Event-Triggered Approximate Byzantine Consensus With Multi-Hop Communication

Liwei Yuan and Hideaki Ishii, Fellow, IEEE

Abstract—In this article, we consider a resilient consensus problem for the multi-agent network where some of the agents are subject to Byzantine attacks and may transmit erroneous state values to their neighbors. In particular, we develop two event-triggered update schemes to tackle this problem as well as reduce the communication for each agent. Our approach is based on the mean subsequence reduced (MSR) algorithm with agents being capable to communicate with multi-hop neighbors through relaying process. Since communication delays are critical in such an environment, we provide necessary graph conditions for the proposed algorithms to perform well with delays in the communication. Moreover, a novel multi-hop relay scheme with event-triggered feature is proposed. It can reduce more transmissions than the conventional one-hop event-triggered algorithm. We also highlight that through multi-hop communication, the network connectivity can be reduced especially in comparison with the common one-hop communication case. Lastly, we show the effectiveness of the proposed algorithms by numerical examples.

Index Terms—Resilient consensus, distributed algorithms, byzantine attack, event-triggered updates, multi-hop communication.

I. INTRODUCTION

Consensus is one of the most fundamental problems in distributed algorithms [1]. Such a problem requires the agents in a network to reach a common value while agents can only interact with their neighbors. As concerns for cyber security have risen in general, multi-agent consensus problems in the presence of adversary agents creating failures and attacks have attracted much attention; see, e.g., [2], [3], [4], [5], [6], [7], [8], [9], [10]. One class of interdisciplinary problems that have been studied in different areas such as computer science, systems control, and signal processing is that of resilient consensus [3], [8], [11], [12], [13]. In these works, the adversary agents are categorized into basically two types: Malicious agents and Byzantine agents. These agents are capable to manipulate their data arbitrarily. Malicious agents are limited as they must broadcast the same messages to their neighbors, while Byzantine agents represent the worst-case and are more capable as they can send individual messages to different neighbors (e.g., [1], [3]). Among the security related works, Byzantine attacks are the most studied type of attacks due to their full capability on manipulating the compromised agents’ behaviors [14], [15], [16], [17]. Various distributed algorithms under Byzantine attacks have been studied for different problems, e.g., binary hypothesis testing [18], [19], resilient synchronization of pulse-coupled oscillators [13], collaborative spectrum sensing in cognitive radio networks [20], and so on.

In this article, we study the approximate Byzantine consensus using a mean subsequence reduced (MSR) algorithm. Such algorithms have been well studied in the fields of fault-tolerant techniques for multi-agent systems (e.g., [3], [11], [21]). A basic assumption in MSR algorithms is the knowledge regarding an upper bound on the maximum number of malicious agents among the neighbors; this bound is denoted by $f$ throughout this article. Then, at each iteration, each node removes the $f$ largest values and $f$ smallest values from neighbors to avoid being influenced by such potentially faulty values. Moreover, the graph property called robustness is shown to be critical for the network structure, guaranteeing the success of resilient consensus algorithms under malicious attacks [3], [12]. In [11], the authors proposed a tight necessary and sufficient condition for Byzantine consensus, where such a condition can also be interpreted using the notion of robustness. However, such robustness requires the network to be relatively dense and complex. Therefore, how to enhance resilience of a sparse network without changing the original network topology has become an urgent problem.

There are several works that tackled this problem by introducing the techniques of multi-hop communication where the nodes relay the data received from their neighbors to others [4], [22], [23]. Such techniques are commonly used in the areas of wireless communication [24], [25], computer science [1], and systems control [26]. It is clear that with multi-hop communication, each node can have more information for updates compared to the one-hop case. Thus, the network may have more resilience against adversary nodes. For instance, the works [27], [28] pursued an approach based on detection of malicious agents in the network; compared to MSR algorithms, which do not have such detection capabilities, the algorithms there are applicable to more sparse networks with the same level of tolerance against malicious agents. Furthermore, in [4], by introducing multi-hop communication in MSR algorithms, the authors solved the Byzantine consensus problem with a weaker condition on network structures compared to that derived...
under the one-hop communication case \[11\]. In \[22\], the authors studied the asynchronous Byzantine consensus based on a flooded algorithm, where nodes relay their values over the entire network. Moreover, in our previous work \[29\], we studied the asynchronous Byzantine consensus using an algorithm which is of less complexity than that in \[22\]. To conclude, through multi-hop communication, the connectivity requirement becomes less stringent for guaranteeing the same level of resilience as for the one-hop case. This is enabled by increasing the amount of data exchanged among agents through message relaying.

In this article, we aim to reduce the transmissions for the agents using the multi-hop weighted MSR algorithm of \[23\] through the use of event-triggered protocols \[30\], \[31\]. The event-trigger technique is widely used in optimization problems and control systems. For instance, the work \[32\] proposed a dynamic gradient update approach for distributed machine learning, which can adaptively skip the gradient calculations and learn with reduced communication and computation. On the other hand, event-based protocols have been developed for conventional consensus without adversary agents in, e.g., \[33\], \[34\], \[35\]. For the case when some agents are under malicious attacks, the work \[36\] proposed two event-based MSR algorithms under one-hop communication to reduce the transmissions. Among these works, event-triggered schemes have shown their effectiveness in reducing the transmissions for the agents even under adversarial environments. Generally, time delay can be a critical factor affecting the performance of agents in the multi-hop communication. It is vital to analyze the performance of the proposed method under the communication delays.

The main contributions of this article can be summarized as follows: First, we introduce two event-triggered protocols to the multi-hop weighted MSR algorithm, which can solve the Byzantine consensus problem with delays in the communication between agents. We show that both event-triggered schemes have different advantages on the consensus error bound over the other scheme. We demonstrate that this depends on the situations of the agent-systems in terms of transmission delay bound and local update frequency.

Second, to the best of our knowledge, our work is the first to consider event-trigger schemes for the multi-hop message relaying for multi-agent consensus. In this context, we find it worthwhile to propose two different models for the relaying of the event-triggered values. The first model is that we separate the event-trigger scheme and the relaying process, i.e., an agent relays the neighbors’ values as soon as it receives them. The second model is more practical and energy-efficient. Here, agents send their own state values along with relayed values only when the difference between the current value and the past communicated value exceeds a given threshold.

Third, through simulations, we show that the agents’ transmissions can be significantly reduced compared to the multi-hop algorithm with the regular communication \[29\] and the one-hop event-triggered algorithm \[36\]. Furthermore, compared to the one-hop MSR algorithm with or without event-triggered protocols \[36\], \[3\], the connectivity requirement for our algorithm is less stringent. Besides, we analyze the performance of our algorithm with delays in communication, which is a case not studied in \[36\]. In addition, we consider the Byzantine attacks, and it is more adversarial than the malicious model studied in \[36\].

The rest of this article is organized as follows. Section II outlines preliminaries on graphs and the system model. Section III presents the event-triggered multi-hop MSR algorithm and the definition of strictly robust graphs with multi-hop communication. In Sections IV and V, we derive a condition under which the two proposed algorithms reach resilient consensus under asynchronous updates with delays. The two algorithms have certain advantages over the other one depending on the situation. Section VI provides numerical examples to show the effectiveness of the proposed algorithms as well as the comparison between the two schemes. Lastly, Section VII concludes the article. A preliminary version of this article appeared as \[37\].

The current article contains all the proofs of the theoretical results and further discussions.

II. PRELIMINARIES

In this section, we introduce the graph notions, the system model, as well as the definition of resilient consensus studied in this article.

A. Network Model

Consider the directed graph \(G = (\mathcal{V}, \mathcal{E})\) consisting of the node set \(\mathcal{V} = \{1, \ldots, n\}\) and the edge set \(\mathcal{E} \subset \mathcal{V} \times \mathcal{V}\). The edge \((j, i) \in \mathcal{E}\) indicates that node \(i\) can get information from node \(j\).

A path from node \(i_1\) to \(i_m\) is a sequence of distinct nodes \((i_1, i_2, \ldots, i_m)\), where \((i_j, i_{j+1}) \in \mathcal{E}\) for \(j = 1, \ldots, m - 1\). Such a path is referred to as an \((m - 1)\)-hop path (or a path of length \(m - 1\)) and also as \((i_1, i_m)\)-path when length is not relevant but the source node \(i_1\) and the destination node \(i_m\) are.

We also say that node \(i_m\) is reachable from node \(i_1\).

For node \(i\), let \(\mathcal{N}_{i\ell}^+\) be the set of nodes that can reach node \(i\) via at most \(\ell\)-hop paths, where \(\ell\) is a positive integer. Also, let \(\mathcal{N}_{i\ell}^-\) be the set of nodes that are reachable from node \(i\) via at most \(\ell\)-hop paths. The \(\ell\)-th power of the graph \(G\), denoted by \(G^\ell\), is a multigraph\(^1\) with the same vertices as \(G\) and a directed edge from node \(j\) to node \(i\) is defined by a path of length at most \(\ell\) from \(j\) to \(i\) in \(G\). The adjacency matrix \(A = [a_{ij}]\) of \(G^\ell\) is given by \(a_{ij} \leq 1\) if \(j \in \mathcal{N}_{i\ell}^+\) and otherwise \(a_{ij} = 0\), where \(a_{ij} > 0\) is a fixed lower bound. We assume that \(\sum_{j=1,j \neq i}^n a_{ij} \leq 1\).

Node \(i_1\) can send messages of its own to its \(\ell\)-hop neighbor \(i_{\ell+1}\) via different paths. We represent a message as a tuple \(m = (w, P)\), where \(w = value(m) \in \mathbb{R}\) is the message content, and \(P = path(m)\) indicates the path via which message \(m\) is transmitted. Moreover, nodes \(i_1\) and \(i_{\ell+1}\) are the message source and destination, respectively. When the source \(i_1\) sends the message, \(P\) is a path vector of length \(\ell + 1\) with the source being \(i_1\) and other entries being empty. Then the one-hop neighbor \(i_2\) receives this message from \(i_1\), and it stores the value of node \(i_1\) for consensus and relays the value of node \(i_1\) to all the one-hop neighbors of \(i_2\) with the second entry of \(P\) being \(i_2\) and other entries unchanged. This relay procedure will continue until every

\(^1\)In a multigraph, two nodes can have multiple edges between them.
entry of $P$ of this message is occupied, i.e., this message reaches node $i_{l+1}$. We denote by $\mathcal{V}(P)$ the set of nodes in $P$.

**B. Update Rule**

In graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the node set $\mathcal{V}$ is partitioned into the set of normal nodes $\mathcal{N}$ and the set of adversary nodes $\mathcal{A}$, where $\left|\mathcal{N}\right| = N$. The partition is unknown to the normal nodes at all times.

In this article, we assume that every normal node $i$ updates its value at least once in every $\theta \geq 1$ steps. The updates can be synchronous ($\theta = 1$) or asynchronous. The update rule for normal agent $i$ is described by

$$x_i[k + 1] = x_i[k] + u_i[k],$$

where $x_i[k] \in \mathbb{R}$ is the state and $u_i[k]$ is the control input. If there is no update at time $k$, then $u_i[k] = 0$. If an update takes place, it is given by

$$u_i[k] = \sum_{j \in \mathcal{N}_i} a_{ij}[k](\hat{x}_j[k] - x_i[k]). \quad (1)$$

Here, $\hat{x}_j[k] \in \mathbb{R}$ is an auxiliary state, representing the last communicated state of node $j$ at time $k$. It is defined as

$$\hat{x}_j[k] = x_j[t_{jk}], \quad k \in \mathcal{K}_{kj},$$

where $t_{0j}, t_{1j}, \ldots$ denote the transmission times of node $j$ determined by the triggering function to be given below. The initial values $x_i[0]$, $x_j[0]$ are given, and $a_{ij}[k]$ is the weight for the edge $(i, j)$. Note that at initial time, $\hat{x}_j[0]$ need not be the same as $x_i[0]$.

Let $a_{ij}[k] = 1 - \sum_{j \in \mathcal{N}_i} a_{ij}[k]$. Assume that $\gamma \leq a_{ij}[k] < 1$ if $a_{ij}[k] \neq 0$ or $i = j$ for $i, j \in \mathcal{V}$, where $0 < \gamma < 1$. In the resilient consensus algorithm to be introduced, the neighbors whose values are used for updates change over time and, hence, the weights $a_{ij}[k]$ are time varying.

We now introduce the triggering function. Denote the error at time $k$ between the updated state $x_i[k + 1]$ and the auxiliary state $\hat{x}_i(k)$ by $e_i[k] = \hat{x}_i[k] - x_i[k + 1]$ for $k \geq 0$. Then, let

$$f_i[k] = |e_i[k]| = (c_0 + c_1[k]), \quad (2)$$

where $c_0 \geq 0$ is a constant and $c_1[k]$ takes nonnegative and decreasing values with $c_1[k] \rightarrow 0$ in finite time. The roles of $c_0$ and $c_1[k]$ are to reduce the triggering frequency, and especially $c_1[k]$ allows the threshold to be large in the initial phase. Each node $i$ will check this function and whenever it finds $f_i[k]$ to be positive, it will transmit its new state $x_i[k + 1]$ to its neighbors.

We employ the control input taking account of possible delays in the transmission. Thus, we extend (1) as

$$u_i[k] = \sum_{j \in \mathcal{N}_i} a_{ij}[k](\hat{x}_j[k] - \tau_{kj}(k) - x_i[k]), \quad (3)$$

where $\hat{x}_j[k]$ denotes the value of node $j$ at time $k$ sent along path $P$ and $\tau_{kj}(k) \in \mathbb{Z}_+$. denotes the delay in this $(j, i)$-path $P$ at time $k$. The delays are time varying and may be different in each path. We assume the common upper bound $\tau$ on any normal path $P$, over which all internal nodes are normal, as

$$0 \leq \tau_{kj}(k) \leq \tau, \quad j \in \mathcal{N}_i^{-}, \quad k \in \mathbb{Z}_+. \quad (4)$$

Although we impose this bound on the delays for message transmissions, the normal nodes need neither the value of this bound nor the information whether a path $P$ is a normal one or not. Also, there is no constraint on the size of $\tau$.

For multi-hop communication under event-trigger schemes, there are different options for the timing of relaying neighbors’ data. Here we propose the following two relay models that can be employed in the proposed multi-hop algorithm:

(i) **Immediate relay model**: Each node immediately relays all the messages received at that time step to its one-hop neighbors.

(ii) **Package relay model**: Each node relays all the recently received messages along with its own values (e.g., in a message package) to its one-hop neighbors when its own event is triggered.

Among the two models, clearly, the package relay model requires less frequent message transmissions and may be a more natural model in the event-based algorithm studied here.

We note however that with this model, it must be guaranteed that transmitted data reach their multi-hop neighbors within $\tau$ time steps as imposed by (4). Hence, for technical reasons, we assume that each agent makes at least one transmission within $\lambda$ time steps after receiving a message to be relayed from a neighbor, where $\lambda$ can be a large number.

This is to cope with the situation where no event is triggered by any of the agents. This can occur since the event triggering function only takes account of the local states. We will illustrate the difference of the effects of the two relay models through simulations later.

**C. Threat Model**

Next, we introduce the threat model studied here.

**Definition 1** (f-total/f-local set): The set of adversary nodes $\mathcal{A}$ is said to be $f$-total if it contains at most $f$ nodes, i.e., $|\mathcal{A}| \leq f$.

Similarly, it is said to be $f$-local (in $l$-hop neighbors) if any normal node $i \in \mathcal{N}$ has at most $f$ adversary nodes as its $l$-hop neighbors, i.e., $|\mathcal{N}_i^{-} \cap \mathcal{A}| \leq f$, $\forall i \in \mathcal{N}$.

**Definition 2** (Byzantine nodes): An adversary node $i \in \mathcal{A}$ is said to be a Byzantine node if it can arbitrarily modify its own value and relayed values, and moreover, it can send different values to its neighbors at each iteration.\(^2\)

As commonly done in the literature, we assume that each normal node knows the value of $f$ and the topology information of the graph up to $l$ hops. In the multi-hop setting, it is important to impose the following assumption.

**Assumption 1**: Each Byzantine node $i$ cannot manipulate the path values in the messages containing its own state $x_i[k]$ and those that it relays.

This is introduced for ease of analysis, but is not a strong constraint. In fact, manipulating message paths can be easily detected and hence does not create problems. For more details, see the discussions in [23].

\(^2\)Here a Byzantine node can also decide not to send any value. This behavior corresponds to the omission/crash model.
D. Resilient Asymptotic Consensus

We now introduce the type of consensus among the normal agents to be sought in this article.

**Definition 3**: Given \( c \geq 0 \), if for any possible sets and behaviors of the adversary agents and any state values of the normal nodes, the following two conditions are satisfied, then we say that the normal agents reach resilient consensus at the error level \( c \):

1) Safety: There exists a bounded safety interval \( S \) determined by the initial values of the normal agents such that \( x_i[k] \in S, \forall i \in N, k \in \mathbb{Z}_+. \)

2) Agreement: For all \( i, j \in N \), it holds that

\[
\limsup_{k \to \infty} |x_i[k] - x_j[k]| \leq c.
\]

The problem of this article is to design distributed algorithms for the normal agents in the network to achieve resilient consensus at a prespecified error level \( c \) when adversarial agents follow the \( f \)-local Byzantine model. Note that because of the event-trigger scheme based on the triggering function in (2), we allow a certain level of error in consensus among the agents. It is clear that with \( c_0 > 0 \) in (2), even if the error between states of neighboring agents are nonzero, the most up-to-date states may not be transmitted over \( l \)-hop communication.

III. EVENT-TRIGGERED ALGORITHM DESIGN

In this section, we outline the structure of the event-triggered multi-hop weighted MSR (MW-MSR) algorithm. Then we define the strictly robust graphs with \( l \) hops, which is crucial for guaranteeing Byzantine consensus [29].

A. Asynchronous Event-Triggered MW-MSR Algorithm

At each time \( k \), each normal node \( i \) follows the three steps for updates as outlined below:

1. **Receive step**: Node \( i \) receives neighbors’ values through different paths (described in (3)) and chooses to update its state or not with an independent probability \( 0 < p_i < 1 \). If it chooses to update, then it proceeds to step 2. Otherwise, it keeps its value as \( x_i[k+1] = x_i[k] \).

2. **Update step**: Node \( i \) updates its value \( x_i[k+1] \) according to Algorithm 1 using the values most recently received from neighbors and its own value \( x_i[k] \).

3. **Transmit step**: Node \( i \) checks the value of \( f_i[k] \) and sets the value of \( \hat{x}_i[k+1] \) as

\[
\hat{x}_i[k+1] = \begin{cases} 
  x_i[k+1], & \text{if } f_i[k] > 0, \\
  x_i[k], & \text{otherwise}.
\end{cases}
\]

Here, the auxiliary variable will be updated only when the current value has varied enough to exceed a threshold, and only at this time the node sends its value and the relayed values over each \( l \)-hop path to node \( j \in N_i^l \).

In the Transmit step and Receive step, the nodes exchange messages with others that are up to \( l \) hops away. Then in the Update step, node \( i \) updates its state using Algorithm 1. Note that the adversary nodes may deviate from this specification as we described earlier.

Algorithm 1: MW-MSR Algorithm

1) At time \( k \), normal node \( i \) obtains the most recently received messages of the nodes in \( N_i^l \) and itself, whose set is denoted by \( M_i[k] \), and sorts the values in \( M_i[k] \) in an increasing order.

2) (a) Define two subsets of \( M_i[k] \) based on the message values:

\[
\overline{M}_i[k] = \{ m \in M_i[k] : \text{value}(m) > x_i[k] \},
\]

\[
\underline{M}_i[k] = \{ m \in M_i[k] : \text{value}(m) < x_i[k] \}.
\]

(b) Then, let \( \overline{R}_i[k] = \overline{M}_i[k] \) if the cardinality of a minimum cover of \( \overline{M}_i[k] \) is less than \( f \), i.e., \( |T(\overline{M}_i[k])| < f \). Otherwise, let \( \overline{R}_i[k] \) be the largest sized subset of \( \overline{M}_i[k] \) such that (i) for all \( m \in \overline{M}_i[k] \setminus \overline{R}_i[k] \) and \( m' \in \overline{R}_i[k] \) we have value\((m) \leq \text{value}(m')\), and (ii) the cardinality of a minimum cover of \( \overline{R}_i[k] \) is exactly \( f \), i.e., \( |T(\overline{R}_i[k])| = f \).

(c) Similarly, we can get \( \underline{R}_i[k] \) from \( \underline{M}_i[k] \), which contains the smallest values.

(d) Finally, let \( R_i[k] = \overline{R}_i[k] \cup \underline{R}_i[k] \).

3) Node \( i \) updates its value as follows:

\[
x_i[k+1] = \sum_{m \in D_i[k]} a_i[k] \text{value}(m),
\]

where \( a_i[k] = 1/|D_i[k]| \) and \( D_i[k] = M_i[k] \setminus R_i[k] \).

One important feature here to further reduce the amount of data in each transmission when an event is triggered is to require that the nodes can send only the relayed values that have changed since last event.

B. The Notion of Strictly Robust Graphs

The notion of graph robustness was first introduced in [3], and it was proved that graph robustness gives a tight condition guaranteeing resilient consensus using MSR-based algorithms. In [23], we generalized this notion to the multi-hop case, where nodes can exchange values with their \( l \)-hop neighbors through different paths. Its definition is as follows.

**Definition 4**: A directed graph \( G = (V, E) \) is said to be \((r, s)\)-robust with \( l \) hops with respect to a given set \( F \subset V \), if for every pair of nonempty disjoint subsets \( V_1, V_2 \subset V \), at least one of the following conditions holds:

1) \( |Z_{V_1}^r| = 0 \); \hspace{2cm} 2) \( |Z_{V_1}^s| = 0 \); \hspace{2cm} 3) \( |Z_{V_1}^r| + |Z_{V_2}^r| \geq s \),

where \( Z_{V_a}^r \) is the set of nodes in \( V_a \) \((a = 1, 2)\) that have at least \( r \) vertex-independent paths of at most \( l \) hops originating from nodes outside \( V_a \) and all these paths do not have any nodes in set \( F \) as intermediate nodes (i.e., the nodes in \( F \) can be source or destination nodes in these paths). Moreover, if the graph \( G \) satisfies this property with respect to any set \( F \) following the \( f \)-total model, then we say that \( G \) is \((r, s)\)-robust with \( l \) hops (under the \( f \)-total model). When \( G \) is \((r, 1)\)-robust with \( l \) hops, we will just say \( G \) is \( r \)-robust with \( l \) hops.
Intuitively speaking, for any set \( F \subset V \) and for node \( i \in V_i \) to have the above-mentioned property, they should satisfy two conditions: (i) At least \( r \) source nodes outside \( V_i \); (ii) at least one independent path of length at most \( l \) hops from each of the \( r \) source nodes to node \( i \), where such a path does not contain any internal nodes from the set \( F \).

To deal with the Byzantine model, we need to focus on the subgraph consisting of only the normal nodes. For the one-hop algorithms in [3] and [11], the graph condition that the normal network is \((f+1)\)-robust is proved to be necessary and sufficient for achieving resilient consensus under \( f \)-total Byzantine model. In [29], we extended this notion to the multi-hop setting and defined it as \( r \)-strictly robust graph with \( l \) hops. Its definition is given as follows.

**Definition 5:** Let \( F \) be a subset of vertices in \( G \) and denote the subgraph of \( G \) induced by vertex set \( V \setminus F \) as \( G_{V \setminus F} \). Then graph \( G \) is said to be \( r \)-strictly robust with \( l \) hops with respect to \( F \) if the induced subgraph \( G_{V \setminus F} \) is \( r \)-robust with \( l \) hops. If graph \( G \) satisfies this property with respect to any set \( F \) following the \( f \)-total/local model, then we say that \( G \) is \( r \)-strictly robust with \( l \) hops under the \( f \)-total/local model. When it is clear from the context, we just say \( G \) is \( r \)-strictly robust with \( l \) hops.

Generally, robustness of a graph increases as the relay range \( l \) increases. See the examples in Fig. 1. Note that graph robustness with multi-hop communication needs to be checked for every possible set \( F \) satisfying the \( f \)-total/local model. For example, the network in Fig. 1(a) is not 2-strictly robust with one hop but is 2-strictly robust with 2 hops. If we remove node 5, for the node sets \( \{1, 2\} \) and \( \{3, 4\} \), neither node 1 nor node 2 has 2 neighbors outside the set \( \{1, 2\} \). The situation is the same for the set \( \{3, 4\} \), and hence, the remaining network is not 2-robust with one hop. Therefore, the 5-node network is not 2-strictly robust with one hop according to Definition 5. However, the situation changes for the two-hop case. If we remove node 5, for the node sets \( \{1, 2\} \) and \( \{3, 4\} \), node 1 has 2 independent two-hop paths originating from the nodes outside of the set \( \{1, 2\} \) (i.e., the paths ‘3 \( \rightarrow \) 2 \( \rightarrow \) 1’ and ‘4 \( \rightarrow \) 1’). After checking all the set partitions, the remaining network satisfies the condition for being 2-robust with 2 hops. We can also check that by removing any node in the 5-node network, the remaining network is 2-robust with 2 hops. Therefore, the 5-node network is 2-strictly robust with 2 hops.

We note that to check whether a given graph satisfies the strict robustness is not difficult for the unbounded path length case, as we discuss in the following: First, the level of robustness is constrained by the in-degrees of the nodes. For instance, to achieve resilient consensus under the \( f \)-total malicious model, the minimum in-degree of the nodes needs to be at least \( 2f \). On the other hand, under the \( f \)-local Byzantine model, the minimum in-degree of the nodes is at least \( 2f+1 \). Second, in our previous work [29], we proved that when the \( l \)-hop communication is unbounded (\( l \geq \ell \)), our graph condition \([(f+1)\)-strictly robust with \( l \) hops] is equivalent to the two conditions in [14] for undirected graphs, which are (i) \( n \geq 3f+1 \) and (ii) the node-connectivity of the graph is \( 2f+1 \). These two conditions are fairly easy to check for undirected graphs. Note that [14] only considered the synchronous Byzantine consensus, but our work can solve the asynchronous Byzantine consensus problem.

### IV. Consensus Analysis

In this section, we first prove the convergence of the asynchronous event-triggered MW-MSR algorithm under Byzantine attacks. Then we discuss the effects of different relay models on the performance of the proposed algorithm.

To prove the convergence, we introduce two kinds of minimum and maximum of the states of the normal agents. Denote the state vector and the transmitted state vector of normal agents at time \( k \) by \( x^N[k] \) and \( \hat{x}^N[k] \), respectively.

First, we denote the minimum and maximum of the states of the normal agents from time \( k - \tau \) to time \( k \) as

\[
\overline{x}_r[k] = \max \{x^N[k], x^N[k-1], \ldots, x^N[k-\tau]\},
\]

\[
\underline{x}_r[k] = \min \{x^N[k], x^N[k-1], \ldots, x^N[k-\tau]\},
\]

respectively. Next, we denote the joint minimum and maximum of the states and the transmitted states of the normal agents from time \( k - \tau \) to time \( k \) respectively, as

\[
\overline{x}_r[k] = \max \{\overline{x}_r[k], \hat{x}^N[k], \ldots, \hat{x}^N[k-\tau]\},
\]

\[
\underline{x}_r[k] = \min \{\underline{x}_r[k], \hat{x}^N[k], \ldots, \hat{x}^N[k-\tau]\}.
\]

We are ready to state the first main result of the article.

**Theorem 1:** Consider a directed graph \( G = (V, E) \) with \( l \)-hop communication, where each normal node updates its value according to the asynchronous event-triggered MW-MSR algorithm. Under the \( f \)-local Byzantine model, the normal nodes reach resilient consensus at an error level \( c \) if and only if the underlying graph is \((f+1)\)-strictly robust with \( l \) hops. Moreover, the safety interval is given by \( S = \overline{x}_r[0], \overline{x}_r[0] \), and the consensus error level \( c \) is achieved if the parameter \( c_0 \) in the triggering function (2) satisfies

\[
c_0 \leq \frac{\lambda N^\theta}{4N^\theta}.\]

**Proof:** (Necessity) This part follows from our previous work [29], which considers the special case without the triggering function, that is, \( c_0 = c_1[k] = 0 \).

(Sufficiency) First, we show by induction that the safety condition is satisfied. Note that the update rule (6) in Algorithm 1 can be rewritten as

\[
x_i[k + 1] = a_i[k]x_i[k] + \sum_{j \in D_i[k]} a_{ij}[k]\hat{x}_j[k - \tau_{ij}[k]],
\]

where \( a_i[k] = 1/|D_i[k]| \). At time \( k = 0 \), it is clear by definition that \( x_i[0], \hat{x}_i[0] \in S \). We first show that \( \overline{x}_r[k] \) is nonincreasing.
in time. From (9), we have $x_i[k+1] \leq \bar{\tau}_r[k]$ for all $i \in \mathcal{N}$ since the values larger than $\bar{\tau}_r[k]$ are ignored in step 2 of Algorithm 1. Moreover, by (5), it follows that $\bar{\tau}_r[k+1] \leq \bar{\tau}_r[k]$ for all $i \in \mathcal{N}$. Together, we have $\bar{\tau}_r[k+1] \leq \bar{\tau}_r[k]$. We can similarly prove that $\bar{\tau}_r[k]$ is nondecreasing in time.

We next show the consensus part. Note that for time $k \in (t^{\ell}_k, t^{\ell+1}_k)$ between two triggering instants, we have $f_i[k] \leq 0$. Moreover, for the neighbor node $j \in \mathcal{N}^z_i$, if $f_j[k] > 0$, then we have $\bar{\tau}_j[k+1] = x_j[k + 1]$. If $f_j[k] \leq 0$, then $\bar{\tau}_j[k+1] = \bar{\tau}_j[k] = x_j[k+1] + e_j[k]$. As a result, it holds $\bar{\tau}_j[k] = x_j[k] + \bar{\tau}_j[k-1]$ for $k \geq 1$, where

$$\bar{\tau}_j[k] = \begin{cases} e_j[k], & \text{if } f_i[k] \leq 0, \\ 0, & \text{otherwise}. \end{cases}$$

Note that

$$|e_j[k]| \leq c_0 + c_1[k], \quad \forall k \geq 0. \quad (10)$$

Then, we can write (9) as

$$x_i[k+1] = a_i[k]x_i[k] + \sum_{j \in M_i[k], R_i[k]} a_i[k](x^P_j[k] - \tau_{ij}^P[k] - \bar{\tau}_j[k] - 1).$$

This can be bounded as

$$x_i[k+1] \leq a_i[k]x_i[k] + \sum_{j \in M_i[k], R_i[k]} a_i[k](x^P_j[k] - \tau_{ij}^P[k] - \bar{\tau}_j[k] - 1)) \leq \bar{\tau}_r[k] + \max_{j \in M_i[k], R_i[k]} \bar{\tau}_j[k] - \tau_{ij}^P[k] - 1).$$

Thus, by (10), letting $c_1[k] = c_0[0]$ for $k < 0$, we have

$$x_i[k+1] \leq \bar{\tau}_r[k] + c_0 + c_1[k - \tau - 1]. \quad (12)$$

Similarly, we have

$$x_i[k+1] \geq \bar{\tau}_r[k] - c_0 - c_1[k - \tau - 1].$$

Let $V[k] = \bar{x}_r[k] - \bar{\tau}_r[k]$. Then, define two sequences by

$$\bar{\tau}_0[k+1] = \bar{\tau}_0[k] + c_0 + c_1[k - \tau - 1],$$

$$x_0[k+1] = x_0[k] - c_0 - c_1[k - \tau - 1], \quad (13)$$

where $\bar{\tau}_0[0] = \bar{\tau}_r[0] - \sigma_0$, and $x_0[0] = x_r[0] + \sigma_0$ with $\sigma_0 = \sigma V[0]$. Then the following inequalities hold:

$$\bar{\tau}_r[k] \leq \bar{\tau}_0[k] + \sigma_0,$$

$$\bar{\tau}_r[k] \geq x_0[k] - \sigma_0. \quad (14)$$

We show $\bar{\tau}_r[k] \leq \bar{\tau}_0[k] + \sigma_0$ by induction, and $\bar{\tau}_r[k] \geq x_0[k] - \sigma_0$ can be proved in a similar way. When $k = 0$, we clearly have $\bar{\tau}_r[0] = \bar{\tau}_0[0] + \sigma_0$. Suppose that (14) holds. Then, we have at time $k + 1$

$$\bar{\tau}_r[k+1] = \max(x^N[k+1], x^N[k], \ldots, x^N[k+1 - \tau]) \leq \bar{\tau}_r[k] + c_0 + c_1[k - \tau - 1] \leq (\bar{\tau}_0[k] + \sigma_0) + c_0 + c_1[k - \tau - 1] = \bar{\tau}_0[k+1] + \sigma_0.$$
The last inequality holds since $\varepsilon + \sigma = 1/2$ and $\varepsilon_0[k] < \varepsilon_0[0] = \varepsilon V[0]$. Thus, we have proved (17).

So far, we have shown that the two sets $Z_1(k, \varepsilon_0[k])$ and $Z_2(k, \varepsilon_0[k])$ are disjoint. Notice that the network is $(f + 1)$-strictly robust with $l$ hops w.r.t. any set $F$ following the $f$-local model and the set of Byzantine nodes $A$ also satisfies the $f$-local model. Hence, the network is $(f + 1)$-strictly robust with $l$ hops w.r.t. the set $A$ and at least one of the conditions in Definition 4 for robustness holds. Therefore, if the two sets are both nonempty, then for these two nonempty disjoint sets $Z_1(k, \varepsilon_0[k])$ and $Z_2(k, \varepsilon_0[k])$, one of them has a normal agent with at least $f + 1$ independent normal paths originating from some normal nodes outside.

Suppose that normal node $i \in Z_1(k, \varepsilon_0[k])$ has the above-mentioned property. A similar argument holds when $i \in Z_2(k, \varepsilon_0[k])$. Now, we go back to the update rule (11) for node $i$ and rewrite it by partitioning the neighbor set into two parts: those that belong to $Z_1(k, \varepsilon_0[k])$ and those that do not. Node $i$ has at least $f + 1$ independent normal paths originating from the normal nodes outside. According to Algorithm 1, it will use at least one value originating from the normal nodes outside $Z_1(k, \varepsilon_0[k])$; thus, we obtain

$$x_i[k + 1] = a_i[k]x_i[k] + \sum_{j \in \mathcal{D}_i[k] \cap Z_1} a_i[k]x_{ij}^P[k] - \tau_{ij}^P[k]$$

$$+ \sum_{j \in \mathcal{D}_i[k] \setminus Z_1} a_i[k]x_{ij}^P[k] - \tau_{ij}^P[k]$$

$$+ \sum_{j \in \mathcal{D}_i[k]} a_i[k]\hat{e}_{ij}[k] - \tau_{ij}^P[k] - 1$$

$$\leq a_i[k]\bar{\pi}_r[k] + \sum_{j \in \mathcal{D}_i[k] \cap Z_1} a_i[k]\bar{\pi}_r[k]$$

$$+ \sum_{j \in \mathcal{D}_i[k] \setminus Z_1} a_i[k](\bar{\pi}_0[k] - \varepsilon_0[k])$$

$$+ \sum_{j \in \mathcal{D}_i[k]} a_i[k]\hat{e}_{ij}[k] - \tau_{ij}^P[k] - 1.$$  

Combining (14) and the fact that $a_i[k]$ is lower bounded by $\gamma$, we have

$$x_i[k + 1] \leq (1 - \gamma)\bar{\pi}_r[k] + \gamma(\bar{\pi}_0[k] - \varepsilon_0[k])$$

$$+ c_0 + c_1[k - \tau - 1]$$

$$\leq (1 - \gamma)(\bar{\pi}_0[k] + \sigma_0) + \gamma(\bar{\pi}_0[k] - \varepsilon_0[k])$$

$$+ c_0 + c_1[k - \tau - 1]$$

$$\leq \bar{\pi}_0[k] + c_0 + c_1[k - \tau - 1] + (1 - \gamma)\sigma_0 - \gamma \varepsilon_0[k]$$

$$= \bar{\pi}_0[k - 1] + \varepsilon_0[k + 1]$$  

(18)

for $k = 0, 1, \ldots, N\theta - 1$, where the first inequality follows from the assumption that $Z_1(k, \varepsilon_0[k])$ is nonempty, and the equality follows from (13) and (15). The relation in (18) shows that once an update happens at node $i$, then this node will move out of $Z_1(k + 1, \varepsilon_0[k + 1])$. It is further noted that inequality (18) also holds for the normal nodes that are not in $Z_1(k, \varepsilon_0[k])$ at time $k$. This indicates that the nodes outside $Z_1(k, \varepsilon_0[k])$ will not move in $Z_1(k + 1, \varepsilon_0[k + 1])$. Similar results hold for the set $Z_2(k + 1, \varepsilon_0[k + 1])$.

Recall that the normal nodes update at least once for every $\theta$ steps. As a result, if the two sets $Z_1(k, \varepsilon_0[k])$ and $Z_2(k, \varepsilon_0[k])$ are both nonempty at time $k$, then after $N\theta$ time steps, all the normal nodes will be out of at least one of them. Suppose that $Z_1(k, \varepsilon_0[k])$ is empty. When such an event occurs at $k = 0$, clearly that $\pi_r[N\theta] \leq \bar{\pi}_0[N\theta] - \varepsilon_0[N\theta]$. From the definition of $V[k]$, we have

$$V[N\theta] = (\bar{\pi}_0[N\theta] - \varepsilon_0[N\theta]) - (\bar{\pi}_0[N\theta] - \sigma_0)$$

$$= \bar{\pi}_0[0] - \varepsilon_0[0] + 2c_0N\theta + 2\sum_{j = 0}^{N\theta - 1} c_1[j] - \varepsilon_0[N\theta] + \sigma_0$$

$$= (\bar{\pi}_0[0] - \sigma_0) - (\bar{\pi}_0[0] + \sigma_0) + 2c_0N\theta + 2\sum_{j = 0}^{N\theta - 1} c_1[j] - \varepsilon_0[N\theta] + \sigma_0$$

$$= V[0] - \sigma V[0] + 2c_0N\theta + 2\sum_{j = 0}^{N\theta - 1} c_1[j]$$

$$= (1 - \gamma N\theta \varepsilon - (1 - \gamma N\theta)\sigma)V[0]$$

$$= (1 - \gamma N\theta (\varepsilon + \sigma))V[0] + 2c_0N\theta + 2\sum_{j = 0}^{N\theta - 1} c_1[j].$$  

(19)

By (16), we have

$$V[hN\theta] \leq \left(1 - \frac{\gamma N\theta}{2}\right) V[0] + 2c_0N\theta + 2\sum_{j = hN\theta - 1}^{N\theta - 1} c_1[j].$$

If there are more updates by node $i$ after time $k = N\theta$, this argument can be extended further as

$$V[hN\theta] \leq \left(1 - \frac{\gamma N\theta}{2}\right) V[(h - 1)N\theta]$$

$$+ 2c_0N\theta + 2\sum_{j = (h - 1)N\theta - 1}^{hN\theta - 1} c_1[j].$$

Hence, we have

$$V[hN\theta] \leq \left(1 - \frac{\gamma N\theta}{2}\right)^h V[0] + \sum_{t = 0}^{h-1} \left(1 - \frac{\gamma N\theta}{2}\right)^{h-1-t}$$

$$\times \left(2c_0N\theta + 2\sum_{j = tN\theta - 1}^{(t+1)N\theta - 1} c_1[j]\right)$$

$$\leq \left(1 - \frac{\gamma N\theta}{2}\right)^h V[0] + 2c_0N\theta \left(1 - \frac{1 - (1 - \gamma N\theta/2)^h}{1 - (1 - \gamma N\theta/2)^h}\right)$$

$$+ 2\sum_{i = 0}^{h-1} \left(1 - \frac{\gamma N\theta}{2}\right)^{h-1-t} \sum_{j = tN\theta - 1}^{(t+1)N\theta - 1} c_1[j].$$  

(20)
Since $c_1[k] \to 0$ in finite time, there exists a finite time $h_0$ such that $c_1[k] = 0, k \geq h_0 N \theta$. Then, for $h \geq h_0$, we can obtain from (20)

$$\limsup_{h \to \infty} V[h N \theta] \leq \frac{2c_0 N \theta}{1 - \frac{\gamma N \theta}{2}} = \frac{4c_0 N \theta}{\gamma N \theta} \leq c.$$ 

The analysis is similar for the dynamics of $V[h N \theta + t], t = 0, 1, \ldots, N \theta - 1$, and we obtain as in (20):

$$\limsup_{h \to \infty} V[h N \theta + t] \leq \frac{4c_0 N \theta}{\gamma N \theta} \leq c. \quad (21)$$

This completes the proof.

This theorem extends our previous result in [29], where the agents do not employ event-triggered scheme and make transmissions and relaying after each update. The topological condition in terms of strict robustness remains the same as expected. However, the proof requires further analysis to take account of the errors arising from infrequent transmission due to the event-based communication.

An interesting feature for agents using the event-triggered scheme 1 is that performance in terms of consensus error is determined by the update frequency represented by the parameter $\theta$. That is, with smaller $\theta$, the updates become more frequent and thus the error in consensus decreases. This can be observed in the very last (21) in the proof and also, equivalently, in (8) in the theorem. On the other hand, we also notice that although delays can slow down the consensus process, they do not affect the consensus error bound. Again, in (8) and (21), neither $l$ nor $\tau$ appears. Similar observations have been made in [12], [23].

More in detail, as we can see from (19), the delays make the consensus error bigger than the one under no delays for every $N \theta$ steps, i.e., the term containing $c_1[k]$ is bigger that the one under no delays. However, when the iteration number is large enough as in (20), the term containing $c_1[k]$ converges to 0 (note that $\tau$ is a finite number), which results in the same error bound $c$ as the one under no delays in the one-hop case [36]. We must highlight that these properties do not hold in general and in fact depend on the structure of the update rules as we will see in the next section.

V. AN ALTERNATIVE EVENT-TRIGGERED SCHEME

In this section, we provide an alternative event-triggered scheme 2, which uses only the event-triggered values in the update laws of the normal nodes.

The new update rule is modified from (9) of scheme 1. In (9), node $i$ uses its current state $x_i[k]$ to obtain the new state $x_i[k+1]$. Therefore, node $i$ needs to store its state $x_i[k]$ at every time step even if the current state does not vary enough to trigger the event. However, in scheme 2, agents use only the triggered values for updates and the current states need not be stored. Hence, scheme 2 uses less storage than scheme 1. Here, scheme 2 is a generalization of the one in [36] which considered the synchronous algorithm with one-hop communication. This type of event-triggered scheme is also motivated by those in [34], [38], [39].

As we discussed in the previous section, it is notable that scheme 2 clearly differs from scheme 1 in the consensus error bound for a given $c_0$ and the parameters that determine it. For scheme 1, the consensus error bound is mainly determined by the update frequency $\theta$, and the upper bound of the transmission delays $\tau$ do not affect the error bound. For scheme 2, the situation becomes the opposite as the consensus error bound is mainly determined by the delay bound $\tau$. As a consequence, depending on the size of these parameters, one scheme may have an advantage over the other. For example, when $\theta$ is small for the agent system (i.e., agents update frequently), scheme 1 may perform better with a smaller error bound. In contrast, when the delay $\tau$ is small in the communication, scheme 2 has a smaller error bound. In fact, we can show by simple calculation that when $\tau + 1 \leq \theta$, the error bound of scheme 2 is strictly smaller than that of scheme 1. More detailed discussions will be given later.

In scheme 2, at each time step $k \in \mathbb{Z}_+$, each normal node $i \in N$ updates its current state by

$$x_i[k+1] = \hat{x}_i[k] + \sum_{j \in D_i[k]} a_i[k](\hat{x}_j^T[k - \tau_{ij}^T[k]] - \hat{x}_i[k]), \quad (22)$$

where $a_i[k] = 1/|D_i[k]|$.

Note that the new state $x_i[k+1]$ is merely used for checking the condition of the triggering function $f_i[k]$ in (2), and it need not be stored for the update in the next time step. Accordingly, in Algorithm 1, the input should be adjusted such that node $i$ uses $\hat{x}_i[k]$ instead of $x_i[k]$ in determining the set $D_i[k]$.

Next, we denote the state vector containing the event-triggered values of normal nodes from time $k - \tau$ to time $k$ as

$$z[k] = [\hat{x}^N[k]^T \hat{x}^N[k-1]^T \cdots \hat{x}^N[k-\tau]^T]^T,$$

which is a $N(\tau + 1)$-dimensional vector for $k \geq 0$. We also define its maximum value and minimum value as $\bar{z}[k] = \max z[k]$ and $\underline{z}[k] = \min z[k]$, respectively.

Then, we are ready to present the main result regarding scheme 2.

Theorem 2: Consider a directed graph $G = (V, E)$ with $l$-hop communication, where each normal node updates its value according to the asynchronous event-triggered scheme 2. Under the $f$-local Byzantine model, the normal nodes reach resilient consensus at an error level $c$ if and only if the underlying graph is $(f + 1)$-strictly robust with $l$ hops. Moreover, the safety interval is given by $S = [\bar{z}[0], \underline{z}[0]]$, and the consensus error level $c$ is achieved if the parameter $c_0$ in the triggering function (2) satisfies

$$c_0 \leq \frac{\gamma N(\tau + 1) - 1}{1 - \frac{\gamma N(\tau + 1)}{1 - \gamma}} c. \quad (23)$$

Proof: (Necessity) This part follows from the proof of Theorem 1.

(Sufficiency) First, we show by induction that the safety condition is satisfied. We can rewrite the update rule (22) as

$$x_i[k+1] = a_i[k] \hat{x}_i[k] + \sum_{j \in D_i[k]} a_i[k] \hat{x}_j^T[k - \tau_{ij}^T[k]], \quad (24)$$
where \( a_i[k] = 1/|D_i[k]| \). At time \( k = 0 \), it is clear by definition that \( x_i[0], \dot{x}_i[0] \in S \). To start with, we must show that \( \overline{x}_r[k] \) is a nonincreasing function of time. By (24), it follows that \( x_i[k+1] \leq \overline{x}_r[k] \) for \( i \in \mathcal{N} \). This is because step 2 of Algorithm 1 will remove values larger than \( \overline{x}_r[k] \). Furthermore, from (5), we have that \( x_i[k+1] \leq \overline{x}_r[k] \) for all \( i \in \mathcal{N} \). By combining these facts, it holds \( \overline{x}_r[k+1] \leq \overline{x}_r[k] \). Similarly, it can be established that \( \overline{x}_r[k] \) is a nonincreasing function of time. Moreover, we can follow a similar argument to show that \( \overline{x}_r[k] \) and \( z[k] \) are, respectively, nonincreasing and nondecreasing in time. Therefore, the safety condition is proven.

Next, we show the consensus condition. We first sort the normal nodes’ values in the vector \( z[k] \) at time \( k \) in increasing order. That is, the values are sorted as \( z_{s_1}[k] \leq z_{s_2}[k] \leq \cdots \leq z_{s_{N+1}}[k] \), where \( s_i[k] \) is the index of the agent taking the \( i \)th smallest value.

We introduce two sequences of conditions for the bound of gaps between two values in the sorted vector. The first condition sequence \( \{A_m\} \) is defined as

\[ A_1 : z_{s_2}[k] - z_{s_1}[k] \leq \left( c_0 + c_1[k] \right)/\gamma. \]

\[ A_2 : z_{s_3}[k] - z_{s_2}[k] \leq \left( c_0 + c_1[k] \right)/\gamma^2. \]

\[ \cdots \]

\[ A_{N(\tau+1)-1} : z_{s_{N(\tau+1)}[k]} - z_{s_{N(\tau+1)-1}}[k] \leq \left( c_0 + c_1[k] \right)/\gamma^{N(\tau+1)-1}. \]

The second condition sequence \( \{B_m\} \) is defined as

\[ B_{N(\tau+1)} : z_{s_{N(\tau+1)}[k]} - z_{s_{N(\tau+1)-1}}[k] \leq \left( c_0 + c_1[k] \right)/\gamma. \]

\[ B_{N(\tau+1)-1} : z_{s_{N(\tau+1)-1}}[k] - z_{s_{N(\tau+1)-2}}[k] \leq \left( c_0 + c_1[k] \right)/\gamma^2. \]

\[ \cdots \]

\[ B_{2} : z_{s_2}[k] - z_{s_1}[k] \leq \left( c_0 + c_1[k] \right)/\gamma^{N(\tau+1)-1}. \]

In the following proof, we will show that all the conditions in the two sequences above must be satisfied when \( k \) tends to infinity. To do this, first, we need to find the minimum \( m \in \{1, \ldots, N(\tau+1)-1\} \) such that condition \( A_m \) is not satisfied, which is defined by \( m_A \). Similarly, we define \( m_B \) as the maximum \( m \in \{2, \ldots, N(\tau+1)\} \) such that condition \( B_m \) is not satisfied. Then, it follows that

\[ z_{s_{m_A+1}}[k] - z_{s_m}[k] \geq c_0 + c_1[k]/\gamma_{m_A}. \]

\[ z_{s_m}[k] - z_{s_{m-1}}[k] \geq c_0 + c_1[k]/\gamma^{N(\tau+1)-m_B+1}. \] (25)

Furthermore, conditions \( A_1 \) to \( A_{m_A-1} \) and \( B_{m_B+1} \) to \( B_{N(\tau+1)} \) are all satisfied. Then, for \( 0 \leq k \leq k' \), we define two sets as follows:

\[ Z_1(k, k') = \{ i \in \mathcal{N} : \dot{x}_i[k'] < z_{s_{m_A}}[k] + c_0 + c_1[k] \}. \]

\[ Z_2(k, k') = \{ i \in \mathcal{N} : \dot{x}_i[k'] > z_{s_{m_B}}[k] - c_0 - c_1[k] \}. \]

Depending on the relationship between \( m_A, m_B \) and \( Z_1(k, k), Z_2(k, k) \), there are several possible cases. In the following, we study them separately.

**Case 1.** \( m_A < m_B \): There are four subcases, denoted by (1-a) to (1-d).

**1-a** \( Z_1(k, k) \neq \emptyset \) and \( Z_2(k, k) \neq \emptyset \): For a normal node \( j \notin Z_1(k, k) \), by definition, the inequality \( \dot{x}_j[k] \geq z_{s_{m_A}}[k] + c_0 + c_1[k] \) holds. Moreover, since the minimum element of \( z[k] \) that exceeds \( z_{s_{m_A}}[k] \) is \( z_{s_{m_A+1}}[k] \), we also have \( \dot{x}_j[k] \geq z_{s_{m_A+1}}[k] \).

If node \( j \) updates (i.e., \( j \notin \mathcal{U}[k] \)) and the event is triggered at node \( j \), then values less than \( z_{s_1}[k] \) will be ignored. Additionally, since the update is based on the convex combination as shown in (24), it holds

\[ \dot{x}_j[k+1] \geq (1-\gamma)z_{s_1}[k] + \gamma z_{s_{m_A+1}}[k]. \] (26)

Since conditions \( A_1 \) to \( A_{m_A-1} \) are satisfied, together, we can bound \( z_{s_1}[k] \) as

\[ z_{s_1}[k] \geq z_{s_{m_A}}[k] - \frac{c_0 + c_1[k]}{\gamma}. \]

\[ \geq z_{s_1}[k] - \left( \frac{1}{\gamma} + \frac{1}{\gamma^2} \right) (c_0 + c_1[k]) \geq \cdots \]

\[ \geq z_{s_{m_A}}[k] - \left( \frac{1}{\gamma} + \frac{1}{\gamma^2} + \cdots + \frac{1}{\gamma^{m_A-1}} \right) (c_0 + c_1[k]). \]

Combining with (26), we have

\[ \dot{x}_j[k+1] \geq z_{s_{m_A}}[k] + \gamma \left( z_{s_{m_A+1}}[k] - z_{s_{m_A}}[k] \right) \]

\[ - \frac{1}{\gamma^{m_A-1}} (c_0 + c_1[k]) + c_0 + c_1[k] \]

\[ > z_{s_{m_A}}[k] + \frac{c_0 + c_1[k]}{\gamma^{m_A}} \]

\[ - \frac{1}{\gamma^{m_A-1}} (c_0 + c_1[k]) + c_0 + c_1[k] \]

\[ = z_{s_{m_A}}[k] + c_0 + c_1[k], \] (27)

where the second inequality is from (25).

On the other hand, for an agent \( j \notin \mathcal{U}[k] \), or for \( j \notin \mathcal{U}[k] \) but the event is not triggered at \( j \), we have \( \dot{x}_j[k+1] = \dot{x}_j[k] \). Then we can derive that

\[ j \notin Z_1(k, k) \Rightarrow j \notin Z_1(k, k+1). \] (28)

Under either case, we conclude that if a normal node is not in \( Z_1(k, k) \), then it will not move into \( Z_1(k', k') \) at time \( k' > k \). Likewise, for \( Z_2(k, k) \), we can show a similar statement.

Furthermore, \( Z_1(k, k) \) and \( Z_2(k, k) \) are disjoint. This is because by using the two inequalities in (25), from \( 1 \leq m_A < m_B \leq N(\tau+1) \) and \( 0 < \gamma \leq 1/2 \), it follows that

\[ z_{s_{m_B}}[k] - z_{s_{m_A}}[k] \]

\[ > \max \left\{ \frac{1}{\gamma^{m_A}}, \frac{1}{\gamma^{N(\tau+1)-m_B+1}} \right\} (c_0 + c_1[k]) \]

\[ \geq 2(c_0 + c_1[k]). \]

Notice that the network is \((f+1)\)-strictly robust with \( l \) hops w.r.t. any set \( F \) following the \((f+1)\)-local model. Use the discussions in the proof of Theorem 1, it holds that for the two nonempty disjoint sets \( Z_1(k, k) \) and \( Z_2(k, k) \), one of them (or both) has a normal agent with at least \( f+1 \) independent normal paths originating from some normal nodes outside.

Suppose that normal node \( i \in Z_1(k, k) \) has the abovementioned property. A similar argument holds when \( i \in Z_2(k, k) \). If
node $i$ chooses to update at time $k \leq k' < k + \theta$ (recall that each normal node should update once in $\theta$ steps), then $\hat{x}_i[k'] = \hat{x}_i[k'^+1] = \hat{x}_i[k'^+1] = x_i[k'^+1]$ for $k \leq k' < k^i$. Thus, it holds that $i \in \mathcal{Z}_1(k, k') \subseteq \mathcal{Z}_1(k, k)$. Due to (28), node $i$ still has the abovementioned property whether some nodes move out of $\mathcal{Z}_1(k, k')$ or not. When node $i$ updates at time $k'$, it should use at least one value outside the set $\mathcal{Z}_1(k, k')$ (i.e., this value is greater than $z_{s_m+A} [k]$). Similar to (26) and (27), we have
\[
\hat{x}_i[k'^+1] \leq (1 - \gamma)z_{s_1}[k] + \gamma z_{s_m+A} [k]
\]
\[
\geq (1 - \gamma)z_{s_1}[k] + \gamma z_{s_m+A} [k]
\]
\[
\geq z_{s_m+A} [k] + c_0 + c_1[k],
\]
indicating that at time $k'^+1$, node $i$ moves out of $\mathcal{Z}_1(k, k)$. This is because by $\hat{x}_i[k'] \leq z_{s_m+A} [k]$, we have
\[
f_i[k'] = |\hat{x}_i[k'] - x_i[k'^+1]| - (c_0 + c_1[k']) > (c_0 + c_1[k]) - (c_0 + c_1[k']) > 0.
\]
Thus after $\theta$ steps, the cardinality of $\mathcal{Z}_1(k, k + \theta)$ is smaller than that of $\mathcal{Z}_1(k, k)$.

Similar results also hold for the case $i \in \mathcal{Z}_2(k, k)$. If $\mathcal{Z}_1(k, k)$ and $\mathcal{Z}_2(k, k)$ are nonempty, we can repeat the above steps until one of them becomes empty. As a result, $\overline{\tau}$ will decrease by $c_0 + c_1[k]$ (or $\overline{\tau}$ will increase by $x_i[k]$) after $N\theta$ steps.

1-b) $\mathcal{Z}_1(k, k) = \emptyset$ and $\mathcal{Z}_2(k, k) \neq \emptyset$: We have $\mathcal{Z}_1(k, k + \theta) = \emptyset$, i.e., $\overline{\tau} = \overline{\tau} + \theta$. This is because by $\hat{x}_i[k'] \leq z_{s_m+A} [k]$, we have
\[
f_i[k'] = |\hat{x}_i[k'] - x_i[k'^+1]| - (c_0 + c_1[k']) > (c_0 + c_1[k]) - (c_0 + c_1[k']) > 0.
\]
Thus after $\theta$ steps, the cardinality of $\mathcal{Z}_1(k, k + \theta)$ is smaller than that of $\mathcal{Z}_1(k, k)$.

In this section, we conduct simulations for networks applying the event-triggered MW-MSR algorithm. For all the simulations, we set the parameters $c_0$ and $c_1[k]$ of the triggering function as $c_0 = 1.215 \times 10^{-2}$ and $c_1[k] = 0.5 \times e^{-0.06(k+2)}$, respectively.

VI. NUMERICAL EXAMPLES

In this section, we conduct simulations for networks applying the event-triggered MW-MSR algorithm. For all the simulations, we set the parameters $c_0$ and $c_1[k]$ of the triggering function as $c_0 = 1.215 \times 10^{-2}$ and $c_1[k] = 0.5 \times e^{-0.06(k+2)}$, respectively.

A. Topology Gap Between One-Hop and Multi-Hop Algorithms

In this part, we show that the proposed algorithm can guarantee resilient consensus in a network where the conventional one-hop algorithm cannot. Consider the network in Fig. 2. This
Fig. 3. Time responses using different event-triggered MSR algorithms.

graph is not 2-strictly robust with one hop under 1-local model. This is because after removing node 3, the remaining graph is not 2-robust with one hop. However, this graph is 2-strictly robust with 2 hops under 1-local model. One can check that after removing any node set satisfying 1-local model (in 2-hop neighbors), the remaining graph is 2-robust with 2 hops.

Suppose that nodes 3 and 17 are Byzantine. Node 3 sends four different values to its four neighbors and node 17 keeps its value fixed. Let the initial states of the normal agents be $x_N[0] = [1015816131772184619101]^T$. According to [3], [36], this graph does not meet the condition for 1-total Byzantine model even for synchronous updates. Thus, resilient consensus cannot be reached. This can be seen in Fig. 3(a) where the normal agents converge to two different values. This is caused by the five red dashed lines indicating the adversarial values. The dots represent the time instants when events are triggered by the normal nodes in their corresponding colors.

Then, we perform simulations for the asynchronous two-hop event-triggered MW-MSR algorithm under the same attacks. Let the normal nodes update synchronously with delays in communication ($\theta = 1$). Moreover, we choose the package relay model, i.e., nodes only relay the messages when events are triggered by their own values. Observe that resilient consensus is achieved as shown in Fig. 3(b). This verifies the effectiveness of the proposed algorithm.

Lastly, we show that the event-triggered algorithm requires less update and communication times than the regular algorithm in our previous work [29]. We apply the MW-MSR algorithm [29] with regular communications under the same attack scenario and initial values of normal agents. The result shows that to achieve the same consensus error level $c = 0.03$, the event-triggered algorithm requires 4.7 times of updates per normal node. The algorithm with regular communications requires 9 times of updates per normal node. We however note that the design of the decaying sequence for the event-triggered threshold can significantly affect the performance of communication saving.

B. The Amount of Transmissions of Different Algorithms

In this part, we show that the amount of transmissions of the proposed algorithm can be further reduced compared to the one-hop algorithm. This time, we consider the network in Fig. 1(b). This graph is 2-strictly robust with one hop, and hence, with 2 hops (see [23]). Node 6 is Byzantine and is capable to send two different values to its neighbors (including different relayed values). Let the initial normal states be $x_N[0] = [246810]^T$. By [3] and Theorem 1, this graph satisfies the condition for 1-total Byzantine model. Thus, resilient consensus can be achieved with both one-hop and two-hop algorithms, and the results are given in Fig. 4.

From Fig. 4, we can also see that the numbers of events of the two-hop algorithm with immediate relays and package relays are both smaller than that of the one-hop algorithm. This is because by introducing the multi-hop communication, each node can have more information of the network, which may result in faster speed of the consensus process and less events. Moreover, observe that the two-hop algorithm with immediate relays has less events than the algorithm with package relays.
Obviously, the immediate relay model is an ideal model and it requires additional communication resources for the relaying process. Note that for this model, each event is accompanied with additional transmissions for relays as each node has three neighbors. In contrast, the package relay model is more realistic and energy-saving since it requires only communication for the events, but reaching consensus may take longer time.

To verify these properties of the algorithms, we further conducted Monte Carlo simulations in the same network for 50 runs by randomly taking initial normal states within [0, 10]. The Byzantine node 6 misbehaves as in the previous simulation. Table I displays the average times of events and transmissions per normal node of the three algorithms. In all runs, consensus is achieved and the results are consistent with our analysis so far. In particular, the package relay model requires the least number of transmissions overall.

C. The Effects of Update Frequency and Transmission Delays on Different Algorithms

In this part, we compare the performance of the proposed schemes 1 and 2 in the same 6-node network. Recall that given $c_0$, the consensus error level $c$ of schemes 1 and 2 are mainly determined by the update frequency $\theta$ and the delay bound $\tau$, respectively. Here, we evaluate the consensus error of schemes 1 and 2 under different situations. In the first situation, we set $\theta = 20$ and $\tau = 0$, i.e., each normal agent updates at least once in 20 steps and there is no delay in the transmissions. In the second situation, we set $\theta = 1$ and $\tau = 10$; this indicates that an normal agent updates at every time step and the bound of delay in the transmissions is 10.

In this simulation, we have $c_0 = 1.215 \times 10^{-3}$ and the remaining parameters are kept the same as the previous simulation. The time responses of the two schemes under the second situation are shown in Figs. 5(a) and 5(b), respectively. They looks similar on the agents’ trajectories. Most importantly, we measure the consensus errors of the two schemes after 1000 time steps. In the first situation, the consensus errors of schemes 1 and 2 are $2.78 \times 10^{-4}$ and $2.34 \times 10^{-4}$, respectively. However, in the second situation, the consensus errors of schemes 1 and 2 are $1.59 \times 10^{-4}$ and $5.99 \times 10^{-3}$, respectively. This result verifies our analysis mentioned before. That is, when $\theta$ is small for the agent system, scheme 1 has a smaller error bound. When the delay $\tau$ is small in the agent system, scheme 2 has a smaller error bound. Hence, if the consensus error level is critical and a lower error level is desired, we can select a better event-triggered update scheme according to the situation of a given agent system.

VII. CONCLUSION

In this article, we have investigated the resilient consensus problem using the event-triggered MSR algorithm with multi-hop communication. We also characterized the network requirement for the proposed algorithms to guarantee resilient consensus with a certain error level. The two proposed event-triggered schemes can be applied in different agent-systems. The scheme with a better performance on the consensus error bound can be chosen based on the transmission delay and the update frequency of the system. By introducing multi-hop communication, even sparse graphs can meet the condition for robustness. Furthermore, the event-triggered scheme provides an effective way to reduce the number of transmissions for the multi-hop communication.



| Algorithms                  | Average events | Average transmissions |
|-----------------------------|----------------|-----------------------|
| One-hop                     | 7.26           | 7.26                  |
| Two-hop with immediate relays| 3.05           | 12.20                 |
| Two-hop with package relays | 6.99           | 6.99                  |

Table I: Average Triggering Times Per Normal Node

Fig. 5. Time responses using different event-triggered MSR algorithms.
[9] S. Kar and J. M. F. Moura, “Distributed consensus algorithms in sensor networks: Quantized data and random link failures,” IEEE Trans. Signal Process., vol. 58, no. 3, pp. 1383–1400, Mar. 2010.

[10] S. Wu and M. G. Rabbat, “Broadcast gossip algorithms for consensus on strongly connected digraphs,” IEEE Trans. Signal Process., vol. 61, no. 16, pp. 3959–3971, Aug. 2013.

[11] N. H. Vaidya, L. Tseng, and G. Liang, “Iterative approximate Byzantine consensus in arbitrary directed graphs,” in Proc. ACM Symp. Princ. Distrib. Comput., 2012, pp. 365–374.

[12] S. M. Dibaji and H. Ishii, “Resilient consensus of second-order agent networks: Asynchronous update rules with delays,” Automatica, vol. 81, pp. 123–132, 2017.

[13] Z. Wang and Y. Wang, “Global synchronization of pulse-coupled oscillator networks under Byzantine attacks,” IEEE Trans. Signal Process., vol. 68, pp. 3158–3168, 2020.

[14] D. Dolev, “The byzantine generals strike again,” J. Algorithms, vol. 3, no. 1, pp. 14–30, 1982.

[15] B. Kailkhura, Y. S. Han, S. Brahma, and P. K. Varshney, “Distributed Bayesian detection in the presence of Byzantine data,” IEEE Trans. Signal Process., vol. 63, no. 19, pp. 5250–5263, Oct. 2015.

[16] A. Vempaty, L. Tong, and P. K. Varshney, “Distributed inference with Byzantine data: State-of-the-art review on data falsification attacks,” IEEE Signal Process. Mag., vol. 30, no. 5, pp. 65–75, Sep. 2013.

[17] S. Marano, V. Matta, and L. Tong, “Distributed detection in the presence of Byzantine attacks,” IEEE Trans. Signal Process., vol. 57, no. 1, pp. 16–29, Jan. 2009.

[18] X. Ren, J. Yan, and Y. Mo, “Binary hypothesis testing with Byzantine sensors: Fundamental tradeoff between security and efficiency,” IEEE Trans. Signal Process., vol. 66, no. 6, pp. 1454–1468, Mar. 2018.

[19] Z. Li, Y. Mo, and F. Hao, “Distributed sequential hypothesis testing with Byzantine sensors,” IEEE Trans. Signal Process., vol. 69, pp. 3044–3058, 2021.

[20] A. S. Rawat, P. Anand, H. Chen, and P. K. Varshney, “Collaborative spectrum sensing in the presence of Byzantine attacks in cognitive radio networks,” IEEE Trans. Signal Process., vol. 59, no. 2, pp. 774–786, Feb. 2011.

[21] M. Azadmanesh and R. Kieckhafer, “Asynchronous approximate agreement in partially connected networks,” Int. J. Parallel Distrib. Syst. Netw., vol. 5, no. 1, pp. 26–34, 2002.

[22] D. Sakavalas, L. Tseng, and N. H. Vaidya, “Asynchronous Byzantine approximate consensus in directed networks,” in Proc. 59th Symp. Princ. Distrib. Comput., 2020, pp. 149–158.

[23] L. Yuan and H. Ishii, “Resilient consensus with multi-hop communication,” in Proc. IEEE Conf. Decis. Control, 2021, pp. 2696–2701.

[24] A. Goldsmith, Wireless Communications. Cambridge, U.K.: Cambridge Univ. Press, 2005.

[25] K. T. Tuong, P. Sartori, and R. W. Heath, “Cooperative algorithms for MIMO amplify-and-forward relay networks,” IEEE Trans. Signal Process., vol. 61, no. 5, pp. 1272–1287, Mar. 2013.

[26] Z. Zhao and Z. Lin, “Global leader-following consensus of a group of general linear systems using bounded controls,” Automatica, vol. 68, pp. 294–304, 2016.

[27] L. Yuan and H. Ishii, “Secure consensus with distributed detection via multi-hop communication,” Automatica, vol. 131, 2021, Art. no. 109775.

[28] C. Zhao, J. He, and J. Chen, “Resilient consensus with mobile detectors against malicious attacks,” IEEE Trans. Signal Inf. Process. Netw., vol. 4, no. 1, pp. 60–69, Mar. 2018.

[29] L. Yuan and H. Ishii, “Asynchronous approximate Byzantine consensus via multi-hop communication,” in Proc. Amer. Control Conf., 2022, pp. 755–760.

[30] W. P. M. H. Heemels, K. H. Johansson, and P. Tabuada, “An introduction to event-triggered and self-triggered control,” in Proc. IEEE Conf. Decis. Control, 2012, pp. 3270–3285.

[31] X. Cao and T. Basar, “Decentralized online convex optimization with event-triggered communications,” IEEE Trans. Signal Process., vol. 69, pp. 284–299, 2021.

[32] T. Chen, G. Giannakis, T. Sun, and W. Yin, “LAG: Lazily aggregated gradient for communication-efficient distributed learning,” in Proc. Adv. Neural Inf. Process. Syst., 2018, pp. 5055–5065.

[33] D. Dimarogonas, E. Frazzoli, and K. H. Johansson, “Distributed event-triggered control for multi-agent systems,” IEEE Trans. Autom. Control, vol. 57, no. 5, pp. 1291–1297, May 2012.

[34] Y. Kadowaki and H. Ishii, “Event-based distributed clock synchronization for wireless sensor networks,” IEEE Trans. Autom. Control, vol. 60, no. 8, pp. 2266–2271, Aug. 2015.

[35] R. K. Mishra and H. Ishii, “Event-triggered control for discrete-time multi-agent average consensus,” Int. J. Robust Nonlinear Control, vol. 33, no. 1, pp. 159–176, 2023.

[36] Y. Wang and H. Ishii, “Resilient consensus through event-based communication,” IEEE Trans. Autom. Control, vol. 7, no. 1, pp. 471–482, Mar. 2020.

[37] L. Yuan and H. Ishii, “Event-triggered approximate Byzantine consensus with multi-hop communication,” in Proc. IEEE Conf. Decis. Control, 2022, pp. 7078–7083.

[38] H. Matsume, Y. Wang, and H. Ishii, “Resilient self/event-triggered consensus based on ternary control,” Nonlinear Anal.: Hybrid Syst., vol. 42, 2021, Art. no. 101091.

[39] F. Xiao and L. Wang, “Asynchronous consensus in continuous-time multi-agent systems with switching topology and time-varying delays,” IEEE Trans. Autom. Control, vol. 53, no. 8, pp. 1804–1816, Sep. 2008.

Liwei Yuan received the M.E. and Ph.D. degrees in computer science from the Tokyo Institute of Technology, Yokohama, Japan, in 2019 and 2022, respectively. He is currently a Postdoctoral Researcher with the College of Electrical and Information Engineering, Hunan University, Changsha, China. His research interests include security in multi-agent systems and distributed algorithms.

Hideaki Ishii (Fellow, IEEE) received the M.Eng. degree from Kyoto University, Kyoto, Japan, in 1998, and the Ph.D. degree from the University of Toronto, Toronto, ON, Canada, in 2002. From 2001 to 2004, he was a Postdoctoral Research Associate with the University of Illinois at Urbana-Champaign, Urbana, IL, USA, and a Research Associate with The University of Tokyo, Tokyo, Japan, from 2004 to 2007. He is currently a Professor with the Department of Computer Science, Tokyo Institute of Technology, Yokohama, Japan. During 2014–2015, he was a Humboldt Research Fellow with the University of Stuttgart, Stuttgart, Germany. He has also held Visiting Positions with CNR-IEIIT, Politecnico di Torino, Turin, Italy, Technical University of Berlin, Berlin, Germany, and City University of Hong Kong, Hong Kong. His research interests include networked control systems, multiagent systems, distributed algorithms, and cyber-security of control systems.

Dr. Ishii was an Associate Editor for Automatica, IEEE CONTROL SYSTEMS LETTERS, IEEE TRANSACTIONS ON AUTOMATIC CONTROL, IEEE TRANSACTIONS ON CONTROL OF NETWORK SYSTEMS, and Mathematics of Control, Signals, and Systems. Since 2017, he has also been the Chair of the IFAC Coordinating Committee on Systems and Signals and was the Chair of the IFAC Technical Committee on Networked Systems from 2011 to 2017. He is the IPC Chair for the IFAC World Congress 2023 to be held in Yokohama, Japan. He was the recipient the IEEE Control Systems Magazine Outstanding Paper Award in 2015.