Decentralized Constrained Optimization: Double Averaging and Gradient Projection

Firooz Shahriari-Mehr, David Bosch and Ashkan Panahi

***Abstract*— In this paper, We consider the convex, finite-sum minimization problem with explicit convex constraints over strongly connected directed graphs. The constraint is an intersection of several convex sets each being known to only one node. To solve this problem, We propose a novel decentralized projected gradient scheme based on local averaging and prove its convergence using only local functions' smoothness. Experimental studies demonstrate the effectiveness of the proposed method in both constrained and unconstrained problems.**

I. Introduction

In the past decade, decentralized optimization techniques have attracted significant interest [1], [2]. In this setting, multiple computing nodes are involved, and there is no coordinator (central) node with which all nodes communicate. A fairly general framework for decentralized optimization problems is given by

$$\min_{\mathbf{x}\in\mathbb{R}^m} \quad f(\mathbf{x}) \triangleq \sum_{\nu=1}^{M} f_\nu(\mathbf{x}), \tag{1}$$

where M is the total number of the nodes in the network, $\mathbf{x} \in \mathbb{R}^m$ is called the global optimization variable, and $f(.)$ is the global objective function which has a finite-sum structure. Here, each node ν has access to its local function $f_\nu : \mathbb{R}^m \to \mathbb{R}$ and communicates to its neighbors $\mathcal{N}_\nu$ to achieve an *optimal consensus solution*. A natural extension to this setup is when $\mathbf{x}$ in problem (1) is required to lie in an intersection of several convex sets, i.e. $\mathbf{x} \in \bigcap_{\nu=1}^{N} S_\nu$, and each constraint S_ν is known only to one node. Applications of this setup, that we refer to as the decentralized constrained optimization problem (DCOP), are ubiquitous, *e.g.* smart grid control [3], [4], optimal energy management [5], sensor networks [6], and support vector machines [7]. However, practical approaches to solving it have not been extensively discussed in the literature. This paper responds to this shortcoming by providing a numerical method to solving DCOP with guaranteed convergence properties.

Node communication is a crucial factor in the design of decentralized optimization techniques, which is represented by either a directed or undirected communication graph. Earlier studies on decentralized techniques considered static undirected graphs, which indicate that each communication link between two nodes is time invariant and bi-directional, meaning that both nodes can send and receive information. This assumption is not compatible with many practical applications, such as broadcast channels with no return link, or communication failures leading to uni-directional links [8]. These problems intrigue researchers to propose decentralized methods considering directed graphs as the underlying communication network, where each communication link in the network is uni-directional. For simplicity, decentralized methods in which a directed graph represents their node communication are called *decentralized directed methods*, throughout this paper. In the same way, we refer to *decentralized undirected methods*. We address the more general case of decentralized directed scenarios, while the undirected cases follow as a special case.

For undirected communication graphs, efficient optimization techniques with provable convergence properties exist based on suitable iterative averaging over neighbor nodes [9]–[11]. The averaging procedure is mathematically represented by the so-called gossip matrices, which are compatible with the network structure, double stochastic, and symmetric [11], [12]. The required gossip matrices can be constructed using Laplacian or Metropolis matrices for undirected graphs. Such gossip matrices are not compatible with directed graphs, as they require asymmetry and finding doubly stochastic matrices is not straightforward, often requiring distributed and iterative numerical procedures such as iterative weight balancing [13]. For this reason, practical schemes utilize row stochastic or column stochastic matrices, instead of using doubly stochastic matrices. In this case, convergence bounds comparable to the undirected scenarios, even in the absence of constraints, are lacking to the best of our knowledge. We further address this issue by proposing a novel double-averaging scheme, similar to the so-called push-pull approach [14], which takes both row and column stochastic matrices into account, and at the same time enjoys superior convergence guarantees.

A. Contributions

The main contributions of the paper are summarized as follows:

- We propose a novel algorithm, called DAGP, to solve the problem of decentralized constraint optimization. Our scheme employs double averaging and projection onto convex sets. It extends the tracking approach, which has been proposed for the first time in [11], [12], to constrained problems and can benefit a fixed step size and fast convergence.
- In contrast to the previously proposed methods in the literature, our method simultaneously considers a directed

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. All the authors are with the CSE department at Chalmers University of Technology, Göteborg, Sweden. {e-mails: Firooz, Davidbos, Ashkan.Panahi@Chalmers.se}

communication graph and individual constraints at each node.

- We show that our technique is applicable to generic constrained convex problems, lacking strong convexity, while maintaining the convergence rate of order $\mathcal{O}(1/\sqrt{n})$, under mild conditions. We are not aware of any decentralized unconstrained method over directed communication graphs with similar established convergence properties.
- We present experiments for constrained decentralized optimization problems on directed graphs, where DAGP outperforms the existing algorithms. We also conduct experiments on unconstrained problems, where DAGP performs similarly to the state of the art, decentralized optimization algorithms.

B. Literature Review

In this section, we review the decentralized methods in the existing literature. We organize our review into three parts: techniques on directed and undirected graphs, and decentralized constrained methods. Several classes of methods exist in the literature that are not in the scope of this paper, *e.g.*, methods considering time-varying graphs [12], [15], local functions with a finite-sum structure [16]–[19], or compressed communication [20], [21].

1) Decentralized optimization over undirected graphs: The algorithms for undirected communication graphs can be divided into several categories. First, the decentralized gradient decent methods including [10], [22] use diminishing step size for convergence to the exact solution of the problem. The diminishing step size leads to practical difficulties with step tuning but establish the convergence rate of $\mathcal{O}(\log n/\sqrt{n})$ in a convex and smooth setting and $\mathcal{O}(\log n/n)$ in a strongly convex and smooth setting. The second category refers to the methods that use gradient tracking technique and leverage the gradient information at all nodes to estimate the gradient of the global function [11], [12], [23]. These methods use fixed step sizes and achieve linear convergence rate, i.e. $\mathcal{O}(\mu^n)$ $\mu < 1$, in a strongly convex and smooth setting. [23] has also shown a sublinear rate of convergence, i.e. $\mathcal{O}(1/n)$, when the functions are not strongly convex. The dual-based methods [24]–[26] are included in the third group. Although these methods are optimal and have linear convergence rate, they need to compute some computationally costly oracles, *e.g.* the gradient of a conjugate function, which is not practical in some applications.

2) Decentralized optimization over directed graphs: Earlier methods for directed problems apply the so-called push-sum protocol [27] to decentralized gradient descent methods to tackle the problem of computing a doubly stochastic gossip matrix for directed graphs [15], [28]. These methods utilize a column stochastic matrix, but a diminishing step size is still vital for their convergence. The methods based on the push-sum protocol converge with order of $\mathcal{O}(\log n/n)$ for smooth and strongly convex functions. To achieve fixed step size, [12], [29] have put push-sum protocol and gradient tracking technique together and have respectively proposed the DEXTRA and Push-DIGing algorithms. These algorithms achieved a linear rate of convergence in a smooth and strongly convex setting. DEXTRA suffers theoretical limitations on the step size, namely a feasible step size might not exist in some cases. [29] has proposed the ADD-OPT algorithm to solve this problem. This algorithm also enjoys linear convergence for strongly convex functions. Recently, methods based on two gossip matrices, one column stochastic and the other one row stochastic, have been proposed in the literature [14], [30], called Push-Pull methods. These methods also have linear convergence in a smooth and strongly convex setting. Our algorithm is similar to push-pull methods as it can be applied with similar underlying matrices.

3) Decentralized Constrained optimization: Despite extensive studies on decentralized optimization, there exist few papers that constraints explicitly. It is worth mentioning that a straightforward approach to solving constrained problems is to add the indicator functions of the constraint sets to the problem, then apply the methods proposed for the unconstrained problem. This approach requires methods that are applicable to non-smooth and non-strongly convex functions with unbounded (sub)gradients due to indicator function characteristics. For this reason, we note that utilizing the previously mentioned methods does not guarantee convergence. [31] is among the first papers incorporating the projection and averaging approaches, but it assumes that the constraint set is identical at all nodes. This leads to a problem when the projection onto the constraint set is not computationally efficient. In response, the projected subgradient algorithm has been proposed, which assumes that the constraint set is different and distributed among all nodes [32]. This paper is similar in setup to ours, but does not provide a precise convergence rate. Moreover, convergence of local variables to a consensus stopping point is proven only in two special cases: when the constraints are identical, or when the graph is fully connected. As the constraint at each node might be an intersection of several constraints, or in some applications, the nodes do not have access to all of their local constraints at each iteration, [33] has proposed a randomized projection scheme. This algorithm suffers from the same limitations as [32], i.e. the proof is only reliable for fully connected networks or a setting with identical constraints at each node. All the above-mentioned methods use a diminishing step size as they do not leverage any gradient tracking technique. Moreover, they assume that the underlying communication graph is undirected. [34] has proposed the DDPS algorithm, which is applicable when the communication graph is directed. However, this algorithm uses diminishing step size as well, and its convergence rate is of order $\mathcal{O}(\log n/\sqrt{n})$. Moreover, it is subject to the restrictive assumption that the constraints are identical. There are also methods with different problem description, such as composite constrained optimization [35]. These methods consider undirected communication graphs, and they are different from our problem, in nature.

C. Paper Outline

The rest of the paper is organized as follows. In the following, some preliminary definitions and notations are introduced. The DAGP algorithm is proposed in section II, along with theoretical convergence analysis. Due to space limitations, the proofs of all Lemmas and Theorems are provided in [36]. Finally, section III is devoted to the numerical studies.

Definition 1 (Normal cone and Projection Operator)**.** For a closed convex set $S \subset \mathbb{R}^n$, the normal cone of S is given by

$$\partial I_S(\mathbf{x}) = \begin{cases} \emptyset & \mathbf{x} \notin S \\ \left\{\mathbf{g} \in \mathbb{R}^n |\ \forall \mathbf{z} \in S,\ \mathbf{g}^T(\mathbf{z}-\mathbf{x}) \leq 0\right\} & \mathbf{x} \in S \end{cases}.$$

Moreover, the projection of a vector $\mathbf{x} \in \mathbb{R}^n$ onto S is computed by

$$P_S(\mathbf{x}) = \arg\min_{\mathbf{y} \in S} \|\mathbf{y}-\mathbf{x}\|_2^2.$$

Definition 2 (Graph Theory)**.** A directed graph is shown by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \ldots, N\}$ is a set of all nodes, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is a set of ordered pairs of distinct nodes, called edges. A directed path between two distinct nodes $u, v \in \mathcal{V}$ is a sequence of nodes $(u = v_0, v_1, \ldots, v_k = v)$ such that each pair (v_i, v_{i+1}) is an edge in $\mathcal{E}$. A graph $\mathcal{G}$ is strongly connected if for any two distinct nodes $u, v \in \mathcal{V}$, there exist a directed path between u and v. The adjacency matrix denoted by $\mathbf{A} = [a_{ij}]$ is an asymmetric matrix, where a_{ij} is $+1$ if $(i,j) \in \mathcal{E}$, and 0 otherwise.

In this paper, each pair shows a communication link between two distinct nodes, and (i,j) is a pair in $\mathcal{E}$, if there is a link from node j to node i. With this intuition, the ith row of an adjacency matrix shows from which nodes it can receive information, and constitute the incoming neighbors of node i called $\mathcal{N}_i^{\text{in}} = \{j|(i,j) \in \mathcal{E}\}$. On the other hand, the ith column of an adjacency matrix shows to which nodes it can send information, and constitute the outgoing neighbors of node i called $\mathcal{N}_i^{\text{out}} = \{j|(j,i) \in \mathcal{E}\}$. The in-degree and out-degree of a node i are defined as the cardinality of $\mathcal{N}_i^{\text{in}}$ and $\mathcal{N}_i^{\text{out}}$, respectively. Consequently, two Laplacian matrices can be defined as

$$\mathbf{L}^{\text{in}} = \mathbf{D}^{\text{in}} - \mathbf{A}, \qquad \mathbf{L}^{\text{out}} = \mathbf{D}^{\text{out}} - \mathbf{A},$$

where $\mathbf{D}^{\text{in}}$ is the in-degree diagonal matrix; that is $d_{ii}^{\text{in}} = \left|\mathcal{N}_i^{\text{in}}\right|$, and $\mathbf{D}^{\text{out}}$ is defined in a similar way. $\mathbf{L}^{\text{in}}$ and $\mathbf{L}^{\text{out}}$ have zero row-sum and column-sum characteristics and their scaled versions are used in this paper.

D. Mathematical Notation

In this paper, bold lowercase and uppercase letters are used to respectively represent vectors and matrices. $w_{\nu u}$ denotes the element at the ν^{th} row and the u^{th} column of the matrix $\mathbf{W}$. $\mathbf{W}^T$ shows the transpose of $\mathbf{W}$, and $\ker(\mathbf{W})$ is its right null space, meaning that $\mathbf{x} \in \ker(\mathbf{W})$, if and only if $\mathbf{W}\mathbf{x} = 0$. $\mathbf{1}_n$ and $\mathbf{0}_n$ respectively denote the $n-$dimensional vectors of all ones and zeros. The index n may be dropped if there is no risk of confusion. Furthermore, $\mathbf{O}$ denotes a matrix with all zero elements. The Euclidean inner product of vectors is denoted by $\langle ., . \rangle$. $\delta_{n,0}$ is the Kronecker delta function. In this paper, subscript generally defines the iteration number, and superscript defines the node number, *e.g.*, $\nabla f_\nu(\mathbf{x}_n^\nu)$ indicates the gradient of the node ν's local function at its local variable at iteration n.

II. Problem setting and Proposed Algorithm

In this section, we propose a new algorithm to solve decentralized constrained optimization problem, considering directed graphs as a communication network between the nodes. The proposed algorithm is called *DAGP* due to Double Averaging and Gradient Projection approaches used in the iterative equations, which are introduced in the subsection II-B.

A. Problem Formulation

Decentralized Constraint Optimization Problem (DCOP) is formulated as

$$\min_{\mathbf{x} \in \mathbb{R}^m} \quad f(\mathbf{x}) \triangleq \sum_{\nu=1}^{M} f_\nu(\mathbf{x}) \quad \text{s.t.} \quad \mathbf{x} \in \bigcap_{\nu=1}^{M} S_\nu, \tag{2}$$

where S_ν is a closed convex set, and the intersection of all these constraint sets is called the feasible set. Note that without loss of generality, the number of constraints is equal to the number of functions. This allows us to assume M nodes in our setting, each having access to one function and one constraint. Note that a different number of constraints can still be considered in our setup, as each node may further access a composite constraint (i.e. an intersection of simpler constraints), or multiple nodes may share identical constraints (i.e. $S_\nu = S_u$) or have trivial constraints $S_\nu = \mathbb{R}^m$. Nevertheless, we merely require the projection operator to the constraint set S_ν of each node to be available, and neglect further possible structures in them.

In decentralized optimization, each node stores and updates a local variable $\mathbf{x}^\nu$ as its solution. The nodes should achieve consensus, i.e. $\mathbf{x}^\nu$s must converge to an equal stopping point $\mathbf{x}^*$, which is further required to be a feasible, optimal solution of (2).

B. DAGP *Algorithm*

The DAGP algorithm does the following updates in each iteration, $\forall \nu \in \mathcal{V}$.

$$\mathbf{z}^\nu = \mathbf{x}_n^\nu - \sum_{u \in \mathcal{N}_\nu^{\text{in}}} w_{\nu u} \mathbf{x}_n^u - \mu \left(\nabla f_\nu(\mathbf{x}_n^\nu) - \mathbf{g}_n^\nu\right) \tag{3}$$

$$\mathbf{x}_{n+1}^\nu = P_{S_\nu}\left(\mathbf{z}^\nu\right) \tag{4}$$

$$\mathbf{g}_{n+1}^\nu = \mathbf{g}_n^\nu + \rho \left[\nabla f_\nu(\mathbf{x}_n^\nu) - \mathbf{g}_n^\nu + \frac{1}{\mu}\left(\mathbf{z}^\nu - \mathbf{x}_{n+1}^\nu\right)\right] + \alpha\left(\mathbf{h}_n^\nu - \mathbf{g}_n^\nu\right) \tag{5}$$

$$\mathbf{h}_{n+1}^\nu = \mathbf{h}_n^\nu - \sum_{u \in \mathcal{N}_\nu^{\text{in}}} q_{\nu u}(\mathbf{h}_n^u - \mathbf{g}_n^u) \tag{6}$$

To interpret the algorithm, consider the above update equations. Take into account that node ν has access to $\left(\mathbf{x}_n^u, \mathbf{h}_n^u - \mathbf{g}_n^u, \forall u \in \mathcal{N}_\nu^{\text{in}}\right)$ as each node u broadcasts this pair

of messages to its out-neighbors. First weighted averaging happens in (3), where each node computes a weighted average of its local variable and its in-neighbors' local variables. This averaging is a basis for achieving consensus, and $\mathbf{W} = [w_{\nu u}]$ must have zero row-sum structure to achieve this objective. Then, the resulting averaged vector is aligned with the negative of the augmented local descent direction $(\nabla f_\nu(\mathbf{x}_n^\nu) - \mathbf{g}_n^\nu)$ scaled by a fixed step size μ. The resulting solution $\mathbf{z}^\nu$ is projected onto the local constraint set in (4). Therefore from the second iteration, the local variables at each iteration lie in their own local constraint set, but not necessarily in the feasible set of the problem in (2). The novelty of this paper is the definition of additional variables $\mathbf{h}_n^\nu$ together with $\mathbf{g}_n^\nu$ variables to push the algorithm towards an optimal consensus solution.

Vectors $\mathbf{g}^\nu$ act as the memory of the algorithm, which preserve and track the previous information of local functions' gradient and *feasible directions*, i.e. ∇f_ν and $\mathbf{z}^\nu - \mathbf{x}^\nu$, respectively. To reach an optimal solution, the gradient and the feasible directions of all nodes must be aggregated. This is achieved by adding the term $\alpha(\mathbf{h}^\nu - \mathbf{g}^\nu)$ to (5), where $\mathbf{h}^\nu$ is updated using (6). $\mathbf{h}^\nu$ propagates the information of the gradient and the feasible directions of other nodes through the second weighted averaging by $\mathbf{Q} = [q_{\nu u}]$. In (6), we further require that $\mathbf{Q}$ is a matrix with zero column-sum structure. Then, $\sum_{\nu\in\mathcal{V}} \mathbf{h}^\nu$ will not change over time. We also require that $\mathbf{h}_0^\nu$s are initialized in a way that they satisfy $\sum_{\nu\in\mathcal{V}} \mathbf{h}^\nu = \mathbf{0}$. The easiest way is to initialize them with zero vectors. In this way, when $\mathbf{g}^\nu$ converges to $\mathbf{h}^\nu$ its sum will also be zero. This in turn, leads to satisfying the optimality condition of the problem as $\mathbf{g}^\nu$s contain gradients and feasible directions of the problem. This is further elaborated in Theorem 1.

C. Convergence Analysis

In this section, we discuss the convergence properties of DAGP. We present two results. First in Theorem 1, we prove that if the iterates of DAGP converge, any stopping point is an optimal and consensus solution of the problem in (2). Then, in Theorem 2, we prove the convergence rate of our proposed algorithm in a smooth and convex setting.

1) Assumptions: We proceed by formalizing the adopted assumptions as follows.

Assumption 1. The nodes will communicate across a strongly connected directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. This assumption guarantees the sufficient information flow between the nodes as there exists a directed path between every two nodes in the graph. As a result, the nodes can achieve consensus.

Assumption 2. The optimization problem (2) is feasible and attains a finite optimal value $f^* = f(x^*)$ at an optimal feasible solution $\mathbf{x}^*$ satsfying the optimality condition:

$$\mathbf{0} \in \sum_{\nu=1}^{M} \partial I_{S_\nu}(\mathbf{x}^*) + \nabla f_\nu(\mathbf{x}^*).$$

Assumption 3. There are two weight matrices $\mathbf{W}$ and $\mathbf{Q}$ with a similar sparsity pattern to the adjacency matrix $\mathbf{A}$ of $\mathcal{G}$. They further satisfy the zero row-sum and zero column-sum structure, respectively. The first one is required for achieving consensus, and the second one is required for proving optimality of the solution. Moreover, we assume that $\ker(\mathbf{Q}) = \ker(\mathbf{W}^T)$ and $\ker(\mathbf{W}) = \operatorname{span}\{\mathbf{1}\}$.

Assumption 4. The functions $f_\nu(.)$ are convex, differentiable and L-smooth.

Now, we define the matrices $\mathbf{R}$ and $\mathbf{P}$ as:

$$\mathbf{R} = \begin{bmatrix} \mathbf{O} & \mathbf{O} & \mathbf{O} & \mathbf{O} \\ \mathbf{I} & \mathbf{O} & \mathbf{O} & \mathbf{O} \\ -\frac{\rho}{\mu}\mathbf{I} & \frac{\rho}{\mu}(\mathbf{I}-\mathbf{W}) & \mathbf{I} & \alpha\mathbf{I} \\ \frac{\rho}{\mu}\mathbf{I} & -\frac{\rho}{\mu}(\mathbf{I}-\mathbf{W}) & \mathbf{O} & (1-\alpha)\mathbf{I}-\mathbf{Q} \end{bmatrix} \tag{7}$$

$$\mathbf{P} = \begin{bmatrix} \mathbf{I} & \mathbf{O} & \mathbf{O} & \mathbf{O} \end{bmatrix}^T. \tag{8}$$

Moreover, for an arbitray positive value of η, we take $\mathbf{S} = [\mathbf{S}_1\ \mathbf{S}_2]$, where

$$\mathbf{S}_1 = \begin{bmatrix} \left(1-\frac{L\mu}{2}\right)\mathbf{I} - M\eta\left(\mathbf{I}-\frac{1}{M}\mathbf{1}\mathbf{1}^T\right) \\ -\frac{1}{2}(\mathbf{I}-\mathbf{W}^T) + \frac{L\mu}{2}\mathbf{I} \\ -\frac{\mu}{2}\mathbf{I} \\ \mathbf{O} \end{bmatrix}$$
$$\mathbf{S}_2 = \begin{bmatrix} -\frac{1}{2}(\mathbf{I}-\mathbf{W}) + \frac{L\mu}{2}\mathbf{I} & -\frac{\mu}{2}\mathbf{I} & \mathbf{O} \\ -\frac{L\mu}{2}\mathbf{I} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & \mathbf{O} \end{bmatrix} \tag{9}$$

Assumption 5. There exist strictly positive constants C, η such that the following limit exists for all $\beta > 0$

$$\lim_{z\to 0} \begin{bmatrix} \mathbf{I} & \mathbf{O} & \mathbf{O} \end{bmatrix} \mathbf{F}^{-1} \begin{bmatrix} -C\mathbf{I} \\ \mathbf{I} \\ \mathbf{O} \end{bmatrix}, \tag{10}$$

where $\mathbf{F}$ is defined as

$$\mathbf{F}(z,\beta) = \begin{bmatrix} \mathbf{S} & z^{-1}\mathbf{I} - \mathbf{R}^T & \mathbf{O} \\ z\mathbf{I} - \mathbf{R} & \mathbf{O} & -\mathbf{P} \\ \mathbf{O} & -\mathbf{P}^T & \beta\mathbf{I} \end{bmatrix}. \tag{11}$$

2) Main Results: Here, we present the main theoretical results and postpone more details to the appendix and [36].

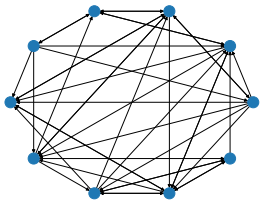
Theorem 1. Let Assumption 3 and 4 hold. If the iterates of DAGP algorithm converge, any stopping point is an optimal and consensus solution of the decentralized constrained optimization problem in (2), i.e. $\mathbf{x}^\nu = \mathbf{x}^*,\ \forall\nu \in \mathcal{V}$, and $\mathbf{x}^*$ satisfies the sufficient optimality conditions.

We also present guarantees for the rate of convergence.

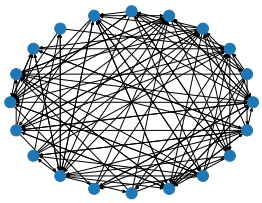
Theorem 2. Let all the assumptions hold. Define $\bar{\mathbf{x}}_N^v = \frac{1}{N}\sum_{n=0}^{N-1} \mathbf{x}_N^v$ and $\bar{\mathbf{x}}_N = \frac{1}{M}\sum_v \bar{\mathbf{x}}_N^v$. Then, $\|\bar{\mathbf{x}}_N - \bar{\mathbf{x}}_N^u\|^2 = O(\frac{1}{N})$, $\operatorname{dist}^2(\bar{\mathbf{x}}_N, S_u) = O(\frac{1}{N})$ and

$$\left|\sum_v f_v(\bar{\mathbf{x}}_N^v) - \sum_v f_v(\mathbf{x}^*)\right| = O\left(\frac{1}{\sqrt{N}}\right). \tag{12}$$

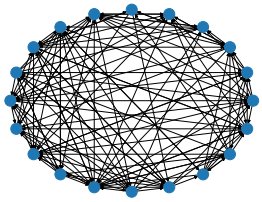
The proof for Theorem 1 is given in the Appendix, while



(a) First setup

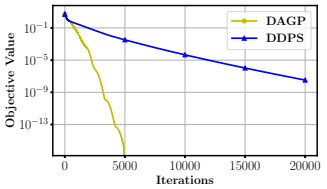


(b) Second setup

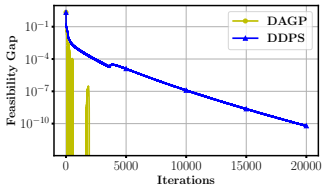


(c) Logistic Regression

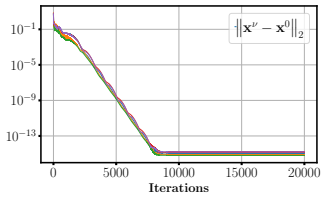
Fig. 1. Directed random graphs used in our experiments.



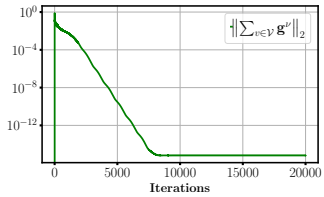
(a) Objective Value



(b) Feasibility Gap



(c) Consensus solution



(d) Optimal solution

Fig. 2. First setup results with $m = 20$ and $M = 10$. Local variables move to a consensus and optimal stopping point in DAGP, while they move to a sub-optimal point in DDPS.

the proof for Theorem 2 is provided in [36].

III. Experimental Results

We evaluate and compare the performance of the DAGP algorithm in two scenarios; decentralized constraint and unconstrained problems. In the first experiment, which contains examples with synthetic data, we consider constraints and examine the convergence and feasibility gap of the DAGP compared to the DDPS algorithm. In the second experiment, we solve the classical logistic regression problem, which is unconstrained. In the latter, we compare our algorithm to the state of the art decentralized unconstrained optimization algorithms over directed graphs, namely the ADD-OPT and Push-Pull algorithms.

In all algorithms used in the first experiment, the parameters are hand-tuned in a way that the algorithms achieve their best performance, leading to a fair comparison. In the real-world logistic regression problem, hand-tuning parameters is not computationally feasible due to the size of the experiment. Therefore, an appropriate step size is selected for all algorithms in the second experiment. Different algorithms use different matrices for averaging. In this paper, we respectively use $\mathbf{L}^{\text{in}}/2\mathbf{d}^{\text{in}}_{max}$ and $\mathbf{L}^{\text{out}}/2\mathbf{d}^{\text{out}}_{max}$ as zero row-sum and zero column-sum matrices, where $\mathbf{d}^{\text{in}}_{max}$ and $\mathbf{d}^{\text{out}}_{max}$ are the largest diagonal elements of $\mathbf{L}^{\text{in}}$ and $\mathbf{L}^{\text{out}}$. By subtracting these matrices from the identity matrix, row stochastic and column stochastic matrices used in this paper are computed. Moreover, random directed and strongly connected graphs are used in our experiments, which are shown in Fig 1. The numerical experiments are described next. We repeated each experiment multiple times, but only one instance from each experiment is presented as the difference between individual runs was minimal.

A. Numerical Results

In this experiment, which contains two setups, we consider synthetic functions and constraints. In our setups, there are M nodes, each having access to one function and one constraint. The nodes communicate over a randomly generated graph and their local variables $\mathbf{x}^\nu$s are of size m, which their initial vectors are generated randomly from zero mean and unit variance normal distribution. The functions are selected to be smooth, but not strongly convex as follows:

$$f_\nu(\mathbf{x}) = \log\big(\cosh(\mathbf{a}_\nu^T\mathbf{x} - b_\nu)\big), \tag{13}$$

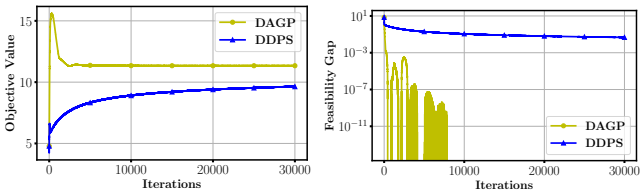
where $\mathbf{a}_\nu$s and b_νs are randomly generated from a zero mean and unit variance normal distribution. Moreover, we choose randomly generated linear constraints $\mathbf{c}_\nu^T\mathbf{x} - d_\nu \leq 0$ since their orthogonal projection operator is simple to compute.[1]

In the first setup, $m = 20$ and $M = 10$, while in the second one, $m = 10$ and $M = 20$. These parameters are chosen since the feasible set in the second experiment is significantly smaller than the feasible set in the first experiment. In both setups, the objective value and the distance to the feasible set, called feasibility gap, are reported, both computed at $\bar{\mathbf{x}} = \sum_{\nu\in\mathcal{V}} \mathbf{x}^\nu$.

The results of these setups are shown respectively in Figures 2 and 3. We observe that $\bar{\mathbf{x}}$ moves completely to the feasible set in our algorithm unlike DDPS in which $\bar{\mathbf{x}}$ becomes only close to the feasible set. Moreover, our algorithm converges faster to the optimal consensus solution in comparison with DDPS algorithm since DDPS needs a diminishing step size. To show that all nodes achieve consensus, the squared norm of $\mathbf{x}_\nu$ at five random nodes is plotted in Fig 2c, where all nodes converge to one stopping point. As described in section II-C, $\sum_{\nu\in\mathcal{V}} \mathbf{g}^\nu$ should become equal to zero to have optimal solution. For this reason, the norm of this variable is computed and shown in Fig 2d, which approaches zero as the algorithm proceeds.

In the first setup, $\mathbf{C} = [\mathbf{c}_i^T] \in \mathbb{R}^{M\times m}$ has a null space, and

[1] In all simulations, $\mathbf{c}_\nu$ and d_ν are selected such that their intersection not being an empty set, *e.g.*, $\mathbf{c}_\nu$s are generated randomly, then d_νs are selected such that $\mathbf{c}_\nu^T\mathbf{x} \leq d_\nu$ for one arbitrary vector $\mathbf{x}$.



(a) Objective Value (b) Feasibility Gap

Fig. 3. Second setup results with $m = 10$ and $M = 20$. As DDPS has not converged to a point in the feasible set, it can achieve less function value.

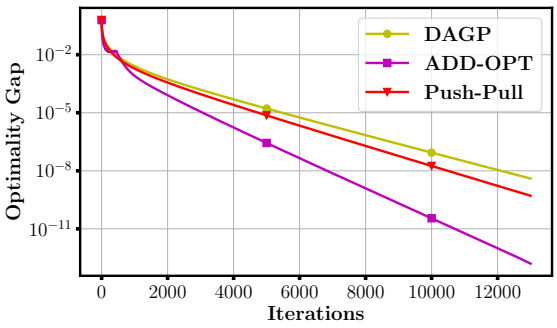


Fig. 4. Convergence rate comparison of decentralized unconstrained algorithms over directed graphs. A fixed step size is used in all algorithms.

the feasible set of its corresponding optimization problem is larger than the feasible set of the second experiment. The linear convergence rate of DAGP in Fig 2a, can be due to a situation where the constraints are not active at the solution, and the algorithm attains the optimal value of the unconstrained version of the problem. Then, the overall unconstrained objective function may be strongly convex, explaining faster convergence. On the other hand, in the second experiment some constraints are active, and the algorithm slows down in converging to a consensus and an optimal solution. In Fig 3a, DDPS achieves a smaller objective value as its local variables remain infeasible.

B. Logistic Regression

For the sake of comparison with other algorithms we examine an unconstrained logistic regression problem. We consider the MNIST [37] dataset restricted to two digits to form a binary classification problem. We once again consider a random directed graph with $M = 20$ nodes as shown in Fig 1c. Total number of $N_s = 10000$ images are used for training the model, i.e. for minimizing the logistic loss function defined as

$$\min_{\mathbf{w}\in\mathbb{R}^{784}} \sum_{i=1}^{N_s} \log\left(1+\exp\left(-y_i\mathbf{x}_i^T\mathbf{w}\right)\right) + \frac{\lambda}{2}\|\mathbf{w}\|_2, \quad (14)$$

where $\{\mathbf{x}_i, y_i\}_{i=1}^{N_s} \subseteq \mathbb{R}^{784} \times \{+1, -1\}$ is the set of training samples, and λ is the regularization parameter, which is chosen to be $1/N_s$. We assume that the training samples are distributed in a balanced way between 20 nodes; as a result, each node has 500 training samples. The loss function at each node f_ν is the collection of terms in (14) associated with the samples of the node ν. The regularization term ensures that this problem is strongly convex, leading to linear rate of convergence for all algorithms.

We compare DAGP with Push-Pull and ADD-OPT. For all algorithms, fixed, similar, and appropriate step size is used. centralized gradient descent is used to determine the optimal value f^* and compare all algorithms objective values with respect to it. The results for optimality gap, defined as $\sum_{\nu\in\mathcal{V}} f_\nu(\bar{\mathbf{x}}) - f^*$, are shown in Fig 4.

As observed, the difference between the convergence rate of these three graphs is minimal. As discussed earlier, optimal step sizes are not used in this experiment due to the size of the problem. Nevertheless, for smaller experiments, we observe that DAGP and Push-Pull can utilize larger step sizes, while ADD-OPT fails to converge with a similar step size. We conclude that by choosing the optimal values of the step size, DAGP and Push-Pull behave similarly, and both outperform ADD-OPT. The practical applicability of our algorithm is immediate, as it is competitive with respect to constrained problems, but also provably capable of solving unconstrained optimization problems, without any modifications to the algorithmic structure.

IV. Conclusion

We introduce the Double Averaging Gradient Projection (DAGP) algorithm designed for solving decentralized optimization problems over directed graphs for problems with and without constraints. In contrast to the existing literature, DAGP allows for different constraints at each node in a directed graph. Previous algorithms such as DDPS [34], require a decreasing step size to solve similar constrained problems, while DAGP requires constant step size by employing a gradient tracking technique. By taking advantage of the projection operator onto convex sets and double averaging, we prove a $\mathcal{O}\left(1/\sqrt{n}\right)$ rate of convergence on constrained problems in a convex and smooth setting. Comparing to the previously proposed algorithms, DAGP is competitive with other decentralized unconstrained directed optimization algorithms, such as ADP-OPT [29] and Push-Pull [38], and greatly outcompetes the previous decentralized constrained optimization algorithms such as DDPS.

References

[1] A. Nedic, "Distributed Gradient Methods for Convex Machine Learning Problems in Networks: Distributed Optimization," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 92–101, 2020.

[2] R. Xin, S. Pu, A. Nedić, and U. A. Khan, "A general framework for decentralized optimization with first-order methods," *Proceedings of the IEEE*, vol. 108, no. 11, pp. 1869–1889, 2020.

[3] M. Alizadeh, X. Li, Z. Wang, A. Scaglione, and R. Melton, "Demand-side management in the smart grid: Information processing for the power switch," *IEEE Signal Processing Magazine*, vol. 29, no. 5, pp. 55–67, 2012.

[4] X. Guan, Z. Xu, and Q.-S. Jia, "Energy-efficient buildings facilitated by microgrid," *IEEE Transactions on smart grid*, vol. 1, no. 3, pp. 243–252, 2010.

[5] H. D. H. Nguyen, T. Narikiyo, and M. Kawanishi, "A distributed optimization method for optimal energy management in smart grid," in *Research Trends and Challenges in Smart Grids*. IntechOpen, 2019.

[6] J. A. Bazerque and G. B. Giannakis, "Distributed spectrum sensing for cognitive radio networks by exploiting sparsity," *IEEE Transactions on Signal Processing*, vol. 58, no. 3, pp. 1847–1862, 2009.

[7] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.
[8] R. Xin and U. A. Khan, "A Linear Algorithm for Optimization over Directed Graphs with Geometric Convergence," *IEEE Control Systems Letters*, vol. 2, no. 3, pp. 313–318, 2018.
[9] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE transactions on information theory*, vol. 52, no. 6, pp. 2508–2530, 2006.
[10] 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.
[11] W. Shi, Q. Ling, G. Wu, and W. Yin, "Extra: An exact first-order algorithm for decentralized consensus optimization," *SIAM Journal on Optimization*, vol. 25, no. 2, pp. 944–966, 2015.
[12] A. Nedic, A. Olshevsky, and W. Shi, "Achieving geometric convergence for distributed optimization over time-varying graphs," *SIAM Journal on Optimization*, vol. 27, no. 4, pp. 2597–2633, 2017.
[13] B. Gharesifard and J. Cortés, "Distributed strategies for generating weight-balanced and doubly stochastic digraphs," *European Journal of Control*, vol. 18, no. 6, pp. 539–557, 2012.
[14] S. Pu, W. Shi, J. Xu, and A. Nedic, "Push-pull gradient methods for distributed optimization in networks," *IEEE Transactions on Automatic Control*, 2020.
[15] A. Nedić and A. Olshevsky, "Distributed optimization over time-varying directed graphs," *IEEE Transactions on Automatic Control*, vol. 60, no. 3, pp. 601–615, 2014.
[16] A. Mokhtari and A. Ribeiro, "DSA: Decentralized double stochastic averaging gradient algorithm," *Journal of Machine Learning Research*, vol. 17, pp. 1–35, 2016.
[17] H. Hendrikx, F. Bach, and L. Massoulié, "An accelerated decentralized stochastic proximal algorithm for finite sums," *arXiv preprint arXiv:1905.11394*, 2019.
[18] H. Hendrikx, F. Bach, and L. Massoulie, "Dual-free stochastic decentralized optimization with variance reduction," *arXiv*, no. NeurIPS, pp. 1–12, 2020.
[19] R. Xin, S. Kar, and U. A. Khan, "Decentralized stochastic optimization and machine learning: A unified variance-reduction framework for robust performance and fast convergence," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 102–113, 2020.
[20] A. Koloskova, S. U. Stich, and M. Jaggi, "Decentralized stochastic optimization and gossip algorithms with compressed communication," *36th International Conference on Machine Learning, ICML 2019*, vol. 2019-June, pp. 6088–6111, 2019.
[21] A. Beznosikov, S. Horváth, P. Richtárik, and M. Safaryan, "On biased compression for distributed learning," *arXiv preprint arXiv:2002.12410*, 2020.
[22] I. Lobel and A. Ozdaglar, "Distributed subgradient methods for convex optimization over random networks," *IEEE Transactions on Automatic Control*, vol. 56, no. 6, pp. 1291–1306, 2010.
[23] G. Qu and N. Li, "Harnessing smoothness to accelerate distributed optimization," *IEEE Transactions on Control of Network Systems*, vol. 5, no. 3, pp. 1245–1260, 2018.
[24] K. Seaman, F. Bach, S. Bubeck, Y. T. Lee, and L. Massoulie, "Optimal algorithms for smooth and strongly convex distributed optimization in networks," *34th International Conference on Machine Learning, ICML 2017*, vol. 6, pp. 4630–4642, 2017.
[25] C. A. Uribe, S. Lee, A. Gasnikov, and A. Nedić, "A dual approach for optimal algorithms in distributed optimization over networks," *Optimization Methods and Software*, pp. 1–40, 2020.
[26] H. Hendrikx, F. Bach, and L. Massoulie, "An optimal algorithm for decentralized finite sum optimization," *arXiv preprint arXiv:2005.10675*, 2020.
[27] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-based computation of aggregate information," in *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.* IEEE, 2003, pp. 482–491.
[28] K. I. Tsianos, S. Lawlor, and M. G. Rabbat, "Push-sum distributed dual averaging for convex optimization," in *2012 ieee 51st ieee conference on decision and control (cdc)*. IEEE, 2012, pp. 5453–5458.
[29] C. Xi, R. Xin, and U. A. Khan, "ADD-OPT: Accelerated Distributed Directed Optimization," *IEEE Transactions on Automatic Control*, vol. 63, no. 5, pp. 1329–1339, 2018.
[30] R. Xin and U. A. Khan, "A linear algorithm for optimization over directed graphs with geometric convergence," *IEEE Control Systems Letters*, vol. 2, no. 3, pp. 315–320, 2018.
[31] S. S. Ram, A. Nedić, and V. V. Veeravalli, "Distributed stochastic subgradient projection algorithms for convex optimization," *Journal of optimization theory and applications*, vol. 147, no. 3, pp. 516–545, 2010.
[32] 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.
[33] S. Lee and A. Nedic, "Distributed random projection algorithm for convex optimization," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 2, pp. 221–229, 2013.
[34] C. Xi and U. A. Khan, "Distributed subgradient projection algorithm over directed graphs," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3986–3992, 2016.
[35] Q. Lü, X. Liao, H. Li, and T. Huang, "A computation-efficient decentralized algorithm for composite constrained optimization," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 6, pp. 774–789, 2020.
[36] F. Shahriari-Mehr, "Decentralized constrained optimization: Double averaging and gradient projection, theorems' proof," https://github.com/Firooz-shahriari/DCOP, 2021.
[37] Y. LeCun and C. Cortes, "Mnist handwritten digit database," 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/
[38] S. Pu, W. Shi, J. Xu, and A. Nedic, "Push-Pull Gradient Methods for Distributed Optimization in Networks," *IEEE Transactions on Automatic Control*, vol. 66, no. 1, pp. 1–16, 2021.

APPENDIX

PROOF OF THEOREM 1

We start by presenting a lemma:

Lemma 1. Let $\ker(\mathbf{W}^T) = \ker(\mathbf{Q})$. Then, $\mathbf{QWx} = \mathbf{0}$ if and only if $\mathbf{x} \in \ker(\mathbf{W})$.

Now, consider an arbitrary stopping point of the algorithm, that is $\mathbf{x}_{n+1}^\nu = \mathbf{x}_n^\nu = \mathbf{x}^\nu$, $\nabla \mathbf{f}_{n+1} = \nabla \mathbf{f}_n = \nabla \mathbf{f}$, $\mathbf{h}_{n+1}^\nu = \mathbf{h}_n^\nu = \mathbf{h}^\nu$ and $\mathbf{g}_{n+1}^\nu = \mathbf{g}_n^\nu = \mathbf{g}^\nu$. We have

$$\mathbf{Z} = \mathbf{X} - \mathbf{WX} - \mu\left(\nabla \mathbf{f} - \mathbf{G}\right) \tag{15}$$

$$\mathbf{X} = \boldsymbol{\mathcal{P}}_S\left(\mathbf{Z}\right) \tag{16}$$

$$\rho\left[\nabla \mathbf{f} - \mathbf{G} + \frac{1}{\mu}\left(\mathbf{Z} - \mathbf{X}\right)\right] + \alpha\left(\mathbf{H} - \mathbf{G}\right) = \mathbf{O} \tag{17}$$

$$\mathbf{Q}\left(\mathbf{H} - \mathbf{G}\right) = \mathbf{O}. \tag{18}$$

Left multiplying (17) by $\mathbf{Q}$, considering (18), we have

$$\mathbf{Q}(\mathbf{G} - \nabla \mathbf{f}) = \frac{1}{\mu}\mathbf{Q}(\mathbf{Z} - \mathbf{X}). \tag{19}$$

Left multiplying (15) by $\mathbf{Q}$, and applying (19) leads to $\mathbf{QWX} = \mathbf{0}$. Therefore, $\mathbf{X} \in \ker(\mathbf{W}) = \mathrm{span}\{\mathbf{1}\}$ based on the result of Lemma 1, which means that $\mathbf{x}^\nu = \mathbf{x}^*$, $\forall \nu \in \mathcal{V}$. As $\mathbf{X} \in \ker(\mathbf{W})$, (15) reduces to $\mathbf{Z} - \mathbf{X} = \mu(\mathbf{G} - \nabla \mathbf{f})$, which leads to $\mathbf{H} = \mathbf{G}$ by incorporating it into (17). Since (6) is designed to preserve the summation of $\mathbf{h}^\nu$s, and each element of $\mathbf{H}$ is initialized with zero vector, we have

$$\mathbf{1}^T\mathbf{G} = \mathbf{1}^T\mathbf{H} = \sum_{\nu \in \mathcal{V}} (\mathbf{h}^\nu)^T = \mathbf{0}^T. \tag{20}$$

From (16), we have $\mathbf{Z} - \mathbf{X} \in \boldsymbol{\partial}\mathbf{I}_S$, consequently, $\mu(\mathbf{G} - \nabla \mathbf{f}) \in \boldsymbol{\partial}\mathbf{I}_S$. As $\boldsymbol{\partial}\mathbf{I}_{S_\nu}$ is a cone, and therefore invariant to scaling, we can write $(\mathbf{G} - \nabla \mathbf{f}) \in \boldsymbol{\partial}\mathbf{I}_S$. Left multiplying by $\mathbf{1}^T$, and moving all the terms to one side, considering (20), we have

$$\mathbf{0} \in \sum_{\nu \in \mathcal{V}} \partial I_{S_\nu}(\mathbf{x}^*) + \nabla f_\nu(\mathbf{x}^*). \tag{21}$$

which shows that $\mathbf{x}^*$ is optimal. ■