DIAMOND: Taming Sample and Communication Complexities in Decentralized Bilevel Optimization

Peiwen Qiu[1] Yining Li[1] Zhuqing Liu[1] Prashant Khanduri[2] Jia Liu[1]
Ness B. Shroff[1] Elizabeth Serena Bentley[3] Kurt Turck[3]
[1]Dept. of ECE, The Ohio State University [2]Dept. of CS, Wayne State University
[3]Air Force Research Laboratory, Information Directorate

***Abstract*—Decentralized bilevel optimization has received increasing attention recently due to its foundational role in many emerging multi-agent learning paradigms (e.g., multi-agent meta-learning and multi-agent reinforcement learning) over peer-to-peer edge networks. However, to work with the limited computation and communication capabilities of edge networks, a major challenge in developing decentralized bilevel optimization techniques is to lower sample and communication complexities. This motivates us to develop a new decentralized bilevel optimization called DIAMOND (decentralized single-timescale stochastic approximation with momentum and gradient-tracking). The contributions of this paper are as follows: i) our DIAMOND algorithm adopts a single-loop structure rather than following the natural double-loop structure of bilevel optimization, which offers low computation and implementation complexity; ii) compared to existing approaches, the DIAMOND algorithm does not require any full gradient evaluations, which further reduces both sample and computational complexities; iii) through a careful integration of momentum information and gradient tracking techniques, we show that the DIAMOND algorithm enjoys $\mathcal{O}(\epsilon^{-3/2})$ in sample and communication complexities for achieving an ϵ-stationary solution, both of which are independent of the dataset sizes and significantly outperform existing works. Extensive experiments also verify our theoretical findings.**

I. Introduction

In recent years, the problem of performing decentralized bilevel optimization over networks has attracted increasing attention. For a peer-to-peer communication network represented by a graph $\mathcal{G} = (\mathcal{N}, \mathcal{L})$, where $\mathcal{N}$ and $\mathcal{L}$ denote the sets of agents and edges with $|\mathcal{N}| = m$, a decentralized bilevel optimization problem can be generally written as follows:

$$\min_{\mathbf{x}\in\mathbb{R}^{d_{\rm up}}} l(\mathbf{x}) = \frac{1}{m}\sum_{i=1}^{m}\left\{l_i(\mathbf{x}) \triangleq \mathbb{E}_{\xi_i}\left[f_i\left(\mathbf{x}, \mathbf{y}_i^*(\mathbf{x}); \xi_i\right)\right]\right\},$$
$$\text{s.t. } \mathbf{y}_i^*(\mathbf{x}) = \underset{\mathbf{y}_i\in\mathbb{R}^{d_{\rm low}}}{\operatorname{argmin}}\left\{g_i(\mathbf{x}, \mathbf{y}_i) \triangleq \mathbb{E}_{\zeta_i}\left[g_i(\mathbf{x}_i, \mathbf{y}_i; \zeta_i)\right]\right\}, \quad (1)$$

where $l_i(\mathbf{x})$ is the local objective function at agent i, $\mathbf{x} \in \mathbb{R}^{d_{\rm up}}$ and $\mathbf{y}_i \in \mathbb{R}^{d_{\rm low}}$ are the global upper-level variables and the local lower-level variables at agent $i \in \{1, \ldots, m\}$, and ξ_i and ζ_i represent the random samples of the upper-level and the lower-level subproblems, respectively.

Problem (1) plays a foundational role for various fundamental multi-agent learning paradigms over decentralized or multi-hop wireless edge networks. For instance, in the well-known actor-critic framework for cooperative multi-agent reinforcement learning (MARL, e.g., [1], [2]), the shared global policy improvement (the actor component) corresponds to the upper-level subproblem in Problem (1), which depends on the optimal solution of a policy evaluation subproblem (the critic component), which corresponds to the lower-level subproblem in Problem (1). Another example can be found in multi-agent meta-learning (also referred to as "learning-to-learn", see, e.g., [3], [4]), where the training of task-specific parameters at each agent can be represented by the lower-level subproblem in Problem (1). The task-specific parameter training is coupled with the shared parameters' optimization, which corresponds to the upper-level subproblem in Problem (1).

It is evident from Problem (1) that the most prominent features of decentralized bilevel optimization are i) "bilevel" and ii) "decentralization." Same as the single-agent bilevel counterpart [5], Problem (1) has a hierarchical structure, where the upper-level subproblem objective value is determined by both the upper-level variable $\mathbf{x}$ and the optimal variables $\{\mathbf{y}_i^*(\mathbf{x})\}_{i=1}^m$ obtained by solving the lower-level subproblems. Due to this bilevel structure, the solution approach for Problem (1) is fundamentally different from the traditional loss minimization in conventional learning problems with a single-level structure. Thus, new algorithm design and analysis techniques are necessary for solving Problem (1). Moreover, instead of only having a single task as in single-agent bilevel optimization problems, one needs to cope with *multiple* lower-level tasks in a *decentralized* fashion in Problem (1). This key difference necessitates new algorithm designs. Thus, solving decentralized bilevel optimization problems over wireless networks needs to address the following technical challenges:

- *Single-Loop or Double-Loop Architecture?* In Problem (1), it is often impractical to asymptotically solve the lower-level problem to optimality. Rather, one typically resorts to using an *approximaion* of $\mathbf{y}^*(\mathbf{x})$, which is obtained by solving the lower-level problems with finite iterations [1], [6]–[11]. However, due to the coordination complexity among agents and training accuracy concerns, the algorithmic architecture choice between "double-loop" [6]–[8] or "single-loop" [1],

This work has been supported in part by NSF grants CAREER CNS-2110259, CNS-2112471, CNS-2102233, CCF-2110252, and AFRL grant FA8750-20-3-1003. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Research Laboratory or the U.S. Government. Distribution A. Approved for public release: Distribution unlimited AFRL-2023-0273 on 18 Jan 2023.

[9]–[11], both of which are widely used in single-agent bilevel optimization, suddenly becomes critical. On one hand, the double-loop architecture naturally follows the bilevel problem structure and executes multiple inner-loop iterations within each outer iteration, which typically yields a more accurate estimation of the lower-level minimizer. However, in a decentralized network setting, this requires a two-timescale implementation with high coordination complexity, as well as high computation and sample complexities in the inner loop. On the other hand, the single-loop architecture iteratively solves the upper- and lower-level subproblems and updates corresponding parameters simultaneously, which is much easier to implement in the decentralized setting. However, it is unclear whether the less accurate inner subproblem solutions could also result in high communication and sample complexities in the overall training process.

- *Achieving Low Communication and Sample Complexities:* Since there is no dedicated centralized server in decentralized bilevel optimization over edge networks, it is infeasible to aggregate the local datasets at the geographically dispersed agents. Hence, it is necessary for the agents to communicate and exchange information with each other to reach a "consensus solution" [12]–[15]. In such scenarios, how to design efficient algorithms to reduce the required rounds of communications (i.e., communication complexity) to reach consensus is one of the most important questions in algorithm design. This is particularly true for deploying decentralized bilevel optimization over wireless edge networks that may have low-speed and unreliable links. Also, due to the fact that the agents (i.e., computing nodes) in many wireless edge networks are fundamentally constrained by computation capabilities (e.g., sensors or smart phones with limited computation speed, energy, and memory), it is important to design efficient algorithms to reduce the required number of samples (i.e., sample complexity). However, achieving low sample and communication complexities are two fundamentally *conflicting* goals. On one hand, the variance of a stochastic gradient highly depends on the number of samples in each mini-batch. The more samples in each mini-batch (i.e., potentially higher sample complexity), the larger the variance, which may imply fewer communication rounds for convergence (i.e., lower communication complexity). On the other hand, if one prefers to use fewer samples per iteration to lower per-iteration sample complexity, the stochastic gradient information is noisier, which could result in more communication rounds to reach certain training accuracy (i.e., higher communication complexity).

In addition to the above challenges, the coupled structure and the inherent non-convexity of the decentralized bilevel optimization problems make it challenging to design algorithms and theoretically analyze their performance. So far, results on designing decentralized bilevel optimization algorithms with low sample, communication, and implementation complexities remains rather limited in the literature (see Section II for more detailed discussions). The main contribution of this paper is that we propose a series of new algorithm design techniques, which overcome the aforementioned challenges and achieve low sample and communication complexities with a *single-loop* structure for decentralized bilevel optimization problems. The key results of this paper are summarized as follows:

- We propose an algorithm called DIAMOND (decentralized single-loop stochastic approximation with momentum and gradient-tracking) for solving decentralized bilevel optimization problems over networks. Our proposed DIAMOND algorithm integrates consensus-based updates with gradient tracking and momentum-based stochastic gradient estimators, which is a carefully designed triple-hybrid approach. We show that this triple-hyrid approach enables the use of a single-loop algorithmic architecture, which significantly reduces the implementation complexity over peer-to-peer edge networks.
- We show that DIAMOND achieves a sample complexity of $\mathcal{O}(\epsilon^{-3/2})$ to find an ϵ-stationary solution for non-convex upper-level optimization objectives. Interestingly, this result matches the state-of-the-art sample complexity of stochastic first-order algorithms for solving single-agent bilevel optimization problems. Meanwhile, the communication complexity of DIAMOND is $\mathcal{O}(\epsilon^{-3/2})$. These results show that DIAMOND strikes a good balance between sample and communication complexities.
- We conduct extensive experiments to validate the theoretical results of the proposed DIAMOND algorithm. Our experimental results show that DIAMOND outperforms other stochastic first-order baseline algorithms in terms of sample complexities in various communication network settings.

The rest of the paper is organized as follows. In Section II, we review related work to provide the necessary background on decentralized and bilevel optimization, and put our work in comparative perspectives. In Section III, we present the system model and the consensus reformulation of decentralized bilevel optimization. In Section IV, we propose the DIAMOND algorithm. We then provide the theoretical convergence analysis of DIAMOND in Section V. Section VI provides numerical results to verify our theoretical findings, and Section VII concludes this paper.

II. Related Work

To facilitate our discussions in subsequent sections, we organize the related work in three parts. First, we survey the approaches for solving single-agent bilevel optimization to provide a contrasting view for decentralized bilevel optimization. Then, we review the literature on decentralized optimization for single-level loss minimization to familiarize readers with the basics of decentralized optimization over networks. Lastly, we provide an in-depth comparison with the most related work on decentralized bilevel optimization.

1) Single-Agent Bilevel Optimization: *1-a) Gradient-Based Approaches:* To our knowledge, single-agent bilevel optimization was first studied in [16]. Since then, several

solution approaches have been proposed, such as 1) penalizing the outer function with the optimality conditions of the inner problem [17], [18]; 2) reformulating the bilevel problem as a single-level problem by replacing the lower-level problem with its optimality conditions [19], [20]; and 3) utilizing gradient-based techniques to iteratively approximate the (stochastic) gradient of the upper-level problem. Gradient-based algorithms for bilevel optimization have gained the most attention due to their ease of analysis. Many gradient-based bilevel optimization algorithms have been proposed, including but not limited to: i) AID-based [4], [21], [22], ii) ITD-based [6], [23], [24], and iii) SGD-based [1], [7], [8], [11]. However, these algorithms were designed for single-agent bilevel optimization problems and not applicable for the decentralized settings.

1-b) Momentum-Based Approaches: Momentum-based approaches enhance the gradient-based algorithms for single-agent bilevel optimization. It has been shown that momentum improves the computation efficiency of stochastic gradient updates both in theory and in practice. Several bilevel optimization algorithms that exploit momentum have been proposed, such as STABLE [9], RSVRB [10], MRBO [8], and SUSTAIN [11]. All of them share a similar single-loop algorithmic architecture. To reach an ϵ-stationary point, STABLE requires an order of $\mathcal{O}\left(\epsilon^{-2}\right)$ samples, while RSVRB, MRBO and SUSTAIN require $\mathcal{O}\left(\epsilon^{-3/2}\right)$ samples. Compared to STABLE, which only uses a momentum-based stochastic gradient estimator for the upper-level subproblems, RSVRB, MRBO, and SUSTAIN all utilize momentum-based stochastic gradient estimators for both the upper- and lower-level subproblems. However, all these momentum-based algorithms are designed for the single-agent bilevel optimization setting. In comparison, we focus on decentralized multi-agent bilevel optimization, and propose the DIAMOND algorithm, which uses momentum-based stochastic gradient estimators in both upper- and lower-levels. Our theoretical result shows that DIAMOND has the sample complexity of $\mathcal{O}\left(\epsilon^{-3/2}\right)$, matching the state-of-the-art result achieved by RSVRB, MRBO and SUSTAIN, but for the *more challenging decentralized setting.*

2) Decentralized Optimization for Single-Level Loss Minimization over Networks: Decentralized optimization for single-level loss minimization over networks traces its roots to the seminal work [25], and has found important applications in many engineering fields, e.g., network resource allocation [26], power networks [27], and robotic networks [28]. One of the most popular methods to solve decentralized optimization problems is the distributed stochastic gradient descent (DSGD) [14], which established the well-known $O(1/\sqrt{T})$ convergence rate with T iterations. Subsequently, [29] showed that DSGD can outperform the centralized SGD counterpart. Recently, various sample- and communication-efficient variants of DSGD have been proposed, e.g., leveraging compression [30], momentum [31], gradient tracking [32], [33], and variance reduction techniques [12], [34], [35]. However, results on solving general decentralized bilevel optimization problems are still limited.

3) Decentralized Bilevel Optimization over Networks: So far, the research on decentralized bilevel optimization remains in its infancy. As mentioned earlier, due to the lack of a centralized server in the decentralized setting, it is natural for us to consider the network-consensus approach [12]–[15] as the solution strategy in this paper. To our knowledge, the most related and the only work that also adopts a consensus-based approach for solving decentralized bilevel optimization problems is reported in [13], which contains two algorithmic variants called INTERACT and SVR-INTERACT. Specifically, INTERACT is a local-full-gradient-based algorithm with gradient tracking and achieves $\left[\mathcal{O}\left(n\epsilon^{-1}\right), \mathcal{O}\left(\epsilon^{-1}\right)\right]$ sample-communication complexity, where n is the size of the dataset at each agent. Recall that the sample and communication complexities of our DIAMOND are both $\mathcal{O}\left(\epsilon^{-3/2}\right)$, which is *independent* of data size. This implies that in the large dataset regime $\Theta(\epsilon^{-1/2})$, which is not uncommon in the era of "big data," INTERACT suffers a higher sample complexity than that of our DIAMOND algorithm.

To lower the sample complexity of INTERACT, SVR-INTERACT leverages variance reduction techniques to retain the same communication complexity as INTERACT, while achieving a lower but still *dataset-dependent* sample complexity.Thus, in the large dataset regime $\Omega(\epsilon^{-1})$, the sample complexity of SVR-INTERACT will be higher than that of DIAMOND. Also, SVR-INTERACT still requires periodic full gradients, while DIAMOND only needs stochastic gradient evaluations. Moreover, due to the variance reduction techniques, SVR-INTERACT has a double-loop algorithmic architecture. By contrast, DIAMOND is *single-loop* structure, which has a lower computational cost and is easier to implement. Apart from these, SVR-INTERACT was designed to solve deterministic bilevel optimization problems, rather than stochastic bilevel optimization problems that has wider applications when the sample size of training data is large (e.g., hyperparameter optimization [36]) or the fresh data is sampled for algorithm iterations (e.g., reinforcement learning [1]).

III. System Model and Problem Formulation

In this section, we will present the network-consensus-based problem reformulation that paves the way for our subsequent algorithm design and analysis. Recall from Problem (1) that we consider a peer-to-peer communication network represented by a graph. Suppose that each agent i can communicate with its set of neighbors denoted by $\mathcal{N}_i \triangleq \{j \in \mathcal{N}, : (i,j) \in \mathcal{L}\}$. To solve Problem (1) in a decentralized fashion, one can rewrite Problem (1) as follows:

$$\begin{aligned}
&\min_{\mathbf{x}_i \in \mathbb{R}^{d_{\rm up}}} l\left(\mathbf{x}\right) = \frac{1}{m}\sum_{i=1}^{m}\left\{l_i\left(\mathbf{x}_i\right) \triangleq \mathbb{E}_{\xi_i}\left[f_i\left(\mathbf{x}_i, \mathbf{y}_i^*\left(\mathbf{x}_i\right); \xi_i\right)\right]\right\},\\
&\text{s.t. } \mathbf{y}_i^*\left(\mathbf{x}_i\right) = \arg\min_{\mathbf{y}_i \in \mathbb{R}^{d_{\rm low}}}\left\{g_i\left(\mathbf{x}_i, \mathbf{y}_i\right) \triangleq \mathbb{E}_{\zeta_i}\left[g_i\left(\mathbf{x}_i, \mathbf{y}_i; \zeta_i\right)\right]\right\},\\
&\quad \mathbf{x}_i = \mathbf{x}_j, \text{ if } (i,j) \in \mathcal{L},
\end{aligned} \tag{2}$$

where $\mathbf{x}_i \in \mathbb{R}^{d_{\rm up}}$, $\forall i$, are the local copies of the global upper-level variables at agent $i \in \{1, \ldots, m\}$ and $g_i(\mathbf{x}_i, \mathbf{y}_i)$ is the

local lower-level loss at agent i. For notation simplicity, we denote $f_i\left(\mathbf{x}_i, \mathbf{y}_i^*\left(\mathbf{x}_i\right)\right) \triangleq \mathbb{E}_{\xi_i}\left[f_i\left(\mathbf{x}_i, \mathbf{y}_i^*\left(\mathbf{x}_i\right); \xi_i\right)\right]$. The equality constraint in Problem (2) ensures that all agents share the same global $\mathbf{x}$-value to achieve the minimization of the upper-level function, hence the name "consensus form" [12]–[15]. We assume that $f_i(\mathbf{x}_i, \mathbf{y}_i^*(\mathbf{x}_i))$ is non-convex in $\mathbf{x}_i$, $\forall i$, in general, and $g_i(\mathbf{x}_i, \mathbf{y}_i)$ is strongly-convex in $\mathbf{y}_i$, $\forall i$, which typically holds in meta-learning, hyper-parameter optimization (see Section VI), and MARL with quadratically-regularized linear critics. Now, we define the notion of ϵ-stationarity that serves as the convergence metric.

Definition 1 (ϵ-Stationary Point)**.** *A stochastic algorithm reaches an ϵ-stationary point $\{\mathbf{x}_i, \mathbf{y}_i, \forall i \in [m]\}$ if*

$$\mathbb{E}\Big[\underbrace{\|\nabla l(\bar{\mathbf{x}})\|^2}_{\substack{\text{Stationarity}\\ \text{Error}}} + \underbrace{\|\mathbf{y}^* - \mathbf{y}\|^2}_{\substack{\text{Lower-Level}\\ \text{Error}}} + \underbrace{\frac{1}{m}\sum\nolimits_{i=1}^{m}\|\mathbf{x}_i - \bar{\mathbf{x}}\|^2}_{\text{Consensus Error}}\Big] \leq \epsilon,$$

where $\bar{\mathbf{x}} \triangleq \frac{1}{m}\sum_{i=1}^{m}\mathbf{x}_i$, $\mathbf{y} \triangleq \left[\mathbf{y}_1^\top, \ldots, \mathbf{y}_m^\top\right]^\top$, and $\mathbf{y}^* \triangleq \left[\mathbf{y}_1^{*\top}, \ldots, \mathbf{y}_m^{*\top}\right]^\top$. The expectation is taken over the randomness of the algorithm.

Next, we formally define the sample complexity and communication complexity of a decentralized algorithm, which are also used in the literature (e.g., [12], [13]).

Definition 2 (Sample Complexity)**.** *The sample complexity is defined as the total number of incremental first-order oracle (IFO) calls required per node for an algorithm to reach an ϵ-stationary point defined in Definition 1, where one IFO call is defined as the evaluation of the stochastic gradient of upper- and lower-level problems at agent $i \in [m]$.*

Definition 3 (Communication Complexity)**.** *The communication complexity is defined as the total rounds of communications required to find an ϵ-stationary point defined in Definition 1, where each node can send and receive local parameters with its neighbors in one communication round.*

IV. The DIAMOND Algorithm

In this section, we present our DIAMOND algorithm for solving Problem (2). Since the agents can communicate with their neighbors through the network to send and receive information (model parameters) and aggregate the received information, we define the consensus weight matrix $\mathbf{M} \in \mathbb{R}^{m\times m}$, where $[\mathbf{M}]_{ij}$ represents the consensus weight over edge $(i,j) \in \mathcal{L}$. We assume that $\mathbf{M}$ satisfies the following:

1) Doubly stochastic: $\sum_{i=1}^{m}[\mathbf{M}]_{ij} = \sum_{j=1}^{m}[\mathbf{M}]_{ij} = 1, \forall i, j$;
2) Symmetric: $[\mathbf{M}]_{ij} = [\mathbf{M}]_{ji}, \forall i, j \in \mathcal{N}$;
3) Network-defined sparsity: $[\mathbf{M}]_{ij} > 0$ if $(i,j) \in \mathcal{L}$; otherwise, $[\mathbf{M}]_{ij} = 0, \forall i, j \in \mathcal{N}$.

The above conditions imply that the eigenvalues of $\mathbf{M}$ are real and thus could be sorted as: $-1 < \lambda_m(\mathbf{M}) \leq \cdots \leq \lambda_2(\mathbf{M}) < \lambda_1(\mathbf{M}) = 1$. We denote the second largest eigenvalue in magnitude of $\mathbf{M}$ as $\lambda \triangleq \max\{|\lambda_2(\mathbf{M})|, |\lambda_m(\mathbf{M})|\}$, which will play an important role in the step-size selection for our proposed algorithm. Note that the choice of $\mathbf{M}$ is not unique. For example, one possible choice of $\mathbf{M}$ that only relies on local information is the Metropolis weights [37]:

$$[\mathbf{M}]_{ij} = \{1 + \max[d(i), d(j)]\}^{-1}, \forall (i,j) \in \mathcal{L},$$
$$[\mathbf{M}]_{ii} = 1 - \sum_{j\neq i}[\mathbf{M}]_{ij}, \forall i \in \mathcal{N},$$

where $d(i) = |\mathcal{N}_i|$ is the degree of agent i.

Using the implicit function theorem, the hypergradient of $l_i(\mathbf{x}_i)$ for a given $\mathbf{x}_i \in \mathbb{R}^{d_{\text{up}}}$ can be evaluated as [6]:

$$\nabla l_i(\mathbf{x}_i) = \nabla_{\mathbf{x}} f_i\left(\mathbf{x}_i, \mathbf{y}_i^*(\mathbf{x}_i)\right) - \nabla^2_{\mathbf{xy}} g_i\left(\mathbf{x}_i, \mathbf{y}_i^*(\mathbf{x}_i)\right) \times \left[\nabla^2_{\mathbf{yy}} g_i\left(\mathbf{x}_i, \mathbf{y}_i^*(\mathbf{x}_i)\right)\right]^{-1} \nabla_{\mathbf{y}} f_i\left(\mathbf{x}_i, \mathbf{y}_i^*(\mathbf{x}_i)\right). \quad (3)$$

Since obtaining $\mathbf{y}_i^*(\mathbf{x}_i)$ in closed-form is difficult, $\bar{\nabla} f_i(\mathbf{x}_i, \mathbf{y}_i)$ is used as a surrogate of $\nabla l_i(\mathbf{x}_i)$ at any $(\mathbf{x}_i, \mathbf{y}_i) \in \mathbb{R}^{d_{\text{up}}\times d_{\text{low}}}$, which is defined as follows [6]:

$$\bar{\nabla} f_i\left(\mathbf{x}_i, \mathbf{y}_i\right) = \nabla_{\mathbf{x}} f_i\left(\mathbf{x}_i, \mathbf{y}_i\right) - \nabla^2_{\mathbf{xy}} g_i\left(\mathbf{x}_i, \mathbf{y}_i\right) \times \left[\nabla^2_{\mathbf{yy}} g_i\left(\mathbf{x}_i, \mathbf{y}_i\right)\right]^{-1} \nabla_{\mathbf{y}} f_i\left(\mathbf{x}_i, \mathbf{y}_i\right). \quad (4)$$

Note that the computation of $\bar{\nabla} f_i\left(\mathbf{x}_i, \mathbf{y}_i\right)$ involves exact Hessian matrix inverse and gradient, which is cumbersome. To avoid this expensive computation, we adopt the biased stochastic gradient estimation of $\bar{\nabla} f_i\left(\mathbf{x}_i, \mathbf{y}_i\right)$ defined as [11]:

$$\hat{\nabla} f_i\left(\mathbf{x}_i, \mathbf{y}_i; \bar{\xi}_i\right) = \nabla_{\mathbf{x}} f_i\left(\mathbf{x}_i, \mathbf{y}_i; \xi_i\right) - \frac{K}{L_g}\nabla^2_{\mathbf{xy}} g_i\left(\mathbf{x}_i, \mathbf{y}_i; \zeta_i^0\right) \times \prod_{j=1}^{k(K)}\left(I - \frac{\nabla^2_{\mathbf{yy}} g_i\left(\mathbf{x}_i, \mathbf{y}_i; \zeta_i^j\right)}{L_g}\right)\nabla_{\mathbf{y}} f_i\left(\mathbf{x}_i, \mathbf{y}_i; \xi_i\right), \quad (5)$$

where $k(K) \sim \mathcal{U}\{0, \cdots, K-1\}$ is a uniform random variable chosen from $\{0, \cdots, K-1\}$. A total of $K+2$ independent samples are collected from the upper- and lower-level distributions π_f and π_g, respectively. We denote all random variables needed in (5) as a $(K{+}3)$-tuple: $\bar{\xi}_i \triangleq \left\{\xi_i, \zeta_i^0, \cdots, \zeta_i^K, k(K)\right\}$, where $\xi_i \sim \pi_f$, $\zeta_i^j \sim \pi_g$, $j = 0, \cdots, K$.

The overall framework of the proposed DIAMOND algorithm for solving the decentralized bilevel optimization problem in (2) is summarized in Algorithm 1. Note that DIAMOND adopts a *single-loop* structure, which reduces the computation and implementation complexities

Algorithm 1 The DIAMOND Algorithm.

Input: Step-sizes α_t, β_t. Momentum coefficients η_t, γ_t.
Initialization: Let $\{\mathbf{x}_{i,-1}, \mathbf{y}_{i,-1}\}_{i=1}^m = \{\mathbf{x}_{-1}, \mathbf{y}_{-1}\}$. Let $\{\mathbf{p}_{i,-1}\}_{i=1}^m = \mathbf{0}$, and $\{\mathbf{v}_{i,-1}\}_{i=1}^m = \mathbf{0}$
for $t = 0$ to $T-1$ **do**
  for each agent $i \in [m]$ **do**
    Estimate the stochastic gradients $\mathbf{p}_{i,t}$ and $\mathbf{v}_{i,t}$ using (6) and (7);
    Track the global gradient $\mathbf{u}_{i,t}$ using (8);
    Update the local model parameters $\mathbf{x}_{i,t+1}$ and $\mathbf{y}_{i,t+1}$ using (9) and (10);
  end for
end for

compared with the double-loop structure. Meanwhile, DIAMOND relies on consensus updates along with gradient tracking and uses momentum-based stochastic gradient estimators, so that it matches state-of-the-art convergence guarantees. Thus, DIAMOND consists of three parts: i) local stochastic gradient estimation, ii) global gradient tracking, and iii) consensus update with stochastic gradient descent (SGD). The details of each part are described as follows:

1) Local Stochastic Gradient Estimation: Each agent estimates the momentum-based upper- and lower-level update directions $\mathbf{p}_{i,t}$ and $\mathbf{v}_{i,t}$ of the upper-level and lower-level functions, respectively, with its local stochastic gradients:

$$\begin{aligned}\mathbf{p}_{i,t} =&\hat{\nabla} f_i\left(\mathbf{x}_{i,t}, \mathbf{y}_{i,t}; \bar{\xi}_i\right) \\ &+ (1-\eta_t)\left(\mathbf{p}_{i,t-1} - \hat{\nabla} f_i\left(\mathbf{x}_{i,t-1}, \mathbf{y}_{i,t-1}; \bar{\xi}_i\right)\right), \quad (6)\\ \mathbf{v}_{i,t} =&\nabla_{\mathbf{y}} g_i\left(\mathbf{x}_{i,t}, \mathbf{y}_{i,t}; \zeta_i\right) \\ &+ (1-\gamma_t)\left(\mathbf{v}_{i,t-1} - \nabla_{\mathbf{y}} \mathbf{g}_i\left(\mathbf{x}_{i,t-1}, \mathbf{y}_{i,t-1}; \zeta_i\right)\right), \quad (7)\end{aligned}$$

where $\eta_t \in [0,1]$ and $\gamma_t \in [0,1]$ are the momentum coefficients. Note that the avoidance of the full gradient computation implies a noisy gradient estimation. DIAMOND utilizes momentum-based stochastic gradient estimations to improve the accuracy of the current gradient estimation, which is similar to [38], [39] for single-level stochastic optimization and [8], [11] for single-agent bilevel optimization.

2) Global Gradient Tracking: Each agent updates the global gradient $\mathbf{u}_{i,t}$ by averaging all its neighbors' gradient estimates $\mathbf{u}_{j,t-1}$, $j \in \mathcal{N}_i$, which is defined as:

$$\mathbf{u}_{i,t} = \sum_{j \in \mathcal{N}_i} [\mathbf{M}]_{ij} \mathbf{u}_{j,t-1} + \mathbf{p}_{i,t} - \mathbf{p}_{i,t-1}. \quad (8)$$

We do not perform gradient tracking for $\mathbf{v}_{i,t}$ since the lower-level $\mathbf{y}$-variables do not require consensus (cf. Problem (2)).

3) Consensus Update with Decentralized SGD: Each agent i updates the upper-level parameters by computing a weighted average of its neighbors' local copies $\mathbf{x}_{j,t}$, $j \in \mathcal{N}_i$ and using the tracked global gradient $\mathbf{u}_{i,t}$ computed in 2):

$$\mathbf{x}_{i,t+1} = \sum_{j \in \mathcal{N}_i} [\mathbf{M}]_{ij} \mathbf{x}_{j,t} - \alpha_t \mathbf{u}_{i,t}, \quad (9)$$

where α_t is the upper-level step-size. The lower-level model parameters $\mathbf{y}_{i,t+1}$ are updated locally by using $\mathbf{v}_{i,t}$:

$$\mathbf{y}_{i,t+1} = \mathbf{y}_{i,t} - \beta_t \mathbf{v}_{i,t}, \quad (10)$$

where β_t is the lower-level step-size. Again, note that consensus is only required for the upper-level $\mathbf{x}_i$-parameters.

V. Theoretical Performance Analysis

In this section, we establish the theoretical convergence guarantees for the DIAMOND algorithm for solving the decentralized bilevel optimization problem in (2). Before describing the proof details, we first highlight three major challenges in the convergence analysis in our DIAMOND algorithm:

a) Momentum-Based Stochastic Gradient Estimation Error of the Lower-Level Subproblem: Although the stochastic gradient estimation $\nabla_{\mathbf{y}} g_i(\mathbf{x}_{i,t}, \mathbf{y}_{i,t}, \zeta_i)$ of the lower-level objective function is unbiased, there exists a bias between $\mathbf{v}_{i,t}$ and the full gradient due to the randomness and the added momentum, which can be written as $\mathbf{e}_{i,t}^g \triangleq \mathbf{v}_{i,t} - \nabla_{\mathbf{y}} g_i\left(\mathbf{x}_{i,t}, \mathbf{y}_{i,t}\right)$.

b) Momentum-Based Stochastic Gradient Estimation Error of the Upper-Level Subproblem: The momentum-based stochastic gradient estimation error of the upper-level subproblem is caused by: i) the randomness of the gradient estimation $\mathbf{p}_{i,t}$, ii) the added momentum, and iii) the approximation $\mathbf{y}_{i,t} \approx \mathbf{y}_{i,t}^*(\mathbf{x}_{i,t})$. We denote this error as: $\mathbf{e}_{i,t}^f \triangleq \mathbf{p}_{i,t} - \bar{\nabla} f_i\left(\mathbf{x}_{i,t}, \mathbf{y}_{i,t}\right) - \mathbf{b}_{i,t}$. where $\mathbf{b}_{i,t} \triangleq \mathbb{E}_{\bar{\xi}_i}[\hat{\nabla} f_i\left(\mathbf{x}_{i,t}, \mathbf{y}_{i,t}; \bar{\xi}_i\right)] - \bar{\nabla} f_i\left(\mathbf{x}_{i,t}, \mathbf{y}_{i,t}\right)$ is the bias.

c) Consensus Error: DIAMOND utilizes a decentralized consensus update for the upper-level model parameters as shown in (8), which inevitably introduces consensus errors.

A. Main Convergence Results

To quantify the convergence rate performance of DIAMOND, we first define a new convergence metric specifically for the decentralized bilevel problem in (2):

$$\mathfrak{M}_t \triangleq \|\nabla l\left(\bar{\mathbf{x}}_t\right)\|^2 + \|\mathbf{x}_t - 1 \otimes \bar{\mathbf{x}}_t\|^2 + \|\mathbf{y}_t^* - \mathbf{y}_t\|^2, \quad (11)$$

where $\bar{\mathbf{x}}_t \triangleq \frac{1}{m}\sum_{i=1}^m \mathbf{x}_{i,t}$, $\mathbf{x}_t \triangleq \left[\mathbf{x}_{1,t}^\top, \ldots, \mathbf{x}_{m,t}^\top\right]^\top$, $\mathbf{y}_t \triangleq \left[\mathbf{y}_{1,t}^\top, \ldots, \mathbf{y}_{m,t}^\top\right]^\top$, and $\mathbf{y}_t^* \triangleq \left[\mathbf{y}_{1,t}^{*\top}, \ldots, \mathbf{y}_{m,t}^{*\top}\right]^\top$. The first term in (11) measures the convergence of the agent-average $\bar{\mathbf{x}}_t$ to a stationary point. The second term in (11) quantifies the consensus error among local copies of the upper-level $\mathbf{x}_t$-parameters. The third term in (11) measures the approximation error caused by the finite iterations of the lower-level optimization across all agents. Clearly, if $\mathfrak{M}_t \to 0$, we can conclude that the algorithm achieves three goals simultaneously: 1) achieving a stationary solution of the decentralized bilevel optimization problem in (2), 2) reaching consensus of upper-level model parameters across all agents, and 3) obtaining optimal solutions to the lower-level subproblem.

Next, we state the following assumptions that are useful for our convergence performance analysis:

Assumption 1 (Upper-Level Objective)**.** *$f_i\left(\mathbf{x}_i, \mathbf{y}_i\right)$ satisfies:*

1) *For any $i \in \{1, \ldots, m\}$ and $\left(\mathbf{x}_i, \mathbf{y}_i\right) \in \mathbb{R}^{d_{\text{up}}} \times \mathbb{R}^{d_{\text{low}}}$, $\nabla_{\mathbf{x}} f_i\left(\mathbf{x}_i, \mathbf{y}_i\right)$ and $\nabla_{\mathbf{y}} f_i\left(\mathbf{x}_i, \mathbf{y}_i\right)$ are Lipschitz continuous with constants $L_{f_x} \geq 0$ and $L_{f_y} \geq 0$, respectively.*
2) *For any $i \in \{1, \ldots, m\}$ and $\left(\mathbf{x}_i, \mathbf{y}_i\right) \in \mathbb{R}^{d_{\text{up}}} \times \mathbb{R}^{d_{\text{low}}}$, we have $\|\nabla_{\mathbf{y}} f_i\left(\mathbf{x}_i, \mathbf{y}_i\right)\| \leq C_{f_y}$ for some constants $C_{f_y} \geq 0$.*

Assumption 2 (Lower-Level Objective)**.** *$g_i\left(\mathbf{x}_i, \mathbf{y}_i\right)$ satisfies:*

1) *For any $\mathbf{x}_i \in \mathbb{R}^{d_{\text{up}}}$ and $\mathbf{y}_i \in \mathbb{R}^{d_{\text{low}}}$, $g_i\left(\mathbf{x}_i, \mathbf{y}_i\right)$ is twice continuously differentiable with respect to $\left(\mathbf{x}_i, \mathbf{y}_i\right)$.*
2) *For any $i \in \{1, \ldots, m\}$ and $\left(\mathbf{x}_i, \mathbf{y}_i\right) \in \mathbb{R}^{d_{\text{up}}} \times \mathbb{R}^{d_{\text{low}}}$, $\nabla_{\mathbf{y}} g_i\left(\mathbf{x}_i, \mathbf{y}_i\right)$ is Lipschitz continuous with constant $L_g \geq 0$.*
3) *For any $i \in \{1, \ldots, m\}$ and $\mathbf{x}_i \in \mathbb{R}^{d_{\text{up}}}$, $g_i\left(\mathbf{x}_i, \cdot\right)$ is μ_g-strongly convex with respect to $\mathbf{y}_i$ for some $\mu_g > 0$.*
4) *For any $i \in \{1, \ldots, m\}$ and $\left(\mathbf{x}_i, \mathbf{y}_i\right) \in \mathbb{R}^{d_{\text{up}}} \times \mathbb{R}^{d_{\text{low}}}$, $\nabla_{\mathbf{xy}}^2 g_i\left(\mathbf{x}_i, \mathbf{y}_i\right)$ and $\nabla_{\mathbf{yy}}^2 g_i\left(\mathbf{x}_i, \mathbf{y}_i\right)$ are Lipschitz continuous with constants $L_{g_{xy}} \geq 0$ and $L_{g_{yy}} \geq 0$, respectively.*

5) *For any* $i \in \{1,\dots,m\}$ *and* $(\mathbf{x}_i,\mathbf{y}_i) \in \mathbb{R}^{d_{\rm up}} \times \mathbb{R}^{d_{\rm low}}$, *we have* $\left\|\nabla^2_{\mathbf{xy}} g_i(\mathbf{x}_i,\mathbf{y}_i)\right\|^2 \le C_{g_{xy}}$ *for some* $C_{g_{xy}} > 0$.

Assumption 3 (Stochastic Objectives)**.** *Assumptions 1 and 2 hold for* $f_i(\mathbf{x}_i,\mathbf{y}_i;\xi_i)$ *and* $g_i(\mathbf{x}_i,\mathbf{y}_i;\zeta_i)$, *for all* $\xi_i \in \operatorname{supp}(\pi_f)$ *and* $\zeta_i \in \operatorname{supp}(\pi_g)$ *where* $\operatorname{supp}(\pi)$ *denotes the support of distribution* π.

Assumption 4 (Stochastic Gradients)**.** *For any* $i \in \{1,\dots,m\}$ *and* $(\mathbf{x}_i,\mathbf{y}_i) \in \mathbb{R}^{d_{\rm up}} \times \mathbb{R}^{d_{\rm low}}$, *the gradient estimators* $\hat{\nabla} f_i\left(\mathbf{x}_i,\mathbf{y}_i;\bar{\xi}_i\right)$ *and* $\nabla_{\mathbf{y}} g_i(\mathbf{x}_i,\mathbf{y}_i;\zeta_i)$ *satisfy:*

1) *There exists a constant* $\sigma_f \ge 0$ *such that* $\mathbb{E}_{\bar{\xi}_i}[\|\hat{\nabla} f_i(\mathbf{x}_i,\mathbf{y}_i;\bar{\xi}_i) - \bar{\nabla} f_i(\mathbf{x}_i,\mathbf{y}_i) - \mathbf{b}_i(\mathbf{x}_i,\mathbf{y}_i)\|^2] \le \sigma_f^2$, *where* $\mathbf{b}_i(\mathbf{x}_i,\mathbf{y}_i) \triangleq \mathbb{E}_{\bar{\xi}_i}[\hat{\nabla} f_i(\mathbf{x}_i,\mathbf{y}_i;\bar{\xi}_i)] - \bar{\nabla} f_i(\mathbf{x}_i,\mathbf{y}_i)$ *is the bias in estimating* $\bar{\nabla} f_i(\mathbf{x}_i,\mathbf{y}_i)$.
2) *There exists a constant* $\sigma_g \ge 0$ *such that* $\mathbb{E}_{\zeta_i}[\|\nabla_{\mathbf{y}} g_i(\mathbf{x}_i,\mathbf{y}_i;\zeta_i) - \nabla_{\mathbf{y}} g_i(\mathbf{x}_i,\mathbf{y}_i)\|^2] \le \sigma_g^2$.

We note that all these assumptions are standard in the bilevel optimization literature (see, e.g., [6], [11], [13]). Next, we state two lemmas on characterizing the Lipschitz constants of the hypergradient $\nabla l_i(\mathbf{x}_i)$ in (3), the approximate gradient $\bar{\nabla} f_i(\mathbf{x}_i,\mathbf{y}_i)$ in (4), the optimal solution $\mathbf{y}_i^*$ of the lower-level problem, and the stochastic gradient estimator $\hat{\nabla} f_i\left(\mathbf{x}_i,\mathbf{y}_i;\bar{\xi}_i\right)$ in (5). These lemmas will be useful in our main convergence results.

Lemma 1 (Ref. [6])**.** *Under Assumptions 1 and 2, we have*

$$\begin{aligned}
&\|\bar{\nabla} f_i(\mathbf{x},\mathbf{y}) - \nabla l_i(\mathbf{x})\| \le L_f \|\mathbf{y}^*(\mathbf{x}) - \mathbf{y}\|,\\
&\|\mathbf{y}_i^*(\mathbf{x}_1) - \mathbf{y}_i^*(\mathbf{x}_2)\| \le L_y \|\mathbf{x}_1 - \mathbf{x}_2\|,\\
&\|\nabla l_i(\mathbf{x}_1) - \nabla l_i(\mathbf{x}_2)\| \le L_l \|\mathbf{x}_1 - \mathbf{x}_2\|,
\end{aligned}$$

for all $i \in \{1,\dots,m\}$, $\mathbf{x},\mathbf{x}_1,\mathbf{x}_2 \in \mathbb{R}^{d_{\rm up}}$ *and* $\mathbf{y} \in \mathbb{R}^{d_{\rm low}}$, *where the Lipschitz constants above are defined as:*

$$\begin{aligned}
&L_f = L_{f_x} + \frac{L_{f_y} C_{g_{xy}}}{\mu_g} + C_{f_y}\left(\frac{L_{g_{xy}}}{\mu_g} + \frac{L_{g_{yy}} C_{g_{xy}}}{\mu_g^2}\right),\\
&L_l = L_f + \frac{L_f C_{g_{xy}}}{\mu_g}, \quad \text{and} \quad L_y = \frac{C_{g_{xy}}}{\mu_g}.
\end{aligned}$$

Lemma 2 (Ref. [11])**.** *Under Assumptions 1, 2 and 3, for any* $(\mathbf{x}_1,\mathbf{y}_1)$, $(\mathbf{x}_2,\mathbf{y}_2) \in \mathbb{R}^{d_{\rm up}} \times \mathbb{R}^{d_{\rm low}}$, *we have*

$$\begin{aligned}
&\mathbb{E}_{\bar{\xi}}\left\|\hat{\nabla} f_i\left(\mathbf{x}_1,\mathbf{y}_1;\bar{\xi}\right) - \hat{\nabla} f_i\left(\mathbf{x}_2,\mathbf{y}_2;\bar{\xi}\right)\right\|\\
&\qquad\qquad \le L_K(\|\mathbf{x}_1 - \mathbf{x}_2\| + \|\mathbf{y}_1 - \mathbf{y}_2\|),
\end{aligned}$$

where $L_K =$

$$2L_{f_x}^2 + \frac{6C_{g_{xy}}^2 L_{f_y}^2 K}{2\mu_g L_g - \mu_g^2} + \frac{6C_{f_y}^2 L_{g_{xy}}^2 K}{2\mu_g L_g - \mu_g^2} + \frac{6C_{g_{xy}}^2 C_{f_y}^2 L_{g_{yy}}^2 K^3}{(L_g - \mu_g)^2\left(2\mu_g L_g - \mu_g^2\right)},$$

and K *is the number of samples required to construct the stochastic gradient estimate in* (5).

The following lemma says that the bias of the stochastic gradient estimator for the upper-level objective defined in (5) decays exponentially fast with respect to the number of samples K that is chosen to approximate the Hessian inverse.

Lemma 3 (Ref. [1])**.** *Under Assumptions 1–3, for any* $i \in \{1,\dots,m\}$ *and* $(\mathbf{x},\mathbf{y}) \in \mathbb{R}^{d_{\rm up}} \times \mathbb{R}^{d_{\rm low}}$, *the bias of the stochastic gradient estimator in* (5) *satisfies:*

$$\|\bar{\nabla} f_i(\mathbf{x}_i,\mathbf{y}_i) - \mathbb{E}[\hat{\nabla} f_i(\mathbf{x}_i,\mathbf{y}_i;\bar{\xi}_i)]\| \le \frac{C_{g_{xy}} C_{f_y}}{\mu_g}\left(1 - \frac{\mu_g}{L_g}\right)^K,$$

where K *is the number of samples required to construct the stochastic gradient estimate in* (5).

Now, based on the convergence metric defined in (11), we state the main convergence result of DIAMOND in Theorem 1:

Theorem 1 (Convergence Rate of DIAMOND)**.** *Under Assumptions 1–4, choose* $K = (L_g/\mu_g)\log\left(C_{g_{xy}} C_{f_y} T/\mu_g\right)$. *Define* $\alpha_t \triangleq (\omega + t)^{-1/3}$ *for* $\omega \ge 2$, $\beta_t \triangleq c_\beta \alpha_t$, $\eta_{t+1} \triangleq c_\eta \alpha_t^2$, *and* $\gamma_{t+1} \triangleq c_\gamma \alpha_t^2$, *where*

$$\begin{aligned}
&c_\beta = \frac{8L_f}{mL_{\mu_g}\bar{c}_y}, \quad c_\eta = \frac{6L_f\bar{c}_\eta + m}{3L_f m(1 - 3\bar{c}_u\bar{c}_\eta)},\\
&c_\gamma = \frac{1}{3L_f} + 8L_g^2 c_\beta^2 + \bar{c}_r\left(\frac{2c_\beta\bar{c}_y}{L_{\mu_g}} + \frac{8L_K^2 c_\beta^2}{\bar{c}_\eta} + 12L_K^2\bar{c}_u c_\beta^2\right),
\end{aligned}$$

and where

$$\begin{aligned}
&\bar{c}_y = \min\left\{\sqrt{\frac{L_f}{2L_y^2 m}}, \sqrt{\frac{L_f}{10L_y^2 m^2}}\right\},\ \bar{c}_u = \min\left\{\frac{1}{48L_K^2}, \frac{1}{3\bar{c}_\eta}\right\},\\
&\bar{c}_\eta = \max\left\{32L_K^2, 160L_K^2 m, \frac{24(\mu_g + L_g)L_K^2 c_\beta}{\bar{c}_y}\right\},\\
&\bar{c}_\gamma = \max\left\{32L_g^2, 160L_g^2 m, \frac{24(\mu_g + L_g)L_g^2 c_\beta}{\bar{c}_y}\right\}.
\end{aligned}$$

Let $\mathfrak{B}_t \triangleq l(\bar{\mathbf{x}}_t) + \bar{c}_y\|\mathbf{y}_t - \mathbf{y}_t^*\|^2 + \bar{c}_x\|\mathbf{x}_t - \mathbf{1}\otimes\bar{\mathbf{x}}_t\|^2 + \bar{c}_u\|\mathbf{u}_t - \mathbf{1}\otimes\bar{\mathbf{u}}_t\|^2$ *with* $\bar{c}_x = \frac{6}{(1-\lambda)\alpha_t}$, $\bar{\mathbf{u}}_t \triangleq \frac{1}{m}\sum_{i=1}^m \mathbf{u}_{i,t}$ *and* $\mathbf{u}_t \triangleq \left[\mathbf{u}_{1,t}^\top,\dots,\mathbf{u}_{m,t}^\top\right]^\top$. *If* $\alpha_t \le \min\{\sqrt{\frac{m}{2L_l^2}}, \frac{\bar{c}_u(1-\lambda)^2}{30}, \frac{\bar{c}_u(1-\lambda)L_{\mu_g}c_\beta}{40L_y^2\bar{c}_y}, \frac{\bar{c}_u(1-\lambda)\bar{c}_\eta}{80L_K^2}, \frac{\bar{c}_u(1-\lambda)\bar{c}_\gamma}{80L_g^2}, \frac{1}{5L_l}, \frac{1}{3L_f}, \sqrt{\frac{(1-\lambda)^2}{120L_K^2}}, \frac{1-\lambda}{240\bar{c}_u L_K^2 m}, \frac{\bar{c}_y(1-\lambda)}{36(\mu_g+L_g)\bar{c}_u L_K^2 c_\beta}, 1-\lambda\}$ *and* $\beta_t \le \frac{1}{\mu_g + L_g}$, *then the sequence* $\{\mathbf{x}_t,\mathbf{y}_t\}$ *generated by Algorithm 1 satisfies:*

$$\begin{aligned}
\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[\mathfrak{M}_t] &= \mathcal{O}\left(\frac{\mathfrak{B}_0 - l^*}{T^{2/3}}\right) + \mathcal{O}\left(\frac{\log(T)\,\sigma_f^2}{T^{2/3}}\right)\\
&\quad + \mathcal{O}\left(\frac{\log(T)\,\sigma_g^2}{T^{2/3}}\right) = \tilde{\mathcal{O}}\left(\frac{1}{T^{2/3}}\right).
\end{aligned}$$

Theorem 1 indicates that the decreasing step-sizes $\{\alpha_t,\beta_t\}$ depend on the Lipschitz constants, the number of agents, and the network topology. Note also that the choice of step-size α_t is directly affected by λ, the second largest eigenvalue in magnitude of the weight matrix $\mathbf{M}$. Further, Theorem 1 immediately implies the following sample complexity and communication complexity of DIAMOND:

Corollary 1 (Sample and Communication Complexities of DIAMOND)**.** *Under the conditions stated in Theorem 1,* DIAMOND *requires* $\mathcal{O}(\epsilon^{-3/2})$ *in sample complexity and communication complexity to reach an* ϵ*-stationary point.*

B. Proofs of the Main Theoretical Results

Due to space limitation, we provide a proof sketch of Theorem 1, which is organized into several key steps:

Step 1) Per-iterate descent of the upper-level objective function: We first bound the per-iterate descent of the upper-level objective function as follows:

Lemma 4. *Under Assumptions 1–2, the following inequality holds for the consecutive iterates of Algorithm 1:*

$$\begin{aligned}\mathbb{E}\left[l\left(\bar{\mathbf{x}}_{t+1}\right)-l\left(\bar{\mathbf{x}}_t\right)\right] \leq \mathbb{E}\Bigg[&-\frac{\alpha_t}{2}\left\|\nabla l\left(\bar{\mathbf{x}}_t\right)\right\|^2+\frac{2\alpha_t}{m}\sum_{i=1}^m\left\|\mathbf{e}_{i,t}^f\right\|^2\\&-\left(\frac{\alpha_t}{2}-\frac{L_l\alpha_t^2}{2}\right)\left\|\bar{\mathbf{u}}_t\right\|^2+\frac{2L_l^2\alpha_t}{m}\sum_{i=1}^m\left\|\bar{\mathbf{x}}_t-\mathbf{x}_{i,t}\right\|^2\\&+\frac{2L_f^2\alpha_t}{m}\sum_{i=1}^m\left\|\mathbf{y}_{i,t}^*-\mathbf{y}_{i,t}\right\|^2+\frac{2\alpha_t}{m}\sum_{i=1}^m\left\|\mathbf{b}_{i,t}\right\|^2\Bigg],\end{aligned}$$

for all $t \in \{0,1,\ldots,T-1\}$, where the expectation is taken over all randomness of the algorithm.

Lemma 4 bounds the expected per-iterate descent of the upper-level objective value, which depends on i) the consensus error of the upper-level parameters $\mathbb{E}[\|\bar{\mathbf{x}}_t-\mathbf{x}_{i,t}\|^2]$, ii) the momentum-based gradient estimation error of the upper-level objective function $\mathbb{E}[\|\mathbf{e}_{i,t}^f\|^2]$ including the bias $\|\mathbf{b}_{i,t}\|^2$, and iii) the approximation gap of the lower-level optimal parameter $\mathbb{E}[\|\mathbf{y}_{i,t}^*-\mathbf{y}_{i,t}\|^2]$, which will be bounded in Step 2).

Step 2) Error bound on $\mathbf{y}^*(\mathbf{x})$: We show that the approximation error of $\mathbf{y}^*(\mathbf{x})$ can be bounded as:

Lemma 5. *Under Assumptions 1 and 2, the following approximation gap of $\mathbf{y}^*(\mathbf{x})$ holds for Algorithm 1:*

$$\begin{aligned}\mathbb{E}\left[\left\|\mathbf{y}_{i,t+1}-\mathbf{y}_{i,t+1}^*\right\|^2\right] \leq \mathbb{E}\Bigg[&\left(1+\frac{1}{c_1}\right)L_y^2\left\|\mathbf{x}_{i,t+1}-\mathbf{x}_{i,t}\right\|^2\\&+(1+c_1)(1+c_0)\left(1-\frac{2\beta_t\mu_g L_g}{\mu_g+L_g}\right)\left\|\mathbf{y}_{i,t}-\mathbf{y}_{i,t}^*\right\|^2\\&+(1+c_1)(1+c_0)\left(\beta_t^2-\frac{2\beta_t}{\mu_g+L_g}\right)\left\|\nabla_{\mathbf{y}}g_i\left(\mathbf{x}_{i,t},\mathbf{y}_{i,t}\right)\right\|^2\\&+(1+c_1)\left(1+\frac{1}{c_0}\right)\beta_t^2\left\|\mathbf{e}_{i,t}^g\right\|^2\Bigg],\end{aligned}$$

for all $i \in \{1,\ldots,m\}$ and $t \in \{0,1,\ldots,T-1\}$ with some positive constants $c_0, c_1 > 0$, where the expectation is taken over all randomness of the algorithm.

Lemma 5 indicates that the approximation error $\mathbf{y}^*(\mathbf{x})$ shrinks if $(1+c_1)(1+c_0)(1-2\beta_t\mu_g L_g/(\mu_g+L_g)) < 1$, and is influenced by the momentum-based stochastic gradient estimation error of the lower-level objective function $\mathbb{E}[\|\mathbf{e}_{i,t}^g\|^2]$. Because of the tightly coupled structure of the bilevel problem, $\mathbb{E}[\|\mathbf{y}_{i,t+1}-\mathbf{y}_{i,t+1}^*\|^2]$ is also affected by the upper-level parameters $\mathbb{E}[\|\mathbf{x}_{i,t+1}-\mathbf{x}_{i,t}\|^2]$, which is in turn affected by the consensus error in decentralized optimization.

Step 3) Shrinking rate of $\mathbf{e}_{i,t}^f$: Next, we bound the stochastic gradient estimation error $\mathbf{e}_{i,t}^f$ of the upper-level objective function as follows:

Lemma 6. *Under Assumptions 1-4, the stochastic gradient estimation error of the upper-level objective function $\mathbf{e}_{i,t}^f$ satisfies the following relationship:*

$$\begin{aligned}\mathbb{E}[\|\mathbf{e}_{i,t+1}^f\|^2] \leqslant&\mathbb{E}[(1-\eta_{t+1})^2\|\mathbf{e}_{i,t}^f\|^2+2\eta_{t+1}^2\sigma_f^2\\&+4(1-\eta_{t+1})^2L_K^2\|\mathbf{x}_{i,t+1}-\mathbf{x}_{i,t}\|^2\\&+8(1-\eta_{t+1})^2L_K^2\beta_t^2\|\mathbf{e}_{i,t}^g\|^2\\&+8(1-\eta_{t+1})^2L_K^2\beta_t^2\|\nabla_{\mathbf{y}}g_i(\mathbf{x}_{i,t},\mathbf{y}_{i,t})\|^2],\end{aligned}$$

for all $i \in \{1,\ldots,m\}$ and $t \in \{0,1,\ldots,T-1\}$, where the expectation is taken over all randomness of the algorithm.

Lemma 6 indicates that the momentum-based stochastic gradient estimation error of the upper-level function $\mathbb{E}[\|\mathbf{e}_{i,t+1}^f\|^2]$ is affected by the consensus error of the upper-level parameters, which is contained in $\mathbb{E}[\|\mathbf{x}_{i,t+1}-\mathbf{x}_{i,t}\|^2]$, the lower-level momentum-based stochastic gradient estimation error $\mathbb{E}[\|\mathbf{e}_{i,t}^g\|^2]$, and the full gradient norm $\mathbb{E}[\|\nabla_{\mathbf{y}}g_i(\mathbf{x}_{i,t},\mathbf{y}_{i,t})\|^2]$.

Step 4) Shrinking rate of $\mathbf{e}_{i,t}^g$: Next, we bound the stochastic gradient estimation error $\mathbf{e}_{i,t}^g$ of the lower-level objective function as follows:

Lemma 7. *Under Assumptions 1-4, the stochastic gradient estimation error of the lower-level function satisfies the following relationship:*

$$\begin{aligned}\mathbb{E}[\|\mathbf{e}_{i,t+1}^g\|^2] \leq&\mathbb{E}[(1-\gamma_{t+1})^2(1+8L_g^2\beta_t^2)\|\mathbf{e}_{i,t}^g\|^2\\&+4(1-\gamma_{t+1})^2L_g^2\|\mathbf{x}_{i,t+1}-\mathbf{x}_{i,t}\|^2+2r_{t+1}^2\sigma_g^2\\&+8(1-\gamma_{t+1})^2L_g^2\beta_t^2\|\nabla_{\mathbf{y}}g_i(\mathbf{x}_{i,t},\mathbf{y}_{i,t})\|^2],\end{aligned}$$

for all $i \in \{1,\ldots,m\}$ and $t \in \{0,1,\ldots,T-1\}$, where the expectation is taken over all randomness of the algorithm.

Lemma 7 shows that, since decentralized bilevel optimization is a composition of a lower-level problem and an upper-level problem, the momentum-based stochastic gradient estimation error of the lower-level function $\mathbb{E}[\|\mathbf{e}_{i,t+1}^g\|^2]$ is influenced by the consensus error of the upper-level parameters, which is contained in $\mathbb{E}[\|\mathbf{x}_{i,t+1}-\mathbf{x}_{i,t}\|^2]$.

Step 5) Iterate contractions: Next, we establish the following iterate contraction results of the DIAMOND algorithm:

Lemma 8. *The following contraction properties of the iterates in Algorithm 1 hold:*

$$\begin{aligned}\left\|\mathbf{x}_{t+1}-\mathbf{1}\otimes\bar{\mathbf{x}}_{t+1}\right\|^2 \leq&(1+c_2)\lambda^2\left\|\mathbf{x}_t-\mathbf{1}\otimes\bar{\mathbf{x}}_t\right\|^2\\&+\left(1+\frac{1}{c_2}\right)\alpha_t^2\left\|\mathbf{u}_t-\mathbf{1}\otimes\bar{\mathbf{u}}_t\right\|^2,\\\left\|\mathbf{u}_{t+1}-\mathbf{1}\otimes\bar{\mathbf{u}}_{t+1}\right\|^2 \leq&(1+c_3)\lambda^2\left\|\mathbf{u}_t-\mathbf{1}\otimes\bar{\mathbf{u}}_t\right\|^2\\&+\left(1+\frac{1}{c_3}\right)\left\|\mathbf{p}_{t+1}-\mathbf{p}_t\right\|^2,\end{aligned}$$

where $c_2, c_3 > 0$ are constants, and $\mathbf{p}_t \triangleq \left[\mathbf{p}_{1,t}^\top, \cdots, \mathbf{p}_{m,t}^\top\right]^\top$. In addition, we have

$$\begin{aligned}
\|\mathbf{x}_{t+1} - \mathbf{x}_t\|^2 \leqslant\ & 8\|\mathbf{x}_t - \mathbf{1}\otimes\bar{\mathbf{x}}_t\|^2 \\
& + 4\alpha_t^2\|\mathbf{u}_t - \mathbf{1}\otimes\bar{\mathbf{u}}_t\|^2 + 4\alpha_t^2 m\|\bar{\mathbf{u}}_t\|^2, \\
\mathbb{E}[\|\mathbf{p}_{t+1} - \mathbf{p}_t\|^2] \leq\ & \mathbb{E}\Bigg[3\eta_{t+1}^2\sum_{i=1}^m\|\mathbf{e}_{i,t}^f\|^2 + 24L_K^2 m\alpha_t^2\|\bar{\mathbf{u}}_t\|^2 \\
& + 12L_K^2\beta_t^2\sum_{i=1}^m\|\nabla_{\mathbf{y}} g_i(\mathbf{x}_{i,t},\mathbf{y}_{i,t})\|^2 + 48L_K^2\|\mathbf{x}_t - \mathbf{1}\otimes\bar{\mathbf{x}}_t\|^2 + \\
& 24L_K^2\alpha_t^2\|\mathbf{u}_t - \mathbf{1}\otimes\bar{\mathbf{u}}_t\|^2 + 12L_K^2\beta_t^2\sum_{i=1}^m\|\mathbf{e}_{i,t}^g\|^2 + 3\eta_{t+1}^2\sigma_f^2 m\Bigg],
\end{aligned}$$

where $\mathbb{E}[\cdot]$ is taken over all randomness of the algorithm.

Note that we only attempt to reach consensus in the upper-level $\mathbf{x}$-variables. Step 5 quantifies the impacts of the consensus error in the iterates of the upper-level $\mathbf{x}$-variables, which is important to analyze the convergence of DIAMOND. Our key idea is to first define $\tilde{\mathbf{M}} = \mathbf{M}\otimes\mathbf{I}_m$. Since $\mathbf{x}_t - \mathbf{1}\otimes\mathbf{x}_t$ is orthogonal to $\mathbf{1}$, which is the eigenvector corresponding to the largest eigenvalue of $\tilde{\mathbf{M}}$, and $\lambda = \max\{|\lambda_2|, |\lambda_m|\}$, we have $\|\tilde{\mathbf{M}}\mathbf{x}_t - \mathbf{1}\otimes\bar{\mathbf{x}}_t\|^2 = \|\tilde{\mathbf{M}}(\mathbf{x}_t - \mathbf{1}\otimes\bar{\mathbf{x}}_t)\|^2 \leq \lambda^2\|\mathbf{x}_t - \mathbf{1}\otimes\bar{\mathbf{x}}_t\|^2$.

Step 6) Decrement of a constructed potential function: Next, we define a potential function W_t as follows:

$$\begin{aligned}
W_t = & \ l(\bar{\mathbf{x}}_t) + \bar{c}_y\|\mathbf{y}_t - \mathbf{y}_t^*\|^2 + \bar{c}_x\|\mathbf{x}_t - \mathbf{1}\otimes\bar{\mathbf{x}}_t\|^2 \\
& + \bar{c}_u\|\mathbf{u}_t - \mathbf{1}\otimes\bar{\mathbf{u}}_t\|^2 + \frac{\|\mathbf{e}_t^f\|^2}{\bar{c}_\eta\alpha_{t-1}} + \frac{\|\mathbf{e}_t^g\|^2}{\bar{c}_\gamma\alpha_{t-1}}. \quad (12)
\end{aligned}$$

Then, we can show the following decrement results for $\{W_t\}$:

Lemma 9. *Choose $c_0 = \frac{\beta_t L_{\mu_g}}{1-2\beta_t L_{\mu_g}}$, $c_1 = \frac{\beta_t L_{\mu_g}}{2(1-\beta_t L_{\mu_g})}$ and $c_2 = c_3 = \frac{1}{\lambda} - 1$. Under the same conditions as in Theorem 1 and using the results in Lemmas 4–8, the iterates generated by Algorithm 1 satisfy:*

$$\begin{aligned}
\mathbb{E}\left[W_{t+1} - W_t\right] \leq \mathbb{E}\Bigg[& -\frac{\alpha_t}{2}\left\|\nabla l\left(\bar{\mathbf{x}}_t\right)\right\|^2 - \frac{2L_f^2}{m}\alpha_t\left\|\mathbf{y}_t^* - \mathbf{y}_t\right\|^2 \\
& - \frac{1}{1-\lambda}\left\|\mathbf{x}_t - \mathbf{1}\otimes\bar{\mathbf{x}}_t\right\|^2 + \frac{2\alpha_t}{m}\sum_{i=1}^m\left\|\mathbf{b}_{i,t}\right\|^2 \\
& + \frac{2m\eta_{t+1}^2}{\bar{c}_\eta\alpha_t}\sigma_f^2 + \frac{2m\gamma_{t+1}^2}{\bar{c}_\gamma\alpha_t}\sigma_g^2 + \frac{3\bar{c}_u m\eta_{t+1}^2}{\alpha_t}\sigma_f^2\Bigg].
\end{aligned}$$

With the proposed potential function and setting the parameters properly, we can make the coefficients of $\mathbb{E}[\|\mathbf{u}_t - \mathbf{1}\otimes\bar{\mathbf{u}}_t\|^2]$, $\mathbb{E}[\|\bar{\mathbf{u}}_t\|^2]$, $\mathbb{E}[\sum_{i=1}^m\|\mathbf{e}_{,t}^f\|^2]$, $\mathbb{E}[\sum_{i=1}^m\|\mathbf{e}_{,t}^g\|^2]$ and $\mathbb{E}[\sum_{i=1}^m\|\nabla_{\mathbf{y}} g_i(\mathbf{x}_{i,t},\mathbf{y}_{i,t})\|^2]$ to be non-positive in the range of α_t and β_t, which leads to the stated result in Lemma 9.

Step 7) Proof of Theorem 1: Note that $\sum_{t=0}^{T-1}\alpha_t^3 \leq \log(T+1)$ with $\omega \geq 1$. Telescoping the result in Lemma 9 from 0 to $T-1$, and multiplying by $2/\alpha_T$ on both sides yields:

$$\begin{aligned}
\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\mathfrak{M}_t\right] \leqslant & \left[\frac{2\left(\mathfrak{B}_0 - l^*\right)}{\alpha_T T} + \frac{4}{m\alpha_T T}\sum_{t=0}^{T-1}\alpha_t E\left[\sum_{i=1}^m\left\|\mathbf{b}_{i,t}\right\|^2\right]\right. \\
& + \frac{4c_\eta^2 m\sigma_f^2}{\bar{c}_\eta}\frac{\log(T+1)}{\alpha_T T} + \frac{4c_\gamma^2 m\sigma_g^2}{\bar{c}_\gamma}\frac{\log(T+1)}{\alpha_T T} + \frac{2\sigma_f^2}{\bar{c}_\eta\alpha_{-1}\alpha_T T} \\
& \left. + 6\bar{c}_u c_\eta^2 m\sigma_f^2\frac{\log(T+1)}{\alpha_T T} + \frac{2\sigma_g^2}{\bar{c}_\gamma\alpha_{-1}\alpha_T T}\right] / \min\left\{\frac{4L_f^2}{m}, \frac{2}{1-\lambda}\right\}.
\end{aligned}$$

Using the fact that $\|\mathbf{b}_{i,t}\| = 1/T$ when $K = (\mu_g/L_g)\log\left(C_{g_{xy}}C_{f_y}T/\mu_g\right)$, we have:

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\mathfrak{M}_t\right] = \mathcal{O}\left(\frac{\mathfrak{B}_0 - l^*}{T^{2/3}}\right) + \widetilde{\mathcal{O}}\left(\frac{\sigma_f^2}{T^{2/3}}\right) + \widetilde{\mathcal{O}}\left(\frac{\sigma_g^2}{T^{2/3}}\right),$$

where $\mathfrak{B}_t \triangleq l(\bar{\mathbf{x}}_t) + \bar{c}_y\|\mathbf{y}_t - \mathbf{y}_t^*\|^2 + \bar{c}_x\|\mathbf{x}_t - \mathbf{1}\otimes\bar{\mathbf{x}}_t\|^2 + \bar{c}_u\|\mathbf{u}_t - \mathbf{1}\otimes\bar{\mathbf{u}}_t\|^2$. This completes the proof of Theorem 1.

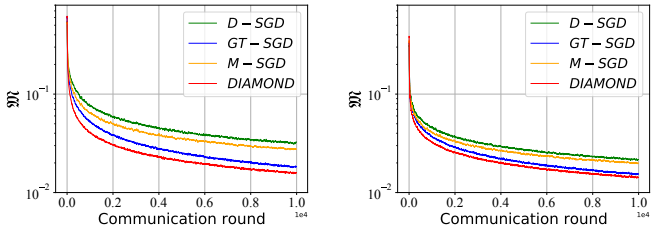
VI. Numerical Results

In this section, we conduct numerical experiments to verify the theoretical results of the proposed DIAMOND algorithm using decentralized meta-learning problems and hyper-parameter optimization problems.

1) Decentralized Meta-Learning: The formulation of the decentralized meta-learning problem is the same as (1), where $\mathbf{x}$ is the common parameters across all agents and $\mathbf{y}_i$ is the task-specific parameters of $\mathcal{T}_i$. The upper-level optimization objective function $f_i(\mathbf{x}, \mathbf{y}_i^*(\mathbf{x}))$ is the loss function related to task $\mathcal{T}_i$ and is non-convex in $\mathbf{x}$. The lower-level optimization objective function $g_i(\mathbf{x}, \mathbf{y}_i)$ satisfies the strongly convex requirements in $\mathbf{y}_i$ by using a strongly convex regularizor $\mathcal{R}(\mathbf{y}_i) = \frac{1}{2}\|\mathbf{y}_i\|_2^2$. The decentralized meta-learning problem aims to learn the common parameters that can be adapted to specific tasks. The classifier is based on a two-hidden-layer fully connected neural network for the decentralized meta-learning problem. The network topology is generated based on Erdös-Rényi random graph by NetworkX [40]. We choose the consensus matrix as $\mathbf{M} = \mathbf{I} - \frac{2\mathbf{V}}{3\rho_{\max}(\mathbf{V})}$, where $\mathbf{V}$ is the Laplacian matrix and $\rho_{\max}$ is the largest eigenvalue of $\mathbf{V}$.

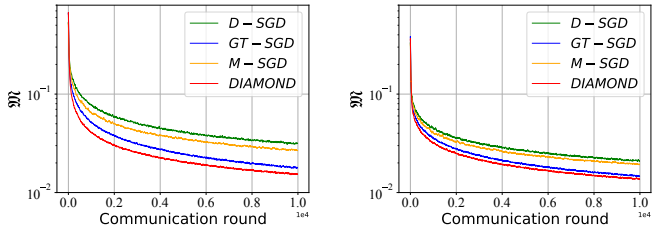
1-a) Comparison between Decentralized Stochastic Algorithms: We conduct decentralized meta-learning on MNIST [41] and CIFAR-10 [42] datasets with 9- and 15-agent networks. The edge connectivity probability is $p_c = 0.3$. We compare the proposed DIAMOND algorithm with three decentralized stochastic bilevel optimization methods:

- **Decentralized Stochastic Gradient Descent (D-SGD) [14]:** Each agent directly updates its local copy of the upper-level parameters $\mathbf{x}_{i,t}$ with the stochastic gradient of the upper-level objective function, i.e., $\mathbf{x}_{i,t} = \sum_{j\in\mathcal{N}_i}[\mathbf{M}]_{ij}\mathbf{x}_{j,t-1} - \alpha_t\hat{\nabla} f_i\left(\mathbf{x}_{i,t}, \mathbf{y}_{i,t}; \bar{\xi}_{i,t}\right)$ and $\mathbf{y}_{i,t} = \mathbf{y}_{i,t-1} - \beta_t\nabla_{\mathbf{y}} g_i\left(\mathbf{x}_{i,t}, \mathbf{y}_{i,t}; \zeta_{i,t}\right)$.
- **Gradient-Tracking Stochastic Gradient Descent (GT-SGD):** In addition to the DSGD update, GT-SGD performs gradient tracking over $\mathbf{x}_{i,t}$-variables. Specifically, GT-SGD updates the upper-level variables



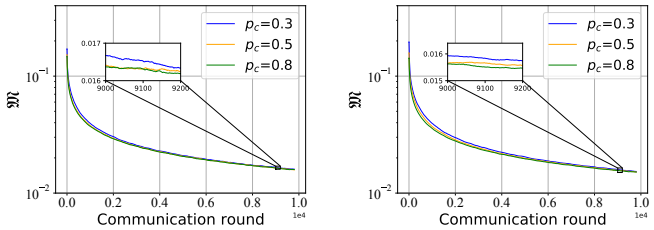
(a) MNIST dataset. (b) CIFAR-10 dataset.

Fig. 1: Convergence performance on the meta-learning problem with 9-agent network.



(a) MNIST dataset. (b) CIFAR-10 dataset.

Fig. 2: Convergence performance on the meta-learning problem with 15-agent network.



(a) 9-agent network. (b) 15-agent network.

Fig. 3: Convergence performance with different edge connectivity probabilities p_c on the meta-learning problem.

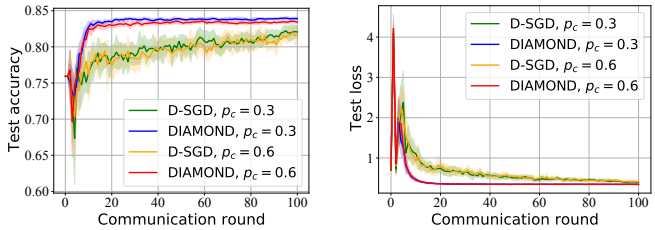


Fig. 4: Comparison between D-SGD and DIAMOND on the hyper-parameter optimization problem.

as: $\mathbf{x}_{i,t+1} = \sum_{j\in\mathcal{N}_i}[\mathbf{M}]_{ij}\mathbf{x}_{j,t} - \alpha_t\mathbf{u}_{i,t}$, where $\mathbf{u}_{i,t} = \sum_{j\in\mathcal{N}_i}[\mathbf{M}]_{ij}\mathbf{u}_{j,t-1} + \hat{\nabla}f_i(\mathbf{x}_{i,t},\mathbf{y}_{i,t};\bar{\xi}_{i,t}) - \hat{\nabla}f_i(\mathbf{x}_{i,t-1},\mathbf{y}_{i,t-1};\bar{\xi}_{i,t-1})$.

- **Momentum Stochastic Gradient Descent (M-SGD):** This algorithm can be viewed as a simplified version of DIAMOND by neglecting gradient tracking. Specifically, we replace the update step (9) by $\mathbf{x}_{i,t+1} = \sum_{j\in\mathcal{N}_i}[\mathbf{M}]_{ij}\mathbf{x}_{j,t} - \alpha_t\mathbf{p}_{i,t}$.

We set the learning rates $\alpha_t = c_\alpha(\omega+t)^{-1/3}$ and $\beta_t = c_\beta\alpha_t$ following Theorem 1, where $c_\alpha = 10$, $\omega = 2$ and $c_\beta = 10$. The momentum coefficients are chosen as $\eta_{t+1} = c_\eta\alpha_t^2$ and $\gamma_{t+1} = c_\gamma\alpha_t^2$ for M-SGD and DIAMOND, respectively, where $c_\eta = c_\gamma = 0.1$. Figs. 1 and 2 illustrate that the DIAMOND algorithm outperforms other algorithms for solving decentralized bilevel optimization problems in terms of the convergence metric in both 9-agent and 15-agent networks, which implies lower sample and communication complexity.

1-b) Impact of connectivity probability: We evaluate the impact of edge connection probability p_c on the performance of DIAMOND with the nine-agent network. We choose p_c from $\{0.3, 0.5, 0.8\}$, and the parameters of learning rate and momentum coefficient are the same as those in the previous setting. As shown in Fig. 3, there is only a slight increase in convergence rate with a higher p_c-value, which reflects that the performance of DIAMOND is not sensitive to the edge connection probability, and the proposed DIAMOND algorithm can adapt to various network settings.

2) Hyper-parameter optimization. Next, we compare DIAMOND with D-SGD using the logistic regression problem [7], [43] with the same formulation as in (1), where $f_i(\mathbf{x},\mathbf{y}_i^*(\mathbf{x})) = \frac{1}{|\mathcal{D}_{\text{val},i}|}\sum_{(\mathbf{a}_j,\mathbf{c}_j)\in\mathcal{D}_{\text{val},i}} Q(\mathbf{a}_j^T\mathbf{y}_i^*,\mathbf{c}_j)$, $g_i(\mathbf{x},\mathbf{y}_i) = \frac{1}{|\mathcal{D}_{\text{tr},i}|}\sum_{(\mathbf{a}_j,\mathbf{c}_j)\in\mathcal{D}_{\text{tr},i}} Q(\mathbf{a}_j^T\mathbf{y}_i,\mathbf{c}_j) + \frac{1}{qp}\sum_{k=1}^{q}\sum_{r=1}^{p}\exp(\mathbf{x}_r)\mathbf{y}_{irk}^2$. Here, $\mathcal{D}_{\text{tr},i}$ and $\mathcal{D}_{\text{val},i}$ are the training and validation datasets for agent i, respectively, Q is the cross-entropy loss, q is the number of classes, and p is the number of features. We use the a9a dataset [44], where $q = 2$ and $p = 123$, and divide the a9a dataset into training, validation, and testing sets, which contain 40%, 40%, and 20% samples, respectively.

We compare the proposed DIAMOND algorithm with D-SGD in terms of test accuracy and test error, using five-agent communication networks. For both DIAMOND and D-SGD, the learning rates are set as $\alpha_t = c_\alpha(\omega+t)^{-1/3}$ and $\beta_t = c_\beta\alpha_t$, where $c_\alpha = 5$, $\omega = 2$, and $c_\beta = 1.5$. Moreover, the momentum related parameters in DIAMOND are chosen as $\eta_{t+1} = c_\eta\alpha_t^2$ and $\gamma_{t+1} = c_\gamma\alpha_t^2$, where $c_\eta = c_\gamma = 0.1$. As shown in Fig. 4, DIAMOND has a faster convergence rate than that of D-SGD over various networks with different edge connection probabilities. This validates the superiority of the proposed DIAMOND algorithm.

VII. Conclusion

In this paper, we proposed the DIAMOND algorithm for decentralized bilevel optimization with non-convex upper-level subproblems and strongly-convex lower-level subproblems. DIAMOND utilizes a network-consensus approach and adopts a single-loop algorithmic architecture along with momentum-based stochastic gradient estimations and gradient tracking techniques. This is contrast to existing related works that use a double-loop architecture, full gradient estimations, or large-batch gradients. We showed that DIAMOND achieves $\mathcal{O}\left(\epsilon^{-3/2}\right)$ in both sample and communication complexities to reach an ϵ-stationary point, outperforming existing works.

We also conducted numerical experiments using meta-learning and hyper-parameter optimization problems to verify our theoretical findings. Future directions include i) to consider Hessian approximation to further reduce computation costs of the Hessian matrix and ii) to extend the proposed algorithm to wireless networks with channel noise and fading.

REFERENCES

[1] M. Hong, H.-T. Wai, Z. Wang, and Z. Yang, "A two-timescale framework for bilevel optimization: Complexity analysis and application to actor-critic," *arXiv preprint arXiv:2007.05170*, 2020.

[2] H. Zhang, W. Chen, Z. Huang, M. Li, Y. Yang, W. Zhang, and J. Wang, "Bi-level actor-critic for multi-agent coordination," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, 2020, pp. 7325–7332.

[3] Y. Liu and R. Liu, "Boml: A modularized bilevel optimization library in python for meta learning," in *2021 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*. IEEE, 2021, pp. 1–2.

[4] A. Rajeswaran, C. Finn, S. M. Kakade, and S. Levine, "Meta-learning with implicit gradients," *Advances in neural information processing systems*, vol. 32, 2019.

[5] S. Dempe, *Foundations of bilevel programming*. Springer Science & Business Media, 2002.

[6] S. Ghadimi and M. Wang, "Approximation methods for bilevel programming," *arXiv preprint arXiv:1802.02246*, 2018.

[7] K. Ji, J. Yang, and Y. Liang, "Bilevel optimization: Convergence analysis and enhanced design," in *International Conference on Machine Learning*. PMLR, 2021, pp. 4882–4892.

[8] J. Yang, K. Ji, and Y. Liang, "Provably faster algorithms for bilevel optimization," *Advances in Neural Information Processing Systems*, vol. 34, pp. 13 670–13 682, 2021.

[9] T. Chen, Y. Sun, and W. Yin, "A single-timescale stochastic bilevel optimization method," *arXiv preprint arXiv:2102.04671*, 2021.

[10] Z. Guo and T. Yang, "Randomized stochastic variance-reduced methods for stochastic bilevel optimization," *arXiv preprint arXiv:2105.02266*, 2021.

[11] P. Khanduri, S. Zeng, M. Hong, H.-T. Wai, Z. Wang, and Z. Yang, "A near-optimal algorithm for stochastic bilevel optimization via double-momentum," *Advances in Neural Information Processing Systems*, vol. 34, pp. 30 271–30 283, 2021.

[12] X. Zhang, J. Liu, Z. Zhu, and E. S. Bentley, "Low sample and communication complexities in decentralized learning: A triple hybrid approach," in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 2021, pp. 1–10.

[13] Z. Liu, X. Zhang, P. Khanduri, S. Lu, and J. Liu, "Interact: Achieving low sample and communication complexities in decentralized bilevel learning over networks," *arXiv preprint arXiv:2207.13283*, 2022.

[14] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Transactions on Automatic Control*, vol. 54, no. 1, pp. 48–61, 2009.

[15] A. Nedic, A. Ozdaglar, and P. A. Parrilo, "Constrained consensus and optimization in multi-agent networks," *IEEE Transactions on Automatic Control*, vol. 55, no. 4, pp. 922–938, 2010.

[16] J. Bracken and J. T. McGill, "Mathematical programs with optimization problems in the constraints," *Operations Research*, vol. 21, no. 1, pp. 37–44, 1973.

[17] C. Shi, J. Lu, and G. Zhang, "An extended kuhn–tucker approach for linear bilevel programming," *Applied Mathematics and Computation*, vol. 162, no. 1, pp. 51–63, 2005.

[18] A. Mehra and J. Hamm, "Penalty method for inversion-free deep bilevel optimization," in *Asian Conference on Machine Learning*. PMLR, 2021, pp. 347–362.

[19] B. Colson, P. Marcotte, and G. Savard, "An overview of bilevel optimization," *Annals of operations research*, vol. 153, no. 1, pp. 235–256, 2007.

[20] G. Kunapuli, K. P. Bennett, J. Hu, and J.-S. Pang, "Classification model selection via bilevel programming," *Optimization Methods & Software*, vol. 23, no. 4, pp. 475–489, 2008.

[21] K. Ji and Y. Liang, "Lower bounds and accelerated algorithms for bilevel optimization," *arXiv preprint arXiv:2102.03926*, 2021.

[22] A. Shaban, C.-A. Cheng, N. Hatch, and B. Boots, "Truncated back-propagation for bilevel optimization," in *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 2019, pp. 1723–1732.

[23] S. Gould, B. Fernando, A. Cherian, P. Anderson, R. S. Cruz, and E. Guo, "On differentiating parameterized argmin and argmax problems with application to bi-level optimization," *arXiv preprint arXiv:1607.05447*, 2016.

[24] F. Pedregosa, "Hyperparameter optimization with approximate gradient," in *International conference on machine learning*. PMLR, 2016, pp. 737–746.

[25] J. N. Tsitsiklis, "Problems in decentralized decision making and computation." Massachusetts Inst of Tech Cambridge Lab for Information and Decision Systems, Tech. Rep., 1984.

[26] Z. Jiang, K. Mukherjee, and S. Sarkar, "On consensus-disagreement tradeoff in distributed optimization," in *2018 Annual American Control Conference (ACC)*. IEEE, 2018, pp. 571–576.

[27] D. S. Callaway and I. A. Hiskens, "Achieving controllability of electric loads," *Proceedings of the IEEE*, vol. 99, no. 1, pp. 184–199, 2010.

[28] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[29] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, "Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[30] X. Zhang, J. Liu, Z. Zhu, and E. S. Bentley, "Compressed distributed gradient descent: Communication-efficient consensus over networks," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 2019, pp. 2431–2439.

[31] H. Yu, R. Jin, and S. Yang, "On the linear speedup analysis of communication efficient momentum sgd for distributed non-convex optimization," in *International Conference on Machine Learning*. PMLR, 2019, pp. 7184–7193.

[32] S. Lu, X. Zhang, H. Sun, and M. Hong, "Gnsd: A gradient-tracking based nonconvex stochastic algorithm for decentralized optimization," in *2019 IEEE Data Science Workshop (DSW)*. IEEE, 2019, pp. 315–321.

[33] S. Pu and A. Nedić, "Distributed stochastic gradient tracking methods," *Mathematical Programming*, vol. 187, no. 1, pp. 409–457, 2021.

[34] R. Xin, U. Khan, and S. Kar, "A hybrid variance-reduced method for decentralized stochastic non-convex optimization," in *International Conference on Machine Learning*. PMLR, 2021, pp. 11 459–11 469.

[35] P. Khanduri, P. Sharma, H. Yang, M. Hong, J. Liu, K. Rajawat, and P. Varshney, "Stem: A stochastic two-sided momentum algorithm achieving near-optimal sample and communication complexities for federated learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 6050–6061, 2021.

[36] L. Franceschi, P. Frasconi, S. Salzo, R. Grazzi, and M. Pontil, "Bilevel programming for hyperparameter optimization and meta-learning," in *International Conference on Machine Learning*. PMLR, 2018, pp. 1568–1577.

[37] K. Zhang, Z. Yang, H. Liu, T. Zhang, and T. Basar, "Fully decentralized multi-agent reinforcement learning with networked agents," in *International Conference on Machine Learning*. PMLR, 2018, pp. 5872–5881.

[38] A. Cutkosky and F. Orabona, "Momentum-based variance reduction in non-convex sgd," *Advances in neural information processing systems*, vol. 32, 2019.

[39] Q. Tran-Dinh, N. H. Pham, D. T. Phan, and L. M. Nguyen, "Hybrid stochastic gradient descent algorithms for stochastic nonconvex optimization," *arXiv preprint arXiv:1905.05920*, 2019.

[40] A. Hagberg, P. Swart, and D. S Chult, "Exploring network structure, dynamics, and function using networkx," Los Alamos National Lab (LANL), Los Alamos, NM (United States), Tech. Rep., 2008.

[41] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[42] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[43] R. Grazzi, L. Franceschi, M. Pontil, and S. Salzo, "On the iteration complexity of hypergradient computation," in *International Conference on Machine Learning*. PMLR, 2020, pp. 3748–3758.

[44] D. Dheeru and E. K. Taniskidou. (2017) UCI machine learning repository. [Online]. Available: https://archive.ics.uci.edu/ml