Distributed Gradient-Tracking Optimization with Packet-Error Resilience in Unreliable Networks

Evagoras Makridis, Sindri Magnússon, and Themistoklis Charalambous

***Abstract*— In this paper, we address the distributed optimization problem over unreliable error-prone directed networks. We propose a distributed gradient-tracking optimization algorithm (referred to as ARQ-OPT), which exploits packet retransmissions via an Automatic Repeat reQuest (ARQ) error control protocol. Nodes utilize acknowledgement messages transmitted over one-bit error-free channels to trigger retransmissions of packets that were previously received in error. This ensures reliable propagation of information throughout the network, even in the presence of packet errors. We analyze the convergence properties of the proposed algorithm, by augmenting the consensus matrices to align with the retransmission mechanism. Subsequently, we show that by appropriately choosing the maximum number of retransmission attempts, ARQ-OPT can achieve B-step consensus contractivity which allow us to establish asymptotic convergence to the unique optimal solution with probability one. Numerical simulations conducted under various channel conditions validate our findings.**

***Index Terms*— distributed optimization, directed graphs, ARQ, time-varying delays, packet-errors, gradient tracking.**

I. Introduction

In recent years, decision-making has shifted from centralized to distributed approaches. Unlike centralized methods relying on a single master agent, distributed optimization enables collaborative optimization through local information exchange over communication networks. Several applications such as robot networks [1], [2], large scale machine learning [3], and sensor network localization [4], to name a few, follow this distributed paradigm. Distributed multi-agent optimization is particularly crucial in scenarios where central coordination is impractical or could lead to system-wide failure due to a single point of failure.

In distributed optimization problems, each node (agent) v_j in a network of $n \geq 2$ nodes, has access only to its own local objective function $f_j(\cdot) : \mathbb{R}^p \to \mathbb{R}$. The collective goal of all the agents is to minimize the average of these objectives

$$\min_{x\in\mathbb{R}^p} f(x) = \frac{1}{n}\sum_{j=1}^{n} f_j(x), \tag{1}$$

where $x \in \mathbb{R}^p$ is the common decision variable to be cooperatively determined via communication and consensus of agents' local decision variables $x_j \in \mathbb{R}^p$[1].

While initial efforts on first-order methods, *i.e.,* gradient-based, distributed optimization primarily focused on networks with bidirectional links [5]–[7], attention has shifted towards scenarios in which information is exchanged over directed links [8]–[10]. This is motivated by mirroring real-world situations in which wireless communication networks are affected by channel fading. With this view in mind, one should model the information exchange of multi-agent networks using directed graphs. Here it is important to mention that, most of the distributed optimization approaches over directed graphs are grounded in the distributed average consensus methods presented in [11] via the *surplus-based consensus* strategy, [12] via the *ratio consensus* algorithm, and [13] via the *push-sum consensus* method.

In more recent works in directed graphs, algorithms like DEXTRA [14], ADD-OPT [15], Push-DIGing [16], and $\mathcal{AB}$/Push-Pull [17]–[19], gained attention due to the enhanced convergence speed they provide by integrating gradient tracking mechanisms. In a parallel line of research, the works in [20]–[22] studied the influence of communication imposed by the unreliability of the communication network, and proposed distributed optimization algorithms to handle delays and/or packet drops in directed graphs.

In this work, we consider a network of agents that coordinate to solve the distributed optimization in (1) over unreliable communication channels in which packet errors may occur. In particular, to cope with the unreliability in the reception of information packets, we derive a distributed optimization algorithm that is based on the well-established communication protocol *Automatic Repeat reQuest (ARQ)* [23], exploited to ensure reliable coordination among the agents in the network. Within this framework, agents can utilize acknowledgement messages transmitted over one-bit error-free channels to trigger retransmissions of packets that were previously received in error, and ensure that no information is lost. To balance the trade-off between convergence speed and packet-error resilience, we additionally exploit the improved convergence speed that gradient tracking offers. Aiming towards this end, we study more robust ways for the design and analysis of distributed optimization algorithms such that they are tailored to real-world scenarios where

E. Makridis, and T. Charalambous are with the Department of Electrical and Computer Engineering, School of Engineering, University of Cyprus, 1678 Nicosia, Cyprus. E-mails: `surname.name@ucy.ac.cy`. T. Charalambous is also a Visiting Professor at the Department of Electrical Engineering and Automation, School of Electrical Engineering, Aalto University, 02150 Espoo, Finland.

Sindri Magnússon is with the Department of Computer and Systems Science, Stockholm University, 164 07 Stockholm, Sweden. E-mail: `sindri.magnusson@dsv.su.se`.

The work of E. Makridis and T. Charalambous has been partly funded by MINERVA, a European Research Council (ERC) project funded under the European Union's Horizon 2022 research and innovation programme (Grant agreement No. 101044629).

[1]To ensure clarity and simplicity of exposition, in this work we consider scalar decision variables, *i.e.,* $x_j \in \mathbb{R}$. Yet, the adaptability of our methods to vector state representations is straightforward, enabling the practical extension to more complex setups.

communication channels are unreliable. To the best of our knowledge, this is the first gradient-tracking distributed optimization algorithm that considers a realistic communication protocol to handle packet errors. Our analysis shows that agents executing the ARQ-OPT algorithm achieve almost sure convergence to the unique optimal solution of the distributed optimization problem in (1) albeit the packet errors affecting the timely availability of the transmitted variables to their neighbors.

II. Preliminaries

A. Notation

We denote the set of real, integer, and natural numbers by $\mathbb{R}$, $\mathbb{Z}$, and $\mathbb{N}$, respectively. Also, we denote the set of nonnegative real numbers by $\mathbb{R}_+$. The set of integer numbers from $a \in \mathbb{Z}$ up to $b \in \mathbb{Z}$ is denoted by $\mathbb{Z}_a^b = \{a, a+1, \dots, b-1, b\}$. We denote vectors by lowercase letters, and matrices by capital letters. The all-ones n-dimensional column vector is denoted by $\mathbf{1}_n$. The identity matrix and the zero matrix (of appropriate dimensions) are denoted by I and 0, respectively.

B. Network model

The interconnection topology of the data layer of the network is modeled by a digraph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Each agent v_j for $j = 1, \dots, n$ is included in the set of digraph nodes $\mathcal{V} = \{v_1, \cdots, v_n\}$. The interactions between agents are included in the set of digraph edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. The total number of edges in the network is denoted by $m = |\mathcal{E}|$. A directed edge denoted by $\varepsilon_{ji} \in \mathcal{E}$ indicates that node v_j receives information from node v_i, *i.e.,* $v_i \to v_j$. The nodes from which node v_j receives information are called in-neighbors of node v_j, and belong to the set $\mathcal{N}_j^{\mathrm{in}} = \{v_i \in \mathcal{V} | \varepsilon_{ji} \in \mathcal{E}\}$. The number of nodes in the in-neighborhood set is called in-degree and is denoted by $d_j^{\mathrm{in}} = |\mathcal{N}_j^{\mathrm{in}}|$. The nodes that receive information from node v_j directly are called out-neighbors of node v_j, and belong to the set $\mathcal{N}_j^{\mathrm{out}} = \{v_l \in \mathcal{V} | \varepsilon_{lj} \in \mathcal{E}\}$. The number of nodes in the out-neighborhood set is called out-degree and is denoted by $d_j^{\mathrm{out}} = |\mathcal{N}_j^{\mathrm{out}}|$. Each node $v_j \in \mathcal{V}$ has immediate access to its own information, *i.e.,* $\varepsilon_{jj} \in \mathcal{E}$ for all $v_j \in \mathcal{V}$, although it is not included in its own neighborhood sets. In $\mathcal{G}$ a node v_i is reachable from a node v_j if there exists a path from v_j to v_i which respects the direction of the edges.

We assume that for each link of the data layer of the network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, there exists a reverse feedback link, considered to be narrowband and error-free, which supports the transmission of 1-bit acknowledgement messages.

C. Packet error control mechanism

In this work, we assume that packet errors are modeled as independent random events over time and across communication links. In particular, a packet transmitted over the directed link $\varepsilon_{lj} \in \mathcal{E}$ at time k might be received in error by agent v_l, with a fixed probability q_{lj}. Each node $v_j \in \mathcal{V}$ employs an ARQ protocol with retransmission limit on its out-going links $\varepsilon_{lj} \in \mathcal{E}$, denoted by $0 \leq \bar{\tau}_{lj} < \infty$. To simplify the notation that follows, consider that the retransmission limit is homogeneous for all links, *i.e.,* $\bar{\tau}_{lj} = \bar{\tau}$ for all $\varepsilon_{lj} \in \mathcal{E}$. This retransmission limit determines the maximum number of retransmissions before considering a packet as dropped. Let $\mathrm{p}_{j,r}(k)$ denote a packet (re)transmitted $r \in \mathbb{Z}_0^{\bar{\tau}}$ times by node v_j at time k. Here, $\mathrm{p}_{j,0}(k)$ represents a fresh packet transmitted for the first time at time k. Now, we define the acknowledgement message indicating whether the packet $\mathrm{p}_{j,r}(k)$ was successfully received by recipient v_l at time k, by $\gamma_{lj,r}(k) \in \{0, 1\}$, *i.e.,*

$$\gamma_{lj,r}(k) = \begin{cases} 0, & \text{if } \mathrm{p}_{j,r}(k) \text{ received in error,} \\ 1, & \text{otherwise.} \end{cases} \tag{2}$$

The acknowledgement messages arrive at the transmitting node v_j at the next time slot. These messages are used by node v_j to compute the retransmission number for each previously (re)transmitted packet. In particular, at time k node v_j sets $\rho_{lj,0}(k) = 0$, and updates the counter for its previously retransmitted packets, for $r \in \mathbb{Z}_1^{\bar{\tau}+1}$ as:

$$\rho_{lj,r}(k) = \begin{cases} \rho_{lj,r-1}(k-1)+1, & \text{if } \gamma_{lj,r-1}(k-1) = 0, \\ \rho_{lj,r-1}(k-1), & \text{otherwise,} \end{cases} \tag{3}$$

Based on the current retransmission counters at time k, the transmitter v_j (re)transmits the packets $\mathrm{p}_{j,r'}(k)$ for which

$$r' \in \{r \in \mathbb{Z}_0^{\bar{\tau}+1} \mid \rho_{lj,r}(k) \leq \bar{\tau} \wedge \gamma_{lj,r-1}(k-1) = 0\} \tag{4}$$

with $\gamma_{lj,-1}(k-1) = 0$ to enforce the transmission of the fresh packet $\mathrm{p}_{j,0}(k)$. In the case where the oldest packet $\mathrm{p}_{j,\bar{\tau}}(k)$ failed to be received by v_l at time k, then v_j considers the packet as dropped, and stops retransmitting this packet. We denote the packet drop indicator variable by

$$\delta_{lj}(k) = \begin{cases} 1, & \text{if } \rho_{lj,\bar{\tau}+1}(k) = \bar{\tau}+1 \wedge \gamma_{lj,\bar{\tau}}(k-1) = 0, \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

The number of consecutive `NACK` messages for a packet, until its successful reception, translates into the delay experienced by the recipient node to obtain the information of that packet. Formally, the delay for a packet initially transmitted at iteration k over the link ε_{lj} is given by:

$$\tau_{lj}(k) \triangleq \min\{r : r \in \mathbb{Z}_0^{\bar{\tau}}, \gamma_{lj,r}(k+r) = 1\}. \tag{6}$$

The probability that the packet $\mathrm{p}_{j,r}(k)$ arrives at node v_l in error at time step $k \geq 0$, is given by:

$$q_{lj} \triangleq \mathbb{P}\big(\gamma_{lj,r}(k) = 0\big), \tag{7}$$

for all $r \in \mathbb{Z}_0^{\bar{\tau}}$. From this we can obtain the probability that the fresh packet $\mathrm{p}_{j,0}(k)$ will experience an integer delay $0 \leq \tau_{lj}(k) \leq \bar{\tau}$ due to retransmissions, which is given by:

$$\mathbb{P}\big(\tau_{lj}(k) = r\big) = q_{lj}^r (1 - q_{lj}), \tag{8}$$

for $r \in \mathbb{Z}_0^{\bar{\tau}}$. The probability that the packet will be dropped is given by:

$$\mathbb{P}\big(\delta_{lj}(k) = 1\big) = 1 - \sum_{r=0}^{\bar{\tau}} \mathbb{P}\big(\tau_{lj}(k) = r\big) = q_{lj}^{\bar{\tau}+1}. \tag{9}$$

D. Assumptions

Before presenting the proposed algorithm, we make the following assumptions that are standard in the distributed optimization literature.

A1. Graph strong connectivity: The directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is strongly connected, *i.e.,* for any pair of nodes $v_i, v_j \in \mathcal{V}$ there exists a directed path from v_i to v_j and a directed path from v_j to v_i.

A2. Smoothness: Each local objective function, f_j, is ℓ_j-smooth, *i.e.,* it is differentiable and has a Lipschitz-continuous gradient. Formally, there exists $\ell_j > 0$ such that

$$\|\nabla f_j(x_1) - \nabla f_j(x_2)\|_2 \le \ell_j \|x_1 - x_2\|_2.$$

Moreover, we can infer that the global function $f = \frac{1}{n}\sum_{j=1}^n f_j$ is $\bar{\ell}$-smooth with $\bar{\ell} = \frac{1}{n}\sum_{j=1}^n \ell_j$.

A3. Strong convexity: Each local objective function, f_j, is μ_j-strongly convex, *i.e.,*

$$f_j(x_1) - f_j(x_2) \le \nabla f_j(x_1)^\top (x_1 - x_2) - \frac{\mu_j}{2}\|x_1 - x_2\|_2^2$$

for any $v_j \in \mathcal{V}$ and x_1 and $x_2 \in \mathbb{R}^p$, and with strong-convexity constant $\mu_j \ge 0$ and at least one $\mu_j > 0$.

A4. B-irreducible packet error probability sequence: Let $P(k)$ denote the time-varying adjacency matrix of an unreliable digraph at time k, where the link $\varepsilon_{ji} \in \mathcal{E}$ that corresponds to the matrix element $p_{ji}(k) \in [0,1]$, experiences errors with a fixed probability q_{ji}. For the sequence $\{P(k)\}$, we assume that all self-loops are absent, *i.e.,* $p_{jj}(k) = 0$ for all $v_j \in \mathcal{V}$. Additionally, for some constant $\epsilon > 0$, we assume that if $p_{ji}(k) \ne 0$, then $p_{ji}(k) \ge \epsilon$ for all $\varepsilon_{ji} \in \mathcal{E}$ and all $k \ge 0$. Under these conditions, the sequence $\{P(k)\}$ is B-irreducible, that is, for some integer $B > 0$, $\sum_{l=k}^{k+B-1} P(l)$ is irreducible for all $k \ge 0$.

Remark 1. *From Assumption A3 we can ensure that the problem in* (1) *has a unique optimal solution*

$$x^\star = \arg\min_{x \in \mathbb{R}^p} f(x).$$

III. Algorithm Development

At each iteration $k \ge 0$, each agent $v_j \in \mathcal{V}$ maintains a local state variable $x_j(k) \in \mathbb{R}$, an auxiliary variable $y_j(k) \in \mathbb{R}$ that asymptotically compensates for the directed network imbalance, and the gradient estimate variable $w_j(k)$. Initially, it sets a sufficiently small gradient step-size $\alpha > 0$ (to maintain the stability of the algorithm), and $x_j(0) \in \mathbb{R}$, $y_j(0) = 1$, $w_j(0) = \nabla f_j\big(x_j(0)\big)$. Subsequently, at each iteration $k \ge 0$ it executes the following processes:

Transmitting: Retransmits the packets, $x_{j,r}(k) - \alpha w_{j,r}(k)$, $y_{j,r}(k)$, and $w_{j,r}(k)$ all scaled by c_{lj}, to its out-neighbors $v_l \in \mathcal{N}_j^{\text{out}}$, for all r satisfying the condition in (4). The packet $x_{j,r}(k)$ corresponds to the state of agent v_j at time $k-r$, *i.e.,* $x_j(k-r)$, resp. for $y_{j,r}(k)$ and $w_{j,r}(k)$. The weight c_{lj} can be assigned by each node $v_j \in \mathcal{V}$, as:

$$c_{lj} = \begin{cases} \frac{1}{1+d_j^{\text{out}}}, & \text{if } v_l \in \mathcal{N}_j^{\text{out}} \text{ or } l = j, \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

Note that, collecting the weights c_{lj} for all $v_l, v_j \in \mathcal{V}$, one can form the matrix $C \in \mathbb{R}_+^{n \times n}$, which is column-stochastic by construction.

Receiving: Receives (possibly delayed) packets from its in-neighbors $v_i \in \mathcal{N}_j^{\text{in}}$. Upon the arrival of these packets, each node v_j processes them as follows:

$$x_j^\rho(k) = \sum_{i=1}^n \sum_{r=0}^{\bar{\tau}} c_{ji}\gamma_{ji,r}(k)\big(x_{i,r}(k) - \alpha w_{i,r}(k)\big), \tag{11a}$$

$$y_j^\rho(k) = \sum_{i=1}^n \sum_{r=0}^{\bar{\tau}} c_{ji}\gamma_{ji,r}(k)\big(y_{i,r}(k)\big), \tag{11b}$$

$$w_j^\rho(k) = \sum_{i=1}^n \sum_{r=0}^{\bar{\tau}} c_{ji}\gamma_{ji,r}(k)\big(w_{i,r}(k)\big). \tag{11c}$$

Updating: Upon the reception and process of the packets that arrived within time slot k, each node $v_j \in \mathcal{V}$ performs local variable update:

$$x_j(k+1) = x_j^\rho(k) + x_j^\delta, \tag{12a}$$

$$y_j(k+1) = y_j^\rho(k) + y_j^\delta, \tag{12b}$$

$$z_j(k+1) = x_j(k+1)/y_j(k+1), \tag{12c}$$

$$w_j(k+1) = w_j^\rho(k) + w_j^\delta + g_j(k+1), \tag{12d}$$

where $g_j(k+1) = \nabla f_j\big(z_j(k+1)\big) - \nabla f_j\big(z_j(k)\big)$, and

$$x_j^\delta(k+1) = \sum_{v_l \in \mathcal{N}_j^{\text{out}}} c_{lj}\delta_{lj}(k)\big(x_{j,\bar{\tau}}(k) - \alpha w_{j,\bar{\tau}}(k)\big), \tag{13a}$$

$$y_j^\delta(k+1) = \sum_{v_l \in \mathcal{N}_j^{\text{out}}} c_{lj}\delta_{lj}(k) y_{j,\bar{\tau}}(k), \tag{13b}$$

$$w_j^\delta(k+1) = \sum_{v_l \in \mathcal{N}_j^{\text{out}}} c_{lj}\delta_{lj}(k) w_{j,\bar{\tau}}(k), \tag{13c}$$

are the packets that are considered dropped, which are able to be recovered through the acknowledgement messages as in (5). The ratio variable $z_j(k+1)$ is used to evaluate the gradient $\nabla f_j\big(z_j(k+1)\big)$.

An overview of the ARQ-OPT algorithm executed at node v_j at each time step k, is summarized in Algorithm 1.

Algorithm 1 – ARQ-OPT at node v_j at time k

```
1:  Set fresh acknowledgement: $\gamma_{lj,-1}(k) = 0$ for all $v_l \in \mathcal{N}_j^{\text{out}}$
2:  Set fresh counter: $\rho_{lj,0}(k) = 0$ for all $v_l \in \mathcal{N}_j^{\text{out}}$
3:  Receive acknowledgements: $\gamma_{lj,r}(k-1)$
4:      from all $v_l \in \mathcal{N}_j^{\text{out}}$ and all $r \in \mathbb{Z}_1^{\bar{\tau}}$
5:  Update counters: $\rho_{lj,r}(k)$ via (3)
6:      for all $v_l \in \mathcal{N}_j^{\text{out}}$ and all $r \in \mathbb{Z}_1^{\bar{\tau}+1}$
7:  (Re)transmit packets: $c_{lj}\big(x_{j,r}(k) - \alpha w_{j,r}(k)\big)$, $c_{lj} y_{j,r}(k)$,
8:      and $c_{lj} w_{j,r}(k)$ to all $v_l \in \mathcal{N}_j^{\text{out}}$ according to condition (4)
9:  Process received packets: as in (11)
10: Send acknowledgements: $\gamma_{ji,r}(k)$ to all $v_i \in \mathcal{N}_j^{\text{in}}$ for $r \in \mathbb{Z}_0^{\bar{\tau}}$
11: Update variables: $x_j(k+1)$, $y_j(k+1)$, $z_j(k+1)$,
12:     and $w_j(k+1)$ via (12)
13: Output: $z_j(k+1)$
```

IV. Analysis

A. Digraph / Adjacency Matrix Augmentation

To model all the possible delayed and dropped transmissions, consider the following augmentation on the original graph $\mathcal{G}$, as constructed in [24]. For each node $v_j \in \mathcal{V}$, we add $\bar{\tau}$ extra virtual nodes, *i.e.,* $\breve{v}_j^{(1)}, \breve{v}_j^{(2)}, \ldots, \breve{v}_j^{(\bar{\tau})}$, that temporarily hold and propagate the delayed information that is destined to arrive at the actual node v_j after $r \in \mathbb{Z}_1^{\bar{\tau}}$ time steps; and $\bar{\tau}$ extra virtual nodes, *i.e.,* $\check{v}_j^{(1)}, \check{v}_j^{(2)}, \ldots, \check{v}_j^{(\bar{\tau})}$, that model the propagation of the information of the dropped packets back to their actual source node v_j. Hence, the augmented digraph $\tilde{\mathcal{G}} = (\tilde{\mathcal{V}}, \tilde{\mathcal{E}})$ that models the interactions using ARQ protocols, consists of $\tilde{n} = n(2\bar{\tau}+1)$ nodes.

Based on the augmented digraph, we stack the local agents' variables $x_j(k)$'s into the vector $x(k)$. Then, concatenating them with the virtual nodes' variables, we get

$$\tilde{x}(k) = \big(x(k)^\top, \breve{x}^\top(k), \check{x}^\top(k)\big)^\top \in \mathbb{R}^{\tilde{n}}, \tag{14a}$$

where $x(k)$, $\breve{x}(k)$, and $\check{x}(k)$ are defined as

$$x(k) = \big(x_1(k), \ldots, x_n(k)\big)^\top \in \mathbb{R}^n,$$
$$\breve{x}(k) = \Big(\breve{x}_1^{(1)}(k), \ldots, \breve{x}_n^{(1)}(k), \ldots, \breve{x}_1^{(\bar{\tau})}(k), \ldots, \breve{x}_n^{(\bar{\tau})}(k)\Big)^\top \in \mathbb{R}^{n\bar{\tau}},$$
$$\check{x}(k) = \Big(\check{x}_1^{(1)}(k), \ldots, \check{x}_n^{(1)}(k), \ldots, \check{x}_1^{(\bar{\tau})}(k), \ldots, \check{x}_n^{(\bar{\tau})}(k)\Big)^\top \in \mathbb{R}^{n\bar{\tau}}.$$

The same notation applies to the virtual nodes' variables for $y(k), z(k), w(k)$, and $g(k)$, to obtain $\tilde{y}(k)$, $\tilde{z}(k)$, $\tilde{w}(k)$, and $\tilde{g}(k)$, respectively, so we omit it for brevity. Note that, all variables corresponding to virtual nodes are initialized at 0 (*e.g.,* $\breve{x}_i^1 = 0$, $\breve{y}_i^1 = 0$, etc.).

Prior to introducing the algorithm in its vector-matrix form, we define the augmented nonnegative random matrix $\tilde{C}(k) \in \mathbb{R}_+^{\tilde{n}\times\tilde{n}}$ which is associated with the augmented digraph that models the exchange of information through the protocol described in Algorithm 1, and is given by:

$$\tilde{C}(k) \triangleq \begin{pmatrix} C^{(0)}(k) & I & 0 & \cdots & I & 0 & \cdots & 0 \\ C^{(1)}(k) & 0 & I & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ C^{(\bar{\tau})}(k) & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 & I & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & I \\ D^{(\bar{\tau})}(k) & 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \end{pmatrix}, \tag{15}$$

where each element of $C^{(r)}(k) \in \mathbb{R}_+^{n\times n}$, for $r \in \mathbb{Z}_0^{\bar{\tau}}$, is determined by

$$[C^{(r)}(k)]_{lj} = \begin{cases} c_{lj}, & \text{if } \tau_{lj}(k) = r,\ \varepsilon_{lj} \in \mathcal{E}, \\ 0, & \text{otherwise.} \end{cases} \tag{16}$$

In other words, if the index r that corresponds to the block matrix $C^{(r)}$ is equal to $\tau_{lj}(k)$, then the augmented link $\tilde{\varepsilon}_{lj} \in \tilde{\mathcal{E}}$ will be weighted by the actual (original) weight c_{lj}; otherwise, its weight will be zero. Matrix $D^{(\bar{\tau})}(k) \in \mathbb{R}_+^{n\times n}$ is a diagonal matrix that models the activation of the packet drop mechanism for packets that exceeded the retransmission limit. Each diagonal element $[D^{(\bar{\tau})}(k)]_{jj}$ corresponds to the buffer of node v_j and is given by

$$[D^{(\bar{\tau})}(k)]_{jj} = \sum_{v_l \in \mathcal{N}_j^{\text{out}}} c_{jj}\delta_{lj}(k), \quad \text{for all } v_j \in \mathcal{V}. \tag{17}$$

Block matrix $D^{(\bar{\tau})}(k)$ activates the virtual links that model the release the information of dropped packets back to their source transmitting nodes via the corresponding virtual nodes of the augmented matrix $\tilde{C}(k)$. Based on the structure of $\tilde{C}(k)$, it is clear to see that column-stochasticity is preserved although the communication links between nodes might be unreliable.

Now by letting $Y(k+1) = \operatorname{diag}\big(\tilde{y}(k+1)\big)$ and $H = \operatorname{diag}\big([\mathbf{1}_n\ \mathbf{0}_{\tilde{n}-n}]\big)$ we are in place to rewrite the algorithm in its vector-matrix form as:

$$\tilde{x}(k+1) = \tilde{C}(k)\big(\tilde{x}(k) - \alpha H\tilde{w}(k)\big), \tag{18a}$$
$$\tilde{y}(k+1) = \tilde{C}(k)\tilde{y}(k), \tag{18b}$$
$$\tilde{z}(k+1) = Y(k+1)^{-1}\tilde{x}(k+1), \tag{18c}$$
$$\tilde{w}(k+1) = \tilde{C}(k)\tilde{w}(k) + \tilde{g}(k+1). \tag{18d}$$

B. Convergence Analysis

Before introducing the main results, consider the following equivalent representation of the aforementioned system, as formulated in [25], by writing the local ratio $z_i(k)$, in terms of $x_i(k)$ and $y_i(k)$, *i.e.,*

$$\begin{aligned} z_i(k+1) &= \sum_{v_j \in \mathcal{N}_i^{\text{in}}} \left(\frac{c_{ij}x_j(k) - \alpha c_{ij}w_j(k)}{\sum_{v_l \in \mathcal{N}_i^{\text{in}}} c_{il}y_l(k)} \right) \\ &= \sum_{v_j \in \mathcal{N}_i^{\text{in}}} \left(\frac{c_{ij}y_j(k)}{\sum_{v_l \in \mathcal{N}_i^{\text{in}}} c_{il}y_l(k)} \right) \left(z_j(k) - \alpha\frac{w_j(k)}{y_j(k)} \right) \\ &= \sum_{v_j \in \mathcal{N}_i^{\text{in}}} r_{ij}(k) \left(z_j(k) - \alpha\frac{w_j(k)}{y_j(k)} \right), \end{aligned} \tag{19}$$

where

$$r_{ij}(k) = \frac{c_{ij}y_j(k)}{\sum_{v_l \in \mathcal{N}_i^{\text{in}}(k)} c_{il}y_l(k)} \tag{20}$$

is the element in the i-th row and j-th column of a row-stochastic matrix $R(k)$. In the case of packet errors, we can obtain an augmented matrix $\tilde{R}(k)$ (which preserves row-stochasticity) by simply replacing the actual weights C with the augmented weights $\tilde{C}(k)$, and the vector $y(k)$ with the augmented vector $\tilde{y}(k)$. Hence, for analysis, one can rewrite the vector-matrix form by utilizing the augmented row-stochastic matrix $\tilde{R}(k)$, of which the elements that are involved in the update of the ratio are independent of $\tilde{x}(k)$ but they are instead dependent on $\tilde{y}(k)$, into the following equivalent form:

$$\tilde{z}(k+1) = \tilde{R}(k)\big(\tilde{z}(k) - \alpha H\tilde{w}(k)\big), \tag{21a}$$
$$\tilde{w}(k+1) = \tilde{C}(k)\tilde{w}(k) + \tilde{g}(k+1). \tag{21b}$$

Now, performing a state transformation from $\tilde{w}(k)$ to $\tilde{u}(k) = \big(\tilde{Y}(k)\big)^{-1}\tilde{w}(k)$, where $\tilde{Y}(k) := \operatorname{diag}(\tilde{y}(k))$ with $\tilde{y}(k+$

$1) = \tilde{C}(k)\tilde{y}(k)$, we can write ARQ-OPT in the following equivalent form

$$\tilde{y}(k+1) = \tilde{C}(k)\tilde{y}(k), \tag{22a}$$

$$\tilde{z}(k+1) = \tilde{R}(k)\big(\tilde{z}(k) - \alpha H\tilde{Y}(k)\tilde{u}(k)\big), \tag{22b}$$

$$\tilde{u}(k+1) = \tilde{R}(k)\tilde{u}(k) + \big(\tilde{Y}(k+1)\big)^{-1}\tilde{g}(k+1), \tag{22c}$$

where $\tilde{R}(k) = \big(\tilde{Y}(k+1)\big)^{-1}\tilde{C}(k)\tilde{Y}(k)$. Having this form at hand, we can proceed with the convergence analysis of the ARQ-OPT algorithm using the form in (22), as in [16], [18]. First, we define the average of the iterate $\tilde{z}(k)$ weighted by the absolute probability sequence of the matrix $\tilde{R}(k)$, as

$$\hat{z}(k) := \sum_{i=1}^{\tilde{n}} \Big[\pi_R(k)\Big]_i \tilde{z}_i(k) = \pi_R^\top(k)\tilde{z}(k). \tag{23}$$

The following lemma shows that there exist absolute probability sequences for the matrices $\tilde{R}(k)$ and $\tilde{C}(k)$.

Lemma 1. *[19, Lemma 3.3 & 3.4] Let Assumptions A1 and A3 hold, and let $\{\tilde{R}(k)\}$ and $\{\tilde{C}(k)\}$ be row-stochastic and column-stochastic matrix sequences, respectively, that are compatible with the augmented digraph $\tilde{\mathcal{G}}(k)$ corresponding to information exchange over the ARQ message exchange mechanism. Then, there exist sequences $\{\pi_R(k)\}$ and $\{\pi_C(k)\}$ of stochastic vectors such that*

$$\pi_R(k) = \pi_R(k+1)\tilde{R}(k), \text{ for all } k \geq 0, \tag{24}$$

$$\pi_C(k+1) = \tilde{C}(k)\pi_C(k), \text{ for all } k \geq 0, \tag{25}$$

with $\pi_C(0) = \frac{1}{\tilde{n}}\mathbf{1}$, and where their elements are positive and bounded from below by $r_{\min} \triangleq \min_{i,j\in\mathcal{V}}\{r_{ij}(k)\}$ and $c_{\min} \triangleq \min_{i,j\in\mathcal{V}}\{c_{ij}(k)\}$, respectively, as

$$\Big[\pi_R(k)\Big]_i \geq \frac{r_{\min}^n}{n}, \ \Big[\pi_C(k)\Big]_i \geq \frac{c_{\min}^n}{n}, \ \text{ for all } v_i \in \tilde{\mathcal{V}}. \tag{26}$$

Proof. The proof is similar to [19, Lemma 3.4] □

Essentially, we would like to show that ARQ-OPT converges to the optimal solution as $k \to \infty$. Note that, the entries of the mixing matrices $\tilde{C}(k)$ and $\tilde{R}(k)$ depend on the packet errors, then they are not guaranteed to be contractive within the time window B. In the sequel we will show that one can properly choose B (*i.e.,* based on the maximum number of retransmission attempts and the channel conditions), such that B-step consensus contractivity holds with high probability. Putting this aside, we proceed with Lemma 2 which states that there exists a time window $B > 0$ over which the state variables of the agents contract.

Lemma 2. *[16, Lemma 13] Let $B \geq \tilde{B}$ and such that $\psi \triangleq Q_1(1-\chi^{\tilde{n}\tilde{B}})^{\frac{B-1}{\tilde{n}\tilde{B}}} < 1$, where $Q_1 \triangleq 2\tilde{n}\frac{1+\chi^{-\tilde{n}\tilde{B}}}{1-\chi^{\tilde{n}\tilde{B}}}$ and $\chi \triangleq \frac{1}{\tilde{n}^{2+\tilde{n}\tilde{B}}}$. Then for any $k \geq B-1$ and any vector $\upsilon \in \mathbb{R}^{\tilde{n}}$, if $\varpi = \tilde{R}(k)\tilde{R}(k-1)\ldots\tilde{R}(k-B+1)\upsilon$, then $\|\varpi\| \leq \psi\|\upsilon\|$.*

We now state the convergence result of ARQ-OPT. For brevity, we provide a sketch of the proof. A detailed proof of the theorem along with the supporting lemmas will be presented in an extended version of this work.

Theorem 1. *Suppose that Assumptions A1-A4 hold, then for a sufficiently small step-size α, the sequence $\{z_j(k)\}$ generated by each agent $v_j \in \mathcal{V}$ executing the ARQ-OPT, converges to the unique optimal solution x^* of the problem in (1) as $k \to \infty$, with probability one.*

Sketch of the proof of Theorem 1:
First consider the quantities $\mathbb{E}\Big[\Big\|\tilde{z}(k) - [x^*\mathbf{1}^\top \ \mathbf{0}^\top]^\top\Big\|\Big]$, $\mathbb{E}\Big[\Big\|\tilde{g}(k)\Big\|\Big]$, $\mathbb{E}\Big[\Big\|\tilde{u}(k) - \mathbf{1}\tilde{y}^\top(k)\tilde{u}(k)\Big\|\Big]$, and $\mathbb{E}\Big[\Big\|\tilde{z}(k) - \mathbf{1}_{\tilde{n}}\hat{z}(k)\Big\|\Big]$, where $\mathbb{E}[\cdot]$ denotes the expected value. To show that these quantities converge to zero, we first need to find an appropriate integer $B > 0$ such that Lemma 2 can be applied. In fact, one should choose B such that the probability that the resulting digraph over B iterations, *i.e.,* $\cup_{l=k}^{k+B-1}\mathcal{G}_l$ for all $k \geq 0$, is strongly connected, is ≥ 0.5. Then we are able to apply Lemma 2 along with the adapted *small gain theorem* in [16] to show that the aforementioned quantities converge to zero. Finally, applying Markov inequality and Borel-Cantelli lemma we are able to show as $k \to \infty$, $z_j(k)$ for all $v_j \in \mathcal{V}$ converges to the unique optimal solution x^*, and the theorem follows.

V. Simulation Results

Consider a network comprised of $n = 10$ agents communicating between each other over directed and unreliable communication links as shown in Fig. 1. In this example, the

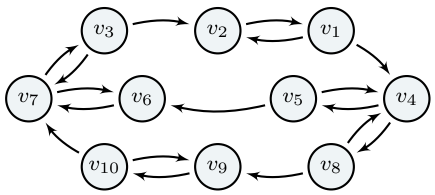

Fig. 1. Digraph comprising of $n = 10$ agents. The digraph incorporates self-loops, yet they are omitted for presentation simplicity.

goal of the agents is to minimize a global cost function for the distributed logistic regression problem,

$$\min_{x\in\mathbb{R}^p} \sum_{i=1}^{10} \frac{1}{\phi_i} \sum_{\ell=1}^{\phi_i} \log(1+\exp(-(\Phi_{i\ell}x)\theta_{i\ell})), \tag{27}$$

where each agent $v_i \in \mathcal{V}$ has its training data $(\Phi_{i\ell}, \theta_{i\ell}) \in \mathbb{R}^p \times \{-1, +1\}$, $\ell = 1, 2, \ldots, \phi_i$, where $\Phi_{i\ell}$ and $\theta_{i\ell}$ denote the feature variable, and binary outcome of agent v_i for the training sample ℓ, respectively. In our example, we assume $p = 1$ feature, and $\phi_i = 150$ data samples, and we generate artificial data of feature vectors $\Phi_{i\ell}$ with outcome $y_{i\ell} = \{-1, 1\}$ from normal distributions with zero mean and standard deviation 0.707 for $y_{i\ell} = -1$ and 1 for $y_{i\ell} = 1$. Each agent v_j chooses the weights of its out-going links, as defined in (10), and executes the ARQ-OPT algorithm. The agents initiate their iterations at $x(0) = [1 \ 2 \ \cdots \ 10]^\top$, $y(0) = [1 \ \cdots \ 1]^\top$ and they set their step-size to $\alpha = 0.01$. The probability that a packet is erroneous is $q \in \{0, 0.3, 0.7\}$ and is assumed to be independent of previous retransmission trials. The retransmission limit is $\bar{\tau} \in \{0, 2, 5\}$, and for simplicity is assumed to be identical for all agents.

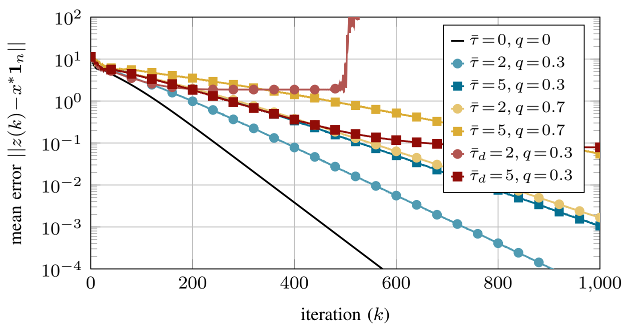

Fig. 2. Mean error $||z(k)-x^{*}\mathbf{1}_n||$ with step-size $\alpha = 0.01$. The upper bound $\bar{\tau}_d \in \{2,5\}$ represents cases where dropped information is discarded.

Fig. 2 depicts the network mean error $\|z(k) - x^{*}\mathbf{1}_n\|$ obtained from 20 Monte Carlo simulations of 1000 iterations, for each packet error probability and retransmission limit configuration. To emphasize the instability issues when the dropped information is discarded, we plot the mean error for $q = 0.3$ and $\bar{\tau}_d \in \{2, 5\}$. As illustrated in Fig. 2, with a higher probability of packet errors and retransmission limit, ARQ-OPT suffers from slow convergence. This is due to the retransmission of packets which introduces information delays in exchange for the convergence to the optimal solution. Conversely, when agents discard dropped packets that exceed the retransmission limit, as observed in the cases of $\bar{\tau}_d = 2$ and $\bar{\tau}_d = 5$ in Fig. 2, they fail to converge.

VI. Conclusions

In this paper, we proposed the ARQ-OPT algorithm to solve distributed optimization problems in error-prone directed networks, by harnessing packet retransmissions to ensure reliable propagation of information across the network. By analyzing its convergence properties and demonstrating its ability to achieve B-step consensus contractivity, we have established that ARQ-OPT can achieve asymptotic convergence to the optimal solution with probability one. Through numerical simulations conducted under various channel conditions, we have validated the performance of ARQ-OPT, confirming its resilience to packet errors.

References

[1] H. Jaleel and J. S. Shamma, "Distributed Optimization for Robot Networks: From Real-Time Convex Optimization to Game-Theoretic Self-Organization," *Proc. of the IEEE*, vol. 108, no. 11, pp. 1953–1967, 2020.

[2] O. Shorinwa, T. Halsted, J. Yu, and M. Schwager, "Distributed Optimization Methods for Multi-Robot Systems: Part 2—A Survey," *IEEE Robot. Autom. Mag.*, vol. 31, no. 3, pp. 154–169, 2024.

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

[4] S. Safavi, U. A. Khan, S. Kar, and J. M. Moura, "Distributed Localization: A Linear Theory," *Proc. of the IEEE*, vol. 106, no. 7, pp. 1204–1223, 2018.

[5] J. Tsitsiklis, D. Bertsekas, and M. Athans, "Distributed Asynchronous Deterministic and Stochastic Gradient Optimization Algorithms," *IEEE Trans. Autom. Control*, vol. 31, no. 9, pp. 803–812, 1986.

[6] A. Nedic and A. Ozdaglar, "Distributed Subgradient Methods for Multi-Agent Optimization," *IEEE Trans. Autom. Control*, vol. 54, no. 1, pp. 48–61, 2009.

[7] J. C. Duchi, A. Agarwal, and M. J. Wainwright, "Dual Averaging for Distributed Optimization: Convergence Analysis and Network Scaling," *IEEE Trans. Autom. Control*, vol. 57, no. 3, pp. 592–606, 2011.

[8] K. I. Tsianos, S. Lawlor, and M. G. Rabbat, "Push-Sum Distributed Dual Averaging for Convex Optimization," in *IEEE Conf. Decision Control*, 2012, pp. 5453–5458.

[9] A. Nedić and A. Olshevsky, "Distributed Optimization over Time-varying Directed Graphs," *IEEE Trans. Autom. Control*, vol. 60, no. 3, pp. 601–615, 2014.

[10] C. Xi, Q. Wu, and U. A. Khan, "On the Distributed Optimization over Directed Networks," *Neurocomputing*, vol. 267, pp. 508–515, 2017.

[11] K. Cai and H. Ishii, "Average Consensus on General Strongly Connected Digraphs," *Automatica*, vol. 48, no. 11, pp. 2750–2761, 2012.

[12] A. D. Domínguez-García and C. N. Hadjicostis, "Distributed Strategies for Average Consensus in Directed Graphs," in *IEEE Conf. Decision Control*, 2011, pp. 2124–2129.

[13] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-Based Computation of Aggregate Information," in *IEEE Symp. Found. Comput. Sci.*, 2003, pp. 482–491.

[14] C. Xi and U. A. Khan, "DEXTRA: A Fast Algorithm for Optimization over Directed Graphs," *IEEE Trans. Autom. Control*, vol. 62, no. 10, pp. 4980–4993, 2017.

[15] C. Xi, R. Xin, and U. A. Khan, "ADD-OPT: Accelerated Distributed Directed Optimization," *IEEE Trans. Autom. Control*, vol. 63, no. 5, pp. 1329–1339, 2017.

[16] A. Nedic, A. Olshevsky, and W. Shi, "Achieving Geometric Convergence for Distributed Optimization over Time-Varying Graphs," *SIAM J. Optim.*, vol. 27, no. 4, pp. 2597–2633, 2017.

[17] R. Xin and U. A. Khan, "A Linear Algorithm for Optimization over Directed Graphs with Geometric Convergence," *IEEE Control Syst. Lett.*, vol. 2, no. 3, pp. 315–320, 2018.

[18] F. Saadatniaki, R. Xin, and U. A. Khan, "Decentralized Optimization over Time-Varying Directed Graphs with Row and Column-Stochastic Matrices," *IEEE Trans. Autom. Control*, vol. 65, no. 11, pp. 4769–4780, 2020.

[19] A. Nedić, D. T. A. Nguyen, and D. T. Nguyen, "AB/Push-Pull Method for Distributed Optimization in Time-Varying Directed Networks," *Optim. Methods Softw.*, pp. 1–28, 2023.

[20] K. I. Tsianos and M. G. Rabbat, "Distributed Consensus and Optimization under Communication Delays," in *Proc. Allerton Conf. Commun., Control, Comput.*, 2011, pp. 974–982.

[21] A. Spiridonoff, A. Olshevsky, and I. C. Paschalidis, "Robust Asynchronous Stochastic Gradient-Push: Asymptotically Optimal and Network-Independent Performance for Strongly Convex Functions," *J. Mach. Learn. Res.*, vol. 21, no. 58, pp. 1–47, 2020.

[22] E. Makridis, G. Oliva, K. R. Narahari, M. Doostmohammadian, U. A. Khan, and T. Charalambous, "Distributed Optimization with Gradient Tracking Over Heterogeneous Delay-Prone Directed Networks," in *Europ. Control Conf.*, 2024, pp. 2312–2319.

[23] S. Lin, D. J. Costello, and M. J. Miller, "Automatic-Repeat-Request Error-Control Schemes," *IEEE Commun. Mag.*, vol. 22, no. 12, pp. 5–17, 1984.

[24] E. Makridis, T. Charalambous, and C. N. Hadjicostis, "ARQ-based Average Consensus over Unreliable Directed Network Topologies," *arXiv preprint arXiv:2209.14699*, 2022.

[25] Y. Lin and J. Liu, "Subgradient-Push is of the Optimal Convergence Rate," in *IEEE Conf. Decision Control*, 2022, pp. 5849–5856.