Proceedings of Machine Learning Research vol 144:1–13, 2021

Optimal Algorithms for Submodular Maximization with Distributed Constraints

Alexander Robey AROBEY1@SEAS.UPENN.EDU

Arman Adibi AADIBI@SEAS.UPENN.EDU

Brent Schlotfeldt BRENTSC@SEAS.UPENN.EDU

Hamed Hassani HASSANI@SEAS.UPENN.EDU

George J. Pappas PAPPASG@SEAS.UPENN.EDU

University of Pennsylvania, Philadelphia, PA 19103, USA

Abstract

We consider a class of discrete optimization problems that aim to maximize a submodular objective function subject to a distributed partition matroid constraint. More precisely, we consider a networked scenario in which multiple agents choose actions from local strategy sets with the goal of maximizing a submodular objective function defined over the set of all possible actions. Given this distributed setting, we develop Constraint-Distributed Continuous Greedy (`CDCG`), a message passing algorithm that converges to the tight $(1-1/e)$ approximation factor of the optimum global solution using only local computation and communication. It is known that a sequential greedy algorithm can only achieve a $1/2$ multiplicative approximation of the optimal solution for this class of problems in the distributed setting. Our framework relies on lifting the discrete problem to a continuous domain and developing a consensus algorithm that achieves the tight $(1-1/e)$ approximation guarantee of the global discrete solution once a proper rounding scheme is applied. We also offer empirical results from a multi-agent area coverage problem to show that the proposed method significantly outperforms the state-of-the-art sequential greedy method.

Keywords: Submodular maximization, partition matroid, distributed optimization

1. Introduction

Recently, the need has arisen to design algorithms that distribute decision making among a collection of agents or computing devices. This need has been motivated by problems from statistics, machine learning and robotics. More specifically, these problems include:

- *(Density estimation)* What is the best way to estimate a non-parametric density function from a distributed dataset? (Hu et al., 2007)
- *(Non-parametric models)* How should we summarize very large datasets in a distributed manner to facilitate Gaussian process regression? (Mirzasoleiman et al., 2016)
- *(Information acquisition)* How should a team of mobile robots acquire information about an environmental process or reduce uncertainty in a mapping task? (Schlotfeldt et al., 2018)

Research toward solving the problems posed in these applications has resulted in a large body of work on topics such as sensing and coverage (Zhong and Cassandras, 2011; Singh et al., 2009), natural language processing (Wei et al., 2013), and learning and statistics (Golovin and Krause, 2011;

© 2021 A. Robey, A. Adibi, B. Schlotfeldt, H. Hassani & G. J. Pappas.

Djolonga et al., 2016). Indeed, inherent to each of these applications is an underlying optimization problem that can be expressed as

$$\begin{aligned} &\text{maximize} \quad f(\mathcal{S}) && \text{(1a)} \\ &\text{subject to} \quad \mathcal{S} \subseteq \mathcal{Y}, \, \mathcal{S} \in \mathcal{I} && \text{(1b)} \end{aligned}$$

where f is a submodular set function (i.e. it has a diminishing-returns property), $\mathcal{Y}$ is a finite set of all decision variables, and $\mathcal{I}$ is a family of allowable subsets of $\mathcal{Y}$. In words, the goal of (1) is to pick a set $\mathcal{S}$ from the family of allowable subsets $\mathcal{I}$ that maximizes the submodular set function f. A wide class of relevant objective functions such as mutual information and weighted coverage are submodular; this has motivated a growing body of work surrounding submodular optimization problems (Mokhtari et al., 2018; Mirzasoleiman et al., 2013; Zhou et al., 2020; Du et al., 2020; Adibi et al., 2020; Chen et al., 2020; Xie et al., 2019).

Intuitively, it is useful to think of the problem in (1) as a distributed n-player game. In this game, each player or agent has a distinct local strategy set of actions. The goal of the game is for each agent to choose at most one action from its own strategy set to maximize a problem-specific notion of reward. Therefore, the problem is *distributed* in the sense that agents can only form a control policy with the actions from their local, distinct strategy sets. To maximize reward, agents are allowed to communicate with their direct neighbors in a bidirectional communication graph. In this way, we might think of these agents as robots that collectively aim to solve a coverage problem in an unknown environment by communicating their sensing actions to their nearest neighbors. Throughout this work, we will refer to this multi-agent game example to elucidate our results.

In this paper, our aim is to study problem (1) in a *distributed* setting, which we will formally introduce in Section 4; this setting differs considerably from the *centralized* setting, which has been studied thoroughly in past work (see Calinescu et al. (2011)). Notably, the distributed setting admits a more challenging problem because agents can only communicate locally with respect to a communication graph. Therefore designing an efficient communication scheme among agents is a concomitant requirement for the distributed setting, whereas in the centralized setting, there is no such desideratum.

Contributions. In this paper, we formulate the general case of maximizing a submodular set function subject to a distributed partition matroid constraint in Problem 1. We then formulate the continuous relaxation of this problem via the multilinear extension in Problem 2. Both of these problems are formally defined in Section 4. To this end, we study the special case of this optimization problem in which each agent can compute the global objective function and the gradient of the objective function; however we assume that each agent only has access to a local, distinct set of actions. Considering these constraints, we develop Constraint-Distributed Continuous Greedy (`CDCG`), a novel algorithm for solving the continuous relaxation of the *distributed* submodular optimization problem that achieves a tight $(1 - 1/e)$ approximation of the optimal solution, which is known to be the best possible approximation unless $\mathbf{P} = \mathbf{NP}$. We offer an analysis of the proposed algorithm and prove that it achieves the tight $(1 - 1/e)$ approximation and that its error term vanishes at a linear rate.

Previous work on the distributed version of this problem can approximate the optimal solution to within a multiplicative factor of $1/2$ via sequential greedy algorithms (Gharesifard and Smith, 2017; Corah and Michael, 2018; Calinescu et al., 2011). Algorithms for different settings, such as the setting of (Mokhtari et al., 2018) in which each node has access to a local objective function which is averaged to form a global objective function, can also achieve the $(1 - 1/e)$ approximation. Similarly, (Calinescu et al., 2011) shows that it is possible to achieve the optimal $(1 - 1/e)$ approximation in the centralized setting. However, to the best of our knowledge the `CDCG` algorithm presented in this paper is the first algorithm that is guaranteed to achieve the $(1-1/e)$ approximation of the optimal solution in this distributed setting.

2. Related work

The optimization problem in (1) has previously been studied in settings that differ significantly from the setting studied in this paper. In particular, (Calinescu et al., 2011) addresses this problem in a centralized setting and shows that a centralized algorithm can obtain the tight $(1 - 1/e)$ approximation of the optimal solution. In this way, (Calinescu et al., 2011) is perhaps the closest to this paper in that both manuscripts introduce algorithms that obtain the tight $(1 - 1/e)$ guarantee for solving the optimization problem in (1) with respect to a particular setting. However, the setting of (Calinescu et al., 2011) is inherently centralized, whereas our setting is *distributed*.

Another similar line of work concerns the so-called "master-worker" model. In this framework, agents solve a distributed optimization problem such as (1) by exchanging local information with a centralized master node. However, this setting also differs from the setting studied in this work in that our results assume an entirely distributed setting with no centralized node (Mirzasoleiman et al., 2013; Barbosa et al., 2015).

Fundamentally, the optimization problem posed in (1) is NP-hard. However, near-optimal solutions to (1) can be approximated by greedy algorithms (Nemhauser et al., 1978; Nemhauser and Wolsey, 1978). In the distributed context, the sequential greedy algorithm (SGA) has been rigorously studied in (Gharesifard and Smith, 2017). This work poses (1) as a communication problem among agents distributed in an directed acyclic graph (DAG) working to optimize a global objective function. The authors of (Gharesifard and Smith, 2017) offer upper and lower bounds on the performance of SGA based on the clique number of the underlying DAG. Building on this, (Corah and Michael, 2018) analyzes the communication redundancy in such an approach and proposes a distributed planning technique that randomly partitions the agents in the DAG. On the other hand, (Grimsman et al., 2018) extends the work of (Gharesifard and Smith, 2017) to a sequential setting in which agents have limited access to the prior decisions of other agents. Extensions of SGA such as the distributed SGA (DSGA) have also been proposed. In particular, (Corah and Michael, 2017, 2019) pose (1) as a multi-robot exploration problem and uses DSGA to quantify the suboptimality incurred by redundant sensing information.

Others have proposed novel algorithms with the goal of avoiding the communication overhead incurred by deploying SGA for a large number of agents. Instead of explicitly solving (1), many of these algorithms seek to solve a continuous relaxation of this problem (Hassani et al., 2017; Mokhtari et al., 2020). This continualization of the problem in (1) was originally introduced in (Calinescu et al., 2011). In particular, (Mokhtari et al., 2018) proposes several gradient ascent-style algorithms for solving a problem akin to (1) in which each agent has access to a local objective function. Similarly, novel algorithms have been developed for solving problems such as unconstrained submodular maximization (Buchbinder et al., 2015) and submodular maximization with matroid constraints (Calinescu et al., 2011; Buchbinder et al., 2014) by first lifting these problems to the continuous domain.

Another notable direction in solving problem (1) has been to define an auxiliary or surrogate function in place of the original submodular objective. For instance, (Clark et al., 2015) introduces a distributed algorithm for maximizing a submodular auxiliary function subject to matroid constraints that obtains the $(1 - 1/e)$ optimal approximation. This approach of defining surrogate functions in place of the submodular objective differs significantly from our approach.

3. Preliminaries

In this section, we review the notation used throughout this paper and state definitions that are necessary for the problem formulations in Section 4.

Notation. Throughout this paper, lowercase bold-face (e.g. $\mathbf{v}$) will denote a vector, while uppercase bold-face (e.g. $\mathbf{W}$) will denote a matrix. The i^{th} component of a vector $\mathbf{v}$ will be denoted v_i; the element in the i^{th} row of the j^{th} column of a matrix $\mathbf{W}$ will be denoted by w_{ij}. The inner product between two vectors $\mathbf{x}$ and $\mathbf{y}$ will be denoted by $\langle \mathbf{x}, \mathbf{y} \rangle$ and the Euclidean norm of a vector $\mathbf{v}$ will be denoted by $||\mathbf{v}||$. Given two vectors $\mathbf{x}$ and $\mathbf{y}$, we define $\mathbf{x} \vee \mathbf{y} = \max(\mathbf{x}, \mathbf{y})$ as the (vector-valued) component-wise maximum between $\mathbf{x}$ and $\mathbf{y}$; similarly, $\mathbf{x} \wedge \mathbf{y} = \min(\mathbf{x}, \mathbf{y})$ will denote the component-wise minimum between $\mathbf{x}$ and $\mathbf{y}$. We will use the notation $\mathbf{0}_n$ to denote an n-dimensional vector in which each component is zero; similarly $\mathbf{1}_n$ will denote an n-dimensional vector in which each component is one. Calligraphic fonts will denote sets (e.g. $\mathcal{Y}$). Given a set $\mathcal{Y}$, $|\mathcal{Y}|$ will denote the cardinality of $\mathcal{Y}$, while $2^{\mathcal{Y}}$ will denote the power set of $\mathcal{Y}$. $1_{\mathcal{Y}} : \mathcal{Y} \mapsto \{0, 1\}$ will represent the indicator function for the set $\mathcal{Y}$. That is, $1_{\mathcal{Y}}$ is the function that takes value one if its argument is an element of $\mathcal{Y}$ and takes value zero otherwise. Finally, $\varnothing$ will denote the null set.

Background and relevant definitions. Let $\mathcal{Y}$ be a finite set and let $f : 2^{\mathcal{Y}} \mapsto \mathbb{R}_+$ be a set function mapping subsets of $\mathcal{Y}$ to the nonnegative real line. In this setting, $\mathcal{Y}$ is commonly referred to as the *ground set*. The function f is called *submodular* if for every $\mathcal{A}, \mathcal{B} \subseteq \mathcal{Y}$,

$$f(\mathcal{A} \cap \mathcal{B}) + f(\mathcal{A} \cup \mathcal{B}) \leq f(\mathcal{A}) + f(\mathcal{B}).$$

In essence, submodularity amounts to f having a so-called diminishing-returns property, meaning that the incremental value of adding a single element to the argument of f is no less than that of adding the same element to a superset of the argument. To illustrate this, we will slightly overburden our notation by defining

$$f(\mathbf{x}|\mathcal{A}) := f(\mathcal{A} \cup \{\mathbf{x}\}) - f(\mathcal{A})$$

as the *marginal reward* of x given $\mathcal{A}$. This gives rise to an equivalent definition of submodularity. In particular, f is said to be submodular if for every $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{Y}$ and $\forall \mathbf{x} \in \mathcal{Y} \backslash \mathcal{B}$,

$$f(\mathbf{x}|\mathcal{B}) \leq f(\mathbf{x}|\mathcal{A}).$$

Throughout this paper, we will consider submodular functions that are also *monotone*, meaning that for every $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{Y}$, $f(\mathcal{A}) \leq f(\mathcal{B})$, and *normalized*, meaning that $f(\varnothing) = 0$.

In practice, one often encounters a constraint on the allowable subsets of the ground set $\mathcal{Y}$ when maximizing a submodular objective function. Concretely, if $\mathcal{I}$ is a nonempty family of allowable subsets of the ground set $\mathcal{Y}$, then the tuple $(\mathcal{Y}, \mathcal{I})$ is a *matroid* if the following criteria are satisfied:

(1) (*Heredity*) For any $\mathcal{A} \subset \mathcal{B} \subset \mathcal{Y}$, if $\mathcal{B} \in \mathcal{I}$, then $\mathcal{A} \in \mathcal{I}$.

(2) (*Augmentation*) For any $\mathcal{A}, \mathcal{B} \in \mathcal{I}$, if $|\mathcal{A}| < |\mathcal{B}|$, then $\exists\, \mathbf{x} \in \mathcal{B} \backslash \mathcal{A}$ such that $\mathcal{A} \cup \{\mathbf{x}\} \in \mathcal{I}$.

Furthermore, if $\mathcal{Y}$ is partitioned into n disjoint sets $\mathcal{Y}_1, \ldots, \mathcal{Y}_n$, then the tuple $(\mathcal{Y}, \mathcal{I})$ is a *partition matroid* if there exists positive integers $\alpha_1, \ldots, \alpha_n$ such that

$$\mathcal{I} \equiv \{\mathcal{A} \,:\, \mathcal{A} \subseteq \mathcal{Y}, |\mathcal{A} \cap \mathcal{Y}_i| \leq \alpha_i \text{ for each } i = 1, \ldots, n\}.$$

Partition matroids are particularly useful when defining the constraints of a distributed optimization problem because they can be used to describe a setting in which a ground set $\mathcal{Y}$ of all possible actions is written as the product of disjoint local action spaces $\mathcal{Y}_i$.

The notion of submodularity can be extended to the continuous domain (Wolsey, 1982). Consider a set $\mathcal{X} = \prod_{i=1}^{n} \mathcal{X}_i$, where $\mathcal{X}_i$ is a compact subset of $\mathbb{R}_+$ for each index $i \in \{1, \ldots, n\}$. We call a continuous function $F : \mathcal{X} \to \mathbb{R}_+$ *submodular* if for all $\mathbf{x}, \mathbf{y} \in \mathcal{X}$,

$$F(\mathbf{x} \vee \mathbf{y}) + F(\mathbf{x} \wedge \mathbf{y}) \leq F(\mathbf{x}) + F(\mathbf{y}).$$

As in the discrete case, we say that a continuous function F is monotone if $\forall \mathbf{x}, \mathbf{y} \in \mathcal{X}$, $\mathbf{x} \preceq \mathbf{y}$ implies that $F(\mathbf{x}) \leq F(\mathbf{y})$. Furthermore, if F is differentiable, we say that F is DR-submodular, where DR stands for "diminishing-returns," if the gradients are *antitone*. That is, $\forall \mathbf{x}, \mathbf{y} \in \mathcal{X}$, F is DR-submodular if $\mathbf{x} \preceq \mathbf{y}$ implies that $\nabla F(\mathbf{x}) \succeq F(\mathbf{y})$.

4. Problem Statement

In this section, we formulate the main problem of this paper: maximizing submodular set functions subject to distributed partition matroid constraints.

Problem 1 (Submodular Maximization Subject to a Distributed Partition Matroid Constraint) *Consider a collection of n agents that form the set $\mathcal{N} = \{1, \ldots, n\}$. Let $f : 2^{\mathcal{Y}} \mapsto \mathbb{R}_+$ be a normalized and monotone submodular set function and let $\mathcal{Y}_1, \ldots, \mathcal{Y}_n$ be a pairwise disjoint partition of a finite ground set $\mathcal{Y}$, wherein each agent $i \in \mathcal{N}$ can only choose actions from its local strategy set $\mathcal{Y}_i$. Furthermore, consider the partition matroid $(\mathcal{Y}, \mathcal{I})$, where*

$$\mathcal{I} := \{\mathcal{S} \subseteq \mathcal{Y} \,:\, |\mathcal{Y}_i \cap \mathcal{S}| \leq 1 \text{ for } i = 1, \ldots, n\}. \tag{2}$$

The problem of submodular maximization subject to a distributed partition matroid constraint is to maximize f by selecting a set $\mathcal{S} \subseteq \mathcal{Y}$ from the family of allowable subsets so that $\mathcal{S} \in \mathcal{I}$. Formally:

$$\begin{aligned} &\text{maximize} && f(\mathcal{S}) && \text{(3a)} \\ &\text{subject to} && \mathcal{S} \in \mathcal{I} && \text{(3b)} \end{aligned}$$

In effect, the distributed partition matroid constraint in Problem 1 enforces that each agent $i \in \mathcal{N}$ can choose at most one action from its local strategy set $\mathcal{Y}_i$. Note that in this setting, each agent can only choose actions from its own local strategy set. Therefore, this problem is distributed in the sense that agents can only determine the actions taken by other agents by directly communicating with one another.

4.1. Sequential greedy algorithm

It is well known that the sequential greedy algorithm (SGA), in which each agent $i \in \mathcal{N}$ chooses an action sequentially based on

$$\mathbf{y}_i = \arg\max_{\mathbf{y} \in \mathcal{Y}_i} f(\mathbf{y}|\mathcal{S}_{i-1}) \tag{4}$$

where $\mathcal{S}_{i-1} = \{\mathbf{y}_1, \ldots, \mathbf{y}_{i-1}\}$, approximates the optimal solution to within a multiplicative factor of $1/2$ (Gharesifard and Smith, 2017). The drawbacks of this algorithm are twofold. Firstly, as we will show, our algorithm achieves the tight $(1 - 1/e)$ approximation of the optimal solution, which is known to be the best possible approximation unless $\mathbf{P} = \mathbf{NP}$. Secondly, as its name suggests, SGA is sequential in nature and therefore it scales very poorly in the number of agents. That is, each agent must wait for each of the previous agents to compute their contribution to the optimal set $\mathcal{S}^*$. Notably, our algorithm does not suffer from this sequential dependence.

4.2. Continuous Extension of Problem 1

Sequential algorithms such as SGA can only achieve a $1/2$ approximation of the optimal solution. To achieve the best possible $(1 - 1/e)$ approximation of the optimal solution, it is necessary to extend Problem 1 to the continuous domain via the so-called *multilinear extension* of the submodular objective function f (Nemhauser et al., 1978). Thus, the method we use in this work to achieve the tight $(1 - 1/e)$ approximation relies on the continualization of Problem 1. Importantly, it has been shown that Problem 1 and the optimization problem engendered by lifting Problem 1 to the continuous domain via this multilinear extension yield the same solution (Calinescu et al., 2011). Furthermore, by applying proper rounding techniques, such as those described in Section 5.1 of (Mokhtari et al., 2018) and in (Calinescu et al., 2011) and (Chekuri et al., 2014) to the continuous relaxation of Problem 1, one can obtain the tight $(1-1/e)$ approximation for Problem 1. Therefore, our approach in this paper will be to lift Problem 1 to the continuous domain. We formulate this problem in the following way:

Problem 2 (Continuous Extension of Problem 1) *Consider the conditions of Problem 1. Define the DR-submodular continuous multilinear extension $F : \mathcal{X} \mapsto \mathbb{R}_+$ of the objective function f in Problem 1 by*

$$F(\mathbf{y}) := \sum_{\mathcal{S} \subseteq \mathcal{Y}} f(\mathcal{S}) \prod_{i \in \mathcal{S}} y_i \prod_{j \notin \mathcal{S}} (1 - y_j) \tag{5}$$

and let $\mathcal{P} \subseteq \mathcal{X}$ be the matroid polytope $\mathcal{P} := \mathbf{conv}\{1_{\mathcal{S}} \ : \ \mathcal{S} \in \mathcal{I}\}$ where $\mathcal{I}$ is the family of sets defined in (2). The continuous relaxation of Problem 1 is formally defined by

$$\text{maximize} \quad F(\mathbf{y}) \tag{6a}$$

$$\text{subject to} \quad \mathbf{y} \in \mathcal{P} \tag{6b}$$

Observe that Problem 2 is *distributed* in the sense that each agent $i \in \mathcal{N}$ is associated with its own distinct continuous strategy space $\mathcal{P}_i$. Formally, the set $\mathcal{P}_i$ is defined as $\mathcal{P}_i := \mathbf{conv}\{1_{\mathcal{S}} \ : \ S \subseteq \mathcal{I}_i\}$, where $\mathcal{I}_i := \{\mathcal{S} \subseteq \mathcal{Y} \ : \ |\mathcal{Y}_i \cap \mathcal{S}| \leq 1\}$. In this way, $\mathcal{P} = \cap_{i=1}^{n} \mathcal{P}_i$. In this way, the sets $\mathcal{P}_i$ play similar roles in Problem 2 as the sets $\mathcal{Y}_i$ do in Problem 1.

Note that Problem 2 is nonconvex, and therefore cannot be solved by classical convex solvers or algorithms. Further, we assume that each agent $i \in \mathcal{N}$ can compute the multilinear extension F of the submodular objective function f in (3a) and the gradient of F.

5. Constraint-Distributed Continuous Greedy

In this section, we present Constraint-Distributed Continuous Greedy (`CDCG`), a decentralized algorithm for solving Problem 2. The pseudo-code of `CDCG` is described in Algorithm 1. At a high level, this algorithm involves updating each agent's local decision variable based on the aggregated belief of a small group of other agents about the best control policy. In essence, inter-agent communication within small groups of agents facilitates local decision making.

For clarity, we introduce a simple framework for the inter-agent communication structure. In `CDCG`, agents $i \in \mathcal{N} = \{1, \dots, n\}$ share their decision variables $\mathbf{y}_i$ with a small subset of *local* agents in $\mathcal{N}$. To encode the notion of locality, suppose that each agent $i \in \mathcal{N}$ is a node in a bidirectional *communication graph* $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ in which $\mathcal{E}$ denotes the set of edges. Given this structure, we assume that each agent $i \in \mathcal{N}$ can only communicate its decision variable $\mathbf{y}_i$ with its direct neighbors in $\mathcal{G}$. Let us denote the neighbor set of agent $i \in \mathcal{N}$ by $\mathcal{N}_i$. Then the set of edges $\mathcal{E}$ can be written $\{(i, j) \ : \ j \in \mathcal{N}_i\}$. We adopt this notation for the remainder of this paper.

5.1. Intuition for the CDCG algorithm

The goal of `CDCG` at a given node $i \in \mathcal{N}$ is to learn the local decision variable $\mathbf{y}_i$. `CDCG` is run at each node in $i \in \mathcal{N}$ to assemble the collection $\{\mathbf{y}_1^T, \dots, \mathbf{y}_n^T\}$ where T is a given positive integer; this collection represents an approximate solution to Problem 2 and guarantees that each agent contributes at most one element to the solution. Then, by applying proper rounding techniques to each element of the collection such as those discussed in (Mokhtari et al., 2018; Calinescu et al., 2011; Chekuri et al., 2014), we obtain a solution to Problem 1. In the proceeding sections, we show that this solution achieves the tight $(1 - 1/e)$ approximation of the optimal solution.

In the analysis of `CDCG`, we add the superscript t to the vectors $\mathbf{v}_i^t$ and $\mathbf{y}_i^t$ defined in Algorithm 1. This superscript denotes the iteration number so that $\mathbf{y}_i^t$ and $\mathbf{v}_i^t$ represent the values of the local variables $\mathbf{y}_i$ and $\mathbf{v}_i$ at iteration $t \in \{1, \dots, T\}$ respectively.

5.2. Description of the steps for CDCG (Algorithm 1)

From the perspective of node $i \in \mathcal{N}$, `CDCG` takes two arguments: nonnegative weights w_{ij} for each $j \in \mathcal{N}_i \cup \{i\}$ and a positive integer T. The weights w_{ij} correspond to the i^{th} row in a doubly-stochastic weight matrix $\mathbf{W}$ and T is the number of iterations for which the algorithm will run. The weight matrix $\mathbf{W}$ is a design parameter of the problem and must fulfill a number of technical requirements that are fully described in Appendix A. Before any computation, the local decision variable $\mathbf{y}_i$ is initialized to the zero vector.

Computation proceeds in T rounds. In each round, the first step is to calculate the gradient of the multilinear extension function F evaluated at the local decision variable $\mathbf{y}_i^{t-1}$ from the previous iteration. Thus, in line 3 of Algorithm 1, we calculate the ascent direction $\mathbf{v}_i^t$ at iteration t in the following way:

$$\mathbf{v}_i^t = \operatorname*{arg\,max}_{\mathbf{x} \in \mathcal{P}_i \cap \mathcal{C}_i} \left\langle \nabla F(\mathbf{y}_i^{t-1}), \mathbf{x} \right\rangle .$$

Intuitively, one can think of $\mathbf{v}_i^t$ as the vector from the set $\mathcal{P}_i \cap \mathcal{C}_i$ that is most aligned with $\nabla F(\mathbf{y}_i^{t-1})$. To define the set $\mathcal{C}_i$, first define the set $\mathcal{J}_i$ as the set of indices of the elements in $\mathcal{Y}$ that correspond

Algorithm 1 Constraint-Distributed Continuous Greedy (`CDCG`) at node i

Require: Weights w_{ij} for each neighbor $j \in \mathcal{N}_i \cup \{i\}$ and number of rounds $T \in \mathbb{Z}_{++}$
Returns: Local solution $\mathbf{y}_i^\star$ for node $i \in \mathcal{N}$ to Problem 1

1: Initialize local vectors $\mathbf{y}_i^0 = \mathbf{0}_{|\mathcal{Y}|}$
2: **for** $t = 1, 2, \ldots, T$ **do**
3: • Calculate an ascent direction for the multilinear extension function F via:

$$\mathbf{v}_i^t \leftarrow \underset{\mathbf{x} \in \mathcal{P}_i \cap \mathcal{C}_i}{\arg\max} \left\langle \nabla F\left(\mathbf{y}_i^{t-1}\right), \mathbf{x} \right\rangle$$

4: • Update the local variable $\mathbf{y}_i^t$ using the ascent direction $\mathbf{v}_i^t$ via:

$$\mathbf{y}_i^t \leftarrow \sum_{j \in \mathcal{N}_i \cup \{i\}} w_{ij} \mathbf{y}_j^{t-1} + \frac{n}{T} \mathbf{v}_i^t$$

5: **end for**
6: $\mathbf{y}_i^\star \leftarrow \texttt{Round}\left(\mathbf{y}_i^T\right)$

to elements in $\mathcal{Y}_i$. Then

$$\mathcal{C}_i := \left\{ \mathbf{x} \in \mathbb{R}_+^{|\mathcal{Y}|} \ : \ x_j = 0 \quad \forall j \notin \mathcal{J}_i \right\}. \tag{7}$$

Using this notation, we can equivalently define $\mathcal{P}_i = \{\mathbf{x} \in \mathbb{R}_+^{|\mathcal{Y}|} \ : \ \sum_{j \in \mathcal{J}_i} x_j \leq 1\}$. Next, in line 4 of Algorithm 1, $\mathbf{y}_i$ is updated by setting

$$\mathbf{y}_i^t = \sum_{j \in \mathcal{N}_i \cup \{i\}} w_{ij} \mathbf{y}_j^{t-1} + \frac{n}{T} \mathbf{v}_i^t.$$

In this way, the governing principle is to collaboratively accumulate the local belief about the optimal decision $\mathbf{y}_i^{t-1}$ and to then move in the approximate direction of steepest ascent from this point.

After T rounds of computation at each node $i \in \mathcal{N}$, we obtain a local decision variable $\mathbf{y}_i^T$ at each node. By applying proper rounding techniques, we obtain a decision variable for each agent $i \in \mathcal{N}$. Rounding in a decentralized manner is discussed in Section 5.1 of (Mokhtari et al., 2018). The rounding techniques of (Mokhtari et al., 2018) build on "pipage rounding" (Calinescu et al., 2011) and "swap rounding" (Chekuri et al., 2014), which are both centralized rounding techniques. The collection of these decision variables form the set $\mathcal{S}^*$, which represents our solution to Problem 1.

6. Convergence Analysis

The main result in this paper is to show that in the distributed setting of Problem 2, `CDCG` achieves a tight $(1 - 1/e)$ multiplicative approximation of the optimal solution. The following theorem summarizes this result.

Theorem 3 *Consider the `CDCG` algorithm described in Algorithm 1. Let $\mathbf{y}^*$ denote the global maximizer of the optimization problem defined in Problem 2, and assume that a positive integer T and a doubly-stochastic weight matrix $\mathbf{W}$ are given. Then provided that the assumptions outlined* *in Appendix A hold, for all nodes $i \in \mathcal{N}$, the local variables $\mathbf{y}_i^T$ obtained after T iterations satisfy*

$$F(\mathbf{y}_i^T) \geq \left(1 - \frac{1}{e}\right) F(\mathbf{y}^*) - \left[\frac{LD^2}{2T} + \frac{LD^2(n^2 + n^{5/2}) + n^{5/2}DG}{T(1-\beta)}\right] \tag{8}$$

where D, G, L, and β are problem-dependent constants that are formally defined in Appendices A and B.

Succinctly, Theorem 3 means that the sequence of local iterates generated by CDCG achieves the optimal approximation ratio $(1 - 1/e)$ and that the error term vanishes at a linear rate of $\mathcal{O}(1/T)$. That is,

$$F(\mathbf{y}_i^T) \geq \left(1 - \frac{1}{e}\right) F(\mathbf{y}^*) - \mathcal{O}\left(\frac{1}{T}\right),$$

which implies that each agent reaches an objective value larger than $(1 - 1/e - \epsilon)\mathbf{y}^*$ after $\mathcal{O}(1/\epsilon)$ rounds of communication. Previous work can only guarantee an objective value of $(1/2)\mathbf{y}^*$ (Gharesifard and Smith, 2017). We provide the proof of this theorem and supporting lemmas in Appendices B and C.

7. Simulation Results

To evaluate the proposed algorithm, we consider a multi-agent area coverage problem. In this setting, each agent $i \in \mathcal{N}$ is constrained to move in a two-dimensional grid. We assume that each agent has a finite radius r so that it can observe those grid points that lie with a square with sidelength $2r + 1$. The objective is for the agents to collectively maximize the cardinality of the union of their observation sets of grid points. In other words, given an initial configuration, the problem is to choose an action for each agent that maximizes the overall coverage of the grid. The top three panels of Figure 1 show various configurations of agents in this two-dimensional grid.

Consider an initial configuration of n agents in states $\mathbf{y}_i \in \mathbb{Z}^2$ for $i \in \{1, \ldots, n\}$ with the dynamic constraint $\mathbf{y}_i^{t+1} = \mathbf{y}_i^t + \mathbf{u}_i^t$, where $\mathbf{u}_i^t$ is a control input from a discrete set

$$\mathcal{U} = \{(0, 1), (0, -1), (-1, 0), (1, 0), (0, 0)\}.$$

Elements from this set represent the admissible actions for each agent in the two-dimensional grid.

In our simulation, we compared the performance of SGA against CDCG on the coverage task posed above for a variable number of agents. For simplicity, we assumed that the underlying communication graph $\mathcal{G}$ used in CDCG was fully connected and that each value in the weight matrix $\mathbf{W}$ was $1/n$. A random initialization for each agent's position and the coverage achieved by CDCG and SGA are shown in the top three panels of Figure 1 respectively. We compared the performance of these algorithms across ten random initializations of starting locations for the agents; the mean performance of each algorithm and the respective standard deviations are shown in the bottom left panel of Figure 1. In each trial, we ran both algorithms 50 times, each of which produced a control input $\mathbf{u}_i$ for each agent. For each initialization, we ran CDCG for $T = 100$ iterations. Note that as the number of agents increases, CDCG is optimal or near optimal in each case; however for larger than eight agents, the performance of SGA begins to fall away from the optimal.

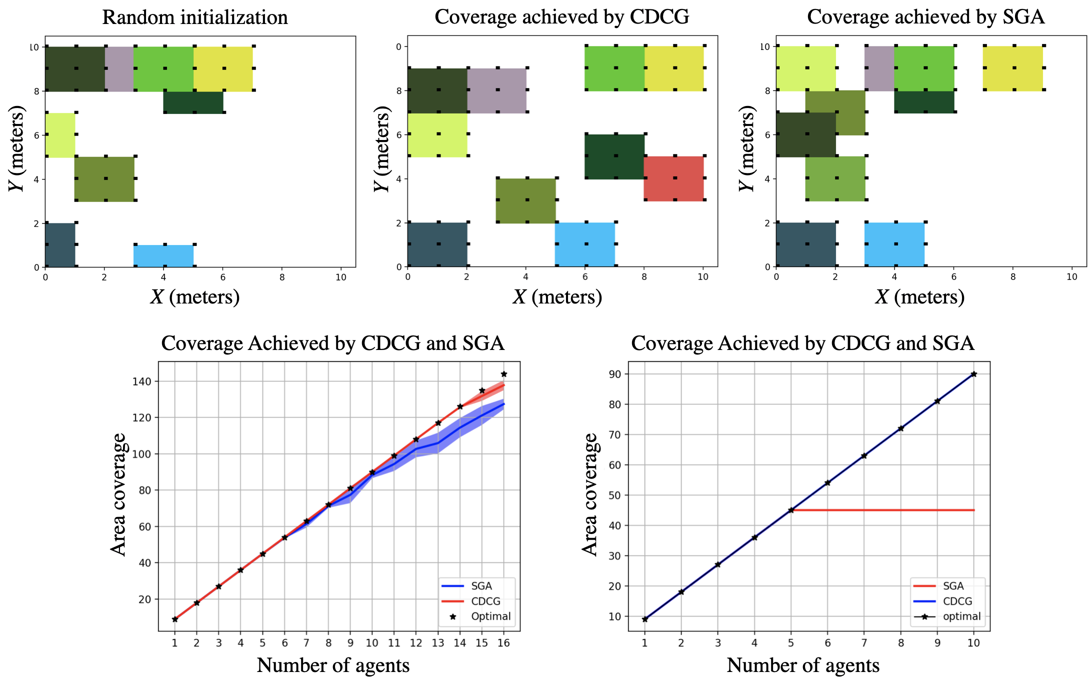

Figure 1: **Area coverage simulation results for `CDCG` and `SGA`.** (Top left) Random initialization of $n = 10$ agents in a 10×10 grid. (Top middle & right) Coverage achieved by `CDCG` (top middle) and `SGA` (top right) from the random initialization shown in the top left panel. (Bottom left) Comparison of the mean coverage achieved by `CDCG` and `SGA` averaged over 10 random initializations. (Bottom right) Comparison of the coverage achieved by `CDCG` and `SGA` for a setting in which each agent's starting point is the center of the grid.

We also compared the coverages achieved by `CDCG` and `SGA` for a setting in which each agent's starting position is the center of the grid. The results of this experiment are shown in the bottom right panel of Figure 1. In this plot, we averaged the performance over 15 independent trials; in each trial, we ran `CDCG` for $T = 100$ iterations. Interestingly, `SGA` converges to a local maximum in this problem, whereas `CDCG` achieves the optimal value.

8. Conclusion

In this work, we described an approach for achieving the optimal approximation to a class of submodular optimization problems subject to a *distributed* partition matroid constraint. The algorithm we proposed outperforms the sequential greedy algorithm in two senses: (1) `CDCG` achieves the tight $(1 - 1/e)$ approximation for the optimal solution whereas SGA can only achieve a $1/2$ approximation; and (2) `CDCG` imposes a limited communication structure on this problem, which allows for significant gains via parallelization. We showed empirically via an area coverage simulation with multiple agents that `CDCG` outperforms the greedy algorithm.

Acknowledgments

This research is supported by NSF award CPS-1837253, NSF CAREER award CIF 1943064, and Air Force Office of Scientific Research Young Investigator Program (AFOSR-YIP) under award FA9550-20-1-0111.

References

Arman Adibi, Aryan Mokhtari, and Hamed Hassani. Submodular meta-learning. *Advances in Neural Information Processing Systems*, 33, 2020.

Rafael Barbosa, Alina Ene, Huy Nguyen, and Justin Ward. The power of randomization: Distributed submodular maximization on massive datasets. In *International Conference on Machine Learning*, pages 1236–1244, 2015.

Niv Buchbinder, Moran Feldman, Joseph Naor, and Roy Schwartz. Submodular maximization with cardinality constraints. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 1433–1452. SIAM, 2014.

Niv Buchbinder, Moran Feldman, Joseph Seffi, and Roy Schwartz. A tight linear time (1/2)-approximation for unconstrained submodular maximization. *SIAM Journal on Computing*, 44 (5):1384–1402, 2015.

Gruia Calinescu, Chandra Chekuri, Martin Pal, and Jan Vondrák. Maximizing a monotone submodular function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011.

Chandra Chekuri, Jan Vondrák, and Rico Zenklusen. Submodular function maximization via the multilinear relaxation and contention resolution schemes. *SIAM Journal on Computing*, 43(6): 1831–1879, 2014.

Lin Chen, Mingrui Zhang, Hamed Hassani, and Amin Karbasi. Black box submodular maximization: Discrete and continuous settings. In *International Conference on Artificial Intelligence and Statistics*, pages 1058–1070, 2020.

Andrew Clark, Basel Alomair, Linda Bushnell, and Radha Poovendran. Scalable and distributed submodular maximization with matroid constraints. In *2015 13th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, pages 435–442. IEEE, 2015.

Micah Corah and Nathan Michael. Efficient online multi-robot exploration via distributed sequential greedy assignment. In *Robotics: Science and Systems*, volume 13, 2017.

Micah Corah and Nathan Michael. Distributed submodular maximization on partition matroids for planning on large sensor networks. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 6792–6799. IEEE, 2018.

Micah Corah and Nathan Michael. Distributed matroid-constrained submodular maximization for multi-robot exploration: Theory and practice. *Autonomous Robots*, 43(2):485–501, 2019.

Josip Djolonga, Sebastian Tschiatschek, and Andreas Krause. Variational inference in mixed probabilistic submodular models. In *Advances in Neural Information Processing Systems*, pages 1759–1767, 2016.

Bin Du, Kun Qian, Christian Claudel, and Dengfeng Sun. Jacobi-style iteration for distributed submodular maximization. *arXiv preprint arXiv:2010.14082*, 2020.

Bahman Gharesifard and Stephen L Smith. Distributed submodular maximization with limited information. *IEEE transactions on control of network systems*, 5(4):1635–1645, 2017.

Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42:427–486, 2011.

David Grimsman, Mohd Shabbir Ali, Joao P Hespanha, and Jason R Marden. The impact of information in greedy submodular maximization. *IEEE Transactions on Control of Network Systems*, 2018.

Hamed Hassani, Mahdi Soltanolkotabi, and Amin Karbasi. Gradient methods for submodular maximization. In *Advances in Neural Information Processing Systems*, pages 5841–5851, 2017.

Yusuo Hu, Hua Chen, Jian-guang Lou, and Jiang Li. Distributed density estimation using non-parametric statistics. In *27th International Conference on Distributed Computing Systems (ICDCS'07)*, pages 28–28. IEEE, 2007.

Baharan Mirzasoleiman, Amin Karbasi, Rik Sarkar, and Andreas Krause. Distributed submodular maximization: Identifying representative elements in massive data. In *Advances in Neural Information Processing Systems*, pages 2049–2057, 2013.

Baharan Mirzasoleiman, Amin Karbasi, Rik Sarkar, and Andreas Krause. Distributed submodular maximization. *The Journal of Machine Learning Research*, 17(1):8330–8373, 2016.

Aryan Mokhtari, Hamed Hassani, and Amin Karbasi. Decentralized submodular maximization: Bridging discrete and continuous settings. *arXiv preprint arXiv:1802.03825*, 2018.

Aryan Mokhtari, Hamed Hassani, and Amin Karbasi. Stochastic conditional gradient methods: From convex minimization to submodular maximization. *Journal of Machine Learning Research*, 21(105):1–49, 2020.

George L Nemhauser and Laurence A Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Mathematics of operations research*, 3(3):177–188, 1978.

George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14(1):265–294, 1978.

Brent Schlotfeldt, Dinesh Thakur, Nikolay Atanasov, Vijay Kumar, and George J Pappas. Anytime planning for decentralized multirobot active information gathering. *IEEE Robotics and Automation Letters*, 3(2):1025–1032, 2018.

Amarjeet Singh, Andreas Krause, Carlos Guestrin, and William J Kaiser. Efficient informative sensing using multiple robots. *Journal of Artificial Intelligence Research*, 34:707–755, 2009.

Kai Wei, Yuzong Liu, Katrin Kirchhoff, and Jeff Bilmes. Using document summarization techniques for speech data subset selection. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 721–726, 2013.

Laurence A Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2(4):385–393, 1982.

Jiahao Xie, Chao Zhang, Zebang Shen, Chao Mi, and Hui Qian. Decentralized gradient tracking for continuous dr-submodular maximization. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2897–2906, 2019.

Minyi Zhong and Christos G Cassandras. Distributed coverage control and data collection with mobile sensor networks. *IEEE Transactions on Automatic Control*, 56(10):2445–2455, 2011.

Lifeng Zhou, Vasileios Tzoumas, George J Pappas, and Pratap Tokekar. Distributed attack-robust submodular maximization for multi-robot planning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2479–2485. IEEE, 2020.