2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Distributed Optimization via Gradient Descent with Event-Triggered Zooming over Quantized Communication

Apostolos I. Rikos, Wei Jiang, Themistoklis Charalambous, and Karl H. Johansson

***Abstract*— In this paper, we study unconstrained distributed optimization strongly convex problems, in which the exchange of information in the network is captured by a directed graph topology over digital channels that have limited capacity (and hence information should be quantized). Distributed methods in which nodes use quantized communication yield a solution at the proximity of the optimal solution, hence reaching an error floor that depends on the quantization level used; the finer the quantization the lower the error floor. However, it is not possible to determine in advance the optimal quantization level that ensures specific performance guarantees (such as achieving an error floor below a predefined threshold). Choosing a very small quantization level that would guarantee the desired performance, requires information packets of very large size, which is not desirable (could increase the probability of packet losses, increase delays, etc) and often not feasible due to the limited capacity of the channels available. In order to obtain a communication-efficient distributed solution and a sufficiently close proximity to the optimal solution, we propose a quantized distributed optimization algorithm that converges in a finite number of steps and is able to adjust the quantization level accordingly. The proposed solution uses a finite-time distributed optimization protocol to find a solution to the problem for a given quantization level in a finite number of steps and keeps refining the quantization level until the difference in the solution between two successive solutions with different quantization levels is below a certain pre-specified threshold. Therefore, the proposed algorithm progressively refines the quantization level, thus eventually achieving low error floor with a reduced communication burden. The performance gains of the proposed algorithm are demonstrated via illustrative examples.**

I. Introduction

The problem of distributed optimization has become increasingly important in recent years due to the rise of large-scale machine learning [1], control [2], and other data-driven applications [3] that involve massive amounts of data.

Most distributed optimization algorithms in current literature assume that nodes exchange real valued messages of infinite precision [4]–[8]. In distributed computing settings, nodes typically communicate with each other over a network that has limited communication bandwidth and latency. This means that exchanging messages with infinite precision can be impractical or even impossible. More specifically, the assumption of infinite-capacity communication channels is unrealistic because it requires the ability to transmit an infinite number of bits per second. Additionally, most distributed algorithms assume the transmission of rational numbers, which however, is only possible over infinite-capacity communication channels.

In order to alleviate the aforementioned limiting assumption, researchers have focused on the scenario where nodes are exchanging quantized[1] messages [9]–[21]. This may lead to a solution to the proximity of the optimal solution that depends on the utilized quantization level. However, most of the proposed works are mainly quantizing values of an asymptotic coordination algorithm. As a consequence, they are only able to exhibit asymptotic convergence to a solution in the proximity of the optimal solution.

A recent work [22] proposed a finite-time communication-efficient algorithm for distributed optimization. However, it is not obvious how coarse/fine the quantization should be. If it is too coarse, the solution to the optimization may lead to an error floor that is considerably large and hence, unacceptable (for the considered application). If it is too fine, then larger packets are needed for communication (which means that the overall system may experience delays, more packet losses, etc). Since the exact solution is not known *a priori* though, it is not possible to know whether the quantization level chosen is sufficient.

Main Contributions. In this paper, we present a novel distributed optimization algorithm aimed at addressing the challenge of quantization level tuning. Our proposed algorithm extends the quantized distributed optimization method in [22] (which converges to an approximate solution within a finite number of iterations). Our key contribution is a strategy that dynamically adjusts the quantization level based on the comparison of error floors resulting from different quantization levels. The proposed strategy allows us to assess the satisfaction of the obtained solution, even in the absence of knowledge about the optimal solution. Our key contributions are the following.

Apostolos I. Rikos is with the Department of Electrical and Computer Engineering, Division of Systems Engineering, Boston University, Boston, MA 02215, US. E-mail: arikos@bu.edu.

Wei Jiang resides in Hong Kong, China. Email: wjiang.lab@gmail.com.

T. Charalambous is with the Department of Electrical and Computer Engineering, School of Engineering, University of Cyprus, 1678 Nicosia, Cyprus. He is also with the Department of Electrical Engineering and Automation, School of Electrical Engineering, Aalto University, Espoo, Finland. E-mail:charalambous.themistoklis@ucy.ac.cy.

K. H. Johansson is with the Division of Decision and Control Systems, KTH Royal Institute of Technology, SE-100 44 Stockholm, Sweden. He is also affiliated with Digital Futures. E-mail: kallej@kth.se.

Part of this work was supported by the Knut and Alice Wallenberg Foundation, the Swedish Research Council, and the Swedish Foundation for Strategic Research. The work of T. Charalambous was partly supported by the European Research Council (ERC) Consolidator Grant MINERVA (Grant agreement No. 101044629).

[1]Quantization is the process of mapping input values from a large set (often a continuous set) to output values in a (countable) smaller set. In quantization, nodes compress (i.e., quantize) their value (of their state or any other stored information), so that they can represent it with a few bits and then transmit it through the channel.

A. We present a distributed optimization algorithm that leverages on gradient descent and fosters efficient communication among nodes through the use of quantized messages; see Algorithm 1. Our algorithm operates by comparing solutions obtained with different quantization levels. If these solutions exceed a predefined threshold, we continue to refine the quantization level, otherwise, we terminate its operation; see for example Fig. 1. While we cannot directly enforce the exact desired accuracy, our algorithm can attain a desired level of accuracy through the selection of an appropriate threshold. For example, by setting the threshold in the order of 10^{-7}, we can guarantee an error floor as low as 10^{-6}. Remarkably, with each iteration of the optimization process, the quantization granularity becomes finer, and the initial conditions approach the vicinity of the optimal state. This behavior resembles a distributed zooming process over the optimization region.

B. We validate the performance of our proposed algorithm through illustrative examples, demonstrating its effectiveness in terms of communication efficiency and the computation of optimal solutions; see Section V. The achieved improvement in communication efficiency is substantial and holds practical significance; see Remark 3.

II. NOTATION AND PRELIMINARIES

Notions. The sets of real, rational, integer and natural numbers are denoted by $\mathbb{R}, \mathbb{Q}, \mathbb{Z}$ and $\mathbb{N}$, respectively. The symbol $\mathbb{Z}_{\geq 0}$ denotes the set of nonnegative integer numbers. The symbol $\mathbb{R}_{\geq 0}$ denotes the set of nonnegative real numbers. The symbol $\mathbb{R}^n_{\geq 0}$ denotes the nonnegative orthant of the n-dimensional real space $\mathbb{R}^n$. Matrices are denoted with capital letters (e.g., A), and vectors with small letters (e.g., x). The transpose of matrix A and vector x are denoted as $A^\top$, $x^\top$, respectively. For any real number $a \in \mathbb{R}$, the floor $\lfloor a \rfloor$ denotes the greatest integer less than or equal to a while the ceiling $\lceil a \rceil$ denotes the least integer greater than or equal to a. For any matrix $A \in \mathbb{R}^{n \times n}$, the a_{ij} denotes the entry in row i and column j. By $\mathbb{1}$, we denote the all-ones vector and by $\mathbb{I}$ the identity matrix of appropriate dimensions. By $\|\cdot\|$, we denote the Euclidean norm of a vector.

Graph Theory. The communication network is captured by a directed graph (digraph) defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. This digraph consists of n ($n \geq 2$) nodes communicating only with their immediate neighbors, and is static (i.e., it does not change over time). In $\mathcal{G}$, the set of nodes is denoted as $\mathcal{V} = \{v_1, v_2, ..., v_n\}$, and the set of edges as $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V} \setminus \{(v_i, v_i) \mid v_i \in \mathcal{V}\}$ (note that self-edges are excluded). The cardinality of the sets of nodes, edges are denoted as $|\mathcal{V}| = n$, $|\mathcal{E}| = m$, respectively. A directed edge from node v_i to node v_l is denoted by $(v_l, v_i) \in \mathcal{E}$, and captures the fact that node v_l can receive information from node v_i (but not the other way around). The subset of nodes that can directly transmit information to node v_i is called the set of in-neighbors of v_i and is represented by $\mathcal{N}_i^- = \{v_j \in \mathcal{V} \mid (v_i, v_j) \in \mathcal{E}\}$. The subset of nodes that can directly receive information from nodes v_i is called the set of out-neighbors of v_i and is represented by $\mathcal{N}_i^+ = \{v_l \in \mathcal{V} \mid (v_l, v_i) \in \mathcal{E}\}$. The *in-degree*, and *out-degree* of v_j and is denoted by $\mathcal{D}_i^- = |\mathcal{N}_i^-|$, $\mathcal{D}_i^+ = |\mathcal{N}_i^+|$, respectively. The diameter D of a digraph is the longest shortest path between any two nodes $v_l, v_i \in \mathcal{V}$. A directed *path* from v_i to v_l of length t exists if we can find a sequence of nodes $i \equiv l_0, l_1, \ldots, l_t \equiv l$ such that $(l_{\tau+1}, l_\tau) \in \mathcal{E}$ for $\tau = 0, 1, \ldots, t-1$. A digraph is *strongly connected* if there exists a directed path from every node v_i to every node v_l, for every $v_i, v_l \in \mathcal{V}$.

Node Operation. Each node $v_i \in \mathcal{V}$ executes a distributed optimization algorithm and a distributed coordination algorithm. For the optimization algorithm (see Algorithm 1 (GraDeZoQuC) below) at each time step k, each node v_i maintains

- its local estimate variable $x_i^{[k]} \in \mathbb{Q}$ (used to calculate the optimal solution),
- γ_β which is the time step during which nodes have converged to a neighborhood of the optimal solution,
- the set S_i which is used to store the γ_β,
- the variable ind_i (used as an indicator of the length of the set S_i),
- the variable flag_i (used to decide whether to terminate the optimization algorithm operation).

For the coordination algorithm (Algorithm 2 (FiTQuAC) below) at each time step k, each node v_i maintains

- the stopping variables M_i, $m_i \in \mathbb{N}$ (used to determine whether convergence has been achieved), and
- the variables $y_i \in \mathbb{Q}$, $c_i^y, c_i^z \in \mathbb{Z}$, and $z_i \in \mathbb{Q}$, (used to communicate with other nodes by either transmitting or receiving messages).

Asymmetric Quantizers. Quantization is a strategy that lessens the number of bits needed to represent information. It is used to compress data before transmission, thus reducing the amount of bandwidth required to transmit messages, and increasing power and computation efficiency. Quantization is mainly used to describe communication constraints and imperfect information exchanges between nodes such as in wireless communication systems, distributed control systems, and sensor networks. The three main types of quantizers are (i) asymmetric, (ii) uniform, and (iii) logarithmic [23]. In this paper we rely on asymmetric quantizers in order to reduce the required communication bandwidth (but our results can also be extended to logarithmic and uniform quantizers). Asymmetric quantizers are defined as

$$q_\Delta^a(\xi) = \left\lfloor \frac{\xi}{\Delta} \right\rfloor, \tag{1}$$

where $\Delta \in \mathbb{Q}$ is the quantization level, $\xi \in \mathbb{R}$ is the value to be quantized, and $q_\Delta^a(\xi) \in \mathbb{Q}$ is the quantized version of ξ with quantization level Δ (note that the superscript "a" indicates that the quantizer is asymmetric.).

The $\max$-consensus algorithm converges to the maximum value among all nodes in a finite number of steps $s_m \leq D$, where D is the network diameter (see, [24, Theorem 5.4]). Similar results hold for the $\min$-consensus algorithm.

III. Problem Formulation

Let us consider a distributed network modeled as a digraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $n = |\mathcal{V}|$ nodes. We assume that each node v_i is endowed with a local cost function $f_i(x) : \mathbb{R}^p \mapsto \mathbb{R}$ only known to itself, and communication channels among nodes have limited capacity and as a result the exact states cannot be communicated if they are irrational. In other words, only quantized values can be transmitted/communicated and thus x can take values that can be expressed as rational numbers.

In this paper we aim to develop a distributed algorithm which allows nodes, despite the communication limitations, to cooperatively solve approximately the following optimization problem, herein called **P1**:

$$\min_{x\in\mathcal{X}} \; F(x_1, x_2, ..., x_n) \equiv \sum_{i=1}^{n} f_i(x_i), \tag{2a}$$

$$\text{s.t. } x_i = x_j, \forall v_i, v_j \in \mathcal{V}, \tag{2b}$$

$$x_i^{[0]} \in \mathcal{X} \subset \mathbb{Q}_{\geq 0}, \forall v_i \in \mathcal{V}, \tag{2c}$$

$$\text{nodes communicate with quantized values,} \tag{2d}$$

$$\text{if } \|f_i(x_i^{[\gamma_{\beta-1}]}) - f_i(x_i^{[\gamma_\beta]})\| \leq \varepsilon_s, \; \forall v_i \in \mathcal{V},$$
$$\text{for any } \varepsilon_s > 0, \text{then terminate operation,} \tag{2e}$$

where $\beta \in \mathbb{N}$, γ_β is the optimization convergence point for which we have $f_i(x_i^{[1+\gamma_\beta]}) = f_i(x_i^{[\gamma_\beta]}) \; \forall v_i \in \mathcal{V}$, $\mathcal{X}$ is the set of feasible values of parameter x, and x^* is the optimal solution of the optimization problem. Eq. (2a) means that we aim to minimize the global cost function which is defined as the sum of the local cost functions in the network. Eq. (2b) means that nodes need to calculate equal optimal solutions. Eq. (2c) means that the initial estimations of nodes belong in a common set. Note that it is not necessary for the initial values of nodes to be rational numbers, i.e., $x_i^{[0]} \in \mathcal{X} \subset \mathbb{Q}_{\geq 0}$. However, nodes can generate a quantized version of their initial states by utilizing the Asymmetric Quantizer presented in Section II. Eq. (2d) means that nodes are transmitting and receiving quantized values with their neighbors since communication channels among nodes have limited bandwidth. Eq. (2e) means that nodes are tracking the improvement of their local cost function between two consecutive convergence points $\gamma_{\beta+1}$ and γ_β. If the improvement of the local cost function of every node is less than a predefined threshold ε_s, then they decide to stop their operation in a distributed way.

Remark 1. *It will be shown later that our algorithm converges to a neighborhood of the optimal solution due to the quantized communication between nodes (see (2d)). Therefore, with γ_β we denote the time step for which all nodes have converged to this neighborhood (i.e., it is the optimization convergence point), and for this reason $f_i(x_i^{[1+\gamma_\beta]}) = f_i(x_i^{[\gamma_\beta]}), \forall v_i \in \mathcal{V}$.*

IV. Distributed Optimization with Zooming over Quantized Communication

In this section, we present a distributed algorithm which solves problem **P1** described in Section III. Before presenting the operation of our proposed algorithm, we make the following assumptions which are necessary for the development of our results.

Assumption 1. *The communication network (described as a digraph) $\mathcal{G}$ is strongly connected.*

Assumption 2. *For every node v_i, the local cost function $f_i(x)$ is smooth and strongly convex. This means that for every node v_i, for every $x_1, x_2 \in \mathcal{X}$,*

- *there exists positive constant L_i such that*

$$\|\nabla f_i(x_1) - \nabla f_i(x_2)\|_2 \leq L_i \|x_1 - x_2\|_2, \tag{3}$$

- *there exists positive constant μ_i such that*

$$f_i(x_2) \geq f_i(x_1) + \nabla f_i(x_1)^\top (x_2 - x_1) + \frac{\mu_i}{2}\|x_2 - x_1\|_2^2. \tag{4}$$

This means that the Lipschitz-continuity and strong-convexity constants of the global cost function F (see (2a)) are L μ, defined as $L = \max\{L_i\}$, and $\mu = \min\{\mu_i\}$.

Assumption 3. *The diameter D (or an upper bound) is known to every node v_i in the network.*

Assumption 1 is a necessary condition so that information from each node can reach every other node in the network, thus all nodes to be able to calculate the optimal solution x^* of $P1$. Assumption 2 is the Lipschitz-continuity condition in (3), and strong-convexity condition in (4). Lipschitz-continuity is a standard assumption in distributed first-order optimization problems (see [25], [26]) and guarantees (i) the existence of the solution x^*, and (ii) that nodes are able to calculate the global optimal minimizer x^* for (2a). Strong-convexity is useful for guaranteeing (i) linear convergence rate, and (ii) that the global function F has no more than one global minimum. Assumption 3 allows each node $v_i \in \mathcal{V}$ to determine whether calculation of a solution x_i that fulfills (2b) has been achieved in a distributed manner.

The intuition of Algorithm 1 (GraDeZoQuC) is the following.

Initialization. Each node v_i maintains an estimate of the optimal solution $x_i^{[0]}$, the desired quantization level Δ, and the refinement constant c_r which is used to refine the quantization level. Quantization level (i) is the same for every node, (ii) allows quantized communication between nodes, and (iii) determines the desired level of precision of the solution. Additionally, each node initializes a set S_i. This set serves as a repository for storing the time steps during which nodes have collectively calculated the neighborhood of the optimal solution according to the utilized quantization level Δ. More specifically, Algorithm 1 converges to a neighborhood of the optimal solution due to the quantized communication between nodes. Each node v_i stores in S_i the optimization time step during which this neighborhood has been reached.

Iteration. At each time step k, each node v_i:

- Updates the estimate of the optimal solution $x_i^{[k+\frac{1}{2}]}$ by performing a gradient descent step towards the negative direction the node's gradient; see Iteration step 1.

- Utilizes Algorithm 2 (FiTQuAC); see Iteration step 2. Algorithm 2 (details of its operation are presented below) allows each node to fulfill (2d), and to calculate in finite time an estimate of the optimal solution $x_i^{[k+1]}$ that fulfills (2b).
- Checks if the calculated estimate of the optimal solution $x_i^{[k+1]}$ is the same as the previous optimization step $x_i^{[k]}$; see Iteration step 3.
- If the above condition holds, then nodes have reached a neighborhood of the optimal solution which depends on the utilized quantization level (i.e., they reached the optimization convergence point for the current quantization level). In this case, node v_i stores the corresponding time step $\gamma_\beta = k$ at the set S_i; see Iteration steps $3a$, $3b$.
- Checks if the difference between the value of its local function at the current optimization convergence point $f_i(x_i^{[\gamma_\beta]})$ and the value of its local function at the previous optimization convergence point $f_i(x_i^{[\gamma_{\beta-1}]})$ is less than a given threshold ε_s; see Iteration step $3c$.
- If the above condition holds, it sets its voting variable equal to 0 (otherwise it sets it to 1). Then nodes are performing a max-Consensus protocol to decide whether they will continue the operation of Algorithm 1; see Iteration step $3d$. The main idea for executing max-Consensus is that if every node finds that the difference between $f_i(x_i^{[\gamma_\beta]})$ and $f_i(x_i^{[\gamma_{\beta-1}]})$ is less than ε_s (signaling convergence) then nodes opt to halt their operation.
- After executing max-Consensus, if at least one node detects that the difference exceeds ε_s (indicating a lack of convergence) then nodes utilize the refinement constant c_r to adjust the quantization level and repeat the algorithm's operation accordingly, otherwise the operation is terminated; see Iteration step $3e$.

Algorithm 2 (FiTQuAC) allows each node to be able to calculate the quantized average of each node's estimate in finite time by processing and transmitting quantized messages, with precision determined by the quantization level. FAQuA algorithm utilizes (i) asymmetric quantization, (ii) quantized averaging, and (iii) a stopping strategy. The intuition of Algorithm 2 (FiTQuAC) is the following. Initially, each node v_i uses an asymmetric quantizer to quantize its state; see Initialization-step 2. Then, at each time step η each node v_i:

- Splits the y_i into z_i equal pieces (the value of some pieces might be greater than others by one); see Iteration-steps 4.1, 4.2.
- Transmits each piece to a randomly selected out-neighbor or to itself; see Iteration-step 4.3.
- Receives the pieces transmitted from its in-neighbors, sums them with y_i and z_i, and repeats the operation; see Iteration-step 4.4.

Finally, every D time steps, each node v_i performs in parallel a max-consensus and a min-consensus operation; see Iteration-steps 1, 2, 5. If the results of the max-consensus and min-consensus have a difference less or equal to one, each node v_i (i) scales the solution according to the quantization level, (ii) stops the operation of Algorithm 2, (iii) uses the value $x_i^{[k+1]}$ to continue the operation of Algorithm 1. Algorithm 2 converges in finite time according to [27, Theorem 1]. It is important to note here that Algorithm 2 (FiTQuAC) runs between every two consecutive optimization steps k and $k+1$ of Algorithm 1 (GraDeZoQuC) (for this reason it uses a different time index λ and not k as Algorithm 1).

Our proposed algorithm is detailed below as Algorithm 1.

Algorithm 1 Gradient Descent with Zoomed Quantized Communication (GraDeZoQuC)

Input: A strongly connected directed graph $\mathcal{G}$ with $n = |\mathcal{V}|$ nodes and $m = |\mathcal{E}|$ edges. Static step-size $\alpha \in \mathbb{R}$, digraph diameter D, initial value $x_i^{[0]}$, local cost function f_i, error bound ε_s, quantization level $\Delta \in \mathbb{Q}$, refinement constant $c_r \in \mathbb{N}$, for every node $v_j \in \mathcal{V}$. Assumptions 1, 2, 3 hold.

Initialization: Each node $v_i \in \mathcal{V}$ sets $\text{ind}_i = 0$, $\beta = \text{ind}_i$, $S_i = \{0\}$.

Iteration: For $k = 0, 1, 2, \ldots$, each node $v_i \in \mathcal{V}$ does the following:

1) $x_i^{[k+\frac{1}{2}]} = x_i^{[k]} - \alpha \nabla f_i(x_i^{[k]})$;
2) $x_i^{[k+1]} =$ Algorithm 2$(x_i^{[k+\frac{1}{2}]}, D, \Delta)$;
3) **if** $x_i^{[k+1]} = x_i^{[k]}$, **then**
   - $3a$) set $\text{ind}_i = \text{ind}_i + 1$, $\beta = \text{ind}_i$, $\gamma_\beta = k$;
   - $3b$) set $S_i = S_i \cup \{\gamma_\beta\}$;
   - $3c$) **if** $\|f_i(x_i^{[\gamma_{\beta-1}]}) - f_i(x_i^{[\gamma_\beta]})\| \leq \varepsilon_s$, **then** set $\text{vot}_i = 0$; **else** set $\text{vot}_i = 1$;
   - $3d$) flag_i = max - Consensus (vot_i);
   - $3e$) **if** $\text{flag}_i = 0$ **then** terminate operation; **else** set $\Delta = \Delta / c_r$ and go to Step 1;

Output: Each node $v_i \in \mathcal{V}$ calculates x_i^* which solves problem **P1** in Section III.

A. Convergence of Algorithm 1

We now analyze the convergence time of Algorithm 1 via the following theorem.

Theorem 1. *Under Assumptions 1–3, when the step-size α satisfies $\alpha \in (\frac{n(\mu+L)}{4\mu L}, \frac{2n}{\mu+L})$ and $\delta \in (0, \frac{n[4\alpha\mu L - n(\mu+L)]}{2\alpha[n(\mu+L)-2\alpha\mu L]})$ where $L = \max\{L_i\}, \mu = \min\{\mu_i\}$, Algorithm 1 generates a sequence of points $\{x^{[k]}\}$ (i.e., the variable $x_i^{[k]}$ of each node $v_i \in \mathcal{V}$) which satisfies*

$$\|\hat{x}^{[k+1]} - x^*\|^2 < \vartheta \|\hat{x}^{[k]} - x^*\|^2 + \mathcal{O}(\Delta^2), \tag{7}$$

where Δ is the quantizer and

$$\vartheta := 2(1 + \frac{\alpha\delta}{n})(1 - \frac{2\alpha\mu L}{n(\mu + L)}) \in (0, 1), \tag{8a}$$

$$\mathcal{O}(\Delta^2) = (8 + 32n^2\hat{\alpha}^2 L^2 + \frac{32n^2\hat{\alpha}L^2}{\delta})\Delta^2. \tag{8b}$$

Proof. The proof follows directly from the proof of [22, Theorem 1], with the difference that the process is restarted under some condition (eq. (2e)). The details are omitted due to space limitations. □

Algorithm 2 Finite-Time Quantized Average Consensus (FiTQuAC)

Input: $x_i^{[k+\frac{1}{2}]}, D, \Delta$.

Initialization: Each node $v_i \in \mathcal{V}$ does the following:

1) Assigns probability b_{li} to each out-neigbor $v_l \in \mathcal{N}_i^+ \cup \{v_i\}$, as follows

$$b_{li} = \begin{cases} \frac{1}{1+\mathcal{D}_i^+}, & \text{if } l = i \text{ or } v_l \in \mathcal{N}_i^+, \\ 0, & \text{if } l \neq i \text{ and } v_l \notin \mathcal{N}_i^+; \end{cases}$$

2) sets $z_i = 2$, $y_i = 2\ q_\Delta^a(x_i^{[k+\frac{1}{2}]})$ (see (1));

Iteration: For $\lambda = 1, 2, \ldots$, each node $v_i \in \mathcal{V}$, does:

1) **if** $\lambda \mod (D) = 1$ **then** $M_i = \lceil y_i/z_i \rceil$, $m_i = \lfloor y_i/z_i \rfloor$;
2) broadcasts M_i, m_i to every $v_l \in \mathcal{N}_i^+$; receives M_j, m_j from every $v_j \in \mathcal{N}_i^-$; sets $M_i = \max_{v_j \in \mathcal{N}_i^- \cup \{v_i\}} M_j$, $m_i = \min_{v_j \in \mathcal{N}_i^- \cup \{v_i\}} m_j$;
3) sets $c_i^z = z_i$;
4) **while** $c_i^z > 1$ **do**
   4.1) $c_i^y = \lfloor y_i \ / \ z_i \rfloor$;
   4.2) sets $y_i = y_i - c_i^y$, $z_i = z_i - 1$, and $c_i^z = c_i^z - 1$;
   4.3) transmits c_i^y to randomly chosen out-neighbor $v_l \in \mathcal{N}_i^+ \cup \{v_i\}$ according to b_{li};
   4.4) receives c_j^y from $v_j \in \mathcal{N}_i^-$ and sets

$$y_i = y_i + \sum_{j=1}^{n} w_{\lambda,ij}^{[r]}\ c_j^y\ , \tag{5}$$

$$z_i = z_i + \sum_{j=1}^{n} w_{\lambda,ij}^{[r]}\ , \tag{6}$$

   where $w_{\lambda,ij}^{[r]} = 1$ when node v_i receives c_i^y, 1 from v_j at time step λ (otherwise $w_{\lambda,ij}^{[r]} = 0$ and v_i receives no message at time step λ from v_j);
5) **if** $\lambda \mod D = 0$ **and** $M_i - m_i \leq 1$ **then** sets $x_i^{[k+1]} = m_i\Delta$ and stops operation.

Output: $x_i^{[k+1]}$.

Remark 2 (Convergence Precision)**.** *The focus of our convergence analysis in Theorem 1 is on the optimization steps performed during the operation of Algorithm 1. As stated, an additional term $\mathcal{O}(\Delta^2)$ appears in (7). This term affects the precision of the calculated optimal solution. While some distributed quantized algorithms in the literature exhibit exact convergence to the optimal solution (e.g., see [9], [16]), our Algorithm 1 adopts an adaptive quantization level to balance communication efficiency and convergence precision. However, by setting $\varepsilon_s = 0$ during Initialization, Algorithm 1 can be adjusted to converge to the exact optimal solution x^* (by refining the quantization level infinitely often). This characteristic is highly important in scenarios where higher precision is crucial. Specifically, Algorithm 1 is able to adjust to specific application requirements by performing a trade-off between communication efficiency and convergence precision. Furthermore, it is worth noting that Algorithm 1 offers distinct advantages particularly in scenarios where communication efficiency is a priority while maintaining satisfactory convergence precision in various applications. As will be shown in Section V, the operational advantages of Algorithm 1 are evident, making it a valuable tool in distributed optimization tasks.*

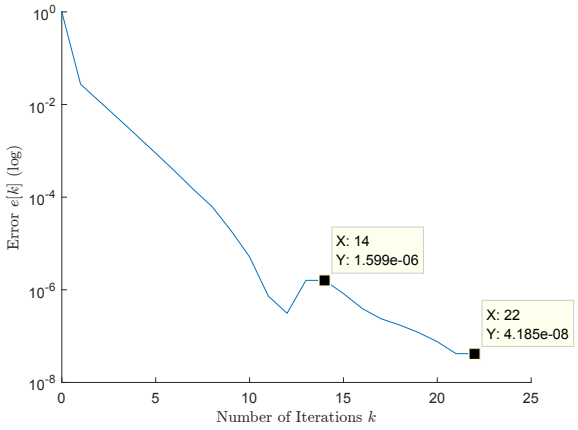

Fig. 1. Execution of Algorithm 1 over a random digraph of 20 nodes.

V. Simulation Results

In this section, we present simulation results in order to demonstrate the operation of Algorithm 1 and its potential advantages. More specifically:

A. We focus on a random digraph of 20 nodes and show how the nodes' states converge to the optimal solution (see Fig. 1). Furthermore, we analyze how the event-triggered zooming (i) leads to a more precise calculation of the optimal solution, and (ii) allows nodes to terminate their operation.

B. We compare the operation of Algorithm 1 against existing algorithms in the literature, and we emphasize on the introduced improvements (see Fig. 2).

For both cases **A.** and **B.** each node v_i is endowed with a local cost function $f_i(x) = \frac{1}{2}\beta_i(x - x_0)^2$. This cost function is smooth and strongly convex. Furthermore, for $f_i(x)$ we have that (i) β_i is initialized as a random integer between 1 and 5 for each node in the network (and characterizes the cost sensitivity of node v_i), and (ii) x_0 is initialized as a random integer between 1 and 5 (and represents the demand of node v_i).

A. Operation over a random digraph of 20 nodes. In Fig. 1, we demonstrate our algorithm over a randomly generated digraph consisted of 20 nodes. For each node v_i we have $\alpha = 0.12$, $x_i^{[0]} \in [1, 5]$, $\varepsilon_s = 0.003$, $\Delta = 0.001$, $c_r = 10$. In Fig. 1, we plot the error $e^{[k]}$ in a logarithmic scale against the number of iterations. The error $e^{[k]}$ is defined as

$$e^{[k]} = \sqrt{\sum_{j=1}^{n} \frac{(x_j^{[k]} - x^*)^2}{(x_j^{[0]} - x^*)^2}}, \tag{9}$$

where x^* is the optimal solution of the problem **P1**.

In Fig. 1 we can see that our algorithm is able to converge to the optimal solution. Furthermore, let us focus at time steps $k = 13, 14$, and $k = 21, 22$. At time steps $k = 13, 14$

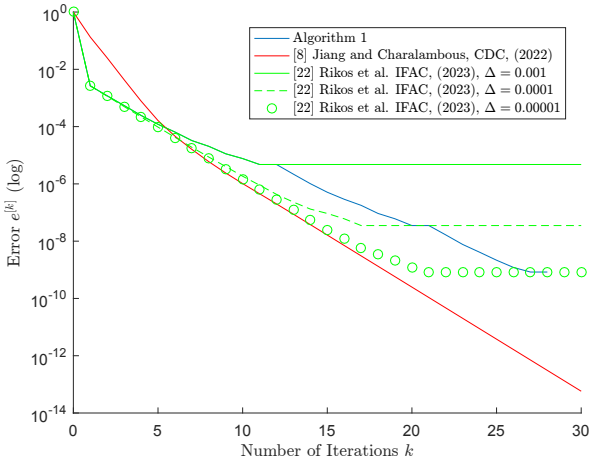

Fig. 2. Comparison of Algorithm 1 against [8], [22] over a random digraph of 20 nodes.

we have that the condition in Iteration Step 3 holds (i.e., $x_i^{[13]} = x_i^{[14]}$ for every $v_i \in \mathcal{V}$), and $e^{[13]} = e^{[14]}$. Therefore, during time step 14, nodes are checking the overall improvement of their local cost functions (i.e., Iteration Step $3c$). Since this condition does not hold for at least one node, they decide to refine the quantization level (i.e., set $\Delta = \Delta/10 = 0.0001$), and continue executing Algorithm 1. At time steps, $k = 14, ..., 21$, nodes are able to approximate the optimal solution with more precision than before since the precision depends on the quantization level (as we showed in Theorem 1). At time steps $k = 21, 22$ we have that the condition in Iteration Step 3 holds again. However, during time step 22 the overall improvement of every nodes' local cost function is less than the given threshold ϵ_s, i.e., $\|f_i(x_i^{[14]}) - f_i(x_i^{[22]})\| \leq \varepsilon_s$, for every $v_i \in \mathcal{V}$ (see Iteration Step $3c$). As a result, nodes decide to terminate the operation at time step $k = 22$ (see Iteration Step $3e$). Note here that a choice of a smaller ϵ_s may lead nodes to refine again the quantization level. This refinement (i.e, $\Delta \leq 0.00001$) will allow them to approximate the optimal solution with even higher precision.

B. Comparison with current literature. In Fig. 2, we compare the operation of Algorithm 1 against [8], [22]. We plot the error $e^{[k]}$ defined in (9). For the operation of the three algorithms, for each node v_i we have $\alpha = 0.12$, $x_i^{[0]} \in [1, 5]$, $\varepsilon_s = 27 \cdot 10^{-7}$, $\Delta = 0.001$, $c_r = 10$ (note that [8] is not utilizing ε_s, Δ, c_r, and [22] is not utilizing ε_s, c_r). Our comparisons focus on:

B-A. The convergence of Algorithm 1 compared to [8], [22].
B-B. The required communication for convergence (in terms of bits per optimization step) of Algorithm 1 compared to [8], [22].

B-A (Convergence). In Fig. 2 we can see that Algorithm 1 converges identically to [22] for optimization steps $k = 0, ..., 12$. However, at time step 12, each node refines the quantization level (because the condition at Iteration Step $3c$ of Algorithm 1 does not hold for at least one node). In this case, for time steps $k > 12$ we can see that Algorithm 1 approximates the optimal solution with higher precision than [22]. This is mainly because [22] utilizes a static quantization level, which is not refined during the operation of the algorithm. Then, at time step $k = 21$, Algorithm 1 refines again the quantization level, obtaining an even more precise estimation of the optimal solution. However, at time step $k = 27$, we have that the condition at Iteration Step $3c$ holds for every node and Algorithm 1 terminates its operation. Finally, in Fig. 2 we can see that [8] exhibits linear convergence rate and is the fastest among the three algorithms. However, during its operation, each node needs to form the Hankel matrix and perform additional computations when the matrix loses rank. This requires the exact values from each node. It means that nodes need to exchange messages of infinite capacity which is practically infeasible and imposes excessive communication requirements over the network. Therefore the main advantage of Algorithm 1 compared to [8], is that nodes exchange quantized values guaranteeing efficient communication.

B-B (Communication). In Fig. 2, let us focus on comparing Algorithm 1 with [22] for $\Delta = 0.00001$ (see green circles line in Fig. 2). Specifically, we will focus on the communication requirements (in terms of total number of bits and bits per optimization time step) for achieving the error $e^{[27]}$ for Algorithm 1 (which is the same as the error $e^{[21]}$ for the algorithm in [22]). The communication bits are calculated as the ceiling of the base-2 logarithm of the transmitted values. For example if node v_i transmits the quantized value α, then the number of bits it transmits is equal to $\lceil \log_2(\alpha) \rceil$. Note that comparing Algorithm 1 with [22] for $\Delta = 0.001$, and $\Delta = 0.0001$ can be shown identically. In Fig. 2, we have that during the operation of [22] for $\Delta = 0.00001$, nodes are utilizing *in total* 800754 bits for communicating with their neighbors. This means that the average communication requirement for each node is $\frac{800754}{(20)(21)} = 1906.55$ bits per optimization time step (since the network consists of 20 nodes which need 21 iterations to converge). During the operation of Algorithm 1, nodes are utilizing $\Delta = 0.001$ for steps $k = 0, ..., 12$, $\Delta = 0.0001$ for steps $k = 13, ..., 21$, and $\Delta = 0.00001$ for steps $k = 22, ..., 27$. For steps $k = 0, ..., 12$, nodes are utilizing *in total* 195607 bits for communicating with their neighbors. For steps $k = 12, ..., 21$, nodes are utilizing *in total* 215635 bits for communicating with their neighbors. For steps $k = 21, ..., 27$, nodes are utilizing *in total* 201044 bits for communicating with their neighbors. The total requirement of bits is 612286 for $k = 0, ..., 27$. This means that the average communication requirement for each node is $\frac{612286}{(20)(27)} = 1133.86$ bits per optimization time step. As a result, Algorithm 1, is able to approximate the optimal solution with precision similar to [22] (for $\Delta = 0.00001$), but its communication requirements are significantly lower in terms of total number of bits and bits per optimization time step.

Remark 3. *During the analysis in **B-B**, we have that Algorithm 1 requires less bits for communication compared to [22] (for $\Delta = 0.00001$) because nodes are utilizing a higher quantization level than [22] for optimization steps*

$k = 1, ..., 21$. *This means that nodes are utilizing less bits to quantize and transmit their states towards their neighboring nodes. However, note here that during the operation of Algorithm 1 we can further improve communication efficiency by shifting the quantization basis after we refine the quantization step. Shifting the quantization basis means changing the location of the quantization levels relative to the states of the nodes. This can be done by adding/subtracting a constant value to the states of the nodes before quantization. This constant value that we can subtract is equal to the optimal solution to which the states of the nodes have converged before refining the quantization level. For example, in Fig. 2, during optimization step* $k = 15$*, node* v_i *will quantize the state* $x_i^{[15]} - x_i^{[\gamma_1]}$ *(where* $x_i^{[\gamma_1]}$ *is equal to* $x_i^{[12]}$*). This strategy increases even further communication efficiency since the states of the nodes can be represented using fewer bits without sacrificing the accuracy of the calculated optimal solutions during the optimization operation. It will be further analyzed at an extended version of our paper.*

VI. Conclusions

In this paper, we considered an unconstrained distributed strongly convex optimization problem, in which the exchange of information is done over digital channels that have limited capacity (and hence information should be quantized). We proposed a distributed algorithm that solves the problem with a solution at a close proximity to the optimal, by progressively refining the quantization level of a node, thus guaranteeing a certain error floor and more efficient communication (smaller packets/reduced number of bits). A simple numerical example shows the performance of our proposed algorithm and highlights the benefits in terms of communication efficiency. More specifically, in the specific example it was shown that the number of bits needed is $\sim 25\%$ less when the quantization level is refined.

References

[1] A. Nedich, "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] G. S. Seyboth, D. V. Dimarogonas, and K. H. Johansson, "Event-based broadcasting for multi-agent average consensus," *Automatica*, vol. 49, no. 1, pp. 245–252, 2013.

[3] S. U. Stich, J. B. Cordonnier, and M. Jaggi, "Sparsified SGD with memory," in *Advances in Neural Information Processing Systems*, vol. 31. Curran Associates, Inc., 2018, pp. 4447–4458.

[4] 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.

[5] 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.

[6] V. Khatana, G. Saraswat, S. Patel, and M. V. Salapaka, "Gradient-consensus method for distributed optimization in directed multi-agent networks," in *2020 American Control Conference (ACC)*, 2020, pp. 4689–4694.

[7] T. Qin, S. R. Etesami, and C. A. Uribe, "Communication-efficient decentralized local SGD over undirected networks," in *2021 60th IEEE Conference on Decision and Control (CDC)*, 2021, pp. 3361–3366.

[8] W. Jiang and T. Charalambous, "A fast finite-time consensus based gradient method for distributed optimization over digraphs," in *IEEE Conference on Decision and Control*, 2022, pp. 6848–6854.

[9] P. Yi and Y. Hong, "Quantized subgradient algorithm and data-rate analysis for distributed optimization," *IEEE Transactions on Control of Network Systems*, vol. 1, no. 4, pp. 380–392, 2014.

[10] C. Huang, H. Li, D. Xia, and L. Xiao, "Quantized subgradient algorithm with limited bandwidth communications for solving distributed optimization over general directed multi-agent networks," *Neurocomputing*, vol. 185, pp. 153–162, 2016.

[11] Y. Pu, M. N. Zeilinger, and C. N. Jones, "Quantization design for distributed optimization," *IEEE Transactions on Automatic Control*, vol. 62, no. 5, pp. 2107–2120, 2017.

[12] H. Li, S. Liu, Y. C. Soh, and L. Xie, "Event-triggered communication and data rate constraint for distributed optimization of multiagent systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 48, no. 11, pp. 1908–1919, 2018.

[13] T. T. Doan, S. T. Maguluri, and J. Romberg, "Fast convergence rates of distributed subgradient methods with adaptive quantization," *IEEE Transactions on Automatic Control*, vol. 66, no. 5, pp. 2191–2205, 2021.

[14] S. Magnusson, H. Shokri-Ghadikolaei, and N. Li, "On maintaining linear convergence of distributed learning and optimization under limited communication," *IEEE Transactions on Signal Processing*, vol. 68, pp. 6101–6116, 2020.

[15] D. Jhunjhunwala, A. Gadhikar, G. Joshi, and Y. C. Eldar, "Adaptive quantization of model updates for communication-efficient federated learning," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 3110–3114.

[16] Y. Kajiyama, N. Hayashi, and S. Takai, "Linear convergence of consensus-based quantized optimization for smooth and strongly convex cost functions," *IEEE Transactions on Automatic Control*, vol. 66, no. 3, pp. 1254–1261, 2021.

[17] J. Liu, Z. Yu, and D. W. C. Ho, "Distributed constrained optimization with delayed subgradient information over time-varying network under adaptive quantization," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2022.

[18] A. Koloskova, S. Stich, and M. Jaggi, "Decentralized stochastic optimization and gossip algorithms with compressed communication," in *Proceedings of the 36th International Conference on Machine Learning*, 2019, pp. 3478–3487.

[19] D. Basu, D. Data, C. Karakus, and S. Diggavi, "Qsparse-local-SGD: Distributed SGD with quantization, sparsification and local computations," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Curran Associates, Inc., 2019.

[20] A. Reisizadeh, A. Mokhtari, H. Hassani, A. Jadbabaie, and R. Pedarsani, "FedPAQ: A communication-efficient federated learning method with periodic averaging and quantization," in *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, 2020, pp. 2021–2031.

[21] B. Li, S. Cen, Y. Chen, and Y. Chi, "Communication-efficient distributed optimization in networks with gradient tracking and variance reduction," in *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, 2020, pp. 1662–1672.

[22] A. I. Rikos, W. Jiang, T. Charalambous, and K. H. Johansson, "Distributed optimization with gradient descent and quantized communication," in *Proceedings of* 22^{nd} *IFAC World Congress*, 2023, pp. 6433–6439.

[23] J. Wei, X. Yi, H. Sandberg, and K. H. Johansson, "Nonlinear consensus protocols with applications to quantized communication and actuation," *IEEE Transactions on Control of Network Systems*, vol. 6, no. 2, pp. 598–608, 2019.

[24] S. Giannini, D. Di Paola, A. Petitti, and A. Rizzo, "On the convergence of the max-consensus protocol with asynchronous updates," in *Proceedings of IEEE Conference on Decision and Control (CDC)*, 2013, pp. 2605–2610.

[25] J. Xu, S. Zhu, Y. C. Soh, and L. Xie, "Convergence of asynchronous distributed gradient methods over stochastic networks," *IEEE Transactions on Automatic Control*, vol. 63, no. 2, pp. 434–448, 2018.

[26] 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.

[27] A. I. Rikos, C. N. Hadjicostis, and K. H. Johansson, "Non-oscillating quantized average consensus over dynamic directed topologies," *Automatica*, vol. 146, p. 110621, 2022.