Distributed Strategy Selection: A Submodular Set Function Maximization Approach

Navid Rezazadeh [a], Solmaz S. Kia [a]

[a] *University of California, Irvine*

Abstract

Joint utility-maximization problems for multi-agent systems often should be addressed by distributed strategy-selection formulation. Constrained by discrete feasible strategy sets, these problems are broadly formulated as NP-hard combinatorial optimization problems. However, in many cases, it is possible to reformulate this class of problems as constrained submodular set function maximization problems which also belong to the NP-hard domain of problems. A prominent example is the problem of multi-agent mobile sensor dispatching over a discrete domain. This paper considers a class of submodular optimization problems that consist of maximization of a monotone and submodular set function subject to a partition matroid constraint over a group of networked agents that communicate over a connected undirected graph. We work with the *value oracle* model. Consequently, the only access of the agents to the utility function is through a black box that returns the utility function value given a specific strategy set. We propose a distributed suboptimal polynomial-time algorithm that enables each agent to obtain its respective strategy via local interactions with its neighboring agents. Our solution is a fully distributed gradient-based algorithm using the submodular set functions' multilinear extension followed by a distributed stochastic Pipage rounding procedure. This algorithm results in a strategy set that when the team utility function is evaluated at the worst case, the utility function value is in $\frac{1}{c}(1-\mathrm{e}^{-c}-O(1/T))$ of the optimal solution with c to be the curvature of the submodular function. An example demonstrates our results.

Key words: Submodular optimization; sensor placement, multilinear extension; distributed optimization;

1 Introduction

Modern industries such as transportation, supply chain, energy, and finance are moving fast towards modular and distributed operations where communicating smart subsystems are expected to coordinate their actions for the optimal operation of the entire system. Optimal strategy selection problems for these networked systems often appear as combinatorial optimization problems where the objective function is a submodular set function. Some example cases include sensor and actuator placement problems [1,2], energy storage placement [3,4], measurement scheduling [5], voltage control in smart grid [6], persistent monitoring via mobile robots [7]. For reasons such as robustness, scalability, privacy preservation, and avoiding a single failure point, these optimal decision-making problems are highly desired to be solved in a distributed manner.

While there has been a plethora of work in developing central solutions for submodular maximization, satisfactory distributed algorithmic solutions for in-network submodular maximization problems where agents communicate over a graph have remained elusive. In this paper, we consider a distributed strategy selection problem that is modeled as submodular maximization subject to *partition matriod*. We seek a distributed solution in which the agents communicate over a connected undirected graph.

Submodular function maximization: A set function $f : 2^{\mathcal{P}} \to \mathbb{R}_{\geq 0}$ defined on the ground set $\mathcal{P}$ is *submodular* if $\forall\, \mathcal{S} \subset \mathcal{T} \subset \mathcal{P}$, and $p \in \mathcal{P} \setminus \mathcal{T}$ we have

$$f(\mathcal{S} \cup \{p\}) - f(\mathcal{S}) \geq f(\mathcal{T} \cup \{p\}) - f(\mathcal{T}). \qquad (1)$$

Submodular set functions naturally possess the diminishing returns property, i.e., the gain of adding a new element p to a set decreases or stays the same as the size of the set increases. Submodularity is an inherent property in many practical utility/objective functions such as weighted coverage functions, facility location service function, entropy, and mutual information functions, which appear in strategy selection problems such as sensor placement, measurement scheduling, workforce hiring, and database sampling [8].

⋆ This work is supported by NSF award IIS-SAS-1724331.
Emails:{nrezazad,solmaz}@uci.edu

Unlike minimization of submodular functions that can be done in polynomial time [9, 10], submodular function maximization problems are NP-hard [11]. Luckily, submodularity is a property of set functions with deep theoretical consequences that enables establishing constant factor approximate (suboptimal solutions) for submodular maximization problems. Research on problems involving the maximization of monotone submodular functions dates back to the work of Nemhauser, Wolsey, and Fisher in the 1970's [11–13]. A fundamental result by Nemhauser et al. [11] establishes that the simple *sequential greedy algorithm* is guaranteed to provide a constant 1/2-approximation factor solution for submodular maximization subject to matroid constraints. The sequential greedy algorithm reaches the final solution by sequentially finding the best current decision based on the decisions made previously and without considering the consequences or interactions with future decisions. These bounds can be made tighter with additional knowledge on the diminishing return property of the submodular objective function quantified by *total curvature* $c \in [0, 1]$. For example [14] shows that the constant factor approximation for submodular maximization subject to a matroid constraint is $\frac{1}{1+c}$.

More recently, another suboptimal solution for submodular maximization subject to matroid constraints with an improved optimality gap is proposed in the literature using the *multilinear* continuous relaxation of a submodular set function [15–19]. The relaxation transforms the discrete problem into a continuous optimization problem with linear constraints. Then, a continuous gradient-based optimization algorithm referred to as *continuous greedy algorithm*, is used to solve the continuous optimization problem. A suboptimal solution for submodular maximization subject to matroid constraint with the improved constant-factor approximation of $(1 - 1/\mathrm{e})$ then is obtained by proper *rounding* of the continuous-domain solution [15, 16]. This approach however requires a central authority to solve the problem. It is worth noting that the literature has shown that for monotone submodular functions, it is computationally hard to approximate this problem within a factor better than $1 - 1/e \approx 0.63\%$ [20].

Distributed submodular function maximization: In multi-agents setting, for example, multi-agent sensor placement problems where the agents are self-organizing autonomous mobile agents with communication and computation capabilities, it is desired to solve the strategy selection problems modeled as constrained submodular maximization problems in a distributed way without involving a central authority. The problems in distributed settings can be divided into two categories: distributed constraint problems and distributed utility problems. In a distributed constraint problem there is a shared utility but each agent has to choose its strategy from a local constraint set that is disjoint from other agents' and is only known to the agent. An example is the heterogeneous coverage problem where each agent has a set of heterogeneous sensors while the area to cover is shared among them. The agents should decide what sensor(s) each to deploy to maximize the area coverage as a team. In distributed utility problems, however, the team's utility function is the sum of the separable local utilities and agents choose their strategies from a shared strategy set. An example case is the optimal Welfare problem [15] where each agent should make strategy choices from a joint set such that the sum of local utilities is maximized. Our focus in this paper is on distributed solution design for a fragmented-constraint submodular maximization problem.

For distributed constraint problems, the sequential greedy algorithm can be implemented in a decentralized way through sequential message-passing or via sequential message-sharing through a cloud [7]. However, a decentralized sequential greedy algorithm comes with communication routing overhead. For agents communicating over a connected graph, implementing sequential message-passing requires finding the Hamiltonian path (a connected path that visits every agent on the graph only once) which is an NP-hard problem to solve. If Hamiltonian path does not exist in a graph, a path that visits the agents in least frequent times should be identified for communication efficient sequential message-passing. Moreover, it is shown that the order of sequence changes the actual approximation factor of the solution obtained by the sequential greedy algorithm [21]. The complexity of finding the sequence that delivers the best solution increases exponentially as the size and the connectivity of the communication network increases. Several attempts have been also undertaken to adapt the sequential greedy algorithm for large-scale submodular maximization problems by reducing the size of the problem through approximations [22] or using several processing units to achieve a faster sequential greedy algorithm, but with some sacrifices on the optimality bound [23–26]. However, these decentralized implementations are mainly intended for parallel processing purposes and are not extendable to decentralized operations when agents communicate over connected graphs.

Some attempts have also been made in devising distributed solutions for submodular maximization using multi-linear extension approaches. For the class of distributed utility functions semi-distributed and fully distributed solutions with an optimality gap close to $(1 - 1/\mathrm{e})$ is studied in [27–29]. However, for the class of distributed constraint problems, the results in the literature are rather limited. For the special class of submodular set functions with curvature $c = 1$, and when each agent is limited to choose only a single strategy from its own strategy set, [30] has proposed an average consensus-based distributed algorithm to the maximization problem over connected graphs. The solution of [30] requires a closed-form expression of the multi-linear extension function. However, the computational complexity of constructing the closed-form of multi-linear extension of a submodular function and its derivatives increases expo-

nentially with the size of the strategy set. Moreover, the result also depends on a centralized rounding scheme.

Statement of contributions: In this paper, motivated by the improved optimality gap of the multilinear continuous relaxation-based algorithms, we develop a distributed implementation of the algorithm of [16] over a connected undirected graph. Particularly, we consider a distributed submodular set function maximization problem formulated by a shared utility function and disjoint strategy sets (fragmented-constraint class). Moreover, in our setup, the agents are allowed to choose multiple strategies from their strategy sets.

We propose a gradient-based algorithm, constructed on multi-linear extension function of a submodular function, which uses a maximum consensus scheme over the communication graph and results in a distributed implementation of the continuous greedy algorithm. The multi-linear extension function of a submodular function is equivalently the expected value of the submodular function evaluated at random sets obtained by picking strategies from the strategy set independently with a probability. This stochastic interpretation allows approximating the multi-linear extension function and its derivatives empirically with a reasonable computational cost via sampling from the strategy set [16]. Furthermore, we carefully analyze the effect of stochastic interpretation of the multilinear extended function and sampling on our algorithm's optimality gap.

We complete our solution by designing a lossless distributed rounding procedure that computes the final suboptimal strategy of each agent. The purpose of a rounding procedures is to convert an optimal solution of a relaxed problem into an approximately optimal solution to the original problem. In a distributed setting, generally an attempt to implement a centralized rounding procedure [16, 31] requires extra coordinating communication between the agents. However, our choice of maximum consensus algorithm as the agreements protocol between the agents removes the need for further communication. We show that after the coordination of the agents through maximum consensus for a given period of time, each agent can use a local randomized Pipage procedure to reach a deterministic set of strategies as its local solution without the necessity to interact with other agents. Furthermore, our algorithm guarantees the resulting global suboptimal strategy lies in the feasible constraint set.

Through rigorous analysis which takes into account the total curvature of the utility function, we show that our proposed distributed algorithm in finite time T achieves, with a known probability, a $\frac{1}{c}(1-\mathrm{e}^{-c})-O(1/T)$ optimality bound, where $1/T$ is the step size of the algorithm and the frequency at which agents communicate over the network. A numerical example demonstrates our results. Using various scenarios, this numerical study highlights the higher computational cost of a solution based on a continuous-relaxation in comparison to the sequential greedy algorithm. But, this high cost is paid off by an improved optimality gap, no overhead to determine message passing sequence and independence of the result from the massage passing sequence.

Notation and definitions: We denote the vectors with bold small font. The p^{th} element of vector $\mathbf{x}$ is denoted by $[\mathbf{x}]_p$. We denote the inner product of two vectors $\mathbf{x}$ and $\mathbf{y}$ with appropriate dimensions by $\mathbf{x}.\mathbf{y}$. We use $\mathbf{0}$ and $\mathbf{1}$ as a vector of zeros and ones respectively, whose dimension is understood from the context. We denote sets with the capital curly font. Given a ground set $\mathcal{P}=\{1,\cdots,n\}$, we define the membership probability vector $\mathbf{x}\in[0,1]^n$ to obtain $\mathcal{R}_{\mathbf{x}}\subset\mathcal{P}$ as a random set where $p\in\mathcal{P}$ is in $\mathcal{R}_{\mathbf{x}}$ with the probability $[\mathbf{x}]_p$. For $\mathcal{R}\subset\mathcal{P}$, $\mathbf{1}_{\mathcal{R}}\in\{0,1\}^n$ is the vector whose p^{th} element is 1 if $p\in\mathcal{R}$ and 0 otherwise; we call $\mathbf{1}_{\mathcal{R}}$ the membership indicator vector of set $\mathcal{R}$. Given a finite countable set $\mathcal{R}\subset\mathbb{R}$ and integer $\kappa, 1\leq\kappa<|\mathcal{R}|$, $\max(\mathcal{R},\kappa)$ returns the κ largest elements of $\mathcal{R}$. For $x\in\mathbb{R}$, $|x|$ is ite absolute value. By overloading the notation, we also use $|\mathcal{R}|$ as the cardinality of set $\mathcal{R}$. For a set function $f:2^{\mathcal{P}}\to\mathbb{R}$, we define $\Delta_f(p|\mathcal{R})=f(\mathcal{R}\cup\{p\})-f(\mathcal{R}),\ \mathcal{R}\subset\mathcal{P}$. A set function is *normalized* if $f(\emptyset)=0$. A set function $f:2^{\mathcal{P}}\to\mathbb{R}_{\geq 0}$ is *monotone increasing* if $f(\mathcal{P}_1)\leq f(\mathcal{P}_2)$ for any $\mathcal{P}_1\subset\mathcal{P}_2\subset\mathcal{P}$.

2 Problem definition and preliminaries

Consider a group of $\mathcal{A}=\{1,...,N\}$ agents with communication and computation capabilities, interacting over a connected undirected graph $\mathcal{G}(\mathcal{A},\mathcal{E})$ where $\mathcal{E}\subset\mathcal{A}\times\mathcal{A}$ is the edge set. Recall that $\mathcal{G}$ is undirected if and only if $(i,j)\in\mathcal{E}$ means that agents i and j can mutually exchange information. An undirected graph is connected if there is a path from each node to every other node in the graph.

Each agent $i\in\mathcal{A}$ has a distinct discrete strategy set $\mathcal{P}_i$, known only to agent i, and wants to choose at most $\kappa_i\in\mathbb{Z}_{>0}$ strategies from $\mathcal{P}_i$ such that a monotone increasing and submodular utility function $f:2^{\mathcal{P}}\to\mathbb{R}_{\geq 0}$, $\mathcal{P}=\bigcup_{i\in\mathcal{A}}\mathcal{P}_i$, evaluated at all the agents' strategy selection is maximized. In other words, the agents aim to solve in a distributed manner the discrete domain optimization problem

$$\max_{\mathcal{R}\in\mathcal{I}} f(\mathcal{R}) \tag{2a}$$

$$\mathcal{I}=\big\{\mathcal{R}\subset\mathcal{P}\ \big|\ |\mathcal{R}\cap\mathcal{P}_i|\leq\kappa_i,\ \forall i\in\mathcal{A}\big\}. \tag{2b}$$

The agents' access to the utility function is through a black box that returns $f(\mathcal{R})$ for any given set $\mathcal{R}\subset\mathcal{P}$ (value oracle model). The constraint set (2b) is a partition matroid, which restricts the number of strategy choices of each agent $i\in\mathcal{A}$ to κ_i. In a distributed solution, each agent $i\in\mathcal{A}$ should obtain its respective component $\mathcal{R}_i^\star\subset\mathcal{P}_i$ of $\mathcal{R}^\star=\cup_{j=1}^N\mathcal{R}_j^\star$, the optimal solution of (2), by interacting only with the agents that are in its communication range.

In the remainder of this paper, without loss of general-

ity, we assume that global strategy set is given by $\mathcal{P} = \bigcup_{i\in\mathcal{A}} \mathcal{P}_i = \{1, \cdots, n\}$. Also, we assume that the agents' local strategies each are non-empty consecutive integers and ordered such that if $\mathcal{P}_i = \{p, p+1, \cdots, q\} \subset \mathcal{P}$, then $p-1 \in \mathcal{P}_{i-1}$ and $q+1 \in \mathcal{P}_{i+1}$.

The distributed solution we propose for solving (2) relies on a multilinear extension relaxation approach and a rounding procedure. In what follows, we introduce the notation and definitions needed for this approach.

2.1 Multilinear relaxation

The utility function f assigns values to all the subsets of $\mathcal{P} = \bigcup_{i\in\mathcal{A}} \mathcal{P}_i = \{1, \cdots, n\}$. Thus, equivalently, we can regard the set value utility function as a function on the Boolean hypercube $\{0,1\}^n$, i.e., $f : \{0,1\}^n \to \mathbb{R}$. For a submodular function $f : 2^{\mathcal{P}} \to \mathbb{R}_{\geq 0}$, its *multilinear extension* $F : [0,1]^n \to \mathbb{R}_{\geq 0}$ in the continuous space is [15]

$$F(\mathbf{x}) = \sum_{\mathcal{R}\subset\mathcal{P}} f(\mathcal{R}) \prod_{p\in\mathcal{R}} [\mathbf{x}]_p \prod_{p\notin\mathcal{R}} (1 - [\mathbf{x}]_p), \quad \mathbf{x} \in [0,1]^n. \quad (3)$$

Given $\mathbf{x} \in [0,1]^n$ we can define $\mathcal{R}_\mathbf{x}$ to be the random subset of $\mathcal{P}$ in which each element $p \in \mathcal{P}$ is included independently with probability $[\mathbf{x}]_p$ and not included with probability $1 - [\mathbf{x}]_p$. Then the multilinear extension F in (3) is simply [15]

$$F(\mathbf{x}) = \mathbb{E}[f(\mathcal{R}_\mathbf{x})], \quad (4)$$

where $\mathbb{E}[.]$ indicates the expected value. Taking the derivatives of $F(\mathbf{x})$ yields [15]

$$\frac{\partial F}{\partial [\mathbf{x}]_p}(\mathbf{x}) = \mathbb{E}[f(\mathcal{R}_\mathbf{x} \cup \{p\}) - f(\mathcal{R}_\mathbf{x} \setminus \{p\})], \quad (5)$$

and

$$\frac{\partial^2 F}{\partial [\mathbf{x}]_p \partial [\mathbf{x}]_q}(\mathbf{x}) = \mathbb{E}[f(\mathcal{R}_\mathbf{x} \cup \{p,q\}) - f(\mathcal{R}_\mathbf{x} \cup \{q\} \setminus \{p\}) - f(\mathcal{R}_\mathbf{x} \cup \{p\} \setminus \{q\}) + f(\mathcal{R}_\mathbf{x} \setminus \{p,q\})]. \quad (6)$$

Multilinear-extension function $F(\mathbf{x})$, expands the function evaluation of the utility function over the space between the vertices of the Boolean hypercube $\{0,1\}^n$. For a solution via a continuous relaxation method, the effect of partition matroid constraint should also be considered. To do so, the *matroid polytope* for partition matroid is defined as

$$\mathcal{M} = \{\mathbf{x} \in [0,1]^n \mid \sum\nolimits_{p\in\mathcal{P}_i} [\mathbf{x}]_p \leq \kappa_i, \forall i \in \mathcal{A}\}. \quad (7)$$

The matroid polytope $\mathcal{M}$ is the convex hull of the vertices of the hypercube $\{0,1\}^n$ that satisfy the partition matroid constraint (2b). Additionally, note that according to (3), $F(\mathbf{x})$ for any $\mathbf{x} \in \mathcal{M}$ is a weighted average of values of F at the vertices of the matriod polytope $\mathcal{M}$. Then, equivalently, $F(\mathbf{x})$ at any $\mathbf{x} \in \mathcal{M}$ is a normalized-weighted average of f on the strategies satisfying constraint (2b). As such,

$$f(\mathcal{R}^\star) \geq F(\mathbf{x}), \quad \mathbf{x} \in \mathcal{M},$$

which is equivalent to $f(\mathcal{R}^\star) = \max_{\mathbf{x}\in\mathcal{M}} F(\mathbf{x})$, where $\mathcal{R}^\star$ is the optimizer of problem (2) [15]. Therefore, solving the continuous domain optimization problem

$$\max_{\mathbf{x}\in\mathcal{M}} F(\mathbf{x}), \quad (8)$$

can lead to finding the $\mathcal{R}^\star$.

A practical implementation of a gradient-based method to solve (8) is achieved by using an Euler discretized implementation with stepsize of $\frac{1}{T}$ in $T \in \mathbb{Z}_{>0}$ steps. A significant challenge in implementing a gradient-based method is the exponential cost of computing the gradient $\nabla F(\mathbf{x})$ whose calculation requires the knowledge of f at each $R \subset 2^{\mathcal{P}}$. The stochastic interpretation (4) of the multilinear extension and its derivatives however offer a mechanism to estimate them with a reasonable computational cost via sampling.

3 Distributed submodular maximization subject to partition matroid

In this section we propose a distributed algorithm for the submodular maximization problem defined in Section 2. Our solution relies on the continuous relaxation of the discrete optimization (2). We first find a suboptimal solution to the relaxed problem and then propose a rounding method to map this solution to a feasible suboptimal solution for (2).

3.1 Distributed Discrete Gradient Ascent Solution

In the distributed setting described in our problem definition, every agent initially has access only to its own strategy set. Let every agent $i \in \mathcal{A}$ maintain and evolve a local copy of the membership probability vector as $\mathbf{x}_i(t) \in \mathbb{R}^n$. Since $\mathcal{P} = \{1, \cdots, n\}$ is sorted agent-wise, we denote $\mathbf{x}_i(t) = [\hat{\mathbf{x}}_{i1}^\top(t), \cdots, \mathbf{x}_{ii}^\top(t), \cdots, \hat{\mathbf{x}}_{iN}^\top(t)]^\top \in \mathbb{R}^n$ where $\mathbf{x}_{ii}(t) \in \mathbb{R}_{\geq 0}^{|\mathcal{P}_i|}$ is the membership probability vector of agent i's own strategy with entries of $[\mathbf{x}_i(t)]_p,\ p \in \mathcal{P}_i$ at iteration $t \in \{0, 1, \cdots, T\}, T \in \mathbb{Z}_{>0}$, while $\hat{\mathbf{x}}_{ij}(t) \in \mathbb{R}_{\geq 0}^{|\mathcal{P}_j|}$ is the local estimate of the membership probability vector of agent j by agent i with entries of $[\mathbf{x}_i(t)]_p,\ p \in \mathcal{P}_j,\ j \in \mathcal{A}\backslash\{i\}$. Every agent $i \in \mathcal{A}$ initializes at $\mathbf{x}_i(0) = \mathbf{0}$ and implements the *propagation* and *update* steps

$$\mathbf{x}_i^-(t+1) = \mathbf{x}_i(t) + \frac{1}{T}\widetilde{\mathbf{v}}_i(t), \quad (9a)$$

$$\mathbf{x}_i(t+1) = \max_{j\in\mathcal{N}_i\cup\{i\}} \mathbf{x}_j^-(t+1), \quad (9b)$$

where

$$\widetilde{\mathbf{v}}_i(t) = \underset{\mathbf{w}\in\mathcal{M}_i}{\operatorname{argmax}}\, \mathbf{w}.\widetilde{\nabla F}(\mathbf{x}_i(t)) \tag{10}$$

with

$$\mathcal{M}_i = \Big\{ \mathbf{w}\in[0,1]^n \Big|\, \mathbf{1}^\top.\mathbf{w} \le \kappa_i \ , [\mathbf{w}]_p = 0,\ \forall p \in \mathcal{P}\backslash\mathcal{P}_i \Big\}. \tag{11}$$

The vector $\widetilde{\nabla F}(\mathbf{x}_i(t))$ is the empirical estimate of $\nabla F(\mathbf{x}_i(t))$. At time step t, each agent $i \in \mathcal{A}$ generates K_i independent samples $\{\mathcal{R}_i^k(t)\}_{k=1}^{K_i}$ of random set $\mathcal{R}_{\mathbf{x}_i(t)}$ drawn according to membership probability vector $\mathbf{x}_i(t)$ from $\mathcal{P}$ and empirically computes gradient vector $\nabla F(\mathbf{x}_i(t)) \in \mathbb{R}^n_{\ge 0}$, which according to (5) is defined element-wise as

$$\Big[\widetilde{\nabla F}(\mathbf{x}_i(t))\Big]_p = \Bigg(\sum_{k=1}^{K_i} f(\mathcal{R}_i^k(t)\cup\{p\}) - f(\mathcal{R}_i^k(t)\setminus\{p\})\Bigg)\Big/ K_i, \tag{12}$$

$p \in \mathcal{P} = \{1,\cdots,n\}$.

In the propagation step (9a) agent i takes a step along a feasible gradient ascent direction in its own local polytope (11). Because the propagation is only based on the local information of agent i, in the update step (9b), the propagated $\mathbf{x}_i^-(t+1)$ of each agent $i \in \mathcal{A}$ is updated by element-wise maximum seeking among its neighbors. Lemma 1 below, whose proof is given in the Appendix D, shows that, as expected,

$$\mathbf{x}_{ii}(t) = \mathbf{x}_{ii}^-(t), \quad i \in \mathcal{A},$$

i.e., the corrected component of $\mathbf{x}_i$ corresponding to agent i itself is the propagated value maintained at agent i, and not the estimated value of any of its neighbors.

Lemma 1 *Let the agents propagate and update their local membership probability vector according to* (9a) *and* (9b). *Let*

$$\bar{\mathbf{x}}(t) = \max_{i\in\mathcal{A}} \mathbf{x}_i(t). \tag{13}$$

Then, at any $t \in \{0,1,\cdots,T\}$, we have

$$\bar{\mathbf{x}}(t) = [\mathbf{x}_{11}^\top(t),\cdots,\mathbf{x}_{NN}^\top(t)]^\top. \tag{14}$$

We interpret $\bar{\mathbf{x}}(t)$ as the *global probability membership vector* of the network. Next lemma, whose proof is given in Appendix D, states that both $\mathbf{x}_i(t)$ and $\bar{\mathbf{x}}(t)$ belong to $\mathcal{M}$ for any $t \in \{0,\cdots,T\}$, i.e., the trajectories of the local and global membership probability vectors never leave the matroid polytope.

Lemma 2 *Let the agents propagate and update their local membership probability vector according to* (9a) *and* (9b). *Then, (a) $\mathbf{x}_i(t)$, $\bar{\mathbf{x}}(t) \in \mathcal{M}$ at any $t \in \{0,1,\cdots,T\}$; (b) $\mathbf{1}.\mathbf{x}_{ii}(T) = \kappa_i$, $\mathbf{1}.\hat{\mathbf{x}}_{ij}(T) \le \kappa_j$, $j \in \mathcal{A}\setminus\{i\}$, and $\mathbf{x}_i(T) \in [0,1]^n$.*

The following result, whose proof is given in Appendix D, establishes the difference between each agent i's local copy of the membership probability vector $\mathbf{x}^i(t)$ and the global probability membership vector of the network $\bar{\mathbf{x}}(t)$. It also shows how the global probability membership vector evolves when agents implement (9). This result is instrumental in establishing proof of Theorem 4.

Proposition 3 *Let the agents propagate and update their local membership probability vector according to* (9a) *and* (9b). *Then, the membership probability $\mathbf{x}_i(t)$, $t \in \{0,\cdots,T\}$, for each agent $i \in \mathcal{A}$ satisfies*

$$0 \le \frac{1}{\kappa}\mathbf{1}.(\bar{\mathbf{x}}(t) - \mathbf{x}_i(t)) \le \frac{1}{T} d(\mathcal{G}), \tag{15a}$$

$$\bar{\mathbf{x}}(t+1) - \bar{\mathbf{x}}(t) = \frac{1}{T}\sum\nolimits_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t), \tag{15b}$$

$$\frac{1}{\kappa_i}\mathbf{1}.(\bar{\mathbf{x}}(t+1) - \bar{\mathbf{x}}(t)) = \frac{1}{T}, \tag{15c}$$

where $\kappa = \sum_{i\in\mathcal{A}} \kappa_i$ and $d(\mathcal{G})$ is the diameter of graph $\mathcal{G}$ and $\bar{\mathbf{x}}(t)$ is given by equation (14).

The following theorem, whose proof is given in Appendix E, quantifies the optimality gap of $F(\bar{\mathbf{x}}(T))$ with respect to the solution of the main problem (2). To characterize this optimality gap we take into account the total curvature of the utility function, defined as

$$c = 1 - \min_{\mathcal{S}\subset\mathcal{P},\ p\notin\mathcal{S}} \frac{\Delta_f(p|\mathcal{S})}{\Delta_f(p|\emptyset)}. \tag{16}$$

The total curvature $c \in [0,1]$ of a submodular set function $f : 2^{\mathcal{P}} \to \mathbb{R}_{\ge 0}$ shows the worst-case increase in the value of the function when member p is added.

Theorem 4 (Optimality gap) *Let the agents propagate and update their local membership probability vector according to* (9). *Let $\kappa = \sum_{i\in\mathcal{A}} \kappa_i$, K_i be number of samples agent i used to compute $\widetilde{\nabla F}(\mathbf{x}_i(t))$, and $\mathcal{R}^\star$ be the optimizer of problem* (2). *Then, with the probability of at least $\Big(\prod_{i\in\mathcal{A}}(1 - 2\mathrm{e}^{-\frac{1}{8T^2}K_i})^{|\mathcal{P}_i|}\Big)^T$, the global probability membership vector $\bar{\mathbf{x}}(T)$ satisfies*

$$F(\bar{\mathbf{x}}(T)) \ge \beta f(\mathcal{R}^\star), \tag{17}$$

where

$$\beta = \frac{1}{c}(1-\mathrm{e}^{-c})(1-(2\,c\,\kappa\,d(\mathcal{G}) + \frac{c\,\kappa}{2} + 1)\frac{\kappa}{T}). \tag{18}$$

and $d(\mathcal{G})$ is the diameter of graph $\mathcal{G}$.

Step 1	Randomized selection	0.15	0.25	0.10	0.20	0.10	0.80	0.05	0.35
	Randomized swapping	0	0.25	0.10	0.20	0.25	0.80	0.05	0.35

Step 2	Randomized selection	0	0.25	0.10	0.20	0.25	0.80	0.05	0.35
	Randomized swapping	0	0.25	0.10	0.20	0.25	1	0.05	0.15

Fig. 1. The first two steps of the stochastic Pipage rounding (19) for an agent i with $\mathbf{x}_{ii}(T) = [0.15, 0.25, 0.1, 0.2, 0.1, 0.8, 0.05, 0.35]^\top$ that should choose two strategies from $\mathcal{P}_i$.

Notice that since

$$1 - 2T\, n\, \mathrm{e}^{-\frac{1}{8T^2}\underline{K}} \leq \Big(\prod\nolimits_{i\in\mathcal{A}}(1 - 2\mathrm{e}^{-\frac{1}{8T^2}K_i})^{|\mathcal{P}_i|}\Big)^T,$$

where $\underline{K} = \min\{K_1, \cdots, K_N\}$, the probability of the bound (17) improves as T, and the number of the samples collected by the agents $\underline{K}$ increase.

3.1.1 Distributed Pipage rounding procedure

The final output of a distributed solver for problem (2) must be a set $\bar{\mathcal{R}}$ that belongs to $\mathcal{I}$ defined in (2b). Recall that strategies corresponding to the vertices of the matroid polytope $\mathcal{M}$ correspond to admissible strategy set $\mathcal{I}$. However, $\bar{\mathbf{x}}(T)$ is a fractional point in $\mathcal{M}$. Moreover, only part of $\bar{\mathbf{x}}(T)$ is available at each agent $i \in \mathcal{A}$. In what follows, we propose a distributed rounding procedure that without any communication among the agents, enables each agent $i \in \mathcal{A}$ to round its $\mathbf{x}_{ii}(T)$, and use this rounded probability membership vector to make its local strategy choice $\bar{\mathcal{R}}_i$ such that $\cup_{i\in\mathcal{A}}\bar{\mathcal{R}}_i = \bar{\mathcal{R}} \in \mathcal{I}$.

Let each agent $i \in \mathcal{A}$ initialize its local rounded membership vector $\mathbf{y}_{ii} \in \mathbb{R}^{|\mathcal{P}_i|}$ at $\mathbf{y}_{ii}(0) = \mathbf{x}_{ii}(T)$. Then, by virtue of Lemma 2, we have $\mathbf{y}_{ii}(0) \in [0,1]^{|\mathcal{P}_i|}$, $i \in \mathcal{A}$. Following a stochastic Pipage rounding procedure, each agent $i \in \mathcal{A}$ at each rounding iteration τ uniformly random selects two fractional elements $[\mathbf{y}_{ii}(\tau)]_p, [\mathbf{y}_{ii}(\tau)]_q$ of $\mathbf{y}_{ii}(\tau)$, i.e., $[\mathbf{y}_{ii}(\tau)]_p, [\mathbf{y}_{ii}(\tau)]_q \in (0,1)$, and performs the randomized swapping/update

$$\begin{cases} [\mathbf{y}_{ii}(\tau+1)]_p = [\mathbf{y}_{ii}(\tau)]_p - \delta_p(\tau), \\ [\mathbf{y}_{ii}(\tau+1)]_q = [\mathbf{y}_{ii}(\tau)]_q + \delta_p(\tau), \end{cases} \text{w.p. } \frac{\delta_q(\tau)}{\delta_p(\tau)+\delta_q(\tau)},$$
$$\begin{cases} [\mathbf{y}_{ii}(\tau+1)]_q = [\mathbf{y}_{ii}(\tau)]_q - \delta_q(\tau), \\ [\mathbf{y}_{ii}(\tau+1)]_p = [\mathbf{y}_{ii}(\tau)]_p + \delta_q(\tau), \end{cases} \text{w.p. } \frac{\delta_p(\tau)}{\delta_p(\tau)+\delta_q(\tau)}, \tag{19}$$

where $\delta_p(\tau) = \min([\mathbf{y}_{ii}(\tau)]_p, 1 - [\mathbf{y}_{ii}(\tau)]_q)$ and $\delta_q(\tau) = \min(1-[\mathbf{y}_{ii}(\tau)]_p, [\mathbf{y}_{ii}(\tau)]_q)$; see Fig. 1 for an illustration. Here, 'w.p.' stands for 'with probability of'. The following proposition whose proof is in Appendix D gives the convergence result of distributed Pipage rounding (19).

Proposition 5 *Starting from $\mathbf{y}_{ii}(0) = \mathbf{x}_{ii}(T)$, let each agent $i \in \mathcal{A}$ implement the rounding procedure (19). Then, $\mathbf{y}_{ii}(|\mathcal{P}_i|) \in \{0,1\}^{|\mathcal{P}_i|}$, and $\mathbf{1}.\mathbf{y}_{ii}(|\mathcal{P}_i|) = \kappa_i$. Moreover, $\bar{\mathbf{y}} = [\mathbf{y}_{11}(|\mathcal{P}_1|), \mathbf{y}_{22}(|\mathcal{P}_2|), \cdots, \mathbf{y}_{NN}(|\mathcal{P}_N|)]$ is a vertex of $\mathcal{M}$.*

It is worth noting that $\mathbf{1}.\mathbf{x}_{ii}(T) = \kappa_i$, guaranteed by Lemma 2 for our proposed algorithm (9), has a significant importance in enabling a rounding procedure without the necessity for coordination among the agents. As we discuss in the numerical example of Section 4, $\mathbf{1}.\mathbf{x}_{ii}(T) = \kappa_i$ is not the case for the distributed continuous greedy algorithm of [30], which uses an average consensus algorithm to coordinate the local probability membership choices of the agents.

Our distributed stochastic Pipage rounding procedure concludes by each agent $i \in \mathcal{A}$ choosing its suboptimal strategy set according to

$$\bar{\mathcal{R}}_i = \mathcal{R}_{\bar{\mathbf{y}}_i}, \quad \text{where} \tag{20}$$
$$\bar{\mathbf{y}}_i = [\mathbf{0}^\top_{|\mathcal{P}_1|\times 1}, \cdots, \mathbf{y}_{ii}(|\mathcal{P}_i|)^\top, \cdots, \mathbf{0}^\top_{|\mathcal{P}_N|\times 1}]^\top,$$

with $\mathbf{y}_{ii}(|\mathcal{P}_i|)$ obtained from (19), initialized at $\mathbf{y}_{ii}(0) = \mathbf{x}_{ii}(T)$. The following result, whose proof is given in Appendix E, shows that our proposed distributed rounding procedure (19) results in a strategy selection (20) that is loss-less in the expected value sense. That is, it results in not only a feasible selected strategy set but also strategies that are selected in a distributed way with no loss in the utility function compared to the fractional solution.

Theorem 6 (Utility evaluation after distributed Pipage rounding) *Let each agents $i \in \mathcal{A}$ choose its strategy set $\bar{\mathcal{R}}_i \subset \mathcal{P}_i$ according to (20). Let $\bar{\mathcal{R}} = \bigcup_{i\in\mathcal{A}}\bar{\mathcal{R}}_i$. Then,*

$$F(\bar{\mathbf{x}}(T)) \leq \mathbb{E}[f(\bar{\mathcal{R}})]. \tag{21}$$

3.2 Distributed implementation

Our proposed suboptimal solution to solve problem (2) consists of iterative propagation step (9a), and update step (9b), which requires local interaction between neighbors to exchange information. After T steps, once $\mathbf{x}_i(T)$ is obtained, each agent $i \in \mathcal{A}$ computes its suboptimal solution from (20) after running Pipage procedure (19) locally for at most $|\mathcal{P}_i|$ steps to compute $\mathbf{y}_{ii}(|\mathcal{P}_i|)$. In the propagation step agents should draw K_i samples of $\mathcal{R}_{\mathbf{x}_i} \subset \mathcal{P}$ to compute $\widetilde{\nabla F}(\mathbf{x}_i(t))$, requiring access to the elements of $\mathcal{P}$ corresponding to the none-zero elements of $\mathbf{x}_i(t)$.

In what follows, by relying on the properties of the updated local copies of the probability membership vector, we outline the information exchange that is needed for implementation of our distributed solution. The resulted implementation is summarized as the distributed multilinear-extension-based iterative greedy algorithm presented as Algorithm 1.

Observe that since in (10) we have $\mathbf{w} \in \mathcal{M}_i$, to carry out the propagation step (9a), each agent should only compute $[\widetilde{\nabla F}(\mathbf{x}_i(t))]_p$, $p \in \mathcal{P}_i$ from (12) using K_i samples of $\{\mathcal{R}_i^k(t)\}_{k=1}^{K_i}$, where each sample satisfies $q \in \mathcal{R}_i^k(t)$ with the probability of $[\mathbf{x}_i]_q$ for all $q \in \mathcal{P}$ for which $[\mathbf{x}_i]_q \neq 0$.

Algorithm 1 Discrete distributed implementation of the continuous greedy algorithm.

Init: $\mathcal{F}_1 \leftarrow \emptyset, \cdots, \mathcal{F}_N \leftarrow \emptyset,\ t \leftarrow 1,$
while $t \leq T$ **do**
  for $i \in \mathcal{A}$ **do**
    Draw K_i sample strategy sets, $\{\mathcal{R}_i^k\}_{k=1}^{K_i}$ such that $q \in \mathcal{R}_i^k$ w.p. α for all $(q, \alpha) \in \mathcal{F}_i$.
    for $p \in \mathcal{P}_i$ **do**
      Compute $[\widetilde{\nabla F}(\mathbf{x}_i(t))]_p \approx \mathbb{E}[f(\mathcal{R} \cup \{p\}) - f(\mathcal{R} \setminus \{p\})]$ using the sample strategy sets of step 4 via (12).
    end for
    compute $\{p_1^\star, \cdots, p_{\kappa_i}^\star\} \subset \mathcal{P}_i$ via (22)
    $\mathcal{F}_i^- \leftarrow \mathcal{F}_i \oplus \{(p_1^\star, \frac{1}{T})\} \oplus \cdots \oplus \{(p_{\kappa_i}^\star, \frac{1}{T})\}$
    Broadcast $\mathcal{F}_i^-$ to the neighbors $\mathcal{N}_i$.
    $\mathcal{F}_i \leftarrow \underset{j \in \mathcal{N}_i \cup \{i\}}{\mathrm{MAX}} \mathcal{F}_j^-$
  end for
  $t \leftarrow t + 1.$
end while
for $i \in \mathcal{A}$ **do**
  $\bar{\mathcal{R}}_i = \{\bar{p}_1, \cdots, \bar{p}_{\kappa_i}\} \leftarrow$ `DistStochPipage`$(\mathcal{F}_i)$
end for
Return $\bar{\mathcal{R}}_1, \cdots, \bar{\mathcal{R}}_N$

Algorithm 2 `DistStochPipage( )`

Input: $\mathcal{F}_i$
Init: $\bar{\mathcal{R}}_i = \emptyset$
while $|\bar{\mathcal{R}}_i| < \kappa_i$ **do**
  pick any $(\alpha_p, p), (\alpha_p, p) \in \mathcal{F}_i$ such that $p, q \in \mathcal{P}_i$
  Set: $\begin{cases} \delta_p = \min\{\alpha_p, 1 - \alpha_q\} \\ \delta_q = \min\{\alpha_q, 1 - \alpha_p\} \end{cases}$
  Update $\begin{cases} \begin{cases} \alpha_p \leftarrow \alpha_p - \delta_p \\ \alpha_q \leftarrow \alpha_q + \delta_p \end{cases} & \text{w.p. } \frac{\delta_q}{\delta_p + \delta_q} \\ \text{or} \\ \begin{cases} \alpha_q \leftarrow \alpha_q - \delta_q \\ \alpha_p \leftarrow \alpha_p + \delta_q \end{cases} & \text{w.p. } \frac{\delta_p}{\delta_p + \delta_q} \end{cases}$
  if $\alpha_p = 1$ then $\bar{\mathcal{R}}_i \leftarrow \bar{\mathcal{R}}_i \cup \{p\}$, $\mathcal{F}_i \leftarrow \mathcal{F}_i \setminus \{(\alpha_p, p)\}$
  if $\alpha_q = 1$ then $\bar{\mathcal{R}}_i \leftarrow \bar{\mathcal{R}}_i \cup \{q\}$, $\mathcal{F}_i \leftarrow \mathcal{F}_i \setminus \{(\alpha_q, q)\}$
end while
Return $\bar{\mathcal{R}}_i$

It follows from submodularity of f that $f(\mathcal{R}_i^k(t) \cup \{p\}) - f(\mathcal{R}_i^k(t)) \setminus \{p\}) \geq 0$. Thus, for any $p \in \mathcal{P}_i$ we have $[\widetilde{\nabla F}(\mathbf{x}_i(t))]_p \geq 0$. Consequently, one realization of $\widetilde{\mathbf{v}}_i(t)$ of optimization problem (10) is $\mathbf{1}_{\{p_1^\star, \cdots, p_{\kappa_i}^\star\}}$, where the set $\{p_1^\star, \cdots, p_{\kappa_i}^\star\} \subset \mathcal{P}_i$ is obtained from

$$\{p_1^\star, \cdots, p_{\kappa_i}^\star\} = \arg\max(\{\widetilde{\nabla F}(\mathbf{x}_i(t))\}_{p \in \mathcal{P}_i}, \kappa_i), \quad (22)$$

i.e., $\{p_1^\star, \cdots, p_{\kappa_i}^\star\} \subset \mathcal{P}_i$ corresponds to the k_i largest elements of $\{\widetilde{\nabla F}(\mathbf{x}_i(t))\}_{p \in \mathcal{P}_i}$.

Using $\widetilde{\mathbf{v}}_i(t) = \mathbf{1}_{\{p_1^\star, \cdots, p_{\kappa_i}^\star\}}$, it follows from (9a) that at each propagation step only the value of κ_i elements of $\mathbf{x}_i(t)$ corresponding to $\{p_1^\star, \cdots, p_{\kappa_i}^\star\}$ changes and increases by $\frac{1}{T}$.

Now let us define the local *information set* of each agent i at time step t as

$$\mathcal{F}_i(t) = \big\{(p, \alpha) \in \mathcal{P} \times [0, 1] \,\big|\, [\mathbf{x}_i(t)]_p \neq 0 \text{ and } \alpha = [\mathbf{x}_i(t)]_p\big\}. \quad (23)$$

$\mathcal{F}_i(t)$ is a set of couples representing which element $p \in \mathcal{P}$ has an associated non-zero probability membership vector element in $\mathbf{x}_i(t)$. Since $\mathbf{x}_i(0) = \mathbf{0}$, then $\mathcal{F}_i(0) = \emptyset$. Introduction of the information set $\mathcal{F}_i(t)$ provides a framework through which the agents only store and communicate the necessary information. Furthermore, it enables the agents to carry out their local computations using the available information in $\mathcal{F}_i(t)$.

As we discussed earlier, since we use $\widetilde{\mathbf{v}}_i(t) = \mathbf{1}_{\{p_1^\star, \cdots, p_{\kappa_i}^\star\}}$ the corresponding realization of propagation rule (9a) over the information set $\mathcal{F}_i(t)$ is

$$\mathcal{F}_i^-(t+1) = \mathcal{F}_i(t) \oplus \{(p_1^\star, \frac{1}{T})\} \oplus \cdots \oplus \{(p_{\kappa_i}^\star, \frac{1}{T})\}, \quad (24)$$

where the addition operator $\oplus$ is defined as follows.

Definition 7 *Given a set $\mathcal{F} \subset \mathcal{P} \times \mathbb{R}$ and a member $(p, \alpha) \in \mathcal{P} \times \mathbb{R}$, we define the addition operator $\oplus$ as $\mathcal{F}' = \mathcal{F} \oplus \{(p, \alpha)\}$ such that*

$$\mathcal{F}' = \begin{cases} \mathcal{F} \cup \{(p, \alpha)\} & (p, \gamma) \notin \mathcal{F}, \\ (\mathcal{F} \setminus \{(p, \gamma)\}) \cup \{(p, \gamma + \alpha)\} & (p, \gamma) \in \mathcal{F}. \end{cases}$$

Per definition operator $\oplus$ inserts $(p_j^\star, \frac{1}{T})$, $j \in \{1, \cdots, \kappa_i\}$ in agent i's information set if there exists no element $(p_j^\star, \alpha)$, $\alpha \neq 0$ in $F_i(t)$; otherwise, operator $\oplus$ pops out $(p_j^\star, \alpha)$ and replaces it with $(p_j^\star, \alpha + \frac{1}{T})$. Therefore, $\mathcal{F}_i^-(t+1)$ is consistent with the realization of $\mathbf{x}_i^-(t+1)$ through the membership probability vector to information set conversion relation (23).

Instead of the agents sharing $\mathbf{x}_i^-(t)$, $i \in \mathcal{A}$ with their neighbors, they can share their local information set with their neighboring agents and execute a max operation over their local and received information sets as

$$\mathcal{F}_i(t+1) = \underset{j \in \mathcal{N}_i \cup \{i\}}{\mathrm{MAX}} \mathcal{F}_j^-(t+1), \quad (25)$$

where the MAX operator is defined as follows.

Definition 8 *Given a collection of sets $\mathcal{F}_i \in \mathcal{P} \times \mathbb{R}$, $i \in \mathcal{A}$, we define the max-operation over these collection as $\underset{i \in \mathcal{A}}{\mathrm{MAX}} \mathcal{F}_i = \{(u, \gamma) \in \mathcal{P} \times \mathbb{R} | (u, \gamma) \in \bar{\mathcal{F}}$ s.t. $\gamma = \max_{(u,\alpha) \in \bar{\mathcal{F}}} \alpha\}$, where $\bar{\mathcal{F}} = \bigcup_{i \in \mathcal{A}} \mathcal{F}_i$.*

Consequently, through the membership probability vector to information set conversion relation (23), $\mathcal{F}_i(t+1)$ is consistent with a realization of $\mathbf{x}_i(t+1)$. Notice that

sharing $\mathcal{F}_i(t)$ not only enables sharing non-zero components of $\mathbf{x}_i^-(t+1)$ but also the corresponding strategies, which are what is needed to preform the next step of the algorithm. Moreover, because at each propagation step of the algorithm only, at most, κ_i zero elements of $\mathbf{x}^i(t)$ will become non-zero when $\mathbf{x}_i^-(t+1)$ is computed, the size of $\mathcal{F}_i$ at worst case grows linearly with $\sum_{j=1}^N \kappa_j$ after each update.

Finally, given the definition of $\mathcal{F}_i(t)$ in (23) and in light of Proposition 5, the stochastic rounding procedure (19) and (20) can be implemented according to Algorithm 2.

In light of the discussion above, Algorithm 1 gives our distributed multilinear extension based suboptimal solution for problem (2). The following theorem, whose proof is given in Appendix E establishes the optimality bound of $f(\bar{\mathcal{R}})$ where $\bar{\mathcal{R}} = \bigcup_{i\in\mathcal{A}}\{\bar{\mathcal{R}}_i\}$ is generated through the decentralized Algorithm 1.

Theorem 9 (Convergence guarantee and suboptimality gap of Algorithm 1) *Let $f : 2^{\mathcal{P}} \to \mathbb{R}_{\geq 0}$ be normalized, monotone increasing and submodular set function. Let $\mathcal{R}^\star$ to be the optimizer of problem (2). Following the distributed Algorithm 1, the admissible strategy set $\bar{\mathcal{R}}$ with probability of at least $1 - 2T\,n\,\mathrm{e}^{-\frac{1}{8T^2}\underline{K}}$, $\underline{K} = \min_{i\in\mathcal{A}} K_i$ satisfies*

$$\mathbb{E}[f(\bar{\mathcal{R}})] \geq \beta\, f(\mathcal{R}^\star),$$

where β is given in (18).

The constant approximation factor β is characterized in terms of the total curvature c of the utility function f. Curvature c represents a measure of the diminishing return of a set function. The curvature of $c = 0$ means that the function is modular, i.e., $f(\{p_1, p_2\}) = f(\{p_1\}) + f(\{p_2\})$, $p_1, p_2 \in \mathcal{P}$. We can see from (18) that when $c = 0$, $\beta = 1$, meaning that for modular functions our algorithm can find the optimal solution in finite time. On the other hand, $c = 1$ means that there is at least a member that adds no value to function f in a special circumstance. Whenever the total curvature is not known, it is rational to assume the worst case scenario and set $c = 1$.

Remark 10 (Extra communication for improved optimality gap) *Replacing the update step (9b) with $\mathbf{x}_i(t+1) = \mathbf{y}_i(d(\mathcal{G}))$ where $\mathbf{y}_i(0) = \mathbf{x}_i^-(t+1)$ and*

$$\mathbf{y}_i(m) = \max_{j\in\mathcal{N}_i\cup\{i\}} \mathbf{y}_j(m-1), \quad m \in \{1, \cdots, d(\mathcal{G})\},$$

i.e., starting with $\mathbf{x}_i^-(t+1)$ and recursively repeating the update step (9b) using the output of the previous recursion for $d(\mathcal{G})$ times, each agent $i \in \mathcal{A}$ arrives at $\mathbf{x}_i(t+1) = \bar{\mathbf{x}}(t+1)$ (recall Lemma 1). Hence, for this revised implementation, following the proof of Theorem 4, we observe that (48) is replaced by $\left|\frac{\partial F}{\partial [\mathbf{x}]_p}(\bar{\mathbf{x}}(t)) - \frac{\partial F}{\partial [\mathbf{x}]_p}(\mathbf{x}_i(t))\right| = 0$, which consequently,

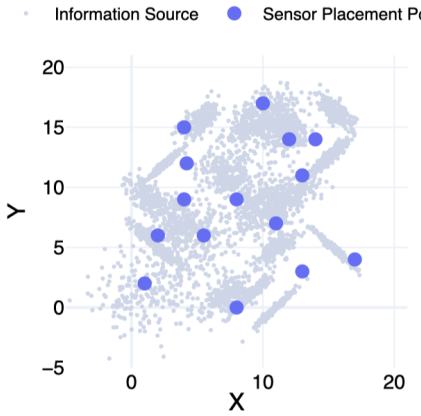

Fig. 2. Set of information sources $\mathcal{D}$ and set of sensor placement point $\mathcal{B}$.

leads to

$$\frac{1}{c}(1-\mathrm{e}^{-c})(1-(\frac{c\,\kappa}{2}+1)\frac{\kappa}{T})f(\mathcal{R}^\star) \leq F(\bar{\mathbf{x}}_{ii}(T)), \quad (26)$$

with the probability of at least $\left(\prod_{i\in\mathcal{A}}(1-2\mathrm{e}^{-\frac{1}{8T^2}K_j})^{|\mathcal{P}_i|}\right)^T$. This improved optimality gap is achieved by $(d(\mathcal{G})-1)T$ extra communication per agent. The optimality bound (26) is the same bound that is achieved by the centralized algorithm of [16]. To implement this revision, Algorithm 1's step 11 (equivalent to (24)) should be replaced by $\mathcal{F}_i = \mathcal{H}_i(d(\mathcal{G}))$, where $\mathcal{H}_i(0) = \mathcal{F}_i^-$, and

$$\mathcal{H}_i(m) = \underset{j\in\mathcal{N}_i\cup\{i\}}{\mathrm{MAX}}\, \mathcal{H}_j^-(m-1), \quad m \in \{1, \cdots, d(\mathcal{G})\}. \quad (27)$$

4 Numerical Example

We demonstrate our algorithm's performance using a multi-agent information harvesting problem. Consider a countable set of information sources $\mathcal{D} \subset \mathbb{R}^2$ that are spread in a two-dimensional area without any prior information on their spread density function. In the same area, a countable set of prespecified information retrieval points $\mathcal{B} \subset \mathbb{R}^2$ are available for placing information harvester devices. We assume that the information is best transferred from an information point $d \in \mathcal{D}$ to a harvester device dispatched at $b \in \mathcal{B}$ if the distance between b and d is minimized. Hence, for each information point $d \in \mathcal{D}$ the closest information retrieval point $b \in \mathcal{B}$ with a deployed device is assigned to harvest information.

Each agent $i \in \mathcal{A}$ is only able to deploy at most κ_i devices to its admissible deployment locations $\mathcal{B}_i \subset \mathcal{B}$, where $\mathcal{B}_1, \cdots, \mathcal{B}_N$ are not necessarily disjoint sets. To make the strategy set of the agents disjoint, we define the deployment strategy of each agent $i \in \mathcal{A}$ as $\mathcal{P}_i = \{(i, b) | b \in \mathcal{B}_i\}$. Note that if $b \in \mathcal{B}_i$ and $b \in \mathcal{B}_j$ then the strategies $(i, b) \in \mathcal{P}_i$ and $(j, b) \in \mathcal{P}_j$ will be placing one sensors from agent $i \in \mathcal{A}$ and one sensor from agent $j \in$

$\mathcal{A}$ at the placement location $b \in \mathcal{B}$. The goal of the agents $\mathcal{A}$ is to each choose a strategy set $\mathcal{R}_i \subset \mathcal{P}_i$, $|\mathcal{R}_i| \leq \kappa_i$ such that cumulative strategy of the team $\mathcal{R} = \bigcup_{i\in\mathcal{A}} \mathcal{R}_i$ results in smallest total distance of information sources to the deployed devices, i.e. minimizing

$$L(\mathcal{R}) = \sum_{d\in\mathcal{D}} \min_{(i,b)\in\mathcal{R}} \|d - b\| . \tag{28}$$

Taking a *phantom* placement location b_0 to be a random point in $\mathbb{R}^2$, the problem can be reformulated as problem (2) where the utility function to maximize is

$$f(\mathcal{R}) = L(\{b_0\}) - L(\mathcal{R} \cup \{b_0\}). \tag{29}$$

This utility function (29) measures the decrease in the loss associated with the active set versus the loss associated with just the phantom placement location and maximizing this function is equivalent to minimizing the loss (28). It is known that the utility function (29) is submodular and monotone increasing [32].

For our numerical study, we consider 4500 information sources spread in a two-dimensional field where there are 15 deployment locations $\mathcal{B} = \{b_1, \cdots, b_{15}\}$, see Fig. 2. We consider a set of ten agents $\mathcal{A} = \{1, \cdots, 10\}$ whose goal is to deploy $\kappa_1 = 2$, $\kappa_2 = 2$, $\kappa_3 = 2$, $\kappa_4 = 2$, $\kappa_5 = 2$, $\kappa_6 = 1$, $\kappa_7 = 1$, $\kappa_6 = 1$, $\kappa_9 = 1$, $\kappa_{10} = 1$ devices at $\mathcal{B}_1 = \{b_1, b_2, b_3, b_5, b_6\}$, $\mathcal{B}_2 = \{b_4, b_5, b_6, b_8, b_9\}$, $\mathcal{B}_3 = \{b_7, b_8, b_9, b_{11}, b_{12}\}$, $\mathcal{B}_4 = \{b_{10}, b_{11}, b_{12}, b_{14}, b_{15}\}$, $\mathcal{B}_5 = \{b_2, b_3, b_{13}, b_{14}, b_{15}\}$, $\mathcal{B}_6 = \{b_2, b_3\}$, $\mathcal{B}_7 = \{b_5, b_6\}$, $\mathcal{B}_8 = \{b_8, b_9\}$, $\mathcal{B}_9 = \{b_{11}, b_{12}\}$, and $\mathcal{B}_{10} = \{b_{14}, b_{15}\}$. Hence the disjoint strategy sets are defined as $\mathcal{P}_1 = \{(1, b_1), (1, b_2), (1, b_3), (1, b_5), (1, b_6)\}$, $\mathcal{P}_2 = \{(2, b_4), (2, b_5), (2, b_6), (2, b_8), (2, b_9)\}$, $\mathcal{P}_3 = \{(3, b_7), (3, b_8), (3, b_9), (3, b_{11}), (3, b_{12})\}$, $\mathcal{P}_4 = \{(4, b_{10}), (4, b_{11}), (4, b_{12}), (4, b_{14}), (4, b_{15})\}$, $\mathcal{P}_5 = \{(5, b_2), (5, b_3), (5, b_{13}), (5, b_{14}), (5, b_{15})\}$, $\mathcal{P}_6 = \{(6, b_2), (6, b_3)\}$, $\mathcal{P}_7 = \{(7, b_5), (7, b_6)\}$, $\mathcal{P}_8 = \{(8, b_8), (8, b_9)\}$, $\mathcal{P}_9 = \{(9, b_{11}), (9, b_{12})\}$, and $\mathcal{P}_{10} = \{(10, b_{14}), (10, b_{15})\}$. Although, the general form of the problem is NP-hard, we have designed our numerical example such the optimal solution is trivial. Recall that to maximize the utility (29) of the group the deployed sensors must be placed such that the distance between the information sources and deployed sensors is minimized. Since there are $|\mathcal{B}| = 15$ deployment locations and $\sum_{j=1}^{5} \kappa_j = 15$ sensors to deploy, the optimal solution is to place the deployed sensors to occupy all the sensor-placement locations. This deployment scenario is only feasible if agent 1 deploys its $\kappa_1 = 2$ devices at locations $\{b_1, b_2\} \subset \mathcal{B}_1$, i.e. $\mathcal{R}_1 = \{(1, b_1), (1, b_2)\}$, agent 2 deploys its $\kappa_2 = 2$ devices at locations $\{b_4, b_5\} \subset \mathcal{B}_2$, i.e. $\mathcal{R}_2 = \{(2, b_4), (2, b_5)\}$, agent 3 deploys its $\kappa_3 = 2$ devices at locations $\{b_7, b_8\} \subset \mathcal{B}_3$, i.e. $\mathcal{R}_3 = \{(3, b_7), (3, b_8)\}$, agent 4 deploys its $\kappa_4 = 2$ devices at locations $\{b_{10}, b_{11}\} \subset \mathcal{B}_4$, i.e. $\mathcal{R}_4 = \{(4, b_{10}), (4, b_{11})\}$, agent 5 deploys its $\kappa_5 = 2$ devices at locations $\{b_{13}, b_{14}\} \subset \mathcal{B}_5$, i.e. $\mathcal{R}_5 = \{(5, b_{13}), (5, b_{141})\}$, agent 6 deploys its $\kappa_6 = 1$ devices at locations $\{b_3\} \subset \mathcal{B}_6$, i.e. $\mathcal{R}_6 = \{(6, b_3)\}$, agent 7 deploys its $\kappa_7 = 1$ devices at locations $\{b_6\} \subset \mathcal{B}_7$, i.e. $\mathcal{R}_7 = \{(7, b_6)\}$, agent 8 deploys its $\kappa_8 = 1$ devices at locations $\{b_9\} \subset \mathcal{B}_8$, i.e. $\mathcal{R}_8 = \{(8, b_9)\}$, agent 9 deploys its $\kappa_9 = 1$ devices at locations $\{b_{12}\} \subset \mathcal{B}_9$, i.e. $\mathcal{R}_9 = \{(9, b_{12})\}$, and agent 10 deploys its $\kappa_{10} = 1$ devices at locations $\{b_{15}\} \subset \mathcal{B}_{10}$, i.e. $\mathcal{R}_{10} = \{(10, b_{15})\}$. This setting allows us to compare the outcome of the suboptimal solutions against the optimal one. It is interesting to notice that the total curvature of utility function (29) is $c = 1$. This is because if we take the strategy set $\mathcal{S} = \{(1, b_1), (2, b_2)\}$ and the strategy $p = (6, b_2)$, since, the strategies $(1, b_2)$ and $(6, b_2)$ place a sensor at the same location then the utility equation (29) results in $\Delta_f(p|\mathcal{S}) = 0$. Thus, given the definition of the total curvature (16), we obtain $c = 1$.

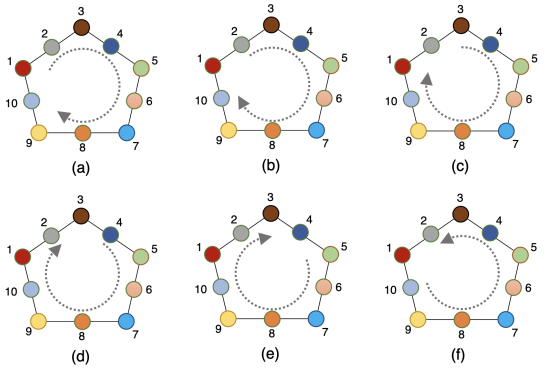

Fig. 3. The communication graph of the agents is a ring graph. Six possible communication sequences to implement a sequential greedy algorithm are shown.

Let the communication topology of the agents be an undirected ring graph, see Fig. 3. First, we solve the problem using Algorithm 1. We generate 50 deployment scenarios, each corresponding to a set of 4500 randomly generated information sources and 15 sensor locations. The results of implementing Algorithm 1 for the different number of samples K_i (all agents use the same number of samples) and iteration number $T = 50$ is shown in Fig. 4. Observe that using a modest number of iterations $T = 50$ and a modest number of samples $K_i = 1000$ Algorithm 1 finds almost the optimal solution in terms of occupying the placement locations; the average number of locations occupied is 14.3. For this setting, the expected outcome of Algorithm 1 over the 50 placement scenarios considered, measured by utility function (29), is at 0.96 of the optimal solution. The run-time of the algorithm, implemented using NumPy library, for each agent is approximately 40 seconds on a computing device with Intel(R) Xeon(R) CPU @ 2.30GHz and 13GB RAM.

Comparison with sequential greedy algorithm: Next, we solve the problem using a decentralized *massage-passing* sequential greedy algorithm following [7,33]. That is, we choose a route `SEQ` that visits all the agents on the communication graph. We then make the agents perform

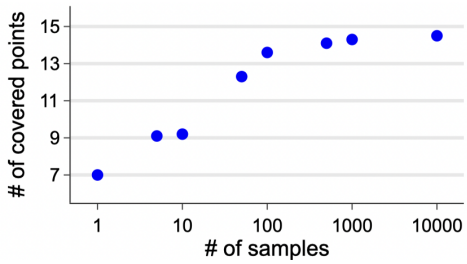

Fig. 4. The average number of sensor placement points covered by deployed sensors when Algorithm 1 is implemented by different sample numbers and $T = 50$.

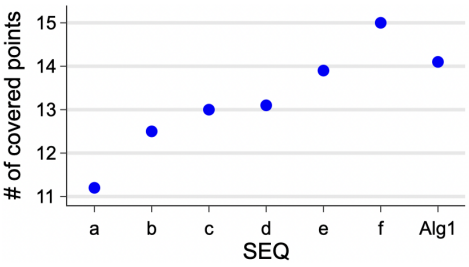

Fig. 5. The average number of covered placement points over 50 different randomly generated information sources and sensor placement locations. The x-axis corresponds to the six `SEQ` in Fig. 3(a)-(f) and Algorithm 1 denoted by Alg1. The y-axis corresponds to the average number of sensor placement points covered by the deployed sensors.

the sequential greedy algorithm by sequential message-passing according to `SEQ`. Fig. 3(a)-(f) gives 6 of the possible `SEQ` depicted by the semi-circular arrow inside the networks. As Fig. 5 shows the performance (measured by the number of occupied placement locations) of the sequential greedy algorithm depends on what `SEQ` agents follow, with `SEQ` of Fig. 3(a) delivering the worst performance. Moreover, the performance measured by utility function (29) for `SEQ` (a),(b),(c),(d),(e), and (f) are respectively 0.78, 0.80, 0.82, 0.83, 0.92, 0.99 of the optimal utility value. We can attribute this inconsistency to the heterogeneity of the agents' sensor numbers. When agents with a larger number of choices pick first, this limits the options of the agents with a lower number of sensors available. However, the performance of Algorithm 1 is regardless of any particular path on the graph since, through its iterative process, the agents get the chance to readjust their choices, see Fig. 6 for a deployment outcome via Algorithm 1 and the sequential greedy algorithm. Intuitively, this explains the better optimality gap of the continuous greedy algorithm over the sequential greedy algorithm. The sequential greedy algorithm has a run-time of less than 1 second for each agent on a device with Intel(R) Xeon(R) CPU @ 2.30GHz and 13GB RAM, which is significantly less than 40 seconds that we reported for our proposed Algorithm 1 using $K_i = 1000$ and $T = 50$. Even though computationally efficient, as we discussed in the introduction, the downsides of the sequential greedy algorithm are in its worse optimality gap, the overhead associated with identifying the message-passing sequence, and the dependence of the results on the message-passing sequence [21].

Comparison with the average consensus based algorithm of [30] : we compare our proposed Algorithm 1, which is based on a maximum consensus communication to the algorithm proposed in [30], which is based on an average consensus communication. Since the algorithm of [30] is designed for when agents choose only one strategy each, we carry out this study for $\kappa_i = 1$ for $i \in \{1, 2, \cdots, 10\}$. Moreover, since no distributed rounding procedures are proposed in [30], we implemented a *central* rounding algorithm based on the contention resolution schemes [34] to achieve a rounded solution from the relaxed solution. Our analysis shows that the two algorithms do not have any significant performance or running time advantage over one another in finding the relaxed solution. Specifically, the given number of iteration $T = 50$ and the given number of samples $K_i = 1000$ (same for all agents) Algorithm 1 and the algorithm in [30] yield the final strategy set $\bar{\mathcal{R}}$ such that $\mathbb{E}[f(\bar{\mathcal{R}})] \geq 0.93 f(\mathcal{R}^\star)$ and $\mathbb{E}[f(\bar{\mathcal{R}})] \geq 0.91 f(\mathcal{R}^\star)$ respectively. Here, the optimal solution was computed by the brute force search. Moreover the running time for both of the algorithms was approximately 40 seconds per agent.

However, we focus our discussion on the trajectory of $\mathbf{x}_i(t)$ on both of Algorithm 1 and the algorithm in [30] to further discuss an inherent characteristic of $\mathbf{x}_i(T)$ resulted from Algorithm 1 which greatly facilitates a distributed rounding procedure. The main difference between the two algorithms however is in how $\mathbf{x}_i(T)$ of each agent $i \in \{1, 2, \cdots, 10\}$ is placed with respect to the edges of the matroid polytope $\mathcal{M}$. Recall that $\mathbf{1}.\mathbf{x}_{ii}(T) = \kappa_i = 1$ is of great importance for the rounding procedure. Let

$$D(t) = \sum\nolimits_{i \in \mathcal{A}} (\mathbf{1}.\mathbf{x}_{ii}(t) - 1)\mathbb{1}((\mathbf{1}.\mathbf{x}_{ii}(t) - 1) \geq 0), \quad (30)$$

where $\mathbb{1} : \mathbb{R} \to \{0, 1\}$ is the indicator function. The value of $D(T)$ shows the deviation of the local component of the membership probability vector $\mathbf{x}_{ii}(T)$ from the edges of matroid polytope $\mathcal{M}$. Figure 7 shows this value for the different numbers of iterations for the two algorithms; Algorithm 1, as ensured by Lemma 2, results in $D(T) = 0$ for any choice of iteration number T. However, this is not the case for Algorithm of [30]. As the results in Figure 7 shows this algorithm seems to satisfy $D(T) = 0$ as the number of the iteration increases. Figure 8 compares the value of $D(t)$ of Algorithm 1 and the algorithm of [30] for $t \in [0, T]$ over 10 different instances of the utility maximization problem (29). The former algorithm is based on maximum consensus and $D(t)$ for this algorithm converges to 0, as predicted by Lemma 2. The latter algorithm is based on average consensus and $D(t)$ converges to a number greater than 0, meaning that $\mathbf{x}_i(T)$ of some of the agents are outside $\mathcal{M}$.

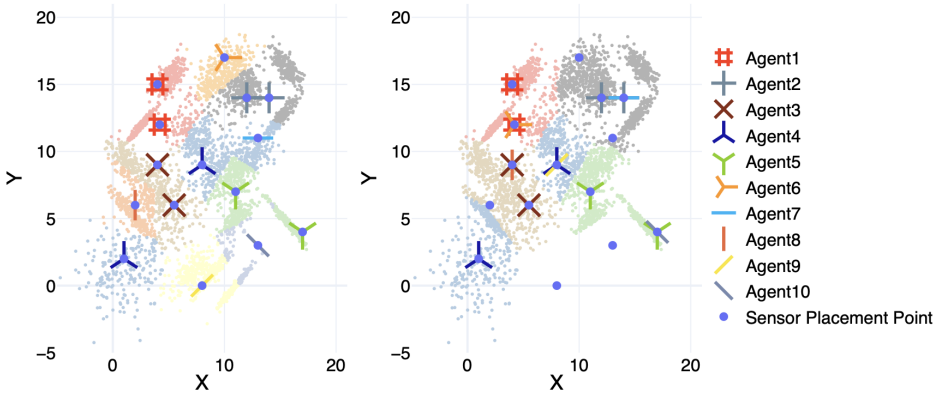

Fig. 6. The left figure shows an instance of the placement result for Algorithm 1 and the right figure shows the placement results for the Sequential Greedy of SEQ (a). Algorithm 1 is able to place the sensors such that all of the placement locations are occupied while the Sequential Greedy of SEQ (a) leaves out three unoccupied placement locations.

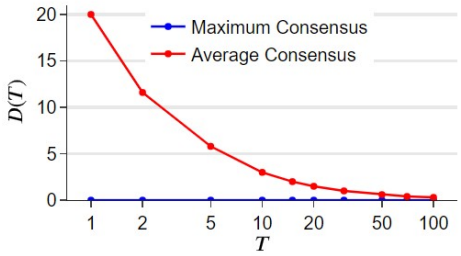

Fig. 7. The deviation of probability vectors $\mathbf{x}_i(T)$, $i \in \mathcal{A}$ from the convex hull $\mathcal{M}$ for average consensus and maximum consensus communication protocols.

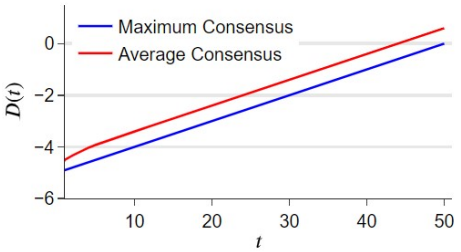

Fig. 8. The value of $D(t)$ calculated using $\mathbf{x}_i(t)$, $i \in \mathcal{A}$ for average consensus and maximum consensus communication protocols. The value of $D(t)$ was calculated by running Algorithm 1 and the algorithm in [30] with $\kappa_i = 1$, $i \in \mathcal{A}$ over 10 different instances of the utility maximization problem (29).

5 Conclusion

We proposed a distributed suboptimal algorithm to solve the problem of maximizing a monotone increasing submodular set function subject to a partition matroid constraint when agents communicate over a connected graph. Our problem of interest was motivated by optimal multi-agent sensor placement problems in discrete space. Our algorithm was a practical decentralization of a multilinear extension-based algorithm that achieves $\frac{1}{c}(1 - \mathrm{e}^{-c} - O(1/T))$ optimally gap, which is an improvement over $\frac{1}{1+c}$ optimality gap that the well-known sequential greedy algorithm achieves. Our algorithm included a distributed continuous greedy algorithm followed by a local rounding procedure that required no inter-agent communication. Through a numerical study, we compared the outcome obtained by our proposed algorithm with a decentralized sequential greedy algorithm that is constructed from assigning a priority sequence to the agents. We showed that the outcome of the sequential greedy algorithm is inconsistent and depends on the sequence. However, our algorithm's outcome, due to its iterative nature intrinsically tended to be consistent, which also explains its better optimally gap over the sequential greedy algorithm. We also compared our algorithm to an existing distributed average consensus-based continuous greedy algorithm. We showed that the main advantage of our proposed algorithm is its strong guarantee of reaching the edges of the constraint set's matroid polytope by all agents in finite time, which is of significance in the Pipage type rounding procedures. Our future work is to study the robustness of our proposed algorithm to message dropout.

References

[1] N. Mehr and R. Horowitz, "A submodular approach for optimal sensor placement in traffic networks," in *American Control Conference*, pp. 6353–6358, 2018.

[2] T. Summers, F. Cortesi, and J. Lygeros, "On submodularity and controllability in complex dynamical networks.," *IEEE Transactions on Control of Network Systems*, vol. 3, no. 1, pp. 91–101, 2016.

[3] J. Qin, I. Yang, and R. Rajagopal, "Submodularity of storage placement optimization in power networks," *IEEE Tran. on Automatic Control*, vol. 64, no. 8, pp. 3268–3283, 2019.

[4] M. Bucciarelli, S. Paoletti, E. Dall'Anese, and A. Vicino, "On the greedy placement of energy storage systems in distribution grids," in *American Control Conference*, 2020.

[5] S. T. Jawaid and S. Smith, "Submodularity and greedy algorithms in sensor scheduling for linear dynamical systems," *Automatica*, vol. 61, pp. 282–288, 2015.

[6] Z. Liu, A. Clark, P. Lee, L. Bushnell, D. Kirschen, and R. Poovendran, "Towards scalable voltage control in smart grid: A submodular optimization approach," in *ACM/IEEE 7th International Conference on Cyber-Physical Systems*, 2016.

[7] N. Rezazadeh and S. S. Kia, "A sub-modular receding horizon solution for mobile multi-agent persistent monitoring," *Automatica*, vol. 127, p. 109460, 2021.

[8] A. Krause and D. Golovin, "Submodular function maximization," in *Tractability: Practical Approaches to Hard Problems* (L. Bordeaux, Y. Hamadi, and P. Kohli, eds.), pp. 71–104, Cambridge, UK: Cambridge University Press, 2014.

[9] A. Schrijver, "A combinatorial algorithm minimizing submodular functions in strongly polynomial time," *Journal of Combinatorial Theory, Series B*, vol. 8, p. 346–355, 2000.

[10] L. F. S. Fujishige and S. Iwata, "A combinatorial, strongly polynomial-time algorithm for minimizing submodular functions," *Journal of the ACM*, vol. 48, no. 4, pp. 761–777, 2001.

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

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

[13] G. L. Nemhauser and L. A. Wolsey, "Best algorithms for approximating the maximum of a submodular set function," *Math. Operations Research*, vol. 3, no. 3, pp. 177–188, 1978.

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

[15] J. Vondrák, "Optimal approximation for the submodular welfare problem in the value oracle model," in *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pp. 67–74, 2008.

[16] J. Vondrák, "Submodularity and curvature: The optimal algorithm (combinatorial optimization and discrete algorithms)," *Research Institute for Mathematical Sciences Kyoto University*, vol. 23, pp. 253–266, 2010.

[17] A. A. Bian, B. Mirzasoleiman, J. Buhmann, and A. Krause, "Guaranteed
non-convex optimization: Submodular maximization over continuous domains," in *Artificial Intelligence and Statistics*, pp. 111–120, 2017.

[18] A. Mokhtari, H. Hassani, and A. Karbasi, "Stochastic conditional gradient methods: From convex minimization to submodular maximization," *Journal of Machine Learning Research*, vol. 21, no. 105, pp. 1–49, 2020.

[19] O. Sadeghi and M. Fazel, "Online continuous dr-submodular maximization with long-term budget constraints," in *International Conference on Artificial Intelligence and Statistics*, pp. 4410–4419, 2020.

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

[21] R. Konda, D. Grimsman, and J. Marden, "Execution order matters in greedy algorithms with limited information," *arXiv preprint arXiv:2111.09154*, 2021.

[22] K. Wei, R. Iyer, and J. Bilmes, "Fast multi-stage submodular maximization," in *International conference on machine learning*, pp. 1494–1502, PMLR, 2014.

[23] B. Mirzasoleiman, A. Karbasi, R. Sarkar, and A. Krause, "Distributed submodular maximization: Identifying representative elements in massive data," in *Advances in Neural Information Processing Systems*, pp. 2049–2057, 2013.

[24] B. Mirzasoleiman, M. Zadimoghaddam, and A. Karbasi, "Fast distributed submodular cover: Public-private data summarization," in *Advances in Neural Information Processing Systems*, pp. 3594–3602, 2016.

[25] R. Kumar, B. Moseley, S. Vassilvitskii, and A. Vattani, "Fast greedy algorithms in mapreduce and streaming," *ACM Transactions on Parallel Computing*, vol. 2, no. 3, pp. 1–22, 2015.

[26] P. S. Raut, O. Sadeghi, and M. Fazel, "Online dr-submodular maximization with stochastic cumulative constraints," *arXiv preprint arXiv:2005.14708*, 2020.

[27] A. Mokhtari, H. Hassani, and A. Karbasi, "Decentralized submodular maximization: Bridging discrete and continuous settings," in *International Conference on Machine Learning*, pp. 3616–3625, 2018.

[28] J. Xie, C. Zhang, Z. Shen, C. Mi, and H. Qian, "Decentralized gradient tracking for continuous dr-submodular maximization," in *International Conference on Artificial Intelligence and Statistics*, pp. 2897–2906, 2019.

[29] L. Ye and S. Sundaram, "Distributed maximization of submodular and approximately submodular functions," in *IEEE Conference on Decision and Control*, pp. 2979–2984, 2020.

[30] A. Robey, A. Adibi, B. Schlotfeldt, J. G. Pappas, and H. Hassani, "Optimal algorithms for submodular maximization with distributed constraints," *Learning for Dynamics and Control*, pp. 150–162, 2021.

[31] A. Ageev and M. Sviridenko, "Pipage rounding: A new method of constructing algorithms with proven performance guarantee," *Journal of Combinatorial Optimization*, vol. 8, no. 3, pp. 307–328, 2004.

[32] R. Gomes and A. Krause, "Budgeted nonparametric learning from data streams," in *ICML*, 2010.

[33] B. Gharesifard and S. Smith, "Distributed submodular maximization with limited information," *IEEE transactions on control of network systems*, vol. 5, no. 4, pp. 1635–1645, 2017.

[34] C. Chekuri, J. Vondrák, and R. Zenklusen, "Submodular function maximization via the multilinear relaxation and contention resolution schemes," *SIAM Journal on Computing*, vol. 43, no. 6, pp. 1831–1879, 2014.

[35] W. Hoeffding, "Probability inequalities for sums of bounded random variables," in *The Collected Works of Wassily Hoeffding*, pp. 409–426, Springer, 1994.

Appendix

In the following sections, we outline the necessary preliminary results and also the proofs of our technical results.

A Central continuous greedy algorithm

As [16] shows, the constrained gradient ascent

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{v}(\mathbf{x}) \;\text{ where }\; \mathbf{v}(\mathbf{x}) = \arg\max_{\mathbf{w}\in\mathcal{M}}(\mathbf{w}.\nabla F(\mathbf{x})), \quad (31)$$

initialized at $\mathbf{x}(0)=\mathbf{0}$ to solve the relaxed problem (8), can lead to a suboptimal solution for problem (2). Since $\mathcal{M}$ is convex and $\mathbf{x}(0)=\mathbf{0}\in\mathcal{M}$, the trajectory $t\mapsto \mathbf{x}$ of (31) belongs to $\mathcal{M}$ for $t\in[0,1]$. The following Lemma provides an essential property of the multilinear extended function F which can be used in quantifying the optimality gap of gradient ascent solver (31).

Lemma 11 (See [16]) *Consider the set value optimization problem* (2). *Suppose $f:\mathcal{P}\to\mathbb{R}_{\geq 0}$ is an increasing and submodular set function with curvature c whose multilinear extension is $F:[0,1]^n\to\mathbb{R}_{\geq 0}$. Let $\mathcal{R}^\star$ be the maximizer of* (2). *Then,*

$$\mathbf{1}_{\mathcal{R}^\star}.\nabla F(\mathbf{x}) \geq f(\mathcal{R}^\star) - c\,F(\mathbf{x}), \quad \forall \mathbf{x}\in\mathcal{M}.$$

Due to Lemma 11, ascent direction in (31) satisfies

$$\frac{\mathrm{d}F}{\mathrm{d}t} = \mathbf{v}(\mathbf{x}).\nabla F(\mathbf{x}) \geq f(\mathcal{R}^\star) - c\,F(\mathbf{x}). \quad (32)$$

From (32), Vondrak [16] shows that (31) results in $F(\mathbf{x}(1))\geq \frac{1}{c}(1-\mathrm{e}^{-c})\,f(\mathcal{R}^\star)$.

B Stochastic estimation of the relaxed functions' gradient

The stochastic interpretation (4) of the multilinear-extension and its derivatives leads to empirical estimation of $\nabla F(\mathbf{x}(t))$ as the equation given by (12). The Chernoff-Hoeffding's inequality can be used to quantify the quality of these estimates given the number of samples.

Theorem 12 (Chernoff-Hoeffding's inequality [35]) *Consider a set of K independent random variables $X_1,...,X_K$ where $a<X_i<b$. Let $S_K=\sum_{i=1}^K X_i$, then*

$$\mathbb{P}[|S_K-\mathbb{E}[S_K]|>\delta] < 2\,\mathrm{e}^{-\frac{2\delta^2}{K(b-a)^2}},$$

for any $\delta\in\mathbb{R}_{\geq 0}$.

The following lemma, whose proof relies on the Chernoff-Hoeffding's inequality, can quantify the quality of estimating the gradient of a multilinear extension function by sampling from the ground set.

Lemma 13 *Consider the set value optimization problem* (2). *Suppose $f:\mathcal{P}\to\mathbb{R}_{\geq 0}$ is an increasing and submodular set function and consider its multilinear extension $F:[0,1]^n\to\mathbb{R}_{\geq 0}$. Let $\widetilde{\nabla F}(\mathbf{x})$ be the estimate of $\nabla F(\mathbf{x})$ that is calculated by taking $K\in\mathbb{Z}_{>0}$ samples of set $\mathcal{R}_{\mathbf{x}}$ according to membership probability vector $\mathbf{x}$. Then,*

$$\left|\left[\widetilde{\nabla F}(\mathbf{x})\right]_p - \frac{\partial F}{\partial[\mathbf{x}]_p}(\mathbf{x})\right| \geq \frac{1}{2T}f(\mathcal{R}^\star), \quad p\in\{1,\cdots,n\}, \quad (33)$$

with the probability of at least $2\mathrm{e}^{-\frac{1}{8T^2}K}$, for any $T\in\mathbb{Z}_{>0}$.

Proof Define the random variable

$$X=\Big((f(\mathcal{R}_{\mathbf{x}}\cup\{p\})-f(\mathcal{R}_{\mathbf{x}}\backslash\{p\}))-\frac{\partial F}{\partial x_p}(\mathbf{x})\Big)\Big/f(\mathcal{R}^\star),$$

and assume that we take K samples from $\mathcal{R}_{\mathbf{x}}$ to construct $\{X_k\}_{k=1}^K$ realization of X. Since f is a submodular function, then we have $(f(\mathcal{R}_{\mathbf{x}}\cup\{p\})-f(\mathcal{R}_{\mathbf{x}}\backslash\{p\}))\leq f(\mathcal{R}^\star)$. Consequently using equation (5), we conclude that $0\leq X\leq 1$. Hence, using Theorem 12, we have $\left|\sum_{k=1}^K X_k-\mathbb{E}[X]\right|\geq\frac{1}{2T}K$ with the probability of at least $2\mathrm{e}^{-\frac{1}{8T^2}K}$. Hence, the estimation accuracy of $\nabla F(\mathbf{x})$, is given by $\left|\left[\widetilde{\nabla F}(\mathbf{x})\right]_p-\frac{\partial F}{\partial[\mathbf{x}]_p}(\mathbf{x})\right|\geq\frac{1}{2T}f(\mathcal{R}^\star)$, $p\in\{1,\cdots,n\}$ with the probability of at least $2\mathrm{e}^{-\frac{1}{8T^2}K}$. □

C Properties of the first and the second order derivatives of the multilinear extension

In this section we derive some auxiliary results on the first and the second order derivatives of the multilinear extension F which are going to be used directly in the proof of the main theorems.

Lemma 14 (First and second derivatives of the multilinear extension) *Let $f:2^{\mathcal{P}}\to\mathbb{R}_{\geq 0}$, $\mathcal{P}=\{1,\cdots,n\}$, be increasing and submodular set function with curvature c, and the multinear extension function $F(\mathbf{x})$ defined in* (3). *Then, $\frac{\partial F}{\partial[\mathbf{x}]_p}\geq(1-c)f(p)$ for all $p\in\mathcal{P}$ and $\mathbf{x}\in[0,1]^n$. Moreover, $-cf(\mathcal{R}^\star)\leq\frac{\partial^2 F}{\partial[\mathbf{x}]_p[\mathbf{x}]_q}\leq 0$ for all $p,q\in\mathcal{P}$ and $\mathbf{x}\in[0,1]^n$.*

Proof The derivative of $F(x)$ can be written as

$$\frac{\partial F}{\partial[\mathbf{x}]_p}(\mathbf{x}) = \Delta_f(p\,|\mathcal{R}_{\mathbf{x}}\setminus\{p\}). \quad (34)$$

Furthermore, by the definition of the total curvature (16) we can write $c\geq 1-\frac{\Delta_f(p|\mathcal{R}_{\mathbf{x}}\setminus p)}{f(p)}$, and by conjunction with equation (34), we have $\frac{\partial F}{\partial[\mathbf{x}]_p}\geq(1-c)f(p)$ which proves the first part of Lemma. Since $p\notin\mathcal{R}_{\mathbf{x}}\cup\{q\}\backslash\{p\}$, therefore by the definition of the total curvature (16) we can write

$$(1-c)f(\{p\})\leq\Delta_f(p|\mathcal{R}_{\mathbf{x}}\cup\{q\}\setminus\{p_i\})\leq f(\{p\}). \quad (35)$$

Moreover, Since $p \notin \mathcal{R}_{\mathbf{x}} \setminus \{p,q\}$, therefore by the definition of the total curvature (16) we can write

$$(1-c)f(\{p\}) \leq \Delta_f(p|\mathcal{R}_{\mathbf{x}} \setminus \{p,q\}) \leq f(\{p\}). \quad (36)$$

Knowing that $\Delta_f(p|\mathcal{R}_{\mathbf{x}} \cup \{q\} \setminus \{p\}) = f(\mathcal{R}_{\mathbf{x}} \cup \{p,q\}) - f(\mathcal{R}_{\mathbf{x}} \cup \{q\} \setminus \{p\})$ and $\Delta_f(p|\mathcal{R}_{\mathbf{x}} \setminus \{p,q\}) = f(\mathcal{R}_{\mathbf{x}} \cup \{p\}) \setminus \{q\}) - f(\mathcal{R}_{\mathbf{x}} \setminus \{p,q\})$, the definition of second order derivative of F (6), we can be written as

$$\frac{\partial F}{\partial [\mathbf{x}]_p \partial [\mathbf{x}]_q} = \mathbb{E}[\Delta_f(p|\mathcal{R}_{\mathbf{x}} \cup \{q\} \setminus \{p\}) - \Delta_f(p|\mathcal{R}_{\mathbf{x}} \setminus \{p,q\})]. \quad (37)$$

Putting (35) and (36) and (37) together in conjunction with submodular property of f results in $-cf(\{p\}) \leq \frac{\partial^2 F}{\partial [\mathbf{x}]_q \partial [\mathbf{x}]_q} \leq 0$. Knowing that $f(\{p\}) \leq f(\mathcal{R}^\star)$ results in proving the second part of Lemma. □

Lemma 15 (Directional Convexity) *Let $f : 2^{\mathcal{P}} \to \mathbb{R}_{\geq 0}$, $\mathcal{P} = \{1, \cdots, n\}$, be monotone increasing and submodular set function with a multinear extension function $F(\mathbf{x})$ defined in* (3)*. Then, for any given $\mathbf{x} \in [0,1]^n$ and $\mathbf{w} \in \{-1, 0, 1\}^n$ where $w_p = 1$, $w_q = -1$ and $w_l = 0$, $l \in \{1, \cdots, n\} \setminus \{p,q\}$ for some $p, q \in \{1, \cdots, n\}$, $F(\mathbf{x} + \lambda \mathbf{w}) : \mathbb{R} \to \mathbb{R}_{\geq 0}$ is a convex function of λ.*

Proof Defining the vector $\mathbf{w} \in \mathbb{R}^n$ and $w_p = 1$, $w_q = -1$ and $w_l = 0$, $l \neq p, q$, then the multilinear extension of set function f in the direction of $\mathbf{w}$ is defined as

$$\begin{aligned} F(\mathbf{x} + \lambda \mathbf{w}) = & \\ \sum_{\mathcal{R} \subset \mathcal{P} \setminus \{p,q\}} & f(\{p\} \cup \mathcal{R})([\mathbf{x}]_p + \lambda)(1 - ([\mathbf{x}]_q - \lambda))\mathbb{P}(\mathcal{R}) + \\ & f(\{q\} \cup \mathcal{R})(1 - ([\mathbf{x}]_p + \lambda))([\mathbf{x}]_q - \lambda)\mathbb{P}(\mathcal{R}) + \\ & f(\mathcal{R})(1 - ([\mathbf{x}]_p + \lambda))(1 - ([\mathbf{x}]_q - \lambda))\mathbb{P}(\mathcal{R}) + \\ & f(\mathcal{R}) - f(\{p,q\} \cup \mathcal{R}))([\mathbf{x}]_p + \lambda)([\mathbf{x}]_q - \lambda)\mathbb{P}(\mathcal{R}). \end{aligned}$$

with $\mathbb{P}(\mathcal{R}) = \prod_{r \in \mathcal{R}} x_r \prod_{r \notin \mathcal{R}} (1 - x_r)$. Taking the second derivative of $F(\mathbf{x} + \lambda \mathbf{w})$ with respect to λ yields

$$\begin{aligned} \frac{\partial^2 F(\mathbf{x} + \lambda \mathbf{w})}{\partial \lambda^2} = & \sum_{\mathcal{R} \subset \mathcal{P} \setminus \{p,q\}} 2\mathbb{P}(\mathcal{R})(f(\{p\} \cup \mathcal{R}) + f(\{q\} \cup \mathcal{R}) \\ & - f(\mathcal{R}) - f(\{p,q\} \cup \mathcal{R})). \end{aligned}$$

The submodularity of f asserts that $\frac{\partial^2 F(\mathbf{x} + \lambda \mathbf{w})}{\partial \lambda^2} \geq 0$ and consequently, $F(\mathbf{x} + \lambda \mathbf{w})$ is a convex function of λ. □

Lemma 16 (Interval Bound of Twice differentiable function) *Consider a twice differentiable function $F(\mathbf{x}) : [0,1]^n \to \mathbb{R}$ which satisfies $\left| \frac{\partial^2 F}{\partial [\mathbf{x}]_p \partial [\mathbf{x}]_q} \right| \leq \alpha$ for any $p, q \in \{1, \cdots, n\}$. Then for any $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^n$ satisfying $\mathbf{x}_2 \geq \mathbf{x}_1$ and $\mathbf{1}.(\mathbf{x}_2 - \mathbf{x}_1) \leq \beta$ we have*

$$\left| \frac{\partial F}{\partial [\mathbf{x}]_p}(\mathbf{x}_1 + \epsilon(\mathbf{x}_2 - \mathbf{x}_1)) - \frac{\partial F}{\partial [\mathbf{x}]_p}(\mathbf{x}_1) \right| \leq \epsilon \alpha \beta, \quad (38a)$$

$$F(\mathbf{x}_2) - F(\mathbf{x}_1) \geq \nabla F(\mathbf{x}_1).(\mathbf{x}_2 - \mathbf{x}_1) - \frac{1}{2}\alpha \beta^2, \quad (38b)$$

for $\epsilon \in [0\ 1]$.

Proof Let $\mathbf{h}_p = [\frac{\partial^2 F}{\partial [\mathbf{x}]_p \partial x_1}, \cdots, \frac{\partial^2 F}{\partial [\mathbf{x}]_p \partial x_n}]^\top$. Then, we can write

$$\begin{aligned} & \left| \frac{\partial F}{\partial [\mathbf{x}]_p}(\mathbf{x}_1 + \epsilon(\mathbf{x}_2 - \mathbf{x}_1)) - \frac{\partial F}{\partial [\mathbf{x}]_p}(\mathbf{x}_1) \right| \\ & = \left| \int_0^\epsilon \mathbf{h}_p(\mathbf{x}_1 + \tau(\mathbf{x}_2 - \mathbf{x}_1)).(\mathbf{x}_2 - \mathbf{x}_1) \mathrm{d}\tau \right| \\ & \leq \int_0^\epsilon \alpha \mathbf{1}.(\mathbf{x}_2 - \mathbf{x}_1) \mathrm{d}\tau = \epsilon \alpha \beta, \end{aligned} \quad (39)$$

Furthermore, $F(\mathbf{x}_2) - F(\mathbf{x}_1) = \int_0^1 \nabla F(\mathbf{x}_1 + \epsilon(\mathbf{x}_2 - \mathbf{x}_1)).(\bar{\mathbf{x}}(t+1) - \bar{\mathbf{x}}(t)) \mathrm{d}\epsilon \geq \int_0^1 (\nabla F(\mathbf{x}_1) - \epsilon \alpha \beta).(\mathbf{x}_2 - \mathbf{x}_1) \mathrm{d}\epsilon = \nabla F(\mathbf{x}_1).(\mathbf{x}_2 - \mathbf{x}_1) - \frac{1}{2}\alpha \beta^2$, with the third line follow from equation (39), which concludes the proof. □

D Convergence Analysis

Proof of Lemma 1 Since f is monotone increasing and submodular, we have $f(\mathcal{R}_{\mathbf{x}_i(t)} \cup \{p\}) - f(\mathcal{R}_{\mathbf{x}_i(t)} \setminus \{p\}) \geq 0$ and hence $\widetilde{\nabla F}(\mathbf{x}_i(t))$ has positive entries $\forall i \in \mathcal{A}$. Thus, $\widetilde{\mathbf{v}}_i(t) \in \mathcal{M}_i$, the optimizer of the optimization (10) has nonnegative entries. Hence, according to the propagation and update rule (9a) and (9b), we can conclude that $\mathbf{x}_{ii}(t)$ has increasing elements and only agent i can update it and other agents only copy this value as $\hat{\mathbf{x}}_{ji}(t)$. Therefore, we can conclude that $[\hat{\mathbf{x}}_{ji}]_p(t) \leq [\mathbf{x}_{ii}]_p(t)$ for all $p \in \mathcal{P}_i$ which concludes the proof. □

Proof of Lemma 2 The proof follows from a mathematical induction argument. The base case $\mathbf{x}_i(0) = \mathbf{0} \in \mathcal{M}$ and $\bar{\mathbf{x}}(0) = \mathbf{0} \in \mathcal{M}$ is trivially true. We take it to be true that at time t and for each agent $i \in \mathcal{A}$ it hold that $\mathbf{x}_i(t) \in \mathcal{M}$ with

$$\mathbf{1}.\mathbf{x}_{ii}(t) = \frac{t}{T}\kappa_i, \text{ and } \mathbf{1}.\hat{\mathbf{x}}_{ij}(t) \leq \frac{t}{T}\kappa_j, \quad j \in \mathcal{A} \setminus \{i\}.$$

for $t < T$ and $\widetilde{\mathbf{v}}_i(t) \in \mathcal{M}_i$ satisfying

$$\sum\nolimits_{p \in \mathcal{P}_i} [\widetilde{\mathbf{v}}_i(t)]_p = \kappa_i, \text{ and } \sum\nolimits_{p \in \mathcal{P}_j} [\widetilde{\mathbf{v}}_i(t)]_p = 0, \ j \in \mathcal{A} \setminus \{i\}.$$

Since $[\widetilde{\nabla F}(\mathbf{x}_i(t))]_p \geq 0 \quad p \in \mathcal{P}$, then by propagation

rule (9a), we establish that

$$\mathbf{1}.\mathbf{x}_{ii}^{-}(t+1) = \frac{t+1}{T}\kappa_i,$$
$$\mathbf{1}.\hat{\mathbf{x}}_{ij}^{-}(t+1) \leq \frac{t}{T}\kappa_j, \quad j \in \mathcal{A}\setminus\{i\}.$$

A a result of $\mathcal{M}_i$, $i \in \mathcal{A}$ being disjoint convex subspaces of $\mathcal{M}$, the update rule (9b) leads to

$$\mathbf{1}.\mathbf{x}_{ii}(t+1) = \frac{t+1}{T}\kappa_i,$$
$$\mathbf{1}.\hat{\mathbf{x}}_{ij}(t+1) \leq \frac{t+1}{T}\kappa_j, \quad j \in \mathcal{A}\setminus\{i\}.$$

Therefore, we conclude that $\mathbf{x}_i(t+1) \in \mathcal{M}$. Moreover, by the definition of $\bar{\mathbf{x}}(t)$ in (14) and $\mathcal{M}_i$, $i \in \mathcal{A}$ being disjoint convex subspaces of $\mathcal{M}$, we deduct that

$$\mathbf{1}.\bar{\mathbf{x}}(t) = \sum\nolimits_{i\in\mathcal{A}} \mathbf{1}.\mathbf{x}_{ii}(t+1) = \frac{t+1}{T}\kappa_i \quad i \in \mathcal{A},$$

for $t < T$ and therefore, $\bar{\mathbf{x}}(t+1) \in \mathcal{M}$. We conclude the proof of (a) by induction and trivially (b) follows. □

Proof of Proposition 3 f is a monotone increasing and submodular set function therefore $f(\mathcal{R}_{\mathbf{x}_i(t)} \cup \{p\}) - f(\mathcal{R}_{\mathbf{x}_i(t)} \setminus \{p\}) \geq 0$ and hence $\widetilde{\nabla F}(\mathbf{x}_i(t))$ has positive entries $\forall i \in \mathcal{A}$. Then, because $\widetilde{\mathbf{v}}_i(t) \in \mathcal{M}_i$, it follows from (10) that $\widetilde{\mathbf{v}}_i(t)$ has non-negative entries, $[\widetilde{\mathbf{v}}_i(t)]_p \geq 0$ which satisfy $\sum_{p\in\mathcal{P}_i}[\widetilde{\mathbf{v}}_i(t)]_p = \kappa_i$. Therefore, it follows from (9a) and Lemma 1 that

$$\mathbf{1}.\mathbf{x}_{ii}(t+1) = \mathbf{1}.\mathbf{x}_{ii}(t) + \frac{\kappa_i}{T}, \quad i \in \mathcal{A}. \tag{40}$$

Using (40) recursively for $d(\mathcal{G})$ steps, we can also write

$$\mathbf{1}.\mathbf{x}_{ii}(t) = \mathbf{1}.\mathbf{x}_{ii}(t-d(\mathcal{G})) + \frac{\kappa_i}{T}d(\mathcal{G}), \quad i \in \mathcal{A}. \tag{41}$$

Furthermore, it follows from Lemma (1) that for all $\forall p \in \mathcal{P}_i$ and any $j \in \mathcal{A}\setminus\{i\}$, we can write

$$[\mathbf{x}_i(t)]_p \geq [\mathbf{x}_j(t)]_p. \tag{42}$$

Also, since every agent $j \in \mathcal{A}\setminus\{i\}$ can be reached from agent $i \in \mathcal{A}$ at most in $d(\mathcal{G})$ hops, it follows from the propagation and update laws (9a) and (9b), for all $\forall p \in \mathcal{P}_i$, for any $j \in \mathcal{A}\setminus\{i\}$ that

$$[\mathbf{x}_j(t)]_p \geq [\mathbf{x}_i(t-d(\mathcal{G}))]_p(t-d(\mathcal{G})). \tag{43}$$

Thus, for $i \in \mathcal{A}$ and $j \in \mathcal{A}\setminus\{i\}$, (42) and (43) result in

$$\mathbf{1}.\mathbf{x}_{ii}(t) \geq \mathbf{1}.\hat{\mathbf{x}}_{ji}(t) \geq \mathbf{1}.\mathbf{x}_{ii}(t-d(\mathcal{G})). \tag{44}$$

Next, we can use (41) and (44) to write

$$\mathbf{1}.\mathbf{x}_{ii}(t) \geq \mathbf{1}.\hat{\mathbf{x}}_{ji}(t) \geq \mathbf{1}.\mathbf{x}_{ii}(t) - \frac{\kappa_i}{T}d(\mathcal{G}), \tag{45}$$

for $i \in \mathcal{A}$ and $j \in \mathcal{A}\setminus\{i\}$. Using (45) for any $i \in \mathcal{A}$ we can write

$$\sum\nolimits_{l\in\mathcal{A}} \mathbf{1}.\mathbf{x}_{ll}(t) \geq \mathbf{1}.\mathbf{x}_{ii}(t) + \sum\nolimits_{j\in\mathcal{A}\setminus\{i\}} \mathbf{1}.\hat{\mathbf{x}}_{ij}(t) \geq \sum\nolimits_{l\in\mathcal{A}} \mathbf{1}.\mathbf{x}_{ll}(t) - \frac{\kappa_l}{T}d(\mathcal{G}). \tag{46}$$

Then, using Lemma 1, from (46) we can write

$$\mathbf{1}.\bar{\mathbf{x}}(t) \geq \mathbf{1}.\mathbf{x}_i(t) \geq \mathbf{1}.\bar{\mathbf{x}}(t) - \frac{\kappa}{T}d(\mathcal{G}),$$

with $\kappa = \sum_{i\in\mathcal{A}}\kappa_i$, which ascertains (15a). Next, note that from Lemma 1, we have $\mathbf{x}_{jj}(t) = \mathbf{x}_{ii}^{-}(t)$ for any $i \in \mathcal{A}$. Then, using (9a) and invoking Lemma 1, we obtain (15b), which, given (40), also ascertains (15c). □

Proof of Proposition 5 Given the definition of $\delta_p(\tau)$ and $\delta_q(\tau)$, at each iteration of (19), either $[\mathbf{y}_{ii}(\tau+1)]_p \in \{0,1\}$ or $[\mathbf{y}_{ii}(\tau+1)]_q \in \{0,1\}$. Moreover, $\mathbf{y}_{ii}(\tau+1) \in [0,1]^{|\mathcal{P}_i|}$. Consequently, $\mathbf{y}_{ii}(|\mathcal{P}_i|) \in \{0,1\}^{|\mathcal{P}_i|}$. Next, note that since $\mathbf{y}_{ii}(0) = \mathbf{x}_{ii}(T)$, $i \in \mathcal{A}$, by virtue of Lemma 2, we have $\mathbf{1}.\mathbf{y}_{ii}(0) = \kappa_i$. Therefore, because (19) is a zero-sum iteration, we have $\mathbf{1}.\mathbf{y}_{ii}(\tau) = \kappa_i$, $i \in \mathcal{A}$ for any $\tau \in \mathbb{Z}_{\geq 0}$, which confirms $\mathbf{1}.\mathbf{y}_{ii}(|\mathcal{P}_i|) = \kappa_i$ and $\bar{\mathbf{y}} \in \mathcal{M}$. Lastly, because $[\mathbf{y}_{ii}(|\mathcal{P}_i|)]_r \in \{0,1\},\ r \in \mathcal{P}_i$ for any $i \in \mathcal{A}$, $\bar{\mathbf{y}}$ is a vertex of $\mathcal{M}$. □

E Proof of the main results

Proof of Theorem 4 Knowing that $\left|\frac{\partial^2 F}{\partial[\mathbf{x}]_p\partial[\mathbf{x}]_q}\right| \leq cf(\mathcal{R}^\star)$ from Lemma 14, and (15c), it follows from Lemma 16 that $F(\bar{\mathbf{x}}(t+1)) - F(\bar{\mathbf{x}}(t)) \geq \nabla F(\bar{\mathbf{x}}(t)).(\bar{\mathbf{x}}(t+1) - \bar{\mathbf{x}}(t)) - \frac{\kappa^2}{2T^2}cf(\mathcal{R}^\star)$, which, given (15b), leads to

$$F(\bar{\mathbf{x}}(t+1)) - F(\bar{\mathbf{x}}(t)) \geq \frac{1}{T}\sum\nolimits_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) - \frac{\kappa^2}{2T^2}cf(\mathcal{R}^\star). \tag{47}$$

By definition, $\bar{\mathbf{x}}(t) \geq \mathbf{x}_i(t)$ for any $\forall i \in \mathcal{A}$. Therefore, given (15a), by invoking Lemma 16, for any $i \in \mathcal{A}$ we can write

$$\left|\frac{\partial F}{\partial[\mathbf{x}]_p}(\bar{\mathbf{x}}(t)) - \frac{\partial F}{\partial[\mathbf{x}]_p}(\mathbf{x}_i(t))\right| \leq \frac{\kappa}{T}d(\mathcal{G})cf(\mathcal{R}^\star), \tag{48}$$

for $p \in \{1,\cdots,n\}$. Recall that at each time step t, the realization of $\widetilde{\mathbf{v}}_i(t)$ in (10) that Algorithm 1 uses for $\{p_1^\star,\cdots,p_{\kappa_i}^\star\} \in \mathcal{P}_i$ is

$$\widetilde{\mathbf{v}}_i(t) = \mathbf{1}_{\{p_1^\star,\cdots,p_{\kappa_i}^\star\}}, \tag{49}$$

for every $i \in \mathcal{A}$. Thus, $\mathbf{1}.\widetilde{\mathbf{v}}_i(t) = \kappa_i, i \in \mathcal{A}$. Consequently, using (48) we can write

$$\sum\nolimits_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) \geq \sum\nolimits_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) - \frac{\kappa^2}{T} d(\mathcal{G}))cf(\mathcal{R}^\star). \quad (50)$$

Next, we let $\bar{\mathbf{v}}_i(t) = \underset{\mathbf{v}\in\mathcal{M}_i}{\operatorname{argmax}}\, \mathbf{v}.\nabla F(\bar{\mathbf{x}}(t))$ and $\hat{\mathbf{v}}_i(t) = \underset{\mathbf{v}\in\mathcal{M}_i}{\operatorname{argmax}}\, \mathbf{v}.\nabla F(\mathbf{x}_i(t))$. Then, using $\hat{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \bar{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t))$ and $\hat{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \tilde{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t))$, $i \in \mathcal{A}$, and (48) we can also write

$$\sum\nolimits_{i\in\mathcal{A}} \hat{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \sum\nolimits_{i\in\mathcal{A}} \bar{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \sum\nolimits_{i\in\mathcal{A}} \bar{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) - \frac{\kappa^2}{T} d(\mathcal{G})cf(\mathcal{R}^\star), \quad (51a)$$

$$\sum\nolimits_{i\in\mathcal{A}} \hat{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \sum\nolimits_{i\in\mathcal{A}} \tilde{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)). \quad (51b)$$

On the other hand, by virtue of Lemma 13, $\left[\widetilde{\nabla F}(\mathbf{x}_i(t))\right]_p$, $p \in \mathcal{P}_i$ that each agent $i \in \mathcal{A}$ uses to solve optimization problem (22) (equivalently (10)) satisfies

$$\left| \left[\widetilde{\nabla F}(\mathbf{x}_i(t))\right]_p - \frac{\partial F}{\partial [\mathbf{x}]_p}(\mathbf{x}_i(t)) \right| \leq \frac{1}{2T} f(\mathcal{R}^\star) \quad (52)$$

with the probability of at least $1–2\mathrm{e}^{-\frac{1}{8T^2}K_i}$. Using (51b) and (52), and because the samples are drawn independently, we obtain

$$\sum_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) \geq \sum_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\widetilde{\nabla F}(\mathbf{x}_i(t)) - \frac{\kappa}{2T} f(\mathcal{R}^\star), \quad (53a)$$

$$\sum\nolimits_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\widetilde{\nabla F}(\mathbf{x}_i(t)) \geq \sum\nolimits_{i\in\mathcal{A}} \hat{\mathbf{v}}_i(t).\widetilde{\nabla F}(\mathbf{x}_i(t)) \geq \sum\nolimits_{i\in\mathcal{A}} \hat{\mathbf{v}}_i(t).\nabla F(\mathbf{x}_i(t)) - \frac{\kappa}{2T} f(\mathcal{R}^\star), \quad (53b)$$

with the probability of at least $\prod_{i\in\mathcal{A}}(1-2\mathrm{e}^{-\frac{1}{8T^2}K_i})^{|\mathcal{P}_i|}$. From (50), (51a),(53a), and (53b) now we can write

$$\sum\nolimits_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) \geq \sum\nolimits_{i\in\mathcal{A}} \bar{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) - (2\kappa d(\mathcal{G})) + 1)\frac{\kappa}{T} f(\mathcal{R}^\star), \quad (54)$$

with the probability of at least $1 - 2\sum_{i\in\mathcal{A}} \mathrm{e}^{-\frac{1}{8T^2}K_i}$.

Next, let $\mathbf{v}_i^\star$ be the projection of $\mathbf{1}_{\mathcal{R}^\star}$ into $\mathcal{M}_i$. Knowing that $\mathcal{M}_i$'s are disjoint sub-spaces of $\mathcal{M}$ covering the whole space then we can write

$$\mathbf{1}_{\mathcal{R}^\star} = \sum\nolimits_{i\in\mathcal{A}} \mathbf{v}_i^\star. \quad (55)$$

Then, using (54), (55), and invoking Lemma 11 and the fact that $\bar{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) \geq \mathbf{v}_i^\star(t).\nabla F(\bar{\mathbf{x}}(t))$ we obtain

$$\begin{aligned}\sum\nolimits_{i\in\mathcal{A}} \widetilde{\mathbf{v}}_i(t).\nabla F(\bar{\mathbf{x}}(t)) \geq \\ \sum\nolimits_{i\in\mathcal{A}} \mathbf{v}_i^\star(t).\nabla F(\bar{\mathbf{x}}(t)) - (2c\kappa d(\mathcal{G})+1)\frac{\kappa}{T} f(\mathcal{R}^\star) = \\ \mathbf{1}_{\mathcal{R}^\star}.\nabla F(\bar{\mathbf{x}}(t)) - (2c\kappa d(\mathcal{G})+1)\frac{\kappa}{T} f(\mathcal{R}^\star) \geq \\ f(\mathcal{R}^\star) - cF(\bar{\mathbf{x}}(t)) - (2c\kappa d(\mathcal{G})+1)\frac{\kappa}{T} f(\mathcal{R}^\star), \end{aligned} \quad (56)$$

with the probability of at least $\prod_{i\in\mathcal{A}}(1-2\mathrm{e}^{-\frac{1}{8T^2}K_i})^{|\mathcal{P}_i|}$. Hence, using (47) and (56), we conclude that

$$F(\bar{\mathbf{x}}(t+1)) - F(\bar{\mathbf{x}}(t)) \geq \frac{1}{T}(f(\mathcal{R}^\star) - cF(\bar{\mathbf{x}}(t)) - (2c\kappa d(\mathcal{G})) + \frac{c\kappa}{2}+1)\frac{\kappa}{T^2} f(\mathcal{R}^\star), \quad (57)$$

with the probability of at least $\prod_{i\in\mathcal{A}}(1-2\mathrm{e}^{-\frac{1}{8T^2}K_i})^{|\mathcal{P}_i|}$. Next, let $g(t) = f(\mathcal{R}^\star) - cF(\bar{\mathbf{x}}(t))$ and $\beta = (2c\kappa d(\mathcal{G})) + \frac{c\kappa}{2}+1)\frac{\kappa}{T^2} f(\mathcal{R}^\star)$, to rewrite (57) as

$$\begin{aligned}(f(\mathcal{R}^\star) - cF(\bar{\mathbf{x}}(t))) - (f(\mathcal{R}^\star) - cF(\bar{\mathbf{x}}(t+1))) = \\ g(t) - g(t+1) \geq \frac{c}{T}(f(\mathcal{R}^\star) - cF(\bar{\mathbf{x}}(t))) - c\beta = \\ \frac{c}{T} g(t) - c\beta. \end{aligned} \quad (58)$$

Then from inequality (58) we get

$$g(t+1) \leq (1 - \frac{c}{T})g(t) + c\beta \quad (59)$$

with the probability of at least $\prod_{i\in\mathcal{A}}(1-2\mathrm{e}^{-\frac{1}{8T^2}K_i})^{|\mathcal{P}_i|}$. Solving for inequality (59) at time T yields

$$\begin{aligned} g(T) &\leq (1 - \frac{c}{T})^T g(0) + \beta \sum\nolimits_{k=0}^{T-1} (1 - \frac{c}{T})^k \\ &= (1 - \frac{c}{T})^T g(0) + T\beta(1 - (1 - \frac{c}{T})^T) \end{aligned} \quad (60)$$

with the probability of at least $\left(\prod_{i\in\mathcal{A}}(1-2\mathrm{e}^{-\frac{1}{8T^2}K_i})^{|\mathcal{P}_i|}\right)^T$. Substituting back $g(T) = f(\mathcal{R}^\star) - cF(\bar{\mathbf{x}}(T))$ and $g(0) = f(\mathcal{R}^\star) - cF(\mathbf{x}(0)) = f(\mathcal{R}^\star)$, in (60) we then obtain

$$\begin{aligned}\frac{1}{c}(1 - (1 - \frac{1}{T})^T)(f(\mathcal{R}^\star) - T\beta) = \\ \frac{1}{c}(1 - (1 - \frac{1}{T})^T)(1 - (2c\kappa d(\mathcal{G}) + \frac{c\kappa}{2}+1)\frac{\kappa}{T})f(\mathcal{R}^\star) \\ \leq F(\bar{\mathbf{x}}(T)), \end{aligned} \quad (61)$$

with the probability of at least $\left(\prod_{i\in\mathcal{A}}(1-2\mathrm{e}^{-\frac{1}{8T^2}K_i})^{|\mathcal{P}_i|}\right)^T$.

By applying $\mathrm{e}^{-c} \geq (1-(1-\frac{c}{T})^T)$, we get

$$\frac{1}{c}(1-\mathrm{e}^{-c})(1-(2c\kappa d(\mathcal{G})+\frac{c\kappa}{2}+1)\frac{\kappa}{T})f(\mathcal{R}^\star) \leq F(\bar{\mathbf{x}}(T)), \tag{62}$$

with the probability of at least $\left(\prod_{i\in\mathcal{A}}(1-2\mathrm{e}^{-\frac{1}{8T^2}K_i})^{|\mathcal{P}_i|}\right)^T$. which concludes the proof. □

Proof of Theorem 6 Consider the distributed Pipage rounding (19). Let τ_i be any arbitrary iteration stage of (19) for agent $i \in \mathcal{A}$. Recall that we partitioned $\mathbf{y}_i$, $i \in \mathcal{A}$ as $\mathbf{y}_i(\tau_i) = [\hat{\mathbf{y}}_{i1}(\tau_i), \cdots, \mathbf{y}_{ii}(\tau_i), \cdots, \hat{\mathbf{y}}_{iN}(\tau_i)]$. Let

$$\mathbf{y}(\tau) = [\mathbf{y}_{11}(\tau_1), \cdots, \mathbf{y}_{ii}(\tau_i+\tau), \cdots, \mathbf{y}_{NN}(\tau_N)]$$

for any $\tau_j \in \mathbb{Z}_{\geq 0}$, $j \in \mathcal{A}$, and arbitrary $i \in \mathcal{A}$.

Distributed Pipage rounding (19) results in

$$\mathbf{y}(\tau+1) = \begin{cases} \mathbf{y}(\tau) + \delta_p(\tau_i)\,\mathbf{z}, & \text{w.p. } \frac{\delta_q(\tau_i)}{\delta_p(\tau_i)+\delta_q(\tau_i)} \in [0,1], \\ \mathbf{y}(\tau) - \delta_q(\tau_i)\,\mathbf{z}, & \text{w.p. } \frac{\delta_p(\tau_i)}{\delta_p(\tau_i)+\delta_q(\tau_i)} \in [0,1], \end{cases}$$

for a $\mathbf{z} \in \{-1,0,1\}^n$ that satisfies $[\mathbf{z}]_p = 1$, $[\mathbf{z}]_q = -1$ and $[\mathbf{z}]_r = 0$, $r \neq p, q$. Next, note that the directional convexity of the multilinear function in Lemma 15 yields

$$F(\mathbf{y}(\tau)) \leq \frac{\delta_q(\tau_i)}{\delta_p(\tau_i)+\delta_q(\tau_i)} F(\mathbf{y}(\tau)+\delta_p(\tau_i)\mathbf{z}) + \frac{\delta_p(\tau_i)}{\delta_p(\tau_i)+\delta_q(\tau_i)} F(\mathbf{y}(\tau)-\delta_q(\tau_i)\mathbf{z}).$$

Hence, we can write

$$F(\mathbf{y}(\tau)) \leq \mathbb{E}[F(\mathbf{y}(\tau+1))|\mathbf{y}(\tau)]. \tag{63}$$

Next, taking expectation with respect to $\mathbf{y}(\tau)$, we get

$$\mathbb{E}[F(\mathbf{y}(\tau))] \leq \mathbb{E}[F(\mathbf{y}(\tau+1))]. \tag{64}$$

Note that because $\mathbf{y}(0)|_{\{\tau_j\}_{j=1}^N=\{\mathbf{0}\}^N} = \bar{\mathbf{x}}(T)$, we have $\mathbb{E}[F(\mathbf{y}(0)|_{\{\tau_j\}_{j=1}^N=\{\mathbf{0}\}^N})] = F(\bar{\mathbf{x}}(T))$. Consequently, since $\mathbf{y}(\tau)$ is defined for any arbitrary $\{\tau_j\}_{j=1}^N$, we can conclude that

$$F(\bar{\mathbf{x}}(T)) \leq \mathbb{E}[F(\bar{\mathbf{y}})], \tag{65}$$

where $\bar{\mathbf{y}} = [\mathbf{y}_{11}(|\mathcal{P}_1|), \mathbf{y}_{22}(|\mathcal{P}_2|), \cdots, \mathbf{y}_{NN}(|\mathcal{P}_N|)]$.

Proposition 5 states that $\bar{y}$ is a vertex of $\mathcal{M}$, therefore $F(\bar{\mathbf{y}}) = f(\mathcal{R}_{\bar{\mathbf{y}}})$. On the other hand, it follows from (20) that $\mathcal{R}_{\bar{\mathbf{y}}} = \bigcup_{i\in\mathcal{A}} \bar{\mathcal{R}}_i$. Consequently, (21) follows from (65). □

Proof of Theorem 9 Given that the information set propagation rules (22), (24), and (25) are a realization of the vector space propagation rules (10), (9a), and (9b), we can conclude that the vector $\mathbf{y} = [\mathbf{y}_1^\top, \cdots, \mathbf{y}_N^\top]^\top$ defined as

$$\begin{cases} [\mathbf{y}]_p = \alpha, & (p,\alpha) \in \mathcal{F}_i(T), \quad p \in \mathcal{P}_i \\ [\mathbf{y}]_p = 0, & \text{Otherwise} \end{cases}$$

is a realization of $\bar{\mathbf{x}}(T)$ and satisfies $F(\bar{\mathbf{x}}(T)) = F(\mathbf{y})$.

Moreover, sampling a single strategy $\bar{p}_i$ according to $\mathbf{y}_i$ out of $\mathcal{P}_i$ is equivalent to sampling rule (19). Noting that $\mathbf{y}$ is a realization of $\bar{\mathbf{x}}(T)$, Lemma 4 and Theorem 6 leads us to concluding the proof. □

[]Navid Rezazadeh is a Research Assistant in Mechanical and Aerospace Engineering Department at the University of California, Irvine, CA, USA. He received Ph.D. and M.Sc in Mechanical and Aerospace Engineering from the University of California, Irvine in 2017 and 2022 respectively. He also received B.Sc in Mechanical Engineering from the Sharif University of Technology, Tehran, Iran, in 2013. He was a recipient of UC Irvine doctoral and Holmes fellowship for his graduate studies. His research is focused on privacy, distributed optimization, distributed decision making and machine learning,

[]Solmaz S. Kia is an Associate Professor of Mechanical and Aerospace Engineering at the University of California, Irvine (UCI), CA, USA. She obtained her Ph.D. degree in Mechanical and Aerospace Engineering from UCI, in 2009, and her M.Sc. and B.Sc. in Aerospace Engineering from the Sharif University of Technology, Iran, in 2004 and 2001, respectively. She was a senior research engineer at SySense Inc., El Segundo, CA from Jun. 2009 to Sep. 2010. She held postdoctoral positions in the Department of Mechanical and Aerospace Engineering at the University of California San Diego and UCI. She was a recipient of the UC president's postdoctoral fellowship in 2012-2014, an NSF CAREER award in 2017, and the best CSM paper award in 2021. Her main research interests, in a broad sense, include distributed optimization/coordination/estimation, nonlinear control theory, and probabilistic robotics.