

ARQ-based Average Consensus over Unreliable Directed Network Topologies

Evagoras Makridis, Themistoklis Charalambous, and Christoforos N. Hadjicostis

Abstract—In this paper, we address the problem of discrete-time average consensus, where agents (nodes) exchange information over unreliable communication links. We enhance the Robustified Ratio Consensus algorithm by embedding the Automatic Repeat ReQuest (ARQ) protocol used for error control of data transmissions, in order to allow the agents to reach asymptotic average consensus even when operating within unreliable directed networks. This strategy, apart from handling time-varying delays induced by retransmissions of erroneous packets (that can be captured by the Robustified Ratio Consensus as well), it is also possible to handle packet drops that occur due to excess of a predefined packet retransmission limit imposed by the ARQ protocol. Invoking the ARQ protocol allows nodes to: (a) exploit the incoming error-free acknowledgement feedback signals to initially acquire or later update their out-degree, (b) know whether a packet has arrived or not, and (c) determine a local upper-bound on the delays which is imposed by the retransmission limit. By augmenting the network's corresponding weighted adjacency matrix, to handle time-varying (yet bounded) delays and possible packet drops, we show that nodes can make use of the proposed algorithm, herein called the ARQ-based Ratio Consensus algorithm, to reach asymptotic average consensus, despite the fact that the communication links are unreliable. To the best of the authors' knowledge, this is the first consensus algorithm that incorporates a communication protocol for error control used in real communication systems with feedback.

Index Terms—average consensus, digraphs, unreliable communication, retransmissions, delays, packet drops, ARQ protocol, ratio consensus.

I. INTRODUCTION

Robust information exchange between computing nodes (usually referred as agents or workers) is essential to apply various estimation and control algorithms in spatially distributed multi-agent systems [1]–[6]. Such distributed systems involve multiple nodes where each individual node has no knowledge of the entire system, but instead it only has some local information. Hence, to achieve certain network-wide (global) objectives, each node needs to either interact with a central master node (in a centralized master-worker architecture), or with its immediate neighbors (in a peer-to-peer decentralized architecture) via a communication network. Centralized topologies involve a master node collecting information

from all other nodes to compute a global decision that achieves a network-wide objective. Decentralized topologies rely on nodes interacting with each other (without the need of a central coordinator) such that a global decision is obtained by combining local information (see [7] for a short review on network topology architectures for distributed computation). In practice, both communication network topologies pose some limitations and constraints on the information flow between nodes, which arise mainly due to network congestion and unreliable communication channels. Such impediments induce transmission delays and packet drops, affecting the performance and reliability of distributed computation algorithms. These issues are even more obvious in centralized network topologies, since the master node in large-scale centralized topologies may become a communication bottleneck, especially when the network resources are fixed and limited, because it needs to maintain communication with all worker nodes in the network [8]. Besides the communication bottleneck at the master node, centralized networks are vulnerable to single points of failure since a potential failure of the master node could lead to the failure of the entire network [7].

Distributed decision-making methods through peer-to-peer network topologies, often rely on nodes interacting with each other to compute a global value (e.g., the average) of a certain common quantity such that a global objective is achieved. The problem where a network of nodes maintain local information which is exchanged with neighboring nodes to compute their average value is called (distributed) average consensus (see [9] for an overview of consensus methods). Under specific conditions on the network topology and the interaction between the nodes, an accurate computation of the average value (i.e., the network-wide average of the agents' initial values) can be obtained, either asymptotically [10]–[12], or in finite time [13]–[15]. Several practical examples limit the exchange of information from bidirectional to directional mainly due to diverse transmission power and interference levels of each individual node in the network. The information flow in such a paradigm is directed meaning that an agent v_j may be able to receive information from another agent v_i , but this does not necessarily imply that v_j is able to send information to v_i . In this configuration, the network topology can be modeled using directed graphs (*digraphs*), where the agents are represented by the graph's nodes, and the communication links that enable information exchange between agents are represented by the graph's edges.

For directed network topologies, the results in [12],

E. Makridis is with Department of Electrical and Computer Engineering, University of Cyprus, Nicosia, Cyprus (e-mail: makridis.evagoras@ucy.ac.cy).

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

C. N. Hadjicostis is with Department of Electrical and Computer Engineering, University of Cyprus, Nicosia, Cyprus, (e-mail: chadjic@ucy.ac.cy)

[16], [17] have shown that agents can asymptotically reach the average. In the presence of computational and/or transmission delays on the information exchange between agents, the authors in [15], [18] proposed *ratio-consensus*-based algorithms, able to reach (asymptotic in the former, finite-time in the latter) average consensus in directed communication networks in the presence of bounded time-invariant and time-varying delays. The ratio consensus algorithm proposed in [19], proved to be able to reach average consensus by computing in an iterative way the ratio of two concurrently running linear iterations with certain initial conditions that are properly specified. An extension of the ratio consensus algorithm is proposed in [20], where nodes exchange messages containing information of their running sums to handle potential loss of packets. They proved, by introducing extra virtual nodes and links describing the aforementioned communication scenarios and topology, that network nodes asymptotically converge to the exact average. For similar scenarios, the authors in [21] proposed a corrective consensus algorithm that enables the practical use of consensus by each node through extra variables, and extra corrective iterations, to prove almost sure convergence to the exact average in the presence of link losses.

In this paper, we cast the problem of discrete-time average consensus over possibly directed network topologies into the framework of the Automatic Repeat reQuest (ARQ) error correction protocol, which is the fundamental block for reliable communication in practical systems. Invoking the ARQ protocol allows nodes to exploit incoming error-free acknowledgement feedback signals to initially acquire or later update their out-degree, know whether a packet has arrived or not, and determine a local upper-bound on the delays, imposed by the ARQ retransmission limit. To the best of our knowledge, this is the first time that a realistic communication error correction protocol currently used in many real-life telecommunication systems, is integrated with distributed consensus iterations to compute the exact average consensus value over unreliable networks. Finally, we provide the proof of asymptotic convergence of the proposed method to the exact average and we evaluate its performance via simulations for different realistic scenarios. The proof is based on augmenting the weighted adjacency matrix that represents the network topology and the interactions between nodes, to model possible arbitrary time-varying (yet bounded) delays and packet drops on the communication links.

The remainder of this paper is organized as follows. Section II introduces the network model, the ARQ error correction protocol, the basic Ratio Consensus algorithm, and the Robustified Ratio Consensus algorithm for directed graphs. Section III describes the ARQ-based Ratio Consensus method for handling unreliable communication such as delays and packet-drops due to retransmissions and excess of a predefined retransmissions limit, respectively. In Section IV we present the augmented digraph representation needed

for the proof of convergence of the proposed method, while in Section V we provide numerical simulations for different packet error probabilities. Finally, in Section VI we provide the concluding remarks and present future directions.

II. BACKGROUND

A. Network Model

Consider n agents (represented by graph's nodes) communicating over a strongly connected network, $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, \dots, v_n\}$ is the set of nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges (representing the communication links between agents). A directed edge $\varepsilon_{ji} \triangleq (v_j, v_i) \in \mathcal{E}$, where $v_j, v_i \in \mathcal{V}$, represent that node v_j can receive information from node v_i , i.e., $v_i \rightarrow v_j$. The nodes that transmit information to node v_j directly are called in-neighbors of node v_j , and belong to the set $\mathcal{N}_j^- = \{u_i \in \mathcal{V} | \varepsilon_{ji} \in \mathcal{E}\}$. The number of nodes in the in-neighborhood is called in-degree and it is represented by the cardinality of the set of in-neighbors, $d_j^- = |\mathcal{N}_j^-|$. The nodes that receive information from node v_j directly are called out-neighbors of node v_j , and belong to the set $\mathcal{N}_j^+ = \{u_l \in \mathcal{V} | \varepsilon_{lj} \in \mathcal{E}\}$. The number of nodes in the out-neighborhood is called out-degree and it is represented by the cardinality of the set of out-neighbors, $d_j^+ = |\mathcal{N}_j^+|$. Note that self-loops are allowed in digraph \mathcal{G} and this implies that the number of in-going links of node v_j are $(d_j^- + 1)$ and similarly its out-going links are $(d_j^+ + 1)$.

B. Ratio Consensus over Directed Graphs

The problem of average consensus involves a number of nodes in a network represented by the digraph \mathcal{G} , that cooperate by exchanging information to compute the network-wide average of their initial values for a certain quantity of interest. Each node v_j maintains two state variables x_k^j (initialized at $x_0^j = V^j$, where V^j is an arbitrary value of node v_j), $z_k^j \in \mathbb{R}$, and an auxiliary scalar variable, y_k^j (initialized at $y_0^j = 1$), at each time step k . The iterative scheme of the ratio consensus algorithm for updating the variables such that average consensus is achieved, is given by:

$$x_{k+1}^j = p_{jj}x_k^j + \sum_{v_i \in \mathcal{N}_j^-} p_{ji}x_k^i, \quad (1a)$$

$$y_{k+1}^j = p_{jj}y_k^j + \sum_{v_i \in \mathcal{N}_j^-} p_{ji}y_k^i, \quad (1b)$$

$$z_{k+1}^j = \frac{x_{k+1}^j}{y_{k+1}^j}, \quad (1c)$$

where the collection of weights $P = \{p_{ji}\} \in \mathbb{R}_+^{n \times n}$ representing the interactions between nodes, forms a column-stochastic matrix. Often, the weight p_{lj} is assigned using the following mechanism:

$$p_{lj} = \begin{cases} \frac{1}{1+d_j^+}, & v_l \in \mathcal{N}_j^+ \cup \{v_j\}, \\ 0, & \text{otherwise.} \end{cases} \quad \forall j. \quad (2)$$

Since each node v_j assigns the weights according to (2), it is required that the nodes have the knowledge of their out-degree. As soon as the weights are assigned by all nodes, the nonnegative weighted column-stochastic adjacency matrix P is formed and (possible) zero-valued entries denote the absence of communication links (edges) between the corresponding nodes in the digraph.

The auxiliary scalar variable y_k is used to asymptotically compute the right Perron eigenvector of P which is not equal to $\mathbf{1}_n$ since P is not row-stochastic [22]. Forcing the auxiliary variable to be initialized at value 1 for each node, we can verify that $\lim_{k \rightarrow \infty} y_k^j = n[\pi_c]^j$ and that $\lim_{k \rightarrow \infty} x_k^j = [\pi_c]^j \sum_{i=1}^n x_0^i$, where π_c is the right eigenvector of P . Hence, the limit of the ratio x_k^j over y_k^j , is the average of the initial values and is given by [19], [23]:

$$\lim_{k \rightarrow \infty} z_k^j = \lim_{k \rightarrow \infty} \frac{x_k^j}{y_k^j} = \frac{[\pi_c]^j \sum_{i=1}^n x_0^i}{n[\pi_c]^j} = \frac{1}{n} \sum_{i=1}^n x_0^i. \quad (3)$$

Remark 1. The Ratio Consensus algorithm assumes that communication links within the network are perfectly reliable, and that each node v_j is aware of its out-degree d_j^+ .

C. Robustified Ratio Consensus over Directed Graphs

In practice, packet transmissions are often delayed due to bad channel conditions and network congestion. To overcome this issue, the authors in [18] proposed a protocol that handles time-varying delays to ensure asymptotic consensus to the exact average, despite the presence of time-varying (yet bounded) delays in the communication links. To get an intuition of their proposed protocol, consider that node v_j at time step k undergoes an *a priori* unknown delay denoted by a bounded positive integer $\tau_k^{ji} \leq \bar{\tau}^{ji} < \infty$. The maximum delay in the network is denoted by $\bar{\tau} = \max\{\bar{\tau}^{ji}\}$. Moreover, the own value of node v_j is always instantly available without delay, i.e., $\tau_k^{jj} = 0, \forall k$. Based on this notation, their proposed strategy involves each node updating its states for each iteration according to:

$$x_{k+1}^j = p_{jj}x_k^j + \sum_{v_i \in \mathcal{N}_j^-} \sum_{r=0}^{\bar{\tau}} p_{ji}x_{k-r}^i \iota_{k-r}^{ji}, \quad (4a)$$

$$y_{k+1}^j = p_{jj}y_k^j + \sum_{v_i \in \mathcal{N}_j^-} \sum_{r=0}^{\bar{\tau}} p_{ji}y_{k-r}^i \iota_{k-r}^{ji}, \quad (4b)$$

$$z_{k+1}^j = \frac{x_{k+1}^j}{y_{k+1}^j}, \quad (4c)$$

where ι_{k-r}^{ji} is an indicator function that captures the bounded delay $\tau_k^{ji} \leq \bar{\tau}^{ji}$ on link ε^{ji} at iteration k , defined as

$$\iota_{k-r}^{ji} = \begin{cases} 1, & \text{if } \tau_k^{ji} = r, \\ 0, & \text{otherwise.} \end{cases} \quad (5)$$

Here it is important to note that, a transmission on link ε^{ji} undergoes an *a priori* unknown delay, for which node v_j is unaware of, but instead, it processes (delayed) packets as soon

as they arrive successfully. Clearly, in the absence of delays, this strategy reduces to the Ratio Consensus algorithm in (1).

Remark 2. The Robustified Ratio Consensus algorithm can handle (possibly) delayed information exchange between nodes, neglecting possible packet drops due to erroneous packets. Moreover, it assumes that each node v_j is aware of its out-degree d_j^+ .

D. Error Correction Protocol - Automatic Repeat reQuest (ARQ)

Several message transmission protocols such as the Transmission Control Protocol (TCP), the High-Level Data Link protocol, and others, utilize Automatic Repeat reQuest (ARQ), an error control protocol for data transmission used to maintain reliable packet transmissions over unreliable communication [24], [25, §6], [26, §5]. ARQ uses error-detection codes, acknowledgment (ACK) or negative acknowledgment (NACK) messages, and timeouts to maintain the reliability of data transmissions. An acknowledgment is a feedback signal sent by the data receiver to notify the data transmitter whether a packet has been successfully received or not. Here it is important to note that the ARQ feedback is sent over a narrowband error-free (with negligible probability of error in the reception of this packet, which is usually of 1 bit) feedback channel that is not used for data transmission, hence the network topology could still be assumed directed, as discussed further in the subsequent section. The ARQ feedback procedure is repeated until a packet is successfully received (without errors) or dropped due to excess of a predefined limit of retransmissions. In other words, the receiver has up to a predefined number of retransmission trials to receive the data packet correctly, otherwise the packet is dropped. A retransmission requested by data receivers introduces a delay of one time slot, since the receiver will get the intended packet as soon as the transmission is successful.

III. ARQ-BASED RATIO CONSENSUS

In this section we propose a distributed Ratio Consensus method, herein called the ARQ-based Ratio Consensus algorithm, to ensure asymptotic convergence to the network-wide average value, based on the ARQ protocol for reliable information exchange (described in §II-D). Under the consensus framework, each node acts as a data transmitter (receiver) when transmitting (receiving) data to (from) its out-neighbors (in-neighbors) utilizing the ARQ protocol described. In particular, at each time slot, all nodes send their values to their adjacent neighbors, and they receive an ACK as soon as the data packets containing the nodes' values are correctly decoded. In contrast, a NACK is fed back to the nodes transmitting the data packets, if the receiving node failed to decode the data packet. Unsuccessful decoding or reception of a data packet can be modeled as transmission delay in the communication link, while possible excess of the ARQ retransmission limit is modeled as a packet loss for the corresponding message.

Each node assigns its self-weight and the weights of its out-neighbors using (2). Here, it is important to mention that each node can acquire its out-degree by summing the number of its incoming ARQ feedback signals (ACK/NACK) which indicates the number of the out-neighbors. This can be established at the initialization, by having nodes broadcasting a dummy packet in order to identify their out-neighbors. As aforementioned, although one could argue that the use of ARQ feedback contradicts the assumed directed network topology, this is not the case because it does not imply bidirectional (undirected) communication, since the ARQ feedback signal is transmitted over a narrowband feedback channel, which cannot support data transmission due to the high data rate required and, hence, larger bandwidth and higher-order modulation. As soon as the weights are assigned, each node updates its own consensus variables using received information from its in-neighbors.

A. Handling Erroneous Transmissions (delays) and Packet Drops

Recall that a negative acknowledgement (NACK), due to erroneous packet reception of information sent by the receiving node v_j , implies that the transmitting node v_i should retransmit the same information in the next time slot, unless the retransmission limit, $\bar{\tau}^{ji}$, has been exceeded. For simplicity of exposition, it is assumed that the communication link from v_i to v_j is stationary. Hence, the probability that a transmitted packet contains an error (detected by the receiver) is constant and it is denoted by q^{ji} . Recall that, a detected error corresponds to a NACK that is sent by the receiver to the transmitter to request a retransmission. In such a case, a delay of one time-step is induced, as shown in Fig 1. For a packet that was initially transmitted at time k on link ε^{ji} , the delay of information is denoted by τ_k^{ji} . During the last trial of a packet transmission (when the maximum retransmission limit has been exceeded, i.e., $\tau_k^{ji} = \bar{\tau}^{ji}$), the transmitting node receives the last ARQ feedback for this particular packet from the receiving node. If the ARQ feedback is NACK (error detected in the receiving packet), then the packet is dropped. Otherwise, the packet is received successfully at the receiving node. Recall that the value of node v_j is always available to itself (self-loop) without delay, $\tau_k^{jj} = 0$. Since all nodes in the network utilize internally an ARQ protocol, it is natural to imply that the delays are bounded by the predefined maximum allowable number of retransmissions, for all $k \geq 0$, i.e., $0 \leq \tau_k^{ji} \leq \bar{\tau}^{ji} \leq \bar{\tau}$, where $\bar{\tau}$ is the maximum delay of all packets sent within the network at all time instants, i.e., $\bar{\tau} = \max\{\bar{\tau}^{ji}\}$.

For simplicity of exposition, we assume that the channel follows a Bernoulli distribution. Hence, the probabilities for different number of consecutive NACKs for a packet initially transmitted at time instant k over link ε^{ji} are defined as

follows:

$$\begin{aligned} \mathbb{P}(\tau_k^{ji} = 0) &= 1 - q^{ji}, & (\text{no error}) \\ \mathbb{P}(\tau_k^{ji} = 1) &= q^{ji}(1 - q^{ji}), & (1 \text{ NACK}) \\ &\vdots \\ \mathbb{P}(\tau_k^{ji} = \bar{\tau}^{ji}) &= (q^{ji})^{\bar{\tau}^{ji}}(1 - q^{ji}), & (\text{max \# of NACKs}) \\ \mathbb{P}(\tau_k^{ji} = \bar{\tau}^{ji} + 1) &= (q^{ji})^{\bar{\tau}^{ji}+1}, & (\text{packet drop}) \end{aligned}$$

where $\mathbb{P}(\cdot)$ refers to the probability that the event in the argument " \cdot " will occur. Recall that a packet may be dropped if the maximum allowable retransmission number ($\bar{\tau}^{ji}$) is exceeded. A successful reception of a packet sent over link ε^{ji} is denoted by the indicator variable $\mathbb{I}_{k+\bar{\tau}^{ji}}^{ji} = 1$, while for a dropped packet is denoted by $\mathbb{I}_{k+\bar{\tau}^{ji}}^{ji} = 0$.

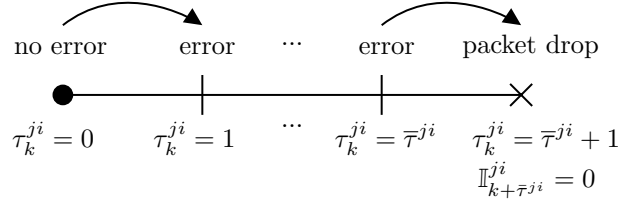


Fig. 1: Erroneous packet propagation.

B. Feedback Schemes for Ratio Consensus

In this work, we consider two different schemes for utilizing the ACK/NACK feedback for the ARQ-based Ratio Consensus algorithm. The idea of these two schemes arise naturally since multiple packets (new or delayed) may be transmitted over the same link ε^{ji} within a time slot. Both schemes involve receivers that feed back ACK/NACK signals for each transmission that arrive. However the former involves multiple individual transmissions and their corresponding ARQ feedback signals, and the latter involves single transmissions that contain the aggregated information and their corresponding ARQ feedback signal (that signifies whether the aggregated packet has been successfully decoded).

1) *Multiple Transmissions Multiple Feedback (MTMF)*: In this scheme, each new packet is sent individually along with possibly retransmitted (delayed) packets. To get an intuition, consider a simple example with two nodes v_1 and v_2 , shown in Fig. 2. During the first time slot (starting at $k = 1$), a new packet (containing a weighted version of the state according to (2)) is sent over link ε^{j1} , where node v_2 detects an error in the received packet, and sends back a NACK. By the second time slot, node v_1 has the NACK feedback sent from v_2 , and retransmits the previous packet w_1^{21} , along with the new packet w_2^{21} . Within this time slot, node v_2 successfully receives the delayed packet w_1^{21} for which it sends an ACK, but it detects an error in the new packet w_2^{21} for which it sends a NACK. The same procedure is followed for the latter transmissions. Note that, this scheme requires transmissions and ARQ feedback signals to be identified.

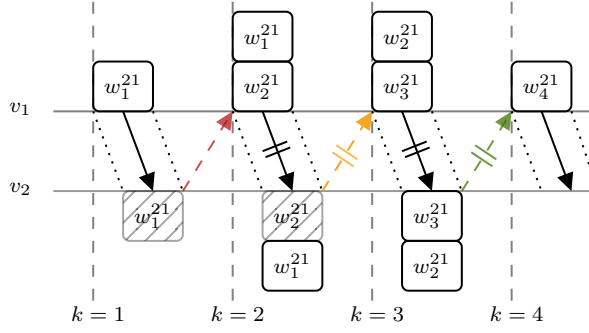


Fig. 2: An individual ARQ feedback (ACK/NACK) is sent for each packet, within each time slot. Red single arrow denote a NACK; orange bus arrow denote multiple ARQ feedback of ACKs and NACKs; green bus arrows denote multiple ACKs; shaded rectangles represent erroneous packets.

2) *Single Transmissions Single Feedback (STSF)*: In this scheme, transmitting nodes aggregate the information (from new and delayed packets) in a single packet to be sent within a single time slot. Hence, receiving nodes need to send back a single ARQ feedback (ACK/NACK) within the same time slot. To get an intuition, consider a simple example with two nodes v_1 and v_2 , shown in Fig. 3. Node v_1 transmits a packet w_1^{21} , to node v_2 within the first time slot. Within the same time slot, node v_2 detects an error and requests a retransmission by sending back a NACK. By the next time slot, node v_1 has the NACK and calculates the summation of the weighted values of the delayed packet and the new packet, in order to transmit a single packet w_2^{21} . Under this scheme, node v_2 sends only a single ARQ feedback to node v_1 , according to the error detection mechanism.

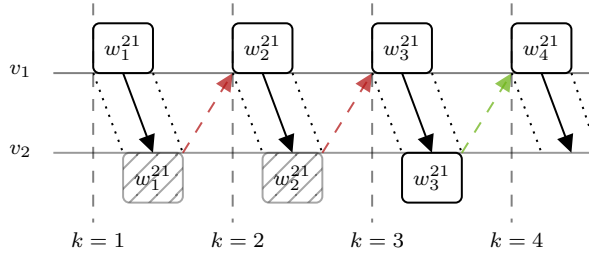


Fig. 3: Receiving node sends an ARQ-feedback (ACK or NACK) for the aggregated information of all received packets, at each time step k . Red arrows denote NACKs; green arrows denote ACKs; shaded rectangles represent erroneous packets.

C. ARQ-based Ratio Consensus Algorithm

Herein, we describe the ARQ-based Ratio Consensus algorithm for the feedback schemes considered in §III-B. In the following steps, we describe MTMF in detail, but one can adjust the algorithm for the STSF scheme. Specifically, each node v_j executes the following steps:

- First, the initial state $x_0^j = V^j$ is determined by the information (often local sensor reading) on which the network of nodes must come to agreement. As soon as

the input has been processed, all the auxiliary variables are initialized at $y_0^j = 1$, $\eta_0^j = 0$, $\psi_0^{ji} = 0$, $\sigma_0^j = 0$, and $\chi_0^{ji} = 0$.

- Second, the out-degree of node v_j is acquired by broadcasting a dummy package and summing the number of received ARQ feedback signals from its in-neighbors. Note that, one could improve the out-degree acquisition accuracy by executing this process several times.
- Node v_j updates its states x_{k+1}^j and y_{k+1}^j using equations in lines 11-12, whenever the actual packet retrasmmissions over link ε^{ji} have not reached the maximum retransmission limit ($\bar{\tau}^{ji}$) imposed by the ARQ protocol.
- In case the retransmission limit of a packet has been reached, then the running sum mechanism, in lines 14-26, is activated. At this stage, the running sum variables σ_{k+1}^j and η_{k+1}^j are updated according to lines 14-15. The accumulated information in the running sum variables of each node v_j is transmitted to its out-neighbors. If the packet arrived at the receiving node v_i without errors, then variables χ_{k+1}^{ji} and ψ_{k+1}^{ji} are updated with the accumulated masses σ_{k+1}^j and η_{k+1}^j respectively. In contrast, if the packet is erroneous, then the auxiliary state variables χ_{k+1}^{ji} and ψ_{k+1}^{ji} remain unchanged. With the next (possibly delayed) transmission, the information held in the auxiliary variables is updated accordingly.

D. Simple Example

To get an intuition of the ARQ-based Ratio Consensus algorithm, we provide the following example with the aid of a graph representation. Here it is important to note that, the augmented graph representation is only used to exploit the modeling and analysis of the algorithm without affecting its actual implementation. More details regarding the augmentation of the digraph are presented in the subsequent section. Consider a digraph of two nodes, shown in Fig. 4, where node v_1 and v_2 utilize an ARQ protocol with maximum retransmission limit (maximum possible delay) $\bar{\tau}^{12} = 1$, and $\bar{\tau}^{21} = 2$, respectively. In this graph representation, we augment the original digraph by adding extra virtual nodes and extra virtual buffer nodes to capture the effect of delays due to retransmissions, and possible packet drops. Initially transmitted information will reach its recipient through virtual nodes (shown in orange squares) that model the induced delays due to retransmissions. In this example, we set the probability that a node receives an erroneous packet to be $q^{ji} = 0.4$. Hence, packets sent from v_1 to v_2 are dropped with probability $(q^{21})^{\bar{\tau}^{21}+1} = 0.4^3 = 0.064$, while packets sent from node v_2 to v_1 , are dropped with probability $(q^{12})^{\bar{\tau}^{12}+1} = 0.4^2 = 0.16$. As soon as the packet drops are realized, then the corresponding edges are activated to model the update of the running sum variables, as described in Algorithm 1. Consider the case where two consecutive NACKs has been sent back from node v_2 to node v_1 . Then, we can model the delay on the initial information by

Algorithm 1 ARQ-based Ratio Consensus

```

1: Input:  $x_0^j = V^j$ 
2: Initialization:  $\sigma_0^j = 0, y_0^j = 1, \eta_0^j = 0,$ 
3:  $\chi_0^{ji} = 0, \psi_0^{ji} = 0, \forall i \in \mathcal{N}_j^-$ 
4: Out-degree acquisition:  $d_j^+ = |\mathcal{N}_j^+|$ 
5: for  $k \geq 0$  : do
6:   Transmit to all  $v_i \in \mathcal{N}_j^+$ :
7:      $x_s^j$  and  $y_s^j, \forall s = k - \bar{\tau}^{ji}, \dots, k$ , for  $k \geq \bar{\tau}^{ji}$ 
8:   Receive from all  $v_i \in \mathcal{N}_j^-$ :
9:      $x_h^i$  and  $y_h^i, \forall h = k - \tau_h^{ji}$ , for  $0 \leq h \leq k$ 
10:  if  $\tau_k^{ji} < \bar{\tau}^{ji}$ 
11:     $x_{k+1}^j = \sum_{v_i \in \mathcal{N}_j^- \cup \{v_j\}} \sum_{r=0}^{\bar{\tau}^{ji}} \iota_{k-r}^{ji} p^{ji} x_{k-r}^i$ 
12:     $y_{k+1}^j = \sum_{v_i \in \mathcal{N}_j^- \cup \{v_j\}} \sum_{r=0}^{\bar{\tau}^{ji}} \iota_{k-r}^{ji} p^{ji} y_{k-r}^i$ 
13:  else
14:     $\sigma_{k+1}^j = \sigma_k^j + p^{jj} x_k^j$ 
15:     $\eta_{k+1}^j = \eta_k^j + p^{jj} y_k^j$ 
16:    Transmit to all  $v_i \in \mathcal{N}_j^+$ :  $\sigma_{k+1}^j$  and  $\eta_{k+1}^j$ 
17:    Receive from all  $v_i \in \mathcal{N}_j^-$ :  $\sigma_{k+1}^i$  and  $\eta_{k+1}^i$ 
18:    if  $\mathbb{I}_{k+\bar{\tau}^{ji}}^{ji} = 1$ 
19:       $\chi_{k+1}^{ji} = \sigma_{k+1}^i$ 
20:       $\psi_{k+1}^{ji} = \eta_{k+1}^i$ 
21:    else
22:       $\chi_{k+1}^{ji} = \chi_k^{ji}$ 
23:       $\psi_{k+1}^{ji} = \psi_k^{ji}$ 
24:    end
25:     $x_{k+1}^j = \sum_{v_i \in \mathcal{N}_j^-} (\chi_{k+1}^{ji} - \chi_k^{ji})$ 
26:     $y_{k+1}^j = \sum_{v_i \in \mathcal{N}_j^-} (\psi_{k+1}^{ji} - \psi_k^{ji})$ 
27:  end
28:  Output:  $z_{k+1}^j = \frac{x_{k+1}^j}{y_{k+1}^j}$ 
29: end for

```

introducing two virtual nodes, $v_{21}^{(1)}$ and $v_{21}^{(2)}$. In the next time slot, the running sum mechanism will be activated to handle possible packet drops since the maximum retransmission limit has been exceeded. For this transmission, the packet drop handling mechanism updates the running sum variables based on the indicator variable $\mathbb{I}_{k+\bar{\tau}^{21}}^{21} = \mathbb{I}_{k+2}^{21}$. If $\mathbb{I}_{k+2}^{21} = 1$, then the initially transmitted information will be received by node v_2 successfully. Otherwise, if node v_2 detects another error in the initially transmitted packet, $\mathbb{I}_{k+2}^{21} = 0$, then the information will propagate to the virtual buffer $v_{21}^{(f)}$ (shown in red square). The information held in $v_{21}^{(f)}$, is released to the actual node v_2 along with the (possibly delayed) packet that is intended to be transmitted in the next time slot. The information release is modeled by activating one of the virtual buffer node's $v_{21}^{(f)}$ outgoing links, \mathbb{I}_{21}^{f} . Packet transmission from node v_2 to node v_1 , is delayed by one time slot which equals to the maximum retransmission limit. However, the packet arrives successfully at node v_1 since no further error was detected in the last retransmission trial.

To emphasize the difference between the ARQ-based

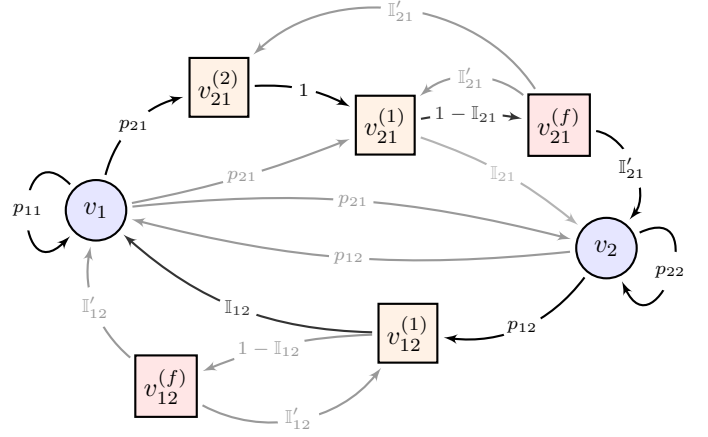


Fig. 4: Two node digraph denoting transmissions using the ARQ-based Ratio Consensus algorithm. A packet transmitted from node v_1 to v_2 is delayed by two time slots (two consecutive NACKs), and is dropped in the subsequent time slot. A packet transmitted from node v_2 to v_1 is delayed by one time slot (one NACK).

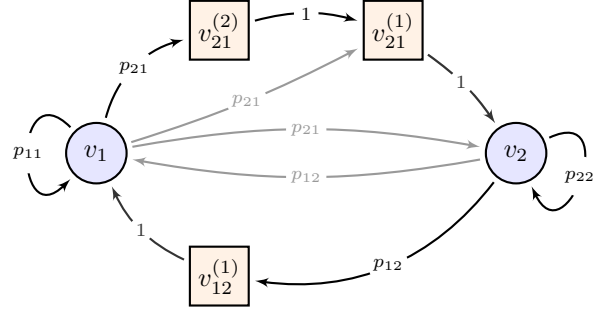


Fig. 5: Two node digraph denoting transmissions using the Robustified Ratio Consensus algorithm. A packet transmitted from node v_1 to v_2 is delayed by two time slots, while from v_2 to v_1 is delayed by one time slot.

Ratio Consensus algorithm and the Robustified Ratio Consensus algorithm (described in §II-C), we introduce the digraph shown in Fig. 5. This digraph models the same communication topology between nodes, as in Fig. 4. However, it is easy to see that the digraph that models the Robustified Ratio Consensus algorithm transmissions, cannot handle packet drops, and hence, the digraph only consists of the actual nodes and the virtual nodes that model the transmission delays. Consequently, nodes utilizing the Robustified Ratio Consensus algorithm, cannot converge to the average of the network's wide initial values if packet drops occur during information exchange.

Remark 3. The ARQ-based Ratio Consensus algorithm can handle (possibly) delayed information exchange between nodes and possible packet drops due to erroneous packets. Moreover, the ARQ error-free acknowledgement feedback signal received by each node v_j is used to determine its out-degree d_j^+ , notified whether a packet has arrived or not, and determine a local upper-bound on the delays, imposed

by the ARQ retransmission limit.

IV. AUGMENTED DIGRAPH REPRESENTATION

In this section, we analyse the convergence of Algorithm 1, by first introducing the random weighted adjacency matrix that corresponds to the delayed and possibly packet dropping communication topology. To simplify the analysis, we consider identical ARQ protocols for each node. This implies that the maximum retransmission limit is the same for all nodes, and therefore the maximum delay on each link $\bar{\tau}^{ji}$, is bounded by the maximum delay in the network, $\bar{\tau}$. Hence, for each link (edge) $\varepsilon_{ji} \in \mathcal{E}$, we add $\bar{\tau}$ extra virtual nodes, $v_{ji}^{(1)}, v_{ji}^{(2)}, \dots, v_{ji}^{(\bar{\tau})}$, where the virtual node $v_{ji}^{(r)}$ holds the information that is destined to arrive at node v_j after r time steps. In case of a packet drop, $\mathbb{I}_k^{ji} = 0$, the delayed information (held in the corresponding virtual node of maximum delay) will propagate in the virtual buffer node of the original node v_j instead of being received by the original node. During the next time slot, the information will be released either to the original node if $\tau_{k+1}^{ji} = 0$, or to the corresponding virtual node if $\tau_{k+1}^{ji} \geq 0$. Note that, the links that correspond to self-loop do not induce delays or packet losses, and hence no virtual nodes for self-loops are considered in the augmented graph representation. Finally, the augmented digraph, $\mathcal{G}^a = (\mathcal{V}^a, \mathcal{E}^a)$, that models the delayed network with packet drops consists of at most $\tilde{n} = |\mathcal{E}|(\bar{\tau} + 1) + n \leq n(n-1)(\bar{\tau} + 1) + n$ nodes $\in \mathcal{V}^a$ where n nodes are real nodes, $|\mathcal{E}|\bar{\tau}$ are virtual nodes due to delays, and $|\mathcal{E}|$ are virtual nodes due to packet losses.

Hence, the consensus iterations performed by Algorithm 1, can be rewritten in matrix form as:

$$\tilde{x}_{k+1} = \Xi_k \tilde{x}_k, \quad (6a)$$

$$\tilde{y}_{k+1} = \Xi_k \tilde{y}_k, \quad (6b)$$

where \tilde{x}_k , and \tilde{y}_k are vectors containing the variables of both the actual and virtual nodes at time instant k . Note that, \tilde{x}_k , and \tilde{y}_k have $\tilde{n} \geq n$ elements, where the first n elements correspond to the actual nodes, while the remaining elements correspond to the virtual nodes of the augmented digraph. Matrix $\Xi_k \in \mathbb{R}_+^{\tilde{n} \times \tilde{n}}$ is a nonnegative random matrix associated with the augmented digraph:

$$\Xi_k \triangleq \begin{pmatrix} P_k^{(0)} & D_k^{\text{succ}} & 0 & \dots & 0 & D_{k+1}^{(0)} \\ P_k^{(1)} & D_k^{\text{exc}} & I & \dots & 0 & D_{k+1}^{(1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ P_k^{(\bar{\tau}-1)} & 0 & 0 & \dots & I & D_{k+1}^{(\bar{\tau}-1)} \\ P_k^{(\bar{\tau})} & 0 & 0 & \dots & 0 & D_{k+1}^{(\bar{\tau})} \\ 0 & D_k^{\text{pd}} & 0 & \dots & 0 & D_{k+1}^{\text{sl}} \end{pmatrix} \quad (7)$$

where each element of $P_k^{(r)}$ is determined by:

$$P_k^{(r)}(j, i) = \begin{cases} P(j, i), & \text{if } \tau_k^{ji} = r, (j, i) \in \mathcal{E}, \\ 0, & \text{otherwise.} \end{cases} \quad (8)$$

If the index r that corresponds to the block matrix $P^{(r)}$ is equal to the packet retransmission number τ_k^{ji} , then the

link ε^{ji} will be weighted by the actual (original) weight p_{ji} , otherwise the weight will be zero.

Clearly the structure of matrix Ξ_k depends on the realized number of retransmissions and packet drops. Block matrix $D_k^{\text{succ}} \in \mathbb{R}_+^{n \times m}$ activates the virtual links that propagate the delayed information to actual nodes. This occurs whenever a packet is successfully received by its destined (actual) node without error, during its last retransmission trial, $\tau_k^{ji} = \bar{\tau}^{ji}$. Block matrix $D_k^{\text{exc}} \in \mathbb{R}_+^{m \times m}$, is a diagonal matrix with its diagonal elements having 1 if $\tau_k^{ji} < \bar{\tau}^{ji}$, and 0 otherwise, for $j, i = 1, \dots, n$ and $j \neq i$. This matrix activates the running sum mechanism whenever the retransmission limit of packet a has been reached. Block matrix $D_k^{\text{pd}} \in \mathbb{R}_+^{m \times m}$ is a diagonal matrix of which its diagonal elements take $1 - \mathbb{I}_{k+\bar{\tau}^{ji}}^{ji}$. This matrix propagates the running sum of each node to its virtual buffer node in, whenever a packet drop occurs. Block matrices $D_{k+1}^{(r)}$, are of appropriate dimensions, and handle the release of information from virtual buffer nodes to the corresponding actual or virtual node. The elements of these matrices are placed in the corresponding row of matrix Ξ_k similarly to (8). In other words, if a packet is dropped, the information will be held in the virtual buffer, until the next successful (possibly delayed) transmission. Block matrix D_{k+1}^{sl} is of appropriate dimensions, and models the self-loops of the virtual buffer nodes, whenever a packet arrives successfully to its intended actual node, without being dropped. Based on the aforementioned properties, it is clear to see that matrix Ξ_k maintains its column-stochasticity property, although the links that establish the transmissions between nodes might be unreliable.

Prior to establishing the convergence of each node to the network-wide average of the initial values, we first introduce some properties of matrices Ξ_k , similar to the ones established in [20] for packet dropping links. Matrix Ξ_k denotes a particular instance from the set of all possible instances of realized delays induced from packet retransmissions, and packet drops. All these possible instances are in the set \mathcal{X} , which consists of a finite number of matrices of identical dimensions $\tilde{n} \times \tilde{n}$, where each matrix in \mathcal{X} corresponds to a distinct instantiation of the packet drop indicator variable \mathbb{I}_k^{ji} and the delay on the links τ_k^{ji} . Thus, each positive element of any matrix Ξ_k is lower bounded by a positive constant $c = \min_{j \in \mathcal{V}} 1/(1 + d_j^+)$, since the (i, j) element of any $\Xi_k \in \mathcal{X}$ can take values of either 0, 1, or $1/(1 + d_j^+)$. Let l be a finite positive integer. Then, a sequence of l matrices in \mathcal{X} , possibly with repetition, exists, and their product in a certain order occurs with a probability greater or equal to some $p_{\min} > 0$. The product of these l matrices (in a certain order), forms a column stochastic matrix, where the rows that correspond to the actual network nodes, contain strictly positive values. Next, we build upon these matrix properties for the analysis of ratio consensus convergence in Lemma 1, Lemma 2, and Theorem 1.

Recall that each node in the network aims at obtaining

consensus to the network-wide average of the initial values:

$$z^* = \frac{\sum_{l \in \mathcal{V}^a} \tilde{x}_0^l}{\sum_{l \in \mathcal{V}^a} \tilde{y}_0^l} = \frac{\sum_{l \in \mathcal{V}} x_0^l}{\sum_{l \in \mathcal{V}} y_0^l} = \frac{1}{n} \sum_{l \in \mathcal{V}} x_0^l, \quad (9)$$

by calculating at each time step k , the ratio:

$$z_k^j = \frac{\tilde{x}_k^j}{\tilde{y}_k^j}, \quad (10)$$

whenever $\tilde{y}_k^j \geq c^l$. To establish the convergence of the proposed algorithm to the exact average of the network-wide initial values, we need to show that the augmented auxiliary variable $\tilde{y}_k^j \geq c^l$ occurs infinitely often, and hence as k goes to infinity, the sequence of ratio computations in (10) converges to the value in (9) almost surely.

Theorem 1. Consider a strongly connected digraph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where each node $v_j \in \mathcal{V}$ has some initial value x_0^j , and $y_0^j = 1$. Let z_k^j (for all $v_j \in \mathcal{V}$ and $k = 0, 1, 2, \dots$) be the output of the iterations in Algorithm 1. Then, the solution to the average consensus can be obtained by each node, by computing:

$$\lim_{k \rightarrow \infty} z_k^j = \frac{x_k^j}{y_k^j} = \frac{\sum_{v_i \in \mathcal{V}} x_i^0}{n}, \quad \forall v_j \in \mathcal{V}. \quad (11)$$

Proof. See Appendix B. ■

V. NUMERICAL EVALUATION

Consider a directed network with each node v_j choosing its out-going links (including its self-loop link) based on (2), which results into the following column-stochastic weighted adjacency matrix:

$$P = \begin{pmatrix} 1/3 & 0 & 0 & 1/2 & 0 \\ 1/3 & 1/3 & 0 & 0 & 0 \\ 1/3 & 1/3 & 1/2 & 0 & 1/3 \\ 0 & 0 & 0 & 1/2 & 1/3 \\ 0 & 1/3 & 1/2 & 0 & 1/3 \end{pmatrix}. \quad (12)$$

The variables of each node x^j , y^j are updated within each time slot using Algorithm 1 with initial values $\mathbf{x}_0 = (x_0^1 \dots x_0^n)^T = (4 \ 5 \ 6 \ 3 \ 2)^T$, and $\mathbf{y}_0 = (y_0^1 \dots y_0^n)^T = (1 \ 1 \ 1 \ 1 \ 1)^T$, respectively. In this example we consider that a transmitted packet may arrive in error at the receiving node with probability q^{ji} . Fig. 6 depicts the absolute consensus error, $|\hat{z} - z^*|$, (\hat{z} is the average ratio among all actual nodes in the network over all simulations) for different packet error probabilities q^{ji} and different maximum retransmission limits (upper-bounds on the delays), for a total of 300 simulations of 100 consensus iterations. As expected, with higher probability of packet error q^{ji} , the absolute consensus error is higher.

To get intuition regarding the convergence of the proposed method, we consider the same ARQ mechanism for each node, with maximum allowable retransmission number set to $\bar{\tau}^{ji} = \bar{\tau} = 2$, which corresponds to the upper bound of the delays on the links. Fig. ?? shows the evolution of the ratio at node v_1 during the first time steps of the execution of the ARQ-based Ratio Consensus algorithm. Node v_1

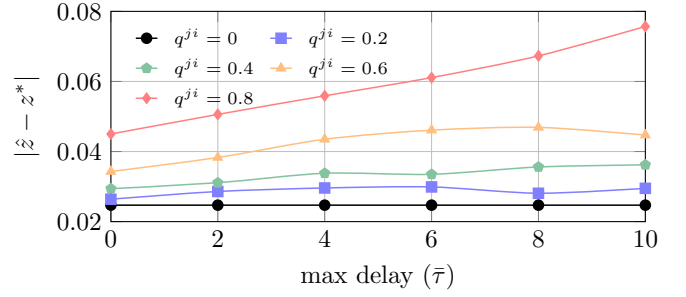


Fig. 6: Absolute consensus error, $|\hat{z} - z^*|$, for different upper bound $\bar{\tau}$ on delays, and probability of erroneous packet.

receives information from node v_4 (and itself), according to the network topology modeled by the weighted adjacency matrix defined in (12). The packet error probability is set at $q^{ji} = 0.6$. As shown in Fig. ??, the ratio z_k^1 is updated as soon as information packets from node v_4 arrive successfully at node v_1 .

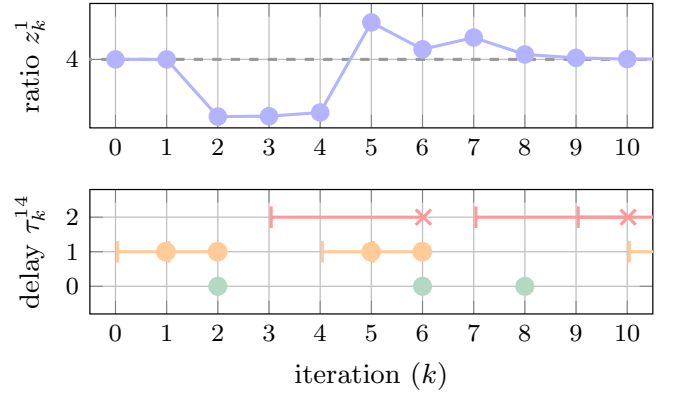


Fig. 7: Upper: Ratio z_k^1 of node v_1 utilizing ARQ-based Ratio Consensus algorithm with packet error probability $q^{ji} = 0.6$. Lower: Snapshot of packet transmissions from node v_4 to node v_1 . Non-delayed transmissions are denoted with green circle marks, while $\tau_k^{14} = 1$, and $\tau_k^{14} = 2$ are denoted with orange and red, respectively. Packet drops are denoted with red cross marks.

Next, we evaluate the ARQ-based Ratio Consensus algorithm for two different communication channel conditions, that are captured by: (a) packet error probability $q^{ji} = 0.2$, and (b) packet error probability $q^{ji} = 0.8$. We assess both cases for two different configurations on the maximum retransmission limit of the ARQ-based Ratio Consensus algorithm, $\bar{\tau} = 2$, and $\bar{\tau} = 10$. Fig. 8 depict the evolution of variables x_k^j and y_k^j of each (non-virtual) node in the network with packet error probability $q^{ji} = 0.2$, and maximum retransmission limit $\bar{\tau} = 2$. Due to the time-varying delays on the links and possible packet drops, we can see that the variables do not converge at a certain value. However, the ratio of each node shown in Fig. 9 converges to the exact average of the nodes' initial values, despite the time-varying delays due to retransmissions, and possible packet drops. The lower plot of Fig. 9 presents the convergence of the ratio z_k^i when the maximum retransmission limit is set to $\bar{\tau} = 10$.

The convergence rate under this configuration is similar to the one with $\bar{\tau} = 2$ since the packet error probability is relatively small, and hence most of the packets will arrive at their destined nodes successfully.

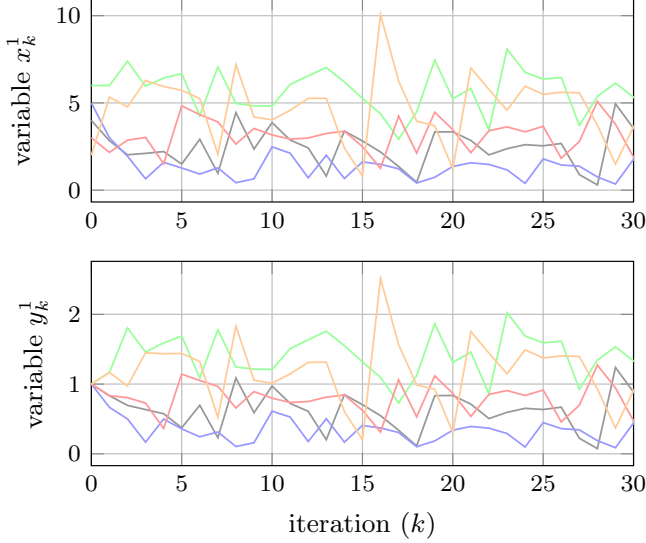


Fig. 8: Variables x_k^j (upper), and y_k^j (lower) of each node v_j , with maximum delay $\bar{\tau} = 2$, and probability of failure $q^{ji} = 0.2$.

will converge to the average consensus value despite the ARQ maximum retransmission limit, and the packet error probability that captures the quality of the communication channels in the directed network.

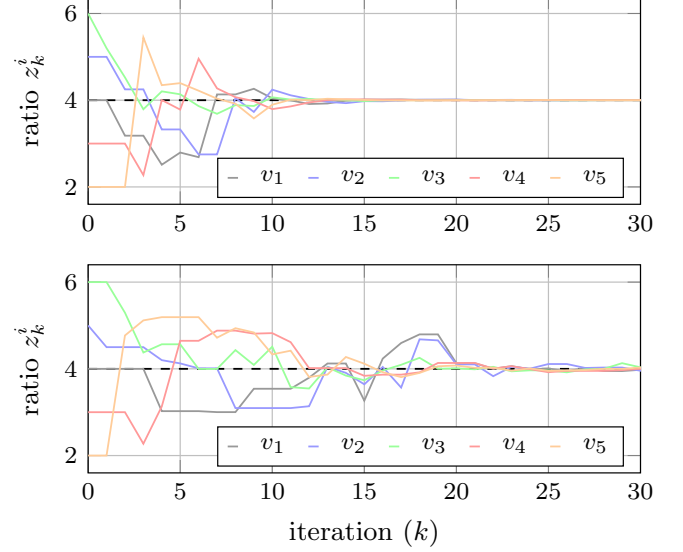


Fig. 10: Ratio z_k^j of each node v_j , with maximum delay $\bar{\tau} = 2$ (upper), and $\bar{\tau} = 10$ (lower), and packet error probability $q^{ji} = 0.8$.

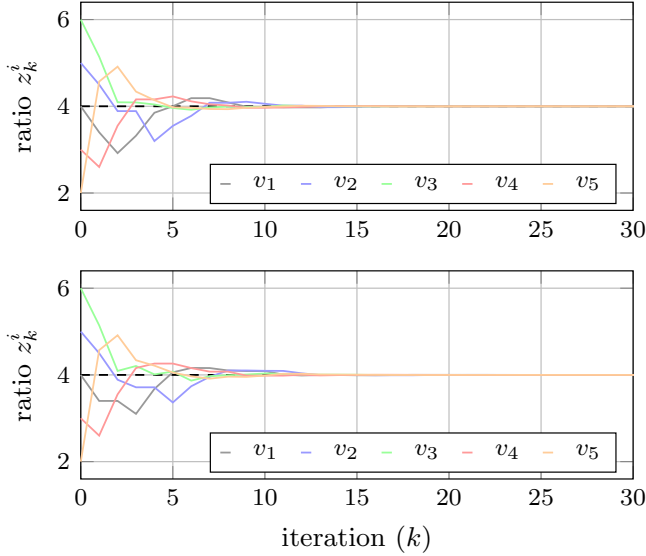


Fig. 9: Ratio z_k^j of each node v_j , with maximum delay $\bar{\tau} = 2$ (upper), and $\bar{\tau} = 10$ (lower), and packet error probability $q^{ji} = 0.2$.

In contrast, when the communication channel is not reliable (which we capture by $q^{ji} = 0.8$), we can see that the convergence to the network-wide average of the nodes' initial values varies with the maximum retransmission limit. In particular, nodes converge faster with $\bar{\tau} = 2$ (upper plot in Fig. 10), while for $\bar{\tau} = 10$ (lower plot in Fig. 10), the convergence to the network-wide average is significantly slower. However, for all the aforementioned cases, the ARQ-based Ratio Consensus algorithm guarantees that each node

VI. CONCLUSIONS AND FUTURE DIRECTIONS

A. Conclusions

In this paper, we proposed a distributed strategy for discrete-time asymptotic average consensus for multi-agent systems, in the presence of time-varying delays induced by packet retransmissions, and packet-dropping communication links. By incorporating the ARQ protocol in our proposed method, the nodes in the network are able to know their out-degrees by utilizing the ARQ feedback signals sent by their receiving nodes. Since the nodes make use of the same ARQ protocol, they also have the knowledge of the global upper-bound on the delays, that is the limit on the number of consecutive retransmissions due to packets detected in error that were intended to arrive at the receiving nodes. We showed that the nodes of a directed strongly connected network can utilize the proposed distributed strategy to reach asymptotic average consensus, in the presence of bounded time-varying delays and packet-dropping communication links.

B. Future Directions

A promising extension of this work to achieve faster convergence to the average consensus value, and maintain higher reliability of packet transmissions, is to devise a ratio consensus algorithm based on the Hybrid-ARQ (HARQ) protocol. This protocol incorporates Forward Error Correction (FEC) with the ARQ protocol, to reduce the frequency of retransmission by correcting the error patterns which occur most frequently, such that the system throughput increases,

and hence the convergence of the average consensus problem converges faster. This extension could exploit the potential of a wide variety of applications of distributed systems that involve wireless transmissions.

APPENDIX A AUXILIARY LEMMAS

Lemma 1 implies that the event $\tilde{y}_k^j \geq c^l$ occurs infinitely often and hence each node computes the ratio consensus estimate in (10) at infinitely many time steps with probability 1. Lemma 2 implies that as k goes to infinity, the sequence of ratio computations in (10) converges to the network-wide average of the initial values z^* in (9) almost surely (with probability 1).

Lemma 1. *Hadjicostis et. al. [20]. Let $\mathcal{K}^j = \{\kappa_1^j, \kappa_2^j, \dots\}$ denote the sequence of time instants when $\tilde{y}_k^j \geq c^l$ and thus node $v_j \in \mathcal{V}$ updates its ratio consensus estimate using (10), where $\kappa_t^j < \kappa_{t+1}^j$, $t \geq 1$. The sequence \mathcal{K}^j contains infinitely many elements with probability 1.*

Lemma 2. *Coefficient of Ergodicity, Hadjicostis et. al. [20]. Let $L_k = \prod_{\ell=0}^{k-1} \Xi_\ell$ be the resulting column stochastic matrix from the product of column stochastic matrices Ξ_ℓ . Then the coefficient of ergodicity $\delta(L_k) := \max_j \max_{i_1, i_2} |L_k(j, i_1) - L_k(j, i_2)|$ converges almost surely to zero.*

APPENDIX B PROOF OF THEOREM 1

Proof. Theorem 1 is established following similar analysis as in [20] *mutatis mutandis*, by replacing the matrix that corresponds to the network topology of the non-delayed case with the augmented matrix that incorporates the ARQ-based communication protocol defined in (7). First, it is clear to see that the evolution of the augmented variables in (6) can be written in the following form:

$$\begin{aligned}\tilde{x}_k &= \Xi_{k-1} \Xi_{k-2} \cdots \Xi_1 \Xi_0 \tilde{x}_0 = \left(\prod_{\ell=0}^{k-1} \Xi_\ell \right) \tilde{x}_0 \equiv L_k \tilde{x}_0, \\ \tilde{y}_k &= \Xi_{k-1} \Xi_{k-2} \cdots \Xi_1 \Xi_0 \tilde{y}_0 = \left(\prod_{\ell=0}^{k-1} \Xi_\ell \right) \tilde{y}_0 \equiv L_k \tilde{y}_0,\end{aligned}$$

where $L_k = \left(\prod_{\ell=0}^{k-1} \Xi_\ell \right)$ is the forward product of column stochastic matrices $\Xi_\ell \in \mathcal{X}$, $\forall \ell = 0, 1, \dots, k-1$. From the above notation, it is clear that for each node $v_j \in \mathcal{V}$, we have $\tilde{x}_k^j = L_k(j, :) \tilde{x}_0$ and $\tilde{y}_k^j = L_k(j, :) \tilde{y}_0$, where $L_k(j, :)$ denotes the j -th row of the forward product matrix L_k . Now consider any $k \geq k_\psi$ such that $y_k^j \geq c^l$. This implies $y_k^j = L_k(j, :) \tilde{y}_0 \geq c^l$ and $\sum_l \tilde{y}_0^l = n$, and hence we know that the maximum entry of $L_k(j, :)$ is at least c^l/n . Moreover, since $\delta(L_k) < \psi$, the columns of matrix L_k are "within ψ " of each other. In particular, the j -th row of L_k can be written as $L_k(j, :) = c_k^j (\mathbf{1}^T + \mathbf{e}_k^T)$, where $\mathbf{1}^T$ is the m -dimensional all-ones row vector, $\mathbf{e}_k^T = [e_k^1, e_k^2, \dots, e_k^m]$ is an m -dimensional row vector that satisfies $e_k^{\max} \equiv \max_l |e_k^l| <$

$\frac{\psi}{2c_k^j}$, and $c_k^j \geq \frac{c^l}{n} - \frac{\psi}{2}$ (assume without loss of generality that $\psi < \frac{c^l}{n}$). Hence, at each node v_j , the ratio is obtained by:

$$\begin{aligned}z_k^j &= \frac{\tilde{x}_k^j}{\tilde{y}_k^j} = \frac{L_k(j, :) \tilde{x}_0}{L_k(j, :) \tilde{y}_0} = \frac{c_k^j (\mathbf{1}^T + \mathbf{e}_k^T) \tilde{x}_0}{c_k^j (\mathbf{1}^T + \mathbf{e}_k^T) \tilde{y}_0} \\ &= \frac{(\mathbf{1}^T + \mathbf{e}_k^T) \tilde{x}_0}{(\mathbf{1}^T + \mathbf{e}_k^T) \tilde{y}_0}.\end{aligned}\quad (13)$$

First, we establish the bounds on the numerator of the above expression as

$$\Sigma^x - e_k^{\max} \Sigma^{|x|} \leq (\mathbf{1}^T + \mathbf{e}_k^T) \tilde{x}_0 \leq \Sigma^x + e_k^{\max} \Sigma^{|x|},$$

where $\Sigma^x = \sum_l x_0^l$, and $\Sigma^{|x|} = \sum_l |x_0^l|$, for $l = 1, \dots, n$. Now, for the bounds on the denominator, we know that $\sum_l y_0^l = n$, since $y_0^l = 1$ for $l = 1, \dots, n$. Hence, we can bound the denominator as

$$n(1 - e_k^{\max}) \leq (\mathbf{1}^T + \mathbf{e}_k^T) \tilde{y}_0 \leq n(1 + e_k^{\max}).$$

Then, the ratio of the bounded numerator and denominator gives:

$$\frac{\Sigma^x - e_k^{\max} \Sigma^{|x|}}{n(1 + e_k^{\max})} \leq \frac{(\mathbf{1}^T + \mathbf{e}_k^T) \tilde{x}_0}{(\mathbf{1}^T + \mathbf{e}_k^T) \tilde{y}_0} \leq \frac{\Sigma^x + e_k^{\max} \Sigma^{|x|}}{n(1 - e_k^{\max})}.$$

Recall that from (9) we have $z^* = \frac{1}{n} \sum_l x_0^l = \frac{\Sigma^x}{n}$, and thus we get:

$$z^* - E_k \leq z_k^j \leq z^* + E_k, \quad (14)$$

where $E_k = z^* (\Sigma^x + \Sigma^{|x|}) e_k^{\max} / \Sigma^x (1 - e_k^{\max})$. To ensure that $z^* - \epsilon \leq z_k^j \leq z^* + \epsilon$ holds whenever $k \geq k_\psi$ and $k \in \mathcal{K}^j$, we can choose $\psi < (2\mu\epsilon / (\Sigma^x + \Sigma^{|x|} + 2n\epsilon))$, such that for any desirable $\epsilon > 0$, $e_k^{\max} < (n\epsilon / (\Sigma^x + \Sigma^{|x|} + n\epsilon))$, holds. Hence, the ratio at each node in (13) converges with probability 1 to z^* as k goes to infinity. ■

REFERENCES

- [1] L. Schenato, "Optimal Estimation in Networked Control Systems subject to Random Delay and Packet Drop," *IEEE Trans. Autom. Control*, vol. 53, no. 5, pp. 1311–1317, 2008.
- [2] M. Pajic, S. Sundaram, G. J. Pappas, and R. Mangharam, "The Wireless Control Network: A New Approach for Control over Networks," *IEEE Trans. Autom. Control*, vol. 56, no. 10, pp. 2305–2318, 2011.
- [3] M. J. Hossain, M. A. Mahmud, F. Milano, S. Bacha, and A. Hably, "Design of Robust Distributed Control for Interconnected Microgrids," *IEEE Trans. Smart Grid*, vol. 7, no. 6, pp. 2724–2735, 2015.
- [4] A. Varghese and D. Tandur, "Wireless Requirements and Challenges in Industry 4.0," in *Int. Conf. Cont. Computing and Inform.*, 2014, pp. 634–638.
- [5] M. Weiner, M. Jorgovanovic, A. Sahai, and B. Nikolić, "Design of a Low-latency, High-reliability Wireless Communication System for Control Applications," in *IEEE Int. Conf. Commun.*, 2014, pp. 3829–3835.
- [6] P. Millán, L. Orihuela, C. Vivas, F. Rubio, D. V. Dimarogonas, and K. H. Johansson, "Sensor-Network-based Robust Distributed Control and Estimation," *Control Eng. Pract.*, vol. 21, no. 9, pp. 1238–1249, 2013.
- [7] A. Nedić, A. Olshevsky, and M. G. Rabbat, "Network Topology and Communication-Computation Tradeoffs in Decentralized Optimization," *Proc. IEEE*, vol. 106, no. 5, pp. 953–976, 2018.

- [8] X. Lian, C. Zhang, H. Zhang, C.-J. Hsieh, W. Zhang, and J. Liu, "Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent," *Adv. in Neur. Inf. Proc. Syst.*, vol. 30, 2017.
- [9] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and Cooperation in Networked Multi-agent Systems," *Proc. IEEE*, vol. 95, no. 1, pp. 215–233, 2007.
- [10] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of Groups of Mobile Autonomous Agents using Nearest Neighbor Rules," *IEEE Trans. Autom. Control*, vol. 48, no. 6, pp. 988–1001, 2003.
- [11] L. Xiao and S. Boyd, "Fast Linear Iterations for Distributed Averaging," *Syst. and Control Lett.*, vol. 53, no. 1, pp. 65–78, 2004.
- [12] K. Cai and H. Ishii, "Average Consensus on General Strongly Connected Digraphs," *Automatica*, vol. 48, no. 11, pp. 2750–2761, 2012.
- [13] A. Y. Kibangou, "Graph Laplacian based Matrix Design for Finite-time Distributed Average Consensus," in *IEEE Amer. Control Conf.*, 2012, pp. 1901–1906.
- [14] J. M. Hendrickx, R. M. Jungers, A. Olshevsky, and G. Vankeerberghen, "Graph Diameter, Eigenvalues, and Minimum-time Consensus," *Automatica*, vol. 50, no. 2, pp. 635–640, 2014.
- [15] T. Charalambous, Y. Yuan, T. Yang, W. Pan, C. N. Hadjicostis, and M. Johansson, "Distributed Finite-time Average Consensus in Digraphs in the Presence of Time Delays," *IEEE Trans. Control Netw. Syst.*, vol. 2, no. 4, pp. 370–381, 2015.
- [16] A. D. Domínguez-García and C. N. Hadjicostis, "Coordination and Control of Distributed Energy Resources for Provision of Ancillary Services," in *IEEE Int. Conf. Smart Grid Commun.*, 2010, pp. 537–542.
- [17] M. Franceschelli, A. Giua, and C. Seatzu, "Distributed Averaging in Sensor Networks based on Broadcast Gossip Algorithms," *IEEE Sensors J.*, vol. 11, no. 3, pp. 808–817, 2010.
- [18] C. N. Hadjicostis and T. Charalambous, "Average Consensus in the Presence of Delays in Directed Graph Topologies," *IEEE Trans. Autom. Control*, vol. 59, no. 3, pp. 763–768, 2013.
- [19] 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.
- [20] C. N. Hadjicostis, N. H. Vaidya, and A. D. Domínguez-García, "Robust Distributed Average Consensus via Exchange of Running Sums," *IEEE Trans. Autom. Control*, vol. 61, no. 6, pp. 1492–1507, 2015.
- [21] Y. Chen, R. Tron, A. Terzis, and R. Vidal, "Corrective Consensus: Converging to the Exact Average," in *IEEE Conf. Decision Control*, 2010, pp. 1221–1228.
- [22] E. Seneta, *Non-negative Matrices and Markov Chains*. Springer, 2006.
- [23] D. Kempe, A. Dobra, and J. Gehrke, "Gossip-based Computation of Aggregate Information," in *IEEE Symp. Found. of Comput. Sci.*, 2003, pp. 482–491.
- [24] S. Lin, D. J. Costello, and M. J. Miller, "Automatic-Repeat-reQuest Error-Control Schemes," *IEEE Commun. Magaz.*, vol. 22, no. 12, pp. 5–17, 1984.
- [25] E. Krouk and S. Semenov, *Modulation and Coding Techniques in Wireless Communications*. John Wiley & Sons, 2011.
- [26] A. Leon-García and I. Widjaja, *Communication Networks: Fundamental Concepts and Key Architectures*. McGraw-Hill New York, 2000, vol. 2.