ABSTRACT
Maximizing a monotone submodular function is a fundamental task in data mining, machine learning, economics, and statistics. In this paper, we present two communication-efficient decentralized online algorithms for the monotone continuous DR-submodular maximization problem, both of which reduce the number of per-function gradient evaluations and per-round communication complexity from $T^{3/2}$ to $1$. The first one, One-shot Decentralized Meta-Frank-Wolfe (Mono-DMFW), achieves a $(1-1/e)$-regret bound of $O(T^{1/2})$. As far as we know, this is the first one-shot and projection-free decentralized online algorithm for monotone continuous DR-submodular maximization. Next, inspired by the non-oblivious boosting function [29], we propose the Decentralized Online Boosting Gradient Ascent (DOBGA) algorithm, which attains a $(1-1/e)$-regret of $O(\sqrt{T})$. To the best of our knowledge, this is the first result to obtain the optimal $O(\sqrt{T})$ against a $(1-1/e)$-approximation with only one gradient inquiry for each local objective function per step. Finally, various experimental results confirm the effectiveness of the proposed methods.

CCS CONCEPTS
• Computing methodologies → Distributed algorithms; • Theory of computation → Online learning theory.

KEYWORDS
distributed data mining, online learning, submodular maximization

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must also be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

1 INTRODUCTION
With the rapid development of digital systems, communication, and sensing technologies [1, 23], numerous large-scale datasets are collected over the networked machines [22]. To deal with large-scale datasets in data-intensive applications, it is urgent to design algorithms in a decentralized manner, where computing units cooperatively optimize the global objective functions throughout local computation and network communications among each other. Clearly, compared to centralized algorithms, decentralized algorithms that efficiently exploiting dispersed computational resources are more scalable. Moreover, decentralized algorithms have advantages in relieving the data privacy risk, as computing nodes often only share very limited local information with each other [4].

Continuous DR-submodular maximization, an important subclass of the non-convex/non-concave optimization, has received considerable attention in recent years, since various applications in data mining, machine learning, economics, and statistics, such as viral marketing [16, 26], revenue maximization [3, 24], non-definite quadratic programming [15], determinantal point processes [17, 18], to name a few, can be modelled using this notion. Apparently, a myriad of the existing literature has studied the centralized and static monotone continuous DR-submodular maximization problems [2, 10, 11, 19]. However, in many real scenarios, the objectives not only are stored in a network of computing nodes but also vary with time [12]. Thus, in this paper, we focus on the decentralized online monotone continuous DR-submodular maximization, where only the stochastic gradient oracles are available.

Recently, Zhu et al. [31] proposed the first decentralized online algorithm for monotone continuous DR-submodular maximization, i.e., the decentralized Meta-Frank-Wolfe (DMFW), which achieves a $(1-1/e)$-approximation guarantee with an expected regret bound of $O(\sqrt{T})$ where $T$ is the time horizon. Noticeably, at each round, DMFW needs to inquire $T^{3/2}$ stochastic gradient estimates for each local objective function and then passes these gradient messages over the network one by one, resulting in a large computation and communication overhead when $T$ is large. In view of this, we consider the following question:
Can we design communication-efficient algorithms for the decentralized online monotone continuous DR-submodular maximization problem which guarantee the tight \((1 - 1/e)\) approximation ratio?

This paper provides an affirmative answer to this question by presenting two communication-efficient decentralized online algorithms for monotone continuous DR-submodular maximization, which both reduce the per-round communication complexity and the gradient evaluations of each local objective function at each round, and ‘Projection-free’ indicates whether the algorithm is projection-free. ‘DMFW’ refers to the algorithm in [31]. ‘Mono-DMFW’ and ‘DOBGA’ are our proposed algorithms.

| Method      | Ratio  | Regret                  | # Com | # Grad | Projection-free |
|-------------|--------|-------------------------|-------|--------|-----------------|
| DMFW        | 1−1/e  | \(O(\sqrt{T})\)         |       |        | Yes             |
| Mono-DMFW   | 1−1/e  | \(O(T^{3/4})\)          | 1     | 1      | Yes             |
| DOBGA       | 1−1/e  | \(O(\sqrt{T})\)         | 1     | 1      | No              |

Table 1: Comparison of decentralized algorithms for online continuous monotone DR-submodular function maximization with stochastic gradient oracles. ‘Ratio’ means approximation ratio; ‘# Com’ means the communication complexity for each node at each round; ‘# Grad’ means the number of stochastic gradient evaluations for each local objective function at each round, and ‘Projection-free’ indicates whether the algorithm is projection-free. ‘DMFW’ refers to the algorithm in [31]. ‘Mono-DMFW’ and ‘DOBGA’ are our proposed algorithms.

Due to limited space, we put some proofs in the full version of our paper which will be released upon acceptance of this paper.

2 RELATED WORKS

Continuous DR-submodular maximization has been extensively investigated since it admits efficient approximate maximization routines. First, in the deterministic setting, [2] presented a variant of Frank-Wolfe for maximizing centralized and static monotone continuous DR-submodular function, which achieves the tight \((1 - 1/e)\)-approximation guarantee. Next, [19] proposed the stochastic continuous greedy algorithm, which attains a \((1 - 1/e)OPT - \epsilon\) after \((O(1/e^2))\) iterations where \(OPT\) is the optimal value. Then, an accelerated stochastic continuous greedy algorithm is presented in [10], which guarantees a \((1 - 1/e)\)-approximation after \(O(1/e^2)\) iterations. In the online setting, [5, 6] first proposed a novel Meta-Frank-Wolfe algorithm achieving the tight \((1 - 1/e)\)-approximation of square-root regret under both deterministic and stochastic settings. After that, [28] proposed the blocking techniques to improve the per-function gradient evaluations of Meta-Frank-Wolfe from \(T^{3/2}\) to \(T^{1/2}\) to \(1\). [11] proceeded to show that the canonical stochastic gradient ascent achieves \((1/2)\)-approximation guarantee. Like the offline gradient ascent algorithm [11], the classical online gradient ascent also attains the suboptimal \((1/2)\)-approximation guarantee. Recently, [29] have proposed an auxiliary function to boost the approximation ratio of the offline and online gradient ascent algorithms from \(1/2\) to \(1 - 1/e\).

A large body of literature has studied decentralized convex optimization problems. [21] first proposed the decentralized gradient descent algorithm, which combines the consensus technique with local gradient descent. [27] proceeded to derive the convergence of decentralized gradient descent. However, these works assume that the objective function is unchanged with time. To tackle the varying objectives, [25] then proposed the framework of decentralized online optimization, where they proved that the decentralized online projected gradient descent algorithm achieves a regret of \(O(\log(T))\) and \(O(\sqrt{T})\) for strongly convex objectives and convex function, respectively. Next, [14] presented the distributed online dual averaging algorithm with the regret of \(O(\sqrt{T})\) for convex objective. Inspired by the online conditional gradient [12], [30] proposed a projection-free decentralized online algorithm for convex optimization and proved a regret bound of \(O(T^{3/4})\). As for the decentralized submodular maximization, in the offline setting, [20] proposed a decentralized continuous greedy algorithm with the tight \((1 - 1/e)\)-approximation guarantee. Next, [7] presented a sample efficient decentralized algorithm for continuous DR-submodular maximization. Under the online scenario, [31] merged the consensus technique and variance reduction technique into the Meta-Frank-Wolfe [6] to propose the decentralized Meta-Frank-Wolfe (DMFW) with an expected \((1 - 1/e)\)-regret of \(O(\sqrt{T})\), which needs to inquire \(T^{3/2}\) stochastic gradients for each local function and communicate these information over network.

We make a comparison between the state-of-the-art DMFW and our algorithms for decentralized online monotone continuous DR-submodular maximization in table 1.
3 PRELIMINARIES

In this section, we introduce some necessary notions and formulate our problem.

3.1 Notations

In this paper, we use \( \mathbb{R} \) and \( \mathbb{R}_+ \) to denote the set of real numbers and non-negative real numbers, respectively. Besides, the lower boldface (e.g. \( x \)) denotes a column vector with suitable dimension and uppercase boldface (e.g. \( A \)) for a matrix. \( x^T \) and \( A^T \) denote the transpose of the vector \( x \) and the matrix \( A \), respectively. Specially, \( I \) and \( 1 \) represent the identity matrix and the vector whose all entries are 1, respectively. We denote the \( i \)-th element of a vector \( x \) as \( (x)_i \). Given two \( n \)-dimensional vector \( x \) and \( y \), we say \( x \geq y \) if \( (x)_i \geq (y)_i \) for all \( i \in [n] \). The product \( (x,y) = \sum_i (x)_i (y)_i \) and the norm \( \|x\| = \sqrt{(x,x)} \). Additionally, for any convex set \( \mathcal{K} \), the radius \( r(\mathcal{K}) = \max_{x \in \mathcal{K}} \|x\| \) and the diameter \( \text{diam}(\mathcal{K}) = \max_{x,y \in \mathcal{K}} \|x - y\| \). For any positive integer \( K \), \( \{K\} \) denotes the set \( \{1, \ldots, K\} \). The symbol \( \otimes \) denotes the Kronecker product.

3.2 Continuous DR-Submodularity

In this subsection, we recall the definition of continuous DR-submodular function. A continuous function \( f: \mathcal{X} \rightarrow \mathbb{R}_+ \) is DR-submodular if

\[
f(x + ze_i) - f(x) \leq \langle \nabla f(x), y + ze_i \rangle - f(y),
\]

where \( e_i \) is the \( i \)-th basic vector, \( x \geq y \) and \( z \in \mathbb{R}_+ \) such that \( x + ze_i \in \mathcal{X} \). Here, \( \mathcal{X} = [0, \rho]^{n} \) such that each \( \rho_i \) is a compact interval in \( \mathbb{R}_+ \). In this paper, we assume \( \mathcal{X} = [0, a_i] \) where \( a_i \in \mathbb{R}_+ \). If the DR-submodular function \( f \) is differentiable, we have \( \nabla f(x) \leq \nabla f(y) \) for any \( x \geq y \). Furthermore, the DR-submodularity of function \( f \) implies that \( f \) is concave along the positive direction [3], i.e., for any \( x, y \in \mathcal{X} \) and \( y \geq x \), it holds that

\[
f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle.
\]

If \( f \) is twice differentiable, the continuous DR-submodularity is equivalent to

\[
\forall i, j \in [n], \forall x \in \mathcal{X}, \frac{\partial^2 f(x)}{\partial x_i \partial x_j} \leq 0.
\]

Moreover, a differentiable function \( f: \mathcal{X} \rightarrow \mathbb{R}_+ \) is called L-smooth if for any \( x, y \in \mathcal{X} \), we have

\[
f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2} \|y - x\|^2,
\]

which implies that the gradient of \( f \) is \( L \)-Lipschitz, i.e.,

\[
\|\nabla f(x) - \nabla f(y)\| \leq L \|x - y\|.
\]

Finally, we say that the function \( f \) is monotone iff \( f(x) \geq f(y) \) for any \( x \geq y \) and \( x, y \in \mathcal{X} \).

3.3 Problem Formulation

In this subsection, we introduce the decentralized online monotone continuous DR-submodular maximization problem. We use the undirected connected graph \( \mathcal{G} = (\mathcal{V}, \mathcal{W}) \) to denote the network of the computing nodes, where \( \mathcal{V} = \{1, \ldots, N\} \) represents the set of nodes and \( \mathcal{W} \subseteq \mathcal{V} \times \mathcal{V} \) denotes the set of edges. The symbol \( \mathcal{N}_i := \{j \in \mathcal{V} \mid (i, j) \in \mathcal{W}\} \) denotes the set of the neighbors of node \( i \). In this paper, we assume that each node \( i \in \mathcal{V} \) only gets access to its local objective function and communicates the information with its neighbors in \( \mathcal{N}_i \). Also, we define \( a_{ij} \geq 0 \) to be the weight that node \( i \) assigns to node \( j \). If \( (i, j) \notin \mathcal{W}, a_{ij} = 0 \). Moreover, the weight matrix \( A = [a_{ij}] \in \mathbb{R}_+^{N \times N} \) satisfies the following assumption.

Assumption 1. The matrix \( A \in \mathbb{R}_+^{N \times N} \) is symmetric and doubly stochastic, i.e., \( A^T = A \) and \( A = 1 \). Regarding the eigenvalue of \( A \), i.e., \( 1 = \lambda_1(A) \geq \lambda_2(A) \ldots \geq \lambda_N(A) \geq -1 \), the \( \beta < 1 \), where \( \beta = \max(|\lambda_2(A)|, |\lambda_N(A)|) \) is the second largest magnitude of the eigenvalues of \( A \).

In a \( T \)-round decentralized online optimization, each node \( i \) first chooses an action \( x_i(t) \) from the constraint set \( \mathcal{K} \) at each round. Then, the adversary reveals a continuous DR-submodular function \( f_i : \mathcal{X} \rightarrow \mathbb{R}_+ \) and feeds back the reward \( f_i(x_i(t)) \) to the node \( i \). The goal of nodes is to online maximize the aggregate continuous DR-submodular function, i.e., \( \max_{x \in \mathcal{K}} \sum_{i=1}^{T} \sum_{t=1}^{T} f_i(x_i(t)) \). Note that, according to [2], maximizing a monotone continuous DR-submodular function subject to a general convex constraint is NP-hard. As a result, we turn to the \( \alpha \)-regret of each node \( j \in \mathcal{V} \) as follows:

\[
R_\alpha(T, j) = \alpha \sup_{x \in \mathcal{K}} \frac{1}{N} \sum_{i=1}^{T} \sum_{t=1}^{T} f_i(x_i(t)) - \frac{1}{N} \sum_{i=1}^{T} \sum_{t=1}^{T} f_i(x_j(t)),
\]

where \( \alpha \) is the approximation ratio. Therefore, in this paper, we aim to design communication-efficient decentralized online algorithms such that 1) at each round, each node \( i \in \mathcal{V} \) only passes messages to its neighbors \( \mathcal{N}_i \), and 2) the \( \alpha \)-regret of each node \( i \in \mathcal{V} \) is sublinear in \( T \), namely, \( \lim_{T \rightarrow \infty} \max_{i \in \mathcal{V}} R_\alpha(T, i)/T = 0 \). In the following sections, we will consider the tight approximation ratio \( \alpha = 1 - 1/e \) [2]. Furthermore, we make the following assumptions throughout this paper.

Assumption 2. The domain \( \mathcal{K} \subseteq \mathcal{X} \) is a bounded convex set.

Assumption 3. Each local function \( f_i : \mathcal{X} \rightarrow \mathbb{R}_+ \) is differentiable, monotone, continuous DR-submodular and \( L \)-smooth, where \( i \in [N] \) and \( t \in [T] \).

Assumption 4. For any \( t \in [T] \) and \( i \in [N] \), there exists a stochastic gradient oracle \( \nabla f_i(x) \) with \( \mathbb{E}((\nabla f_i(x))|x) = \nabla f_i(x) \) and \( \mathbb{E}((||\nabla f_i(x) - \nabla f_i(x)||)^2) \leq \sigma^2 \).

4 ONE-SHOT DECENTRALIZED META-FRANK-WOLFE ALGORITHM

To begin, we review the detailed results of DMFW algorithm [31], which is the first projection-free algorithm for the decentralized online monotone continuous DR-submodular maximization problem. The DMFW needs to inquire stochastic gradients at \( K \) different points for each local function and then sequentially passes these gradients information to the neighboring nodes, which will incur the \( K \) amounts of communications over the network at each round. [31] also have verified that the DMFW achieves a \((1 - 1/e)\)-regret of \( O(\sqrt{T} + \frac{\tau}{K^{1/3}}) \). In order to attain the lowest \( O(\sqrt{T}) \)-regret bound, we usually set the \( K = T^{3/7} \), which will incur huge gradient evaluations and prohibitive communication complexity when \( T \) is large.

As a result, in this section, we propose the one-shot decentralized Meta-Frank-Wolfe algorithm (Momo-DMFW), which reduces the
Algorithm 1 One-Shot Decentralized Meta-Frank-Wolfe (Mono-DMFW)

1: **Input:** Time horizon $T$, the number of nodes $N$, blocking parameter $Q$ and $K$ where $T = KQ$, weight matrix $A = \{a_{ij}\} \in \mathbb{R}^{N \times N}$, step size $\eta_k, \forall k \in [K]$, parameter $\gamma$, constraint set $\mathcal{K}$.

2: **Output:** $\{x_i(t) : i \in [N], t \in [T]\}$.

3: Initialize $K$ online linear optimization oracles over $\mathcal{K}$, $E_i^{(1)}, \ldots, E_i^{(K)}$ for each node $i \in [N]$.

4: For any $q \in [Q]$ and $i \in [N]$, initialize $d_i^{(0)}(q) = x_i^{(0)}(q) = g_i^{(0)}(q) = 0$.

5: for $q = 1, \ldots, Q$ do

6: for $k = 1, \ldots, K$ do

7: for $t = 1, \ldots, N$ do

8: Receive the update direction $d_i^{(k)}(q)$ which is the output of oracle $E_i^{(k)}$.

9: $x_i^{(k)}(q) = \sum_{j \in N_i \cup \{i\}} a_{ij} x_j^{(k-1)}(q) + \gamma v_i^{(k)}(q)$.

10: end for

11: end for

12: for $i = 1, \ldots, N$ do

13: for $t = (q-1)K + 1, \ldots, qK$ do

14: Node $i$ plays $x_i(t) = x_i^{(k)}(q)$ to get reward $f_{i,j}(x_i(t))$ and observes the stochastic gradient information of $f_{i,j}$.

15: end for

16: Generate a random permutation $\{t_i^{(1)}(q), \ldots, t_i^{(K)}(q)\}$ for $(q-1)K + 1, \ldots, qK$.

17: for $k = 1, \ldots, K$ do

18: $g_i^{(k)}(q) = (1-\eta_k)g_i^{(k-1)}(q) + \eta_k \nabla f_i^{(k)}(q,i)(x_i^{(k)}(q))$.

19: $d_i^{(k)}(q) = (1-\gamma) \sum_{j \in N_i \cup \{i\}} a_{ij} d_j^{(k-1)}(q) + \gamma g_i^{(k)}(q)$.

20: Feed back $\{d_i^{(k)}(q), v_i^{(k)}(q)\}$ as the payoff to be received by oracle $E_i^{(k)}$.

21: end for

22: end for

23: end for

number of both per-round communication complexity and per-function gradient evaluations from $K$ to $1$.

To obtain the one-shot decentralized algorithm, we adopt the blocking procedure in [28]. For each node $i \in \mathcal{N}$, we divide the $T$ objective functions $f_{1,i}, \ldots, f_{Q,i}$ into $Q$ equisized blocks of size $K$ where $T = KQ$. For instance, $\{f(Q-1)K+1,i, \ldots, fQK,i\}$ are included in the $q$-th block of node $i$. Then, we define the average function in the $q$-th block of node $i$ as $\bar{f}_{q,i} = \frac{1}{K} \sum_{k=1}^{K} f_{q,i}^{(k)}$ where $q \in [Q]$. If we view the $f_{1,i}, \ldots, f_{Q,i}$ as the virtual local objective functions for node $i$, we could obtain a new $Q$-round decentralized online problem. That is, after each node $i$ first chooses an action $y_i(q) \in \mathcal{K}$ at each round, the environment reveals the average reward function $\bar{f}_{q,i}$ and feeds back the reward $\bar{f}_{q,i}(y_i(q))$ to the node $i$. The goal of us is also to minimize the $\alpha$-regret $R_d(Q,i) = \alpha \sup_{x \in \mathcal{K}} \frac{1}{T} \sum_{t=1}^{T} \sum_{i \in \mathcal{I}} \bar{f}_{q,i}(x) - \frac{1}{T} \sum_{t=1}^{T} \sum_{i \in \mathcal{I}} \bar{f}_{q,i}(y_i(q))$.

Note that, if we play the same action $y_i(q) \in \mathcal{K}$ for the all upcoming objective functions in the $q$-th block of node $i$, the $\alpha$-regret of the original $T$-round decentralized online problem is $K$ times as that of the new $Q$-round game, i.e., $R_d(T,i) = K R_d(Q,i)$. Therefore, we have $R_d(T,i)/T = K R_d(Q,i)/T = R_d(Q,i)/Q$, meaning that an algorithm with sublinear regret for the new $Q$-round problem could provide a feasible solution with sublinear regret for the original $T$-round game. Since each local average function $\bar{f}_{q,i}$ is also monotone and continuous DR-submodular, a simple way to design a sublinear algorithm for the new $Q$-round problem is directly carrying the DMFW algorithm.

Similarly, DMFW algorithm needs to inquire stochastic gradients at $K$ different points for each local average function $\bar{f}_{q,i}$ and then sequentially communicates these gradients’ information with the neighboring nodes. Noticeably, in $q$-th block of each node $i \in [N]$, there exist $K$ different unbiased gradient oracles $\{\nabla f_{(q-1)K+1,i}, \ldots, \nabla f_{QK,i}\}$. Next, we present a method to generate stochastic gradients of $\bar{f}_{q,i}$ at $K$ different points throughout these $K$ gradient oracles $\{\nabla f_{(q-1)K+1,i}, \ldots, \nabla f_{QK,i}\}$. To be precise, let $\{t_1^{(1)}(q), \ldots, t_1^{(K)}(q)\}$ be a random permutation of the indices $(q-1)K+1, \ldots, qK)$ of node $i$. Then, for each $t_1^{(k)}(q)$ where $k \in [K]$, we have $\nabla f_{t_1^{(1)}(q),i}(x) = \bar{f}_{q,i}(x)$ and $\nabla f_{t_1^{(k)}(q),i}(x) = \nabla f_{t_1^{(k)}(q),i}(x)$. As a result, with only one gradient evaluation per function $f_{t_1^{(k)}(q),i}$, where $k \in [K]$, we can obtain $K$ unbiased stochastic gradients of the virtual objective function $\bar{f}_{q,i}$. In this way, for each node $i \in [N]$, we only need to inquire one gradient estimate of the local function $f_{t_1^{(1)}(q),i}$ and share this message with the neighbors, which successfully reduce the required number of both per-round communication complexity and per-function gradient evaluations from $K$ to $1$. Merging this blocking technique into the DMFW, we get the one-shot decentralized Meta-Frank-Wolfe (Mono-DMFW) in Algorithm 1.

In Algorithm 1, each node $i \in [N]$ at round $q$ keeps track of two local variables $x_i^{(k)}(q)$ and $d_i^{(k)}(q)$ which are iteratively updated using the information of the neighboring nodes. After receiving the local update direction $v_i^{(k)}(q)$, node $i \in [N]$ updates their local variable $x_i^{(k-1)}(q)$ by averaging their local and neighboring decisions and ascends in the direction $v_i^{(k)}(q)$ with stepsize $1/K$ (see line 9), i.e.,

$$x_i^{(k)}(q) = \sum_{j \in N_i \cup \{i\}} a_{ij} x_j^{(k-1)}(q) + \frac{1}{K} v_i^{(k)}(q),$$

where $a_{ij}$ is the weight that node $i$ assigns to node $j$. Similarly, we update local gradient approximation vector $d_i^{(k)}(q)$ according to the following rule (see line 19):

$$d_i^{(k)}(q) = (1-\gamma) \sum_{j \in N_i \cup \{i\}} a_{ij} d_j^{(k-1)}(q) + \gamma g_i^{(k)}(q),$$

where we view $g_i^{(k)}(q)$ as an approximation to the gradient of $\bar{f}_{q,i}$. We will show that Mono-DMFW achieves a $(1 - 1/e)$-regret bound of $O(T^{4/5})$. Before that, we first state the following assumption.

**Assumption 5.** For any linear maximization oracle $E_i^{(k)}$, the regret at horizon $t$ is at most $M_0 t^3$, where $M_0$ is a parameter. **Remark 1.** Notably, this is a basic assumption in the online meta-framework [5, 6]. Moreover, numerous online algorithms [13] can...
achieve $O(\sqrt{t})$-regret for linear objectives at horizon $t$, such as Regularized Follow the Leader and Online Gradient Descent.

**Theorem 1.** Under Assumption 1-5 and $\|\nabla f_i(x)\| \leq G$, if we set $\eta_t = \frac{k}{(t+3/2)}$ when $1 \leq t \leq \frac{K}{2} + 1$, and when $\frac{K}{2} + 2 \leq t \leq K$, $\eta_t = \frac{k}{(K + t + 1/2)}$, we have, for each node $j \in [N]$, $\mathbb{E}[R_j(T, j)] \leq NG\text{diam}(\mathcal{K})\frac{Q(1 + \log(K + 1))}{Y} + C_2Q(K^{2/3} + \frac{K\beta}{1 - (1 - \beta)\beta} + C_3K\sqrt{Q},$

where $C_1 = \frac{L_1\eta_t(K)}{2} + \frac{N\eta_t(K)\text{diam}(\mathcal{K})(G + G)}{1 - \beta}$, $C_2 = \text{diam}(\mathcal{K})(2G + L_1\eta_t(K))$, $C_3 = 2\text{diam}(\mathcal{K})\sqrt{Q}$, $C_4 = \text{diam}(\mathcal{K})\sqrt{Q}$, $C_5 = M_0N$, $C_6 = \max\{5^{2/3}G, 32(2G + L_1\eta_t(K))^{2/3}, 2.25(2G + L_1\eta_t(K))^{2/3}\}$, and $\alpha = 1 - 1/e$.

**Remark 2.** According to Theorem 1, if we set $K = T^{3/5}, Q = T^{2/5}$, and $Y = 1/(T^{1/5})$, Mono-DMFW attains the $(1 - 1/e)$-regret of $O(T^{3/5})$. As far as we know, this is the first projection-free decentralized online algorithm to achieve $(1-1/e)$ approximation ratio with communication complexity 1 for each node at each round, which greatly reduces the $T^{3/2}$ communication complexity of DMFW [31].

### 4.1 Convergence Analysis

This subsection provides a brief introduction to the theoretical guarantee of our proposed Mono-DMFW, i.e., Theorem 1. To begin, we define some auxiliary symbols for Algorithm 1: $\vec{x}^{(k)}(q) = \sum_{i=1}^{N} x_i^{(k)}(q)$, $\vec{c}(k) = \sum_{i=1}^{N} c_i^{(k)}(q)$, and $\vec{y}(q) = \sum_{i=1}^{N} y_i(q)$. First, we establish an upper bound of the distance between $\vec{x}^{(k+1)}(q)$ and $\vec{y}^{(k)}(q)$.

**Lemma 1.** Under Assumption 1, for each $0 \leq k \leq K - 1$, and $q \in [Q]$, we have $\|\vec{y}^{(k+1)}(q) - \vec{y}^{(k)}(q)\| \leq r(\mathcal{K})/K$.

**Proof.**

\[
\begin{align*}
\vec{x}^{(k+1)}(q) &= \frac{\sum_{i=1}^{N} x_i^{(k+1)}(q)}{N} \\
&= \frac{\sum_{j=1}^{N} (\sum_{i \in N_j(i)} a_{ij}x_j^{(k)}(q) + v_j^{(k+1)}(q))}{N} \\
&= \frac{\sum_{j=1}^{N} a_{ij}x_j^{(k)}(q) + \sum_{i=1}^{N} v_i^{(k+1)}(q)}{K} \\
&= \frac{\vec{x}^{(k)}(q) + \sum_{i=1}^{N} v_i^{(k+1)}(q)}{K}.
\end{align*}
\]

Due to $v_j^{(k+1)}(q) \in \mathcal{K}$, $\|\vec{x}^{(k+1)}(q) - \vec{x}^{(k)}(q)\| \leq \frac{\sum_{i=1}^{N} \|v_i^{(k+1)}(q)\|}{\sqrt{N}} \leq r(\mathcal{K})/K$.

Next, we derive the upper bound of the sum of the expected gap between the local estimate $x_i^{(k)}(q)$ and its average vector $\vec{x}^{(k)}(q)$.

**Lemma 2.** Under Assumption 1, for any $k \in [K]$ and $q \in [Q]$, we could conclude that

\[
\begin{equation}
\sqrt{\sum_{i=1}^{N} \|x_i^{(k)}(q) - \bar{x}^{(k)}(q)\|^2} \leq \frac{\sqrt{N}r(\mathcal{K})}{K(1 - \beta)}.
\end{equation}
\]

where $\beta$ is the second largest magnitude of the eigenvalues of matrix $A$, i.e., $\beta = \max(\lambda_2(A), |\lambda_3(A)|)$.

**Proof.** In order to verify Lemma 2, we define two auxiliary concatenation vectors:

\[
\begin{align*}
\vec{x}^{(k)}(q) &= [x_1^{(k)}(q); \ldots; x_N^{(k)}(q)] \in \mathbb{R}^{Nn}, \\
\vec{v}^{(k)}(q) &= [v_1^{(k)}(q); \ldots; v_N^{(k)}(q)] \in \mathbb{R}^{Nn}.
\end{align*}
\]

According to Algorithm 1,

\[
\begin{equation}
\vec{x}^{(k)}(q) = \left( A \otimes I \right) \vec{x}^{(k-1)}(q) + \vec{v}^{(k)}(q)/K
\end{equation}
\]

where $I$ is $n$-dimensional identity matrix and the second equality comes from the iteration. Simultaneously, we could get the average concatenation vector:

\[
\begin{equation}
\left( \frac{1}{N} A \otimes I \right) \vec{x}^{(k)}(q) = [x_1^{(k)}(q); \ldots; x_N^{(k)}(q)]
\end{equation}
\]

\[
\begin{equation}
= \sum_{s=1}^{k} \left( \frac{1}{N} A \otimes I \right) \vec{v}^{(s)}(q)/K,
\end{equation}
\]

where $1$ is $N$-dimensional vector, every element of which is 1; and the second equality follows from $1^T A = 1^T$. Finally,

\[
\begin{equation}
\begin{aligned}
\sum_{i=1}^{N} \|x_i^{(k)}(q) - \bar{x}^{(k)}(q)\|^2 &= \|x^{(k)}(q) - \left( \frac{1}{N} A \otimes I \right) \vec{x}^{(k)}(q)\|^2 \\
&= \|\sum_{s=1}^{k} \left( \frac{1}{N} A \otimes I \right) \vec{v}^{(s)}(q)/K\|^2 \\
&\leq \sum_{s=1}^{k} \|\frac{1}{N} A \otimes I \vec{v}^{(s)}(q)/K\|^2 \\
&\leq \sum_{s=1}^{k} \beta^{k-s} \frac{\sqrt{N}r(\mathcal{K})}{K(1 - \beta)},
\end{aligned}
\end{equation}
\]

where the second inequality follows from $\|\frac{1}{N} A \otimes I \vec{v}^{(s)}(q)/K\| \leq \beta^{k-s} [20]$. 

**Like variance reduction techniques in [19], we also use $g_i^{(k)}(q)$ to approximate the exact gradient $\nabla f_i x_i^{(k)}(q)$ in Algorithm 1 (See Line 18). To verify the approximation errors, we first prove the bound of $\mathbb{E}[\|g_i^{(k)}(q)\|^2]$ as follows.

**Lemma 3.** Under Assumption 1-4 and $\|\nabla f_i x_i(q)\| \leq G$, for any $k \in [K]$, $q \in [Q]$ and $i \in [N]$, we have $\mathbb{E}[\|g_i^{(k)}(q)\|^2] \leq 2(\sigma^2 + G^2)$. 

3334
Lemma 4.\[ E(\|\hat{\mathbf{d}}_i^{(k)}(q)\|_2^2) \leq 2(\sigma^2 + G^2), \]

where the first inequality comes from \( \eta_k \leq 1 \), and the final inequality follows from the \( f_i^{(m)}(q,i) \) is \( G \)-Lipschitz and Assumption 4.

Then, if \( k = m \), \( E(\|\hat{\mathbf{d}}_i^{(m)}(q)\|_2^2) \leq 2(\sigma^2 + G^2) \), we have

\[ E(\|\hat{\mathbf{d}}_i^{(m+1)}(q)\|_2^2) \leq E(\|\hat{\mathbf{d}}_i^{(m)}(q)\|_2^2) \leq 2(\sigma^2 + G^2), \]

where the first inequality comes from the Cauchy–Schwarz inequality; the second inequality follows from the fact that \( \|\hat{\mathbf{d}}_i^{(m)}(q)\|_2^2 \leq 2(\sigma^2 + G^2) \) and \( E(\|\hat{\mathbf{d}}_i^{(m)}(q)\|_2^2) \leq 2(\sigma^2 + G^2). \]

With Lemma 3, we can establish an upper bound of the sum of the expected distance between the vectors \( \hat{\mathbf{d}}_i^{(k)}(q) \) and its average vector \( \bar{\mathbf{d}}_i^{(k)}(q) \).

Lemma 4. Under Assumption 1-4 and \( \|\nabla f_{i,j}(\mathbf{x})\| \leq G, \) for any \( k \in [K] \) and \( q \in [Q] \), we could conclude that

\[ E\left(\sqrt{\sum_{i=1}^{N} \|\hat{\mathbf{d}}_i^{(k)}(q) - \bar{\mathbf{d}}_i^{(k)}(q)\|_2^2}\right) \leq \frac{\sqrt{2N(\sigma^2 + G^2)}}{1 - (1 - \gamma)\beta}. \]

Proof. We first define some variables:

\[ \mathbf{d}_i^{(k)}(q) = [\hat{d}_1^{(k)}(q); \ldots; \hat{d}_N^{(k)}(q)] \in \mathbb{R}^{Nn}, \]

\[ \mathbf{g}_i^{(k)}(q) = [\mathbf{g}_1^{(k)}(q); \ldots; \mathbf{g}_N^{(k)}(q)] \in \mathbb{R}^{Nn}. \]

According to Algorithm 1, we have

\[ \mathbf{d}_i^{(k)}(q) = (1 - \gamma)(A \otimes 1)\mathbf{d}_i^{(k-1)}(q) + \gamma \mathbf{g}_i^{(k)}(q) \]

\[ = (1 - \gamma)(A \otimes 1)\mathbf{d}_i^{(k-1)}(q) + \gamma \sum_{s=1}^{k} (1 - \gamma)^{k-s} \mathbf{A} \otimes 1 \mathbf{g}_i^{(s)}(q), \]

where I is a \( n \)-dimensional identity matrix. Simultaneously, we have

\[ \left( \frac{11T}{N} \otimes 1 \right) \mathbf{d}_i^{(k)}(q) = [\bar{d}_1^{(k)}(q); \ldots; \bar{d}_N^{(k)}(q)] \]

\[ = \gamma \sum_{s=1}^{k} (1 - \gamma)^{k-s} \left( \frac{11T}{N} \otimes 1 \right) \mathbf{g}_i^{(s)}(q), \]

where I is \( N \)-dimensional vector, whose element is 1, and the second equality follows from \( A \mathbf{F} = \mathbf{F} \).

Finally,

\[ E\left(\sum_{i=1}^{N} \|\mathbf{d}_i^{(k)}(q) - \bar{\mathbf{d}}_i^{(k)}(q)\|_2^2\right) \leq \frac{2(\sigma^2 + G^2)}{1 - (1 - \gamma)\beta}. \]

Before presenting the regret bound, we denote the average function as \( \tilde{f}_{q,i} = \frac{\sum_{k=1}^{K} f_{q,i,k}(x)}{K} \) denotes the average function of the remaining \( (K-k) \) functions for node i in q-th block, where \( 0 \leq k \leq K - 1 \).

Like the aforementioned insight, we actually view Algorithm 1 as DMFW [31] on a new q-round decentralized game with local objective functions \( f_{i,k}, \ldots, f_{i,K} \) for each node i. Therefore, from the results of DMFW [31], we can conclude that

Lemma 5. Under Assumption 3,

\[ (1 - 1/e) \sum_{q=1}^{Q} \sum_{i=1}^{N} \mathbf{f}_{q,i}(\mathbf{x}) - \sum_{q=1}^{Q} \sum_{i=1}^{N} E(\bar{f}_{q,i}(\mathbf{x}(\mathbf{K})(q))) \leq \frac{O K q^2 (K)}{2K} + \frac{\text{diam}(\mathcal{K})}{NK} \sum_{q=1}^{Q} \sum_{i=1}^{N} \sum_{j=1}^{N} \|\nabla f_{q,i,j}(\mathbf{x}(q)) - \mathbf{N} \mathbf{d}(\mathbf{x}(q))\|, \]

where \( \mathbf{x} = \arg \max_{x \in \mathcal{K}} \sum_{i=1}^{N} \sum_{j=1}^{N} f_{q,i,j}(x) \).

Noticably, for each \( t \in [7] \) in the q-th block, we pay \( x_i(t) = x_i^{(K)(q)} \) for each node i. Thus, if setting \( \mathbf{x}(t) = \sum_{i=1}^{N} x_i(t) / N \), we know

\[ \sum_{i=1}^{N} f_{q,i}(\mathbf{x}(t)) - \sum_{i=1}^{N} E(\bar{f}_{q,i}(\mathbf{x}(t))) \]

\[ = NK \left( \sum_{q=1}^{Q} \sum_{i=1}^{N} \mathbf{f}_{q,i}(\mathbf{x}) - \sum_{q=1}^{Q} \sum_{i=1}^{N} E(\bar{f}_{q,i}(\mathbf{x}(\mathbf{K})(q))) \right). \]
Therefore, from Lemma 5, we could get

\[ \sum_{i=1}^{T} \sum_{j=1}^{N} (1 - 1/e) f_{i,j}(\mathbf{x}^*) - \sum_{i=1}^{T} \sum_{j=1}^{N} \mathbb{E}(f_{i,j}(\bar{x}(t))) \leq N \sum_{q=1}^{Q} \sum_{t=1}^{T} \mathbb{E}(\bar{d}^{(q)}(q), x^* - \bar{v}^{(q)}(q))) + \frac{L N Q r^2(\mathcal{K})}{2} \]

\[ + \text{diam}(\mathcal{K}) \sum_{q=1}^{Q} \sum_{t=1}^{T} \mathbb{E}(\sum_{i=1}^{N} \nabla f_{q,i}(\bar{x}(t)), - \bar{d}^{(q)}(q))) \]

where \( \bar{x}(t) = \sum_{i=1}^{N} x_i(t)/N. \)

From the Lemma 6, for controlling the value \( \sum_{t=1}^{T} \sum_{i=1}^{N} (1 - 1/e) f_{i,j}(\mathbf{x}^*) - \sum_{t=1}^{T} \sum_{i=1}^{N} \mathbb{E}(f_{i,j}(\bar{x}(t))) \), we next seek to bound the RHS of Equation (3):

\[ \eta_t = \frac{2}{(t+1)^{2/3}} \text{ when } 1 \leq t \leq \frac{K}{2} + 1, \text{ and when } \frac{K}{2} + 2 \leq t \leq K, \]

\[ \eta_t = \frac{1}{(K-t+1)^{2/3}}. \]

\[ \sum_{q=1}^{Q} \sum_{t=1}^{T} \mathbb{E}(\sum_{i=1}^{N} \nabla f_{q,i}(\bar{x}(t)), - \bar{d}^{(q)}(q))) \leq \text{diam}(\mathcal{K}) Q \sqrt{\frac{\sqrt{2}(\mathcal{G}^2 + \mathcal{G}^2)}{1 - (1-\gamma)\beta}} + \frac{M_0 K \sqrt{Q}}{1 - \beta} \]

and

\[ \sum_{q=1}^{Q} \sum_{t=1}^{T} \mathbb{E}(\sum_{i=1}^{N} \nabla f_{q,i}(\bar{x}(t)), - \bar{d}^{(q)}(q))) \leq \frac{Q N G \sqrt{\frac{\sqrt{2}(\mathcal{G}^2 + \mathcal{G}^2)}{1 - \beta}} + \frac{N Q (2G + L r(\mathcal{K}))(K + 1)}{1 - \beta} + 2N Q \sqrt{C_1 K^{2/3}}}{1 - \beta} \]

where \( C_1 = \max\{2/3, 4(G^2 + \sigma^2) + 32(2G + L r(\mathcal{K}))^2, 2.25(G^2 + \sigma^2) + 7(2G + L r(\mathcal{K}))^{2/3} \}. \)

By merging Lemma 7 into Lemma 6, we could bound the gap between \( \sum_{t=1}^{T} \sum_{i=1}^{N} \mathbb{E}(f_{i,j}(\bar{x}(t))) \) and the \( (1 - 1/e) \)-optimal value. Note that, instead of \( \bar{x}(t) \), the definition of regret in Equation 1 hopes to compare the sum of reward at point \( x_j(t) \) for each \( f_{i,j} \). Thus, we also need to control the gap \( \sum_{t=1}^{T} \sum_{i=1}^{N} \mathbb{E}(f_{i,j}(\bar{x}(t)) - f_{i,j}(x_j(t))) \), i.e.,

\[ \text{Lemma 8. Under Assumption 1-5 and } \mathbb{E}(\nabla f_{i,j}(\mathbf{x}(t))) \leq G, \text{ for each } j \in [N], \]

\[ \sum_{i=1}^{N} \sum_{t=1}^{T} \mathbb{E}(f_{i,j}(\bar{x}(t)) - f_{i,j}(x_j(t))) \leq \frac{G N^{3/2} r(\mathcal{K}) Q}{1 - \beta}. \]

From the Lemma 6-8, we finally get the outcome of Theorem 1.

5 DECENTRALIZED ONLINE BOOSTING GRADIENT ASCENT

In sharp contrast with DMFW algorithm [31], Algorithm 1 trades the convergence rate for the lower communication overheads. In this section, we will present a faster one-shot decentralized online algorithm for monotone continuous DR-submodular maximization problem, i.e., the decentralized online boosting gradient ascent (DOBGA).

We begin by reviewing the online projected gradient algorithms. In the centralized setting, [32] first proposed the online gradient descent algorithm and derived the corresponding regret bounds for (strongly) convex objective functions. Then, [6] proved that the classical online projected gradient method achieves a suboptimal \((1/2)\)-approximation guarantee for online monotone continuous DR-submodular maximization problem. Based on a novel auxiliary function, [29] proceeded to present a variant of the online gradient ascent algorithm to boost the approximation ratio from \( 1/2 \) to \( 1/2 \). As for the decentralized scenarios, [25] first exhibited a decentralized online projected gradient method, which achieves a square-root regret for convex objectives.

As we know, the decentralized online projected gradient method is naturally one-shot, meaning that every node only needs to inquire one gradient evaluation and communicate with neighbors once at each round. However, we cannot directly apply the decentralized online projected gradient method [25] for the monotone continuous DR-submodular maximization problem, since it will suffer the same suboptimal \((1/2)\)-approximation guarantee as its centralized counterpart [6]. To achieve the optimal \((1-1/e)\) approximation ratio, we will design a new decentralized online gradient ascent algorithm based on the boosting auxiliary function.

5.1 Boosting Auxiliary Function

In this subsection, we first review some concepts and results about the novel boosting auxiliary function in [29]. For each monotone continuous DR-submodular function \( f : X \to \mathbb{R}^+ \) with \( f(0) = 0 \), we define its boosting auxiliary function as:

\[ F(x) = \int_{0}^{1} \frac{e^{x-1}}{z} f(z \ast x) dz. \]

Here, for any fixed \( x \in X \), the boosting function \( F \) allocates different weights \( e^{x-1}/z \) to the function values \( f(z \ast x) \). Next, we demonstrate some important properties of \( F \).

\[ \text{Lemma 9 ([29]). If the monotone continuous DR-submodular } f \text{ is differentiable, for any } x, y \in X, \text{ we have } \langle y - x, \nabla F(x) \rangle \geq (1 - 1/e) f(y) - f(x). \]

**Remark 3.** First, we review the definition of the stationary point on domain \( \mathcal{K} \). A point \( x \in \mathcal{K} \) is called a stationary point for function \( g \) over the domain \( \mathcal{K} \) iff \( \max_{y \in \mathcal{K}} \langle \nabla g(x), y - x \rangle \leq 0 \). As a result, this lemma demonstrates that any stationary point of auxiliary function \( F \) on the domain \( \mathcal{K} \) can attain \((1 - 1/e)\)-approximation of the global maximum of \( f \) over \( \mathcal{K} \). However, according to [11], \( \langle y - x, \nabla f(x) \rangle \geq \frac{1}{2} f(y) - f(x) \) which implies that the stationary point of \( f \) itself only provides a \((1/2)\)-approximation guarantee.

**Remark 4.** Moreover, the projected gradient ascent method with small step size usually converges to a stationary point, which sheds light on the possibility to use auxiliary function \( F \) to improve the performance of the classical gradient ascent algorithm for monotone continuous DR-submodular maximization. Similarly, in this paper, we hope to use the auxiliary function to boost the approximation ratio of the decentralized online gradient ascent algorithm.
Algorithm 2 Decentralized Online Boosting Gradient Ascent (DOBGA)

1: Input: Time horizon $T$, the number of nodes $N$, weight matrix $\Lambda = [a_{ij}] \in \mathbb{R}^{N \times N}$, step size $\eta_t$, $\forall t \in [T]$ and constraint set $\mathcal{K}$
2: Output: $\{x_i(t) : i \in [N], t \in [T]\}$
3: Initialize a point $x_i(1) \in \mathcal{K}$ for any $i \in [N]$
4: for $t = 1, \ldots, T$ do
5: for $i = 1, \ldots, N$ do
6: Node $i$ plays $x_i(t)$ to get reward $f_{i,i}(x_i(t))$ and observes the stochastic gradient information of $f_{i,i}$.
7: Generate a sample $z_i(t)$ of the random variable $Z \in [0, 1]$ where $\Pr(Z \leq z) = e^{z - 1 - e}$. 
8: Query the stochastic gradient $\nabla f_{i,i}(z_i(t) \ast x_i(t))$.
9: $y_i(t + 1) = \sum_{j \in N_i \cup \{i\}} a_{ij}x_j(t) + \eta_t(1 - 1/e)\nabla f_{i,i}(z_i(t) \ast x_i(t))$.
10: $x_i(t + 1) = \arg \min_{x \in \mathcal{K}} \|z - y_i(t + 1)\|_2$
11: end for
12: end for

From Equation (4), we know that $\nabla F(x) = \int_0^1 e^{z - 1} \nabla f(z \ast x) \, dz$. Generally speaking, it is impossible to directly compute the $\nabla F(x)$ by this equation. Therefore, we proceed by showing how to estimate $\nabla F(x)$ using an unbiased stochastic oracle $\nabla \tilde{f}(x)$. We first generate a sample $z$ from a random variable $Z \in [0, 1]$ where $\Pr(Z \leq z) = e^{z - 1 - e}$. Next, we inquire the stochastic gradient at point $z \ast x$ of the original function $f$, i.e., $\nabla \tilde{f}(z \ast x)$. Then, we use the $(1 - 1/e)\nabla \tilde{f}(z \ast x)$ to estimate the $\nabla F(x)$. In Appendix B, we will show the estimate $(1 - 1/e)\nabla \tilde{f}(z \ast x)$ is unbiased under Assumption 4.

5.2 Decentralized Online Boosting Gradient Ascent

In this subsection, we first assume that, for each local function $f_{i,i}$, $f_{i,i}(0) = 0$. Note that the shift of each local function will not affect our theoretical analysis and algorithm. Also, we need to have a local auxiliary function of each local $f_{i,i}$ as $F_{i,i}(x) = \int_0^1 e^{z - 1} f_{i,i}(z \ast x) \, dz$. Next, we introduce our proposed decentralized online boosting gradient ascent (DOBGA) as shown in Algorithm 2.

In Algorithm 2, each node $i \in [N]$ first generates a sample $z_i(t)$ and inquires the stochastic gradient at point $z_i(t) \ast x_i(t)$ to compute the stochastic gradient of boosting auxiliary function $F_{i,i}$ (See line 7-8). Then, DOBGA algorithm aggregates the actions of neighboring nodes and its own decision, namely, $\sum_{j \in N_i \cup \{i\}} a_{ij}x_j(t)$, and pushes the aggregated information along the stochastic boosting gradient with stepsize $\eta_t$ (See line 9). Finally, the node $i$ updates its action via the projection operation (See line 10). Obviously, in Algorithm 2, node $i$ only inquires one stochastic gradient evaluation and communicates with neighboring nodes once at each round. We proceed to show the regret bound for Algorithm 2.

THEOREM 2. Under Assumption 1-4, if $\eta_t \geq \eta_{t+1}$ for each $1 \leq t < [T]$ in Algorithm 2 and $\|\nabla F_{i,i}(x)\|_2 \leq C_1$, then we have, for each node $j \in [N]$,

$$\mathbb{E}[R_j(T, j)] \leq \frac{N \text{diam}(\mathcal{K})}{2\eta_T} + C_2 \sum_{t=1}^T \eta_t,$$

where $C_2 = 2NG^2 + \frac{4NG^2}{1 - \rho} + 2G^2 (N + N^2)/(1 - \rho)^2$ and $\alpha = 1 - 1/e$.

Remark 5. If we set stepsize $\eta_t = \frac{1}{\sqrt{T}}$, DOBGA achieves the $(1 - 1/e)$-regret of $O(\sqrt{T})$. To the best of our knowledge, this is the first decentralized online algorithm to obtain $O(\sqrt{T})$ regret bound against a $(1 - 1/e)$-approximation with $O(1)$ gradient inquiry and communication complexity for each node $i \in [N]$ per step.

6 NUMERICAL EXPERIMENTS

In this section, we will empirically compare the following algorithms.

Decentralized Meta-Frank-Wolfe (DMFW): Algorithm 1 in [31] with $K = T^{3/2}$, $\eta_k = 2/K^{2/3}$ and $\gamma_k = 1/K^{1/2}$.

One-Shot Decentralized Meta-Frank-Wolfe (Mono-DMFW): Our proposed Algorithm 1 with $\eta_k = \frac{2}{(k+2)^2}$ for any $1 \leq k = \frac{K}{T} + 1$ and $\gamma_k = \frac{1.5}{(K+2)^2}$ for any $K + 2 \leq k \leq K$, $\gamma = 1/T^{1/5}$, $K = T^{3/5}$ as well as $Q = T^{2/5}$.

Decentralized Online Boosting Gradient Ascent (DOBGA): Our proposed Algorithm 2 with $\eta_t = 1/\sqrt{T}$. Furthermore, we use the average of 5 independent stochastic gradients to estimate each $\nabla F_{i,i}$ at every iteration.

We evaluate these algorithms on the MovieLens dataset [9] for the movie recommendation task. This dataset consists of 10 million 5-star ratings by 72,000 users for 10,000 movies. All Ratings are made with half-star increment. Note that all experiments are performed in Python 3.6.5 using CVX optimization tool [8] on a Windows 10 machine with 32GB RAM and Intel(R)255 Core(TM) i7-9700K CPU.

To begin, we split the first $T \times b$ users into disjoint and equally-sized sets $\mathcal{U}_0, \ldots, \mathcal{U}_T$, i.e., $|\mathcal{U}_t| = b$ for any $t \in [T]$. At the $t$-th round, we assume the data of the users $\mathcal{U}_t$ is distributed equally over the network of nodes, which means that, at round $t$, each node $i \in [N]$ only can get access to the data of a partition of $\mathcal{U}_t$. Furthermore, we use $\mathcal{U}_t$ to denote the users stored in node $i$ at round $t$. In this experiment, our goal is to find a set of $k$ movies that are satisfactory to all considered users.

Precisely, let $r_{u,m}$ denote the rating of user $u$ for movie $m$. For each user $u$, we consider a well-motivated facility location objective function $g_u(S) = \max_{m \in S} r_{u,m}$, where $S$ is a subset of the movies. Such a function shows how much the user $u$ is satisfied with the movies in the subset $S$. Due to that each $\mathcal{U}_t$ contains multiple users, the facility location function associated with node $i$ at round $t$ is $g_{i,t}(S) = \sum_{u \in \mathcal{U}_t} g_u(S)$. Following [31], at round $t$, we also consider setting the objective function of node $i$ as the multilayer extension of the facility location function $g_{i,t}$, i.e., $f_{i,t}(x) = \sum_{S \subseteq [n]} g_{i,t}(S) \prod_{j \in S} x_j \prod_{j \notin [n]} x_j (1 - x_j)$, where we encode the movies as the set $\{1, 2, \ldots, n\}$ and $x \in \{0, 1\}^n$. According to [31], $f_{i,t}$ is a monotone continuous DR-submodular function.

In the first experiment, we plan to recommend a set of ten movies with high objective value from the first 200 movies at each round, so $n = 200$ and the constraint set is $\mathcal{K} = \{x \in \mathbb{R}^n : 0 \leq x \leq 1, \sum x \leq 10\}$. We also consider a network of 30 nodes and set $b = 60$ and $T = 200$. The stochastic gradient is obtained by imposing the
Gaussian noise, i.e., $\tilde{f}_i(x) = f_i(x) + O(0,1)$ for any $t \in [T]$, $i \in [N]$ and $x \in [0,1]^N$, where $N(0,1)$ is standard multivariate normal distribution. As for the network topology, we consider three different choices: A complete graph, a cycle graph, and an Erdos-Renyi random graph (with an average degree of 3). The matrix $\mathbf{A}$ is chosen in the following manner. If the edge $(i,j)$ is an edge of the graph, let $a_{ij} = 1/(1+\max(d_i,d_j))$ where $d_i$ and $d_j$ are the degree of node $i$ and $j$, respectively. If $(i,j)$ is not an edge of the graph and $i \neq j$, then $a_{ij} = 0$. Finally, we set $a_{ii} = 1 - \sum_{j \in N_i} a_{ij}$. According to Assumption 1, $\mathbf{A}$ is satisfied with Assumption 1.

We present the trend of the ratio between $(1-1/e)$-regret and timestamp in the Figure 1(a)-(c) and report the running time in Table 2. From Figure 1(a)-(c), we observe that the Mono-DMFW and DMFW algorithm exhibit the lowest and fastest convergence rate in all three graphs, respectively. This phenomenon is in accord with our theoretical analysis (See Table 1). When the iteration index is large, the DOBGA achieves nearly the same $(1-1/e)$-regret with DMFW. Despite the better empirical performance of DMFW, both Mono-DMFW and DOBGA efficiently speed up the computation. As shown in Table 2, our proposed DOBGA and Mono-DMFW can be 400 and 3000 times faster than DMFW, respectively, in all three graphs.

In Figure 1(d)-1(e), we present how the network topology affects the performance of the Mono-DMFW and DOBGA algorithms. We can observe that both algorithms converges faster on the complete graph than the cycle graph and ER graph.

Following [31], we also explore how the number of nodes influences Algorithm 1 and Algorithm 2. In this experiment, we consider the complete graph with different number of nodes and set $b = 128$. As shown in Figure 1(f)-(g), when the number of nodes increases, the $(1-1/e)$-regret of both Mono-DMFW and DOBGA decrease more slowly, which confirms our theoretical results.

### 7 CONCLUSION

In this paper, we propose two communication-efficient decentralized online algorithms for optimizing the monotone continuous DR-submodular maximization problem over the network, both of which improve the communication complexity and the number of per-function gradient evaluations from $T^{3/2}$ to 1. First, we develop the Mono-MFW algorithm, achieving a $(1-1/e)$-regret bound of $O(T^{1/2})$. Then, we present a variant of the decentralized online gradient ascent, namely, DOBGA algorithm, which attains a $(1-1/e)$-regret of $O(\sqrt{T})$. Numerical experiments demonstrate the effectiveness of the proposed algorithms.

### 8 ACKNOWLEDGEMENTS

Zhang and Yang’s research is supported in part by the Hong Kong Research Grants Council (ECS 21214720) and Alibaba Group through Alibaba Innovative Research (AIR) Program.
REFERENCES

[1] Mervat Abu-Elkheir, Mohammad Hayajneh, and Najah Abu Ali. Data management for the internet of things: Design primitives and solution. Sensors, 13(11):15582–15612, 2013.

[2] Andrew An Bian, Baharan Mirzasoleiman, Joachim M Buhmann, and Andreas Krause. Guaranteed non-convex optimization: Submodular maximization over continuous domains. In Artificial Intelligence and Statistics, pages 111–120. PMLR, 2017.

[3] Yatao Bian, Joachim M Buhmann, and Andreas Krause. Continuous submodular function maximization. arXiv preprint arXiv:2006.13474, 2020.

[4] 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.

[5] Lin Chen, Christopher Harshaw, Hamed Hassani, and Amin Karbasi. Projection-free online optimization with stochastic gradient: From convexity to submodularity. In International Conference on Machine Learning, pages 814–823. PMLR, 2018.

[6] Lin Chen, Hamed Hassani, and Amin Karbasi. Online continuous submodular maximization. In International Conference on Artificial Intelligence and Statistics, pages 1896–1905. PMLR, 2018.

[7] Hongchang Gao, Hanzi Xu, and Slobodan Vucetic. Sample efficient decentralized stochastic frank-wolfe methods for continuous dr-submodular maximization. In IJCAI, pages 3501–3507, 2021.

[8] Michael Grant and Stephen Boyd. Cvx: Matlab software for disciplined convex programming, version 2.1, 2014.

[9] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. ACM Transactions on Interactive Intelligent Systems, 5(4):1–19, 2015.

[10] Hamed Hassani, Amin Karbasi, Aryan Mokhtari, and Zebang Shen. Stochastic conditional gradient++: (non) convex minimization and continuous submodular maximization. SIAM Journal on Optimization, 30(4):3315–3344, 2020.

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

[12] Elad Hazan et al. Introduction to online convex optimization. Foundations and Trends® in Optimization, 2(3–4):157–325, 2016.

[13] Elad Hazan, Kfir Yehuda Levy, and Shai Shalev-Shwartz. On graduated optimization for stochastic non-convex problems. In International Conference on Machine Learning, pages 1833–1841. PMLR, 2016.

[14] Saghar Hosseini, Airlie Chapman, and Mehran Mesbahi. Online distributed optimization via dual averaging. In 52nd IEEE Conference on Decision and Control, pages 1484–1489. IEEE, 2013.

[15] Shinji Ito and Ryohei Fujimaki. Large-scale price optimization via network flow. Advances in Neural Information Processing Systems, 29, 2016.

[16] David Kempe, Jon Kleinberg, and Eva Tardos. Maximizing the spread of influence through a social network. In Proceedings of the ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 137–146, 2003.

[17] Siddharth Mitra, Moran Feldman, and Amin Karbasi. Submodular+ concave. In Advances in Neural Information Processing Systems, 2021.

[18] Aryan Mokhtari, Hamed Hassani, and Amin Karbasi. Decentralized submodular maximization: Bridging discrete and continuous settings. In International Conference on Machine Learning, pages 3616–3625. PMLR, 2018.

[19] Angella Nedic and Anamn Ozdaglar. Distributed subgradient methods for multi-agent optimization. IEEE Transactions on Automatic Control, 54(1):48–61, 2009.

[20] 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.

[21] Michael Rabbat and Robert Nowak. Distributed optimization in sensor networks. In Proceedings of the 3rd international symposium on Information processing in sensor networks, pages 20–27, 2004.

[22] Qixin Zhang et al. Determinantal point processes for machine learning. Foundations and Trends® in Machine Learning, 5(2–3):125–286, 2012.

[23] Fangzheng Xie and Yi Ma. Projection-free distributed online learning in networks. In International Conference on Machine Learning, pages 4054–4062. PMLR, 2017.

[24] Junlong Zhu, Qingtao Wu, Mingchuan Zhang, Ruijuan Zheng, and Keqin Li. Continuous influence maximization. In Proceedings of the 3rd international symposium on Information processing in sensor networks, pages 111–120. PMLR, 2011.

[25] Beng Yan, Shreyas Sundaram, SVN Vishwanathan, and Yuan Qi. Distributed autonomous online learning: Regret and intrinsic privacy-preserving properties. IEEE Transactions on Knowledge and Data Engineering, 29(4):2483–2495, 2012.

[26] Kun Yuan, Qing Ling, and Wotao Yin. On the convergence of decentralized gradient descent. SIAM Journal on Optimization, 26(3):1835–1854, 2016.

[27] Mingrui Zhang, Lin Chen, Hamed Hassani, and Amin Karbasi. Online continuous submodular maximization: From full-information to bandit feedback. In Advances in Neural Information Processing Systems, pages 9206–9217, 2019.

[28] Qixin Zhang, Zengde Deng, Zaiyi Chen, Haoyuan Hu, and Yu Yang. Continuous submodular maximization: Boosting via non-oblivious function. In Advances in Neural Information Processing Systems, 2021.

[29] Qixin Zhang, Zaiyi Chen, Haoyuan Hu, and Yu Yang. Stochastic continuous submodular maximization: Boosting via non-oblivious function. In International Conference on Machine Learning, pages 2616–26134. PMLR, 2022.

[30] Wenpeng Zhang, Peilin Zhao, Wenwu Zhu, Steven CH Hoi, and Tong Zhang. Projection-free distributed online learning in networks. In International Conference on Machine Learning, pages 4054–4062. PMLR, 2017.

[31] Junlong Zhu, Qingtao Wu, Mingchuan Zhang, Ruijuan Zheng, and Keqin Li. Projection-free decentralized online learning for submodular maximization over time-varying networks. Journal of Machine Learning Research, 22(51):1–42, 2021.

[32] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In International Conference on Machine Learning, pages 928–936, 2003.