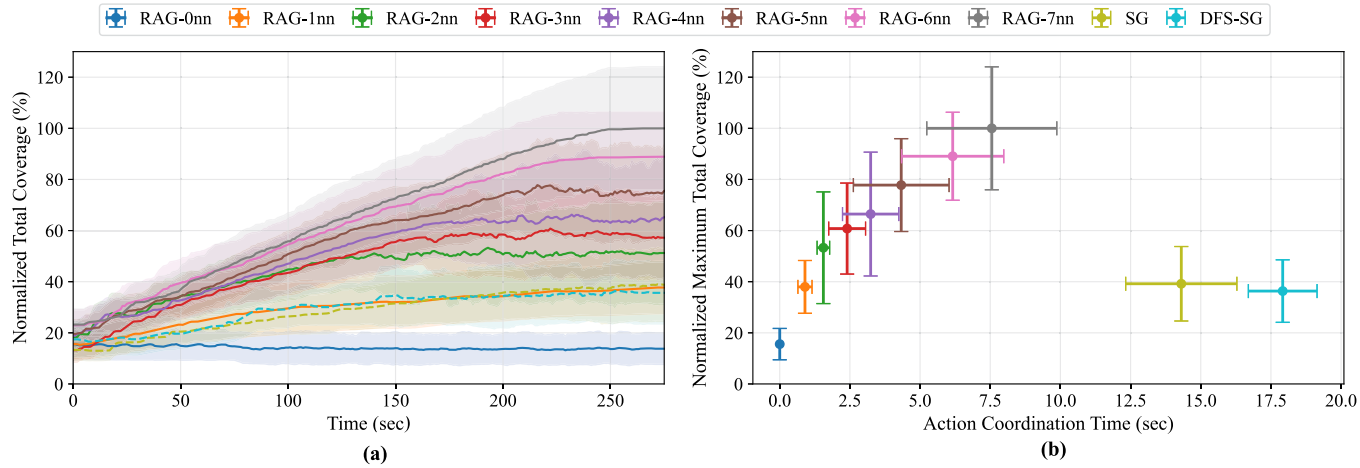


Fig. 8. Comparative Analysis of Coverage Performance with 15 robots for 0.25 Mb/s data rate. The reported times include ROS1 delays of up to 0.81 s per action coordination step. (a) Total road-area coverage versus time. (b) Maximum total coverage versus action coordination time.

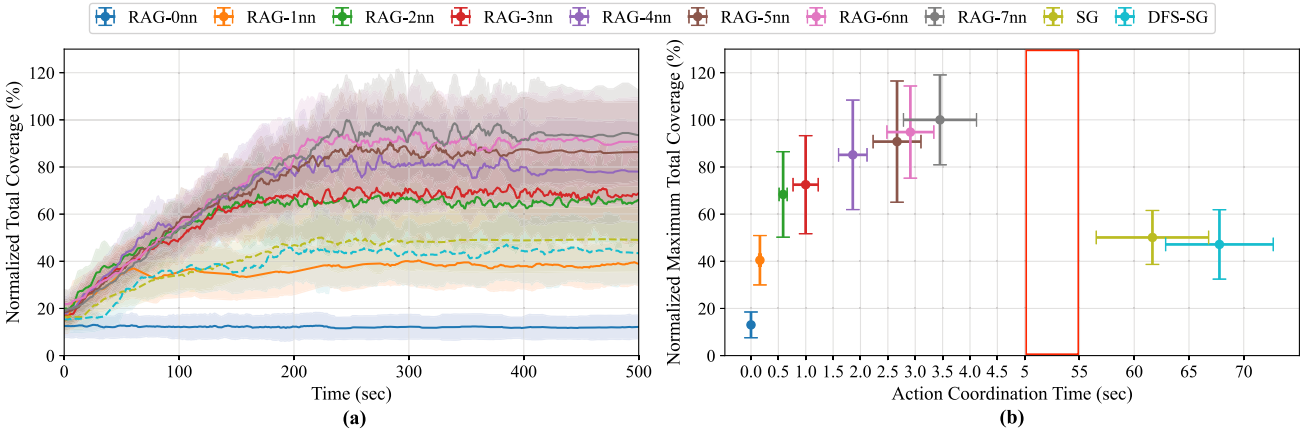


Fig. 9. Comparative analysis of coverage performance with 45 robots for 100 Mb/s data rate. The reported times include ROS1 delays of up to 3.78 s per action coordination step.

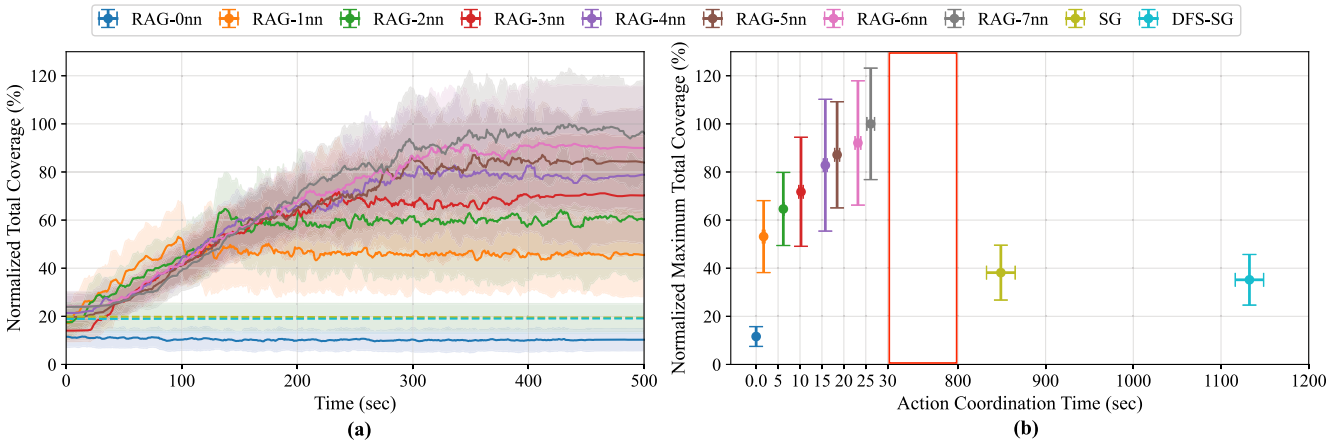


Fig. 10. Comparative analysis of coverage performance with 45 robots for 0.25 Mb/s data rate. The reported times include ROS1 delays of up to 6.54 s per action coordination step.

Results: The results are summarized in Fig. 9 (100 Mb/s) and 10 (0.25 Mb/s). RAG maintains superior mean coverage, with all qualitative observations from the 15-robot case applying here. Per Figs. 9(b) and 10(b), RAG's coordination times scaled linearly compared to the 15-robot case. In contrast, SG's and DFS-SG's coordination times scaled super-cubically, requiring now minutes per action coordination step at 100 Mb/s and tens of minutes at 0.25 Mb/s. The results demonstrate the linear scalability of RAG and cubic scalability of the state of the art, as predicted by our theoretical analysis (see Table I).

VIII. Conclusion

We introduced a distributed submodular optimization paradigm, RAG that enables scalable and near-optimal coordination over robot mesh networks. The framework applies to any distributed submodular optimization task. In this article, we applied it to active information-gathering, demonstrating RAG's performance in simulated scenarios of road detection with up to 45 robots. In the simulations, RAG enabled real-time planning, up to 3 orders of magnitude faster than competitive near-optimal algorithms, while also achieving superior mean coverage performance. To enable the simulations, we extended the high-fidelity and photo-realistic AirSim by enabling a scalable collaborative autonomy pipeline to tens of robots while simulating realistic r2r communication messages and speeds.

In our future work, we will apply RAG to distributed metric-semantic SLAM where the r2r communication messages scale to MBs [8], [57], 40 times larger than the messages of 25 KB, we considered for the road detection task in this article.

We also plan the following algorithmic contributions.

1) RAG assumes synchronous communication. Although RAG can be modified to handle such cases, e.g., by instructing each robot to execute its action without first waiting for all other robots to select actions, its near-optimality guarantees provided in this article may no longer hold. Our future work will extend our theoretical and algorithmic analysis beyond the above limitations.
2) We will also extend our results to handle effective task execution over long time horizons. For example, in collaborative mapping over long time horizons, the team needs to stay updated on the areas that have been mapped, such that the current plans do not repeat past actions. This is in addition to the focus of this paper that only the current plans among the robots do not overlap. To this end, we will handle network connectivity constraints.
3) Finally, we will enhance our simulator by integrating the simulation of realistic communication channels and protocols toward communication-aware and -efficient coordination algorithms.

Appendix A

Proof of Proposition 1: Consider robot $i \in \mathcal{N}$ and two disjoint robot sets $\mathcal{B}_1, \mathcal{B}_2 \subseteq \mathcal{N} \setminus \{i\}$, we have

$$\begin{aligned}&\mathsf{coin}_{f,i}(\mathcal{B}_1 \cup \mathcal{B}_2) - \mathsf{coin}_{f,i}(\mathcal{B}_1)\\ &= f(a_i \,|\, \{a_j\}_{j \in \mathcal{B}_1 \cup \mathcal{B}_2}) - f(a_i \,|\, \{a_j\}_{j \in \mathcal{B}_1}) \leq 0 \end{aligned} \tag{20}$$

where the inequality holds since f is submodular. Hence, $\mathsf{coin}_{f,i}(\mathcal{N}_i)$ is nonincreasing in $\mathcal{N}_i$. Therefore, for any $i \in \mathcal{N}$, $\mathsf{coin}_{f,i}(\mathcal{N}_i)$ achieves the lower bound $\mathsf{coin}_{f,i}(\mathcal{N}_i) \geq \mathsf{coin}_{f,i}(\mathcal{N} \setminus \{i\}) = f(a_i) - f(a_i \,|\, \emptyset) = 0$. For the upper bound

$$\mathsf{coin}_{f,i}(\mathcal{N}_i) \leq \mathsf{coin}_{f,i}(\emptyset) = f(a_i) - f(a_i \,|\, \mathcal{A}_{\mathcal{N}\setminus\{i\}}) \tag{21}$$

$$\leq \kappa_f f(a_i) \tag{22}$$

where the first inequality holds since $\mathsf{coin}_{f,i}$ is nonincreasing, and the second inequality holds from (8). □

Appendix B

We first prove Theorem 1, and then present and prove the bounds of RAG when f is submodular or approximately submodular instead of second-order submodular.

A. Proof of Theorem 1

We index each agent in $\mathcal{N}$ per its selecting order in RAG, i.e., agent $i \in \mathcal{N} \triangleq \{1, \dots, |\mathcal{N}|\}$ is the ith agent to select an action during the execution of RAG. If multiple agents select actions simultaneously, then we index them randomly. We use also the notation.

1) $\mathcal{A}_{\mathcal{X}}^{\mathsf{RAG}} \triangleq \{a_i^{\mathsf{RAG}}\}_{i \in \mathcal{X}}$ for any $\mathcal{X} \subseteq \mathcal{N}$, i.e., $\mathcal{A}_{\mathcal{X}}^{\mathsf{RAG}}$ is the set of actions selected by the agents in $\mathcal{X}$.

Then, we have

$$f(\mathcal{A}^{\mathsf{OPT}})$$

$$= f(\mathcal{A}^{\mathsf{OPT}} \cup \mathcal{A}^{\mathsf{RAG}}) - \sum_{i \in \mathcal{N}} f(a_i^{\mathsf{RAG}} \,|\, \mathcal{A}^{\mathsf{OPT}} \cup \mathcal{A}_{[i-1]}^{\mathsf{RAG}}) \tag{23}$$

$$\leq f(\mathcal{A}^{\mathsf{RAG}}) + \sum_{i \in \mathcal{N}} f(a_i^{\mathsf{OPT}} \,|\, \mathcal{A}_{\mathcal{N}_i \cap [i-1]}^{\mathsf{RAG}}) - (1-\kappa_f) \sum_{i \in \mathcal{N}} f(a_i^{\mathsf{RAG}} \,|\, \mathcal{A}_{\mathcal{N}_i \cap [i-1]}^{\mathsf{RAG}}) \tag{24}$$

$$\leq f(\mathcal{A}^{\mathsf{RAG}}) + \kappa_f \sum_{i \in \mathcal{N}} f(a_i^{\mathsf{RAG}} \,|\, \mathcal{A}_{\mathcal{N}_i \cap [i-1]}^{\mathsf{RAG}}) \tag{25}$$

$$= (1+\kappa_f) f(\mathcal{A}^{\mathsf{RAG}}) + \kappa_f \sum_{i \in \mathcal{N}} \left[f(a_i^{\mathsf{RAG}} \,|\, \mathcal{A}_{\mathcal{N}_i \cap [i-1]}^{\mathsf{RAG}}) - f(a_i^{\mathsf{RAG}} \,|\, \mathcal{A}_{[i-1]}^{\mathsf{RAG}}) \right] \tag{26}$$

$$\leq (1+\kappa_f) f(\mathcal{A}^{\mathsf{RAG}}) + \kappa_f \sum_{i \in \mathcal{N}} \left[f(a_i^{\mathsf{RAG}}) - f(a_i^{\mathsf{RAG}} \,|\, \mathcal{A}_{[i-1] \setminus \mathcal{N}_i}^{\mathsf{RAG}}) \right] \tag{27}$$

$$\leq (1+\kappa_f) f(\mathcal{A}^{\mathsf{RAG}}) + \kappa_f \sum_{i \in \mathcal{N}} \underbrace{\left[f(a_i^{\mathsf{RAG}}) - f(a_i^{\mathsf{RAG}} \,|\, \mathcal{A}_{\mathcal{N}_i^c}^{\mathsf{RAG}}) \right]}_{\mathsf{coin}_{f,i}(\mathcal{N}_i)} \tag{28}$$

where (23) holds from telescoping the sums; (24) holds from f being submodular and

$$1 - \kappa_f \leq \frac{f(a_i^{\mathsf{RAG}} \,|\, \mathcal{A}_{\mathcal{N} \setminus \{i\}}^{\mathsf{RAG}})}{f(a_i^{\mathsf{RAG}})} \leq \frac{f(a_i^{\mathsf{RAG}} \,|\, \mathcal{A}^{\mathsf{OPT}} \cup \mathcal{A}_{[i-1]}^{\mathsf{RAG}})}{f(a_i^{\mathsf{RAG}} \,|\, \mathcal{A}_{\mathcal{N}_i \cap [i-1]}^{\mathsf{RAG}})} \tag{29}$$

per the definition of κ_f; (25) holds since RAG selects a_i^{RAG} greedily; (26) holds from telescoping the sums; (27) holds from f being second-order submodular; and (28) holds from f being submodular. Therefore, (12) holds.

Then, in the fully centralized case where $\mathcal{N}_i^c = \emptyset, \forall i \in \mathcal{N}$, we have $\mathsf{coin}_{f,i}(\mathcal{N}_i) = f(a_i^{\mathsf{RAG}}) - f(a_i^{\mathsf{RAG}}) = 0$. Hence, (13) follows (28).

Finally, in the fully decentralized case where $\mathcal{N}_i^c = \mathcal{N} \setminus \{i\}, \forall i \in \mathcal{N}$, we have

$$\sum_{i \in \mathcal{N}} \mathsf{coin}_{f,i}(\mathcal{N}_i) = \sum_{i \in \mathcal{N}} f(a_i^{\mathsf{RAG}}) - f(a_i^{\mathsf{RAG}} \,|\, \mathcal{A}_{\mathcal{N} \setminus \{i\}}^{\mathsf{RAG}})$$

$$\leq \kappa_f \sum_{i \in \mathcal{N}} f(a_i^{\mathsf{RAG}}) \tag{30}$$

$$\leq \frac{\kappa_f}{1-\kappa_f} f(\mathcal{A}^{\mathsf{RAG}}) \tag{31}$$

where (30) holds from (8), and (31) holds from [58, Lemma 2.1]. Combining (28) and (31), the lower bound in (15) can be proved. □

B. Suboptimality bounds of RAG for submodular or approximately submodular f

We present the bounds in Corollary 4. To this end, we use the following definition.

Definition 5 (Total curvature [9], [10]): Consider $f : 2^{\mathcal{V}} \mapsto \mathbb{R}$ is nondecreasing. Then, fs total curvature is defined as

$$c_f \triangleq 1 - \min_{v \in \mathcal{V}} \min_{\mathcal{A}, \mathcal{B} \subseteq \mathcal{V} \setminus \{v\}} \frac{f(\{v\} \cup \mathcal{A}) - f(\mathcal{A})}{f(\{v\} \cup \mathcal{B}) - f(\mathcal{B})}. \tag{32}$$

Similarly to κ_f, we have $c_f \in [0,1]$. When f is submodular, then $c_f = \kappa_f$. Generally, if $c_f = 0$, then f is modular, while if $c_f = 1$, then (32) implies the assumption that f is nondecreasing. In [59], any monotone f with total curvature c_f is called c_f-submodular, as repeated as follows.[10]

Theorem 4 (Suboptimality Bounds for Submodular and Approximately Submodular Functions): If f is not second-order submodular, $\mathcal{A}^{\mathsf{RAG}}$ enjoys the following approximation bounds:

1) if f is nondecreasing submodular, then

$$\frac{f(\mathcal{A}^{\mathsf{RAG}})}{f(\mathcal{A}^{\mathsf{OPT}})} \geq \begin{cases} \frac{1}{1+\kappa_f}, & \mathcal{G} \text{ is fully centralized} \\ 1-\kappa_f, & \mathcal{G} \text{ is not fully centralized} \end{cases} \tag{33}$$

2) if f is nondecreasing c_f-submodular, then

$$\frac{f(\mathcal{A}^{\mathsf{RAG}})}{f(\mathcal{A}^{\mathsf{OPT}})} \geq \begin{cases} \frac{1-c_f}{1+c_f-c_f^2}, & \mathcal{G} \text{ is fully centralized} \\ (1-c_f)^2, & \mathcal{G} \text{ is not fully centralized.} \end{cases} \tag{34}$$

C. Proof of Corollary 4

We present the proof separately for each case. First, when f is submodular with $\mathcal{G}$ being fully centralized, RAG has the same $1/(1+\kappa_f)$ bound as in Theorem 1, following from (26).

When f is submodular, and $\mathcal{G}$ is not fully centralized, RAG provides the same bound as the lower bound in (15)

$$f(\mathcal{A}^{\mathsf{OPT}}) \leq f(\mathcal{A}^{\mathsf{RAG}}) + \kappa_f \sum_{i \in \mathcal{N}} f(a_i^{\mathsf{RAG}} \,|\, \mathcal{A}_{\mathcal{N}_i \cap [i-1]}^{\mathsf{RAG}})$$

$$\leq f(\mathcal{A}^{\mathsf{RAG}}) + \frac{\kappa_f}{1-\kappa_f} \sum_{i \in \mathcal{N}} f(a_i^{\mathsf{RAG}} \,|\, \mathcal{A}_{[i-1]}^{\mathsf{RAG}}) \tag{35}$$

$$= \left(1 + \frac{\kappa_f}{1-\kappa_f}\right) f(\mathcal{A}^{\mathsf{RAG}}) \tag{36}$$

where (35) holds from (25) and the definition of κ_f.

When f is c_f-submodular, and $\mathcal{G}$ is fully centralized

$$f(\mathcal{A}^{\mathsf{OPT}})$$

[10] Lehmann et al. [59] defined c_f-submodularity by considering in (32) $\mathcal{A} \subseteq \mathcal{B}$ instead of $\mathcal{A} \subseteq \mathcal{V}$. Generally, nonsubmodular but monotone functions have been referred to as *approximately* or *weakly* submodular [60], [61], names that have also been adopted for the definition of c_f in [59], e.g., in [62] and [63].

$$= f(\mathcal{A}^{\mathsf{OPT}} \cup \mathcal{A}^{\mathsf{RAG}}) - \sum_{i\in\mathcal{N}} f(a_i^{\mathsf{RAG}} \,|\, \mathcal{A}^{\mathsf{OPT}} \cup \mathcal{A}^{\mathsf{RAG}}_{[i-1]})$$

$$\leq f(\mathcal{A}^{\mathsf{RAG}}) + \frac{1}{1-c_f} \sum_{i\in\mathcal{N}} f(a_i^{\mathsf{OPT}} \,|\, \mathcal{A}^{\mathsf{RAG}}_{\mathcal{N}_i\cap[i-1]}) - (1-c_f) \sum_{i\in\mathcal{N}} f(a_i^{\mathsf{RAG}} \,|\, \mathcal{A}^{\mathsf{RAG}}_{\mathcal{N}_i\cap[i-1]}) \tag{37}$$

$$\leq f(\mathcal{A}^{\mathsf{RAG}}) + \left[\frac{1}{1-c_f} - (1-c_f)\right] \sum_{i\in\mathcal{N}} f(a_i^{\mathsf{RAG}} \,|\, \mathcal{A}^{\mathsf{RAG}}_{\mathcal{N}_i\cap[i-1]}) \tag{38}$$

$$= \left(c_f + \frac{1}{1-c_f}\right) f(\mathcal{A}^{\mathsf{RAG}}) \tag{39}$$

where (37) holds from (23) and Definition 5, (38) holds since RAG selects a_i^{RAG} greedily, and (39) holds since $\mathcal{N}_i \cap [i-1] = [i-1]$ for $\mathcal{G}$ being fully centralized.

When f is c_f-submodular, and $\mathcal{G}$ is not fully centralized

$$f(\mathcal{A}^{\mathsf{OPT}})$$

$$\leq f(\mathcal{A}^{\mathsf{RAG}}) + \left[\frac{1}{1-c_f} - (1-c_f)\right] \sum_{i\in\mathcal{N}} f(a_i^{\mathsf{RAG}} \,|\, \mathcal{A}^{\mathsf{RAG}}_{\mathcal{N}_i\cap[i-1]})$$

$$\leq f(\mathcal{A}^{\mathsf{RAG}}) + \left[\frac{1}{(1-c_f)^2} - 1\right] \sum_{i\in\mathcal{N}} f(a_i^{\mathsf{RAG}} \,|\, \mathcal{A}^{\mathsf{RAG}}_{[i-1]}) \tag{40}$$

$$= \frac{1}{(1-c_f)^2} f(\mathcal{A}^{\mathsf{RAG}}) \tag{41}$$

where (40) holds from (38) and Definition 5. □

Appendix C

We provide the proofs regarding the a posteriori suboptimality bound of RAG.

A. Proof of Theorem 2

Since $\mathcal{I}_i = \mathcal{N}_i \cap [i-1]$, (18) follows (26), and thus, Theorem 2 is proved. Notice that f only needs to be submodular instead of second-order submodular for Theorem 2 to hold. □

B. Proof of Theorem 2

Let $\delta_i(\mathcal{I}_i) \triangleq f(a_i^{(\mathcal{I}_i)} \,|\, \mathcal{A}_{\mathcal{I}_i})$, where $\mathcal{A}_{\mathcal{I}_i}$ is the actions selected by $\mathcal{I}_i$ per RAG, and $a_i^{(\mathcal{I}_i)}$ is the action selected by i per RAG, i.e., greedily. Therefore, proving $\delta_i(\mathcal{I}_i)$ is nonincreasing and approximately supermodular in $\mathcal{I}_i, \forall i \in \mathcal{N}$, will be sufficient in proving Theorem 2.

We start with the nonincreasing property by proving δ_i is nonincreasing. For disjoint sets $\mathcal{B}_1, \mathcal{B}_2 \subseteq \mathcal{N} \setminus \{i\}$, we have $\mathcal{A}_{\mathcal{B}_1} \subseteq \mathcal{A}_{\mathcal{B}_1\cup\mathcal{B}_2}$ and, thus

$$\delta_i(\mathcal{B}_1) = f(a_i^{(\mathcal{B}_1)} \,|\, \mathcal{A}_{\mathcal{B}_1}) \geq f(a_i^{(\mathcal{B}_1\cup\mathcal{B}_2)} \,|\, \mathcal{A}_{\mathcal{B}_1}) \tag{42}$$

$$\geq f(a_i^{(\mathcal{B}_1\cup\mathcal{B}_2)} \,|\, \mathcal{A}_{\mathcal{B}_1\cup\mathcal{B}_2}) = \delta_i(\mathcal{B}_1 \cup \mathcal{B}_2) \tag{43}$$

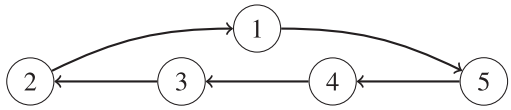


Fig. 11. Example of a directed communication graph with the worst-case agent order where each message needs to traverse $|\mathcal{N}| - 2$ edges.

where (42) holds since RAG selects $a_i^{(\mathcal{B}_1)}$ greedily given $\mathcal{A}_{\mathcal{B}_1}$, and (43) holds since $\mathcal{A}_{\mathcal{B}_1} \subseteq \mathcal{A}_{\mathcal{B}_1\cup\mathcal{B}_2}$. To prove the approximate supermodularity of δ_i, we will first prove another function δ_i' is supermodular, then show that $\delta_i(\mathcal{S}) \leq \delta_i'(\mathcal{S}) \leq \delta_i(\mathcal{S}) + \epsilon, \forall \mathcal{S} \subseteq \mathcal{N} \setminus \{i\}$ [64]. In particular, let us define $\delta_i'(\mathcal{I}_i) \triangleq f(a \,|\, \mathcal{A}_{\mathcal{I}_i})$, where $\mathcal{A}_{\mathcal{I}_i}$ is the actions selected per RAG by $\mathcal{I}_i$ as in the definition of δ_i, but $a \in \mathcal{V}_i$ is an arbitrary fixed action. Consider robot set $\mathcal{S} \subseteq \mathcal{N} \setminus \{i\}$ other than $\mathcal{B}_1, \mathcal{B}_2$, then

$$\delta_i'(\mathcal{S} \,|\, \mathcal{B}_1) - \delta_i'(\mathcal{S} \,|\, \mathcal{B}_1 \cup \mathcal{B}_2)$$

$$= \delta_i'(\mathcal{S} \cup \mathcal{B}_1) - \delta_i'(\mathcal{B}_1) - \delta_i'(\mathcal{S} \cup \mathcal{B}_1 \cup \mathcal{B}_2) + \delta_i'(\mathcal{B}_1 \cup \mathcal{B}_2)$$

$$= f(a \,|\, \mathcal{A}_{\mathcal{S}\cup\mathcal{B}_1}) - f(a \,|\, \mathcal{A}_{\mathcal{B}_1}) - f(a \,|\, \mathcal{A}_{\mathcal{S}\cup\mathcal{B}_1\cup\mathcal{B}_2}) + f(a \,|\, \mathcal{A}_{\mathcal{B}_1\cup\mathcal{B}_2}) \leq 0 \tag{44}$$

where the inequality holds since f is second-order submodular. Hence, δ_i' is supermodular. Then, we have

$$\delta_i(\mathcal{S}) - \delta_i'(\mathcal{S}) = f(a_i^{(\mathcal{S})} \,|\, \mathcal{A}_{\mathcal{S}}) - f(a \,|\, \mathcal{A}_{\mathcal{S}}) \leq f(a_i^{(\mathcal{S})}) - (1-\kappa_f) f(a) \tag{45}$$

which holds from the submodularity of f and (8)

$$1 - \kappa_f \leq \frac{f(a \,|\, \mathcal{A}_{\mathcal{N}\setminus\{i\}})}{f(a)} \leq \frac{f(a \,|\, \mathcal{A}_{\mathcal{S}})}{f(a)}. \tag{46}$$

Also, $\delta_i(\mathcal{S}) - \delta_i'(\mathcal{S}) \geq f(a_i^{(\mathcal{S})} \,|\, \mathcal{A}_{\mathcal{S}}) - f(a_i^{(\mathcal{S})} \,|\, \mathcal{A}_{\mathcal{S}}) = 0$ since RAG selects $a_i^{(\mathcal{S})}$ greedily. All in all, $\delta_i'(\mathcal{S}) \leq \delta_i(\mathcal{S}) \leq \delta_i'(\mathcal{S}) + \epsilon$ holds true with

$$\epsilon = \min_{a_2\in\mathcal{V}_i} \max_{a_1\in\mathcal{V}_i} [f(a_1) - (1-\kappa_f) f(a_2)] \tag{47}$$

that is, δ_i is approximately supermodular. □

Appendix D

We provide the proof of the communication time of the algorithm in [37] and [38]: the worst-case communication time of the two methods occurs when, for example, if the informational DAG $\mathcal{G}'$ is complete, i.e., each agent i requires information from $[i-1]$, then

1) for undirected graphs $\mathcal{G}$, e.g., when agent 1 locates in the center of a line graph and the rest are ordered alternately extending outward from the center to both ends of the line (e.g., $5 \leftrightarrow 3 \leftrightarrow 1 \leftrightarrow 2 \leftrightarrow 4 \leftrightarrow 6$), which leads to $\sum_{i=1}^{|\mathcal{N}|-1} \tau_c \times i = 1/2\, \tau_c\, |\mathcal{N}|\, (|\mathcal{N}| - 1)$;
2) for directed graphs $\mathcal{G}$, e.g., when $\mathcal{G}$ is a one-directional acyclic graph yet the agents' order increases in the other direction (see Fig. 11), and every agent i needs to traverse all other agents to send information to $i+1$, which leads to $\sum_{i=1}^{|\mathcal{N}|-1} \tau_c \times (|\mathcal{N}| - 2) = \tau_c\, (|\mathcal{N}| - 1)(|\mathcal{N}| - 2)$.

Therefore, the worst-case communication time is $O(\tau_c |\mathcal{N}|^2)$ for both undirected and directed $\mathcal{G}$.

REFERENCES

[1] N. Atanasov, J. Le Ny, K. Daniilidis, and G. J. Pappas, "Decentralized active information acquisition: Theory and application to multi-robot SLAM," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2015, pp. 4775–4782.

[2] P. Dames, P. Tokekar, and V. Kumar, "Detecting, localizing, and tracking an unknown number of moving targets using a team of mobile robots," *Int. J. Robot. Res.*, vol. 36, no. 13/14, pp. 1540–1553, 2017.

[3] M. Corah and N. Michael, "Scalable distributed planning for multi-robot, multi-target tracking," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 437–444.

[4] U. Feige, "A threshold of $ln(n)$ for approximating set cover," *J. ACM*, vol. 45, no. 4, pp. 634–652, 1998.

[5] Y. Sung et al., "A survey of decision-theoretic approaches for robotic environmental monitoring," *Foundations Trends Robot.*, vol. 11, no. 4, pp. 225–315, 2023.

[6] J. Bilmes, "Submodularity in machine learning and artificial intelligence," 2022, *arXiv:2202.00132*.

[7] A. Clark, B. Alomair, L. Bushnell, and R. Poovendran, *Submodularity in Dynamics and Control of Networked Systems*. Berlin, Germany: Springer, 2016.

[8] X. Liu et al., "Slideslam: Sparse, lightweight, decentralized metric-semantic SLAM for multi-robot navigation," 2024, *arXiv:2406.17249*.

[9] M. Sviridenko, J. Vondrák, and J. Ward, "Optimal approximation for submodular and supermodular optimization with bounded curvature," 2013, *arXiv:1311.4728*.

[10] M. Sviridenko, J. Vondrák, and J. Ward, "Optimal approximation for submodular and supermodular optimization with bounded curvature," *Math. Operations Res.*, vol. 42, no. 4, pp. 1197–1218, 2017.

[11] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey, "An analysis of approximations for maximizing submodular set functions—II," in *Polyhedral Combinatorics*. Heidelberg, Berlin: Springer, 1978, pp. 73–87.

[12] A. Krause, A. Singh, and C. Guestrin, "Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies," *J. Mach. Learn. Res.*, vol. 9, pp. 235–284, 2008.

[13] A. Singh, A. Krause, C. Guestrin, and W. J. Kaiser, "Efficient informative sensing using multiple robots," *J. Artif. Intell. Res.*, vol. 34, pp. 707–755, 2009.

[14] P. Tokekar, V. Isler, and A. Franchi, "Multi-target visual tracking with aerial robots," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2014, pp. 3067–3072.

[15] M. Corah and N. Michael, "Distributed matroid-constrained submodular maximization for multi-robot exploration: Theory and practice," *Auton. Robots*, vol. 43, no. 2, pp. 485–501, 2019.

[16] M. Lauri, J. Pajarinen, J. Peters, and S. Frintrop, "Multi-sensor next-best-view planning as matroid-constrained submodular maximization," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 5323–5330, Oct. 2020.

[17] B. Schlotfeldt, V. Tzoumas, and G. J. Pappas, "Resilient active information acquisition with teams of robots," *IEEE Trans. Robot. (TRO)*, vol. 38, no. 1, pp. 244–261, Feb. 2022.

[18] B. Biggs, J. McMahon, P. Baldoni, and D. J. Stilwell, "Non-submodular maximization via the greedy algorithm and the effects of limited information in multi-agent execution," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 10118–10125.

[19] O. Akcin, O. Unuvar, O. Ure, and S. P. Chinchali, "Fleet active learning: A submodular maximization approach," in *Proc. 7th Annu. Conf. Robot Learn.*, 2023, pp. 1378–1399.

[20] X. Cai, B. Schlotfeldt, K. Khosoussi, N. Atanasov, G. J. Pappas, and J. P. How, "Energy-aware, collision-free information gathering for heterogeneous robot teams," *IEEE Trans. Robot.*, vol. 39, no. 4, pp. 2585–2602, Aug. 2023.

[21] A. J. Smith and G. A. Hollinger, "Distributed inference-based multi-robot exploration," *Auton. Robots*, vol. 42, no. 8, pp. 1651–1668, 2018.

[22] J. Yu et al., "SMMR-explore: Submap-based multi-robot exploration system with multi-robot multi-target potential field exploration method," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2021, pp. 8779–8785.

[23] Y. Gao et al., "Meeting-merging-mission: A multi-robot coordinate framework for large-scale communication-limited exploration," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 13700–13707.

[24] B. Zhou, H. Xu, and S. Shen, "RACER: Rapid collaborative exploration with a decentralized multi-UAV system," *IEEE Trans. Robot.*, vol. 39, no. 3, pp. 1816–1835, Jun. 2023.

[25] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Proc. 11th Int. Conf. Field Service Robot.: Results*, 2018, pp. 621–635.

[26] S. Hughes, R. Martin, M. Corah, and S. Scherer, "Multi-robot planning for filming groups of moving actors leveraging submodularity and pixel density," in *Proc. IEEE Conf. Decis. Contr.*, 2024, pp. 5401–5408.

[27] Z. Xu and V. Tzoumas, "Resource-aware distributed submodular maximization: A paradigm for multi-robot decision-making," in *Proc. IEEE Conf. Decis. Control*, 2022, pp. 5959–5966.

[28] Z. Xu and V. Tzoumas, "Performance-aware self-configurable multi-agent networks: A distributed submodular approach for simultaneous coordination and network design," in *Proc. IEEE Conf. Decis. Control*, 2024, pp. 5393–5400.

[29] Q. Wu et al., "A comprehensive overview on 5G-and-beyond networks with UAVs: From communications to sensing and intelligence," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 10, pp. 2912–2945, Oct. 2021.

[30] A. Robey, A. Adibi, B. Schlotfeldt, H. Hassani, and G. J. Pappas, "Optimal algorithms for submodular maximization with distributed constraints," in *Proc. Learn. Dyn. Cont.*, 2021, pp. 150–162.

[31] R. Konda, D. Grimsman, and J. R. Marden, "Execution order matters in greedy algorithms with limited information," in *Proc. Amer. Control Conf.*, 2022, pp. 1305–1310.

[32] B. Yamauchi, "Decentralized coordination for multirobot exploration," *Robot. Auton. Syst.*, vol. 29, no. 2-3, pp. 111–118, 1999.

[33] R. Zlot, A. Stentz, M. B. Dias, and S. Thayer, "Multi-robot exploration controlled by a market economy," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2002, vol. 3, pp. 3016–3023.

[34] M. Berhault et al., "Robot exploration with combinatorial auctions," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2003, vol. 2, pp. 1957–1962.

[35] W. Sheng, Q. Yang, J. Tan, and N. Xi, "Distributed multi-robot coordination in area exploration," *Robot. Auton. Syst.*, vol. 54, no. 12, pp. 945–955, 2006.

[36] L. Klodt and V. Willert, "Equitable workload partitioning for multi-robot exploration through pairwise optimization," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 2809–2816.

[37] B. Gharesifard and S. L. Smith, "Distributed submodular maximization with limited information," *IEEE Trans. Control Netw. Syst.*, vol. 5, no. 4, pp. 1635–1645, Dec. 2018.

[38] D. Grimsman, M. S. Ali, J. P. Hespanha, and J. R. Marden, "The impact of information in distributed submodular maximization," *IEEE Trans. Control Netw. Syst.*, vol. 6, no. 4, pp. 1334–1343, Dec. 2019.

[39] F. F. Lizzio, E. Capello, and G. Guglieri, "Implementation and performance evaluation of a consensus protocol for multi-UAV formation with communication delay," in *Proc. Int. Conf. Unmanned Aircr. Syst.*, 2022, pp. 1592–1600.

[40] S. Baidya, Z. Shaikh, and M. Levorato, "FlyNetSim: An open source synchronized UAV network simulator based on NS-3 and ArduPilot," in *Proc. 21st ACM Int. Conf. Model., Anal. Simul. Wireless Mobile Syst.*, 2018, pp. 37–45.

[41] M. Calvo-Fullana, D. Mox, A. Pyattaev, J. Fink, V. Kumar, and A. Ribeiro, "ROS-NetSim: A framework for the integration of robotic and network simulators," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 1120–1127, Apr. 2021.

[42] S. Acharya, M. Bharatheesha, Y. Simmhan, and B. Amrutur, "A co-simulation framework for communication and control in autonomous multi-robot systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2023, pp. 11087–11094.

[43] M. Selden, J. Zhou, F. Campos, N. Lambert, D. Drew, and K. S. Pister, "BotNet: A simulator for studying the effects of accurate communication models on multi-agent and swarm control," in *Proc. Int. Symp. Multi-Robot Multi-Agent Syst.*, 2021, pp. 101–109.

[44] M. Corah and N. Michael, "Distributed submodular maximization on partition matroids for planning on large sensor networks," in *Proc. IEEE Conf. Decis. Control*, 2018, pp. 6792–6799.

[45] A. Downie, B. Gharesifard, and S. L. Smith, "Submodular maximization with limited function access," *IEEE Trans. Autom. Control*, vol. 68, no. 9, pp. 5522–5535, Sep. 2023.

[46] Y. Crama, P. L. Hammer, and R. Holzman, "A characterization of a cone of Pseudo-Boolean functions via supermodularity-type inequalities," in *Quantitative Methoden in Den Wirtschaftswissenschaften*. Berlin, Germany: Springer, 1989, pp. 53–55.

[47] S. Foldes and P. L. Hammer, "Submodularity, supermodularity, and higher-order monotonicities of pseudo-Boolean functions," *Math. Operations Res.*, vol. 30, no. 2, pp. 453–461, 2005.
[48] J. Liu, L. Zhou, P. Tokekar, and R. K. Williams, "Distributed resilient submodular action selection in adversarial environments," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 5832–5839, Jul. 2021.
[49] O. S. Oubbati, M. Atiquzzaman, P. Lorenz, M. H. Tareque, and M. S. Hossain, "Routing in flying ad hoc networks: Survey, constraints, and future challenge perspectives," *IEEE Access*, vol. 7, pp. 81057–81105, 2019.
[50] 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 Appl. Math.*, vol. 7, no. 3, pp. 251–274, 1984.
[51] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York, NY, USA: Wiley, 2012.
[52] N. Rezazadeh and S. S. Kia, "Distributed strategy selection: A submodular set function maximization approach," *Automatica*, vol. 153, 2023, Art. no. 111000.
[53] B. Du, K. Qian, C. Claudel, and D. Sun, "Jacobi-style iteration for distributed submodular maximization," *IEEE Trans. Autom. Control*, vol. 67, no. 9, pp. 4687–4702, Sep. 2022.
[54] G. Calinescu, C. Chekuri, M. Pál, and J. Vondrák, "Maximizing a monotone submodular function subject to a matroid constraint," *SIAM J. Comput.*, vol. 40, no. 6, pp. 1740–1766, 2011.
[55] C. Godsil and G. F. Royle, *Algebraic Graph Theory*, vol. 207. Berlin, Germany: Springer, 2001.
[56] M. Mesbahi and M. Egerstedt, *Graph Theoretic Methods in Multiagent Networks*. Princeton, NJ, USA: Princeton Univ. Press, 2010.
[57] Y. Tian, Y. Chang, F. H. Arias, C. Nieto-Granda, J. P. How, and L. Carlone, "Kimera-multi: Robust, distributed, dense metric-semantic slam for multi-robot systems," *IEEE Trans. Robot.*, vol. 38, no. 4, pp. 2022–2038, Aug. 2022.
[58] R. K. Iyer, S. Jegelka, and J. A. Bilmes, "Curvature and optimal algorithms for learning and minimizing submodular functions," in *Adv. Neural Inf. Process. Syst.*, 2013, pp. 2742–2750.
[59] B. Lehmann, D. Lehmann, and N. Nisan, "Combinatorial auctions with decreasing marginal utilities," *Games Econ. Behav.*, vol. 55, no. 2, pp. 270–296, 2006.
[60] A. Krause and V. Cevher, "Submodular dictionary selection for sparse representation," in *Proc. Int. Conf. Mach. Learn.*, 2010, pp. 567–574.
[61] E. R. Elenberg et al., "Restricted strong convexity implies weak submodularity," *Ann. Statist.*, vol. 46, no. 6B, pp. 3539–3568, 2018.
[62] L. F. Chamon and A. Ribeiro, "Near-optimality of greedy set selection in the sampling of graph signals," in *Proc. IEEE Glob. Conf. Signal Inf. Process.*, 2016, pp. 1265–1269.
[63] B. Guo, O. Karaca, T. Summers, and M. Kamgarpour, "Actuator placement for optimizing network performance under controllability constraints," in *Proc. IEEE Conf. Decis. Control*, 2019, pp. 7140–7147.
[64] T. Horel and Y. Singer, "Maximization of approximately submodular functions," *Adv. Neural Inf. Process. Syst.*, 2016, vol. 29, pp. 3053–3061.

Zirui Xu (Graduate Student Member, IEEE) received the B.Eng. degree in automation from Northeastern University, Shenyang, China, in 2018 and the M.S. degree in electrical and computer engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 2020. He is currently working toward the Ph.D. degree in aerospace engineering with the Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI, USA.

His research focuses on theories and algorithms for scalable and reliable coordination of distributed multirobot systems in resource-constrained, unstructured, and untrustworthy environments.

Sandilya Sai Garimella (Graduate Student Member, IEEE) received the B.S.E. degree in mechanical engineering and the M.S.E. degree in robotics from the University of Michigan, Ann Arbor, MI, USA, in 2022 and 2024, respectively. He is currently working toward the Ph.D. degree in robotics with the Institute for Robotics and Intelligent Machines (IRIM), Georgia Institute of Technology, Atlanta, GA, USA.

His research interests lie in multirobot autonomy, with a focus on algorithms for spatial perception, active information acquisition, and efficient multiagent decision-making.

Vasileios Tzoumas (Senior Member, IEEE) received the Diploma in electrical and computer engineering from the National Technical University of Athens, Athens, Greece, in 2012, the Ph.D. degree in electrical and systems engineering from the University of Pennsylvania, Philadelphia, PA, USA, in 2018, the Master of Arts in statistics from the Wharton School of Business, University of Pennsylvania, in 2016, the Master of Science in electrical engineering from the University of Pennsylvania, in 2016.

He is currently an Assistant Professor with the Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI, USA. He was with the Massachusetts Institute of Technology (MIT), with the Department of Aeronautics and Astronautics, and with the Laboratory for Information and Decision Systems (LIDS), where he was a Research Scientist (2019–2020) and a Postdoctoral associate (2018–2019). He works on algorithms and innovative hardware for scalable and reliable cyber-physical systems in resource-constrained, uncertain, and contested environments via resource-aware decision-making, online learning, and resilient adaptation.

Dr. Vasileios is a recipient of an NSF CAREER Award, the Best Paper Award in Robot Vision at the 2020 IEEE International Conference on Robotics and Automation (ICRA), an Honorable Mention from the 2020 IEEE ROBOTICS AND AUTOMATION LETTERS (RA-L), and was a Best Student Paper Award finalist at the 2017 IEEE Conference in Decision and Control (CDC).