A Discrete-Time Reputation-Based Resilient Consensus Algorithm for Synchronous or Asynchronous Communications

Guilherme Ramos ©, Daniel Silvestre ©, and Carlos Silvestre ©, Senior Member, IEEE

Abstract—We tackle the problem of a set of agents achieving resilient consensus in the presence of attacked agents. We present a discrete-time reputation-based consensus algorithm for synchronous and asynchronous networks by developing a local strategy where, at each time, each agent assigns a reputation (between zero and one) to each neighbor. The reputation is then used to weigh the neighbors’ values in the update of its state. Under mild assumptions, we show that: 1) the proposed method converges exponentially to the consensus of the regular agents; 2) if a regular agent identifies a neighbor as an attacked node, then it is indeed an attacked node; 3) if the consensus value of the normal nodes differs from that of any of the attacked nodes’ values, then the reputation that a regular agent assigns to the attacked neighbors goes to zero. Further, we extend our method to achieve resilience in scenarios where there are noisy nodes, dynamic networks, and stochastic node selection. Finally, we illustrate our algorithm with several examples, delineating some attacks that can be dealt with by the current proposal but not by the state-of-the-art approaches.

Index Terms—Agents and autonomous systems, computer networks, fault tolerant systems, reputation systems.

I. INTRODUCTION

In the last decades, much work has been devoted to networked control systems with a focus on cybersecurity aspects. Communications over shared mediums potentiate the exploitation of vulnerabilities that may result in consequences sometimes catastrophic. A central problem in networked multiagent systems is the so-called consensus problem, where a set of agents interacting locally through a communication network attempts to reach a common value. In other words, the final value is the outcome of running a distributed algorithm among nodes that can communicate according to the network topology. Therefore, the problem of consensus is the basilar stone of a multitude of applications, ranging in areas such as optimization [1], [2]; motion coordination tasks—flocking and leader following [3]; rendezvous problems [4]; computer networks resource allocation [5]; and the computation of the relative importance of web pages, by PageRank like algorithm [6].

Notwithstanding, the consensus problem further appears as a critical subproblem of major applications. A distributed Kalman filter, based on two consensus systems, was proposed in [7] to estimate the 2-D motion of a target. The work was experimentally assessed in [8] to estimate the motion of a real robot. A crucial property to ensure in any consensus algorithm is its ability to overcome abnormal situations, i.e., achieve resilient consensus.

Resilient consensus: Each agent in a network that works as expected must filter erroneous information and be capable of reaching the consensus value that results from legit network information. A new fault-tolerant algorithm to accomplish an approximate Byzantine consensus in asynchronous networks was introduced in [9]. The method also applies to synchronous networks, to networks with communication paths with delay, and the authors extend the results to time-varying networks. Subsequently, authors performed the convergence rate analysis of the fault-tolerant consensus algorithm of [9] in [10].

The work in [11] introduced a system to tackle worst-case and stochastic faults in the scenario of gossip consensus. The developed method may be integrated into the consensus algorithm to achieve resilient consensus, making the nodes converge to a steady-state belonging to the set resulting from the intersection of the estimates that each agent perceives for the remaining nodes [12]. The technique was extended to a wider family of gossip algorithms in [13]. These methods converge, and they offer a theoretical bound on the attacker signal that we can compute a priori. However, their computational complexity for the worst-case undetectable attack makes the approach inviable. For detecting attacks, their computational complexity is worse than that of the new algorithm proposed in this work. For attackers’ isolation, the computational complexity in [13], [14], and [15] is exponential, contrasting with the proposed polynomial-time approach.

In [16], two fault-tolerant and parameter-independent consensus algorithms were introduced to deal with misbehaving agents. One of the approaches adaptively estimates, using a fault-detection scheme, how many faulty agents are in the network. Whenever there are $f$ faulty agents, the authors’ method converges if the network of nonfaulty nodes is $(f + 1)$-robust. The other approach consists of a nonparametric Mean-Subsequence-Reduced algorithm, converging when both the network of nonfaulty agents is $(f + 1)$-robust and the nonfaulty agents possess the same amount of in-neighbors. In [17], the authors characterized the limitations on the performance of any distributed optimization algorithm in the presence of adversaries. In [18], the authors explored the scenario where each regular agent in a network refreshes its state based on local information, using a designed feedback law, and that malicious nodes update their state arbitrarily. The authors gave an algorithm for the consensus of second-order sampled-data multiagent systems. When the network has sufficient connectivity, the authors proposed a resilient consensus method, in which each node ignores the information of neighbors having large/small position values, considering that although the global topology of the network is
unknown to regular agents, they know a priori the maximum number of malicious neighbor nodes. In [19], the authors extended the previous work, showing a consensus algorithm for clock synchronization in wireless sensor networks.

Subsequently, in [20], each agent might have a particular threshold for the number of extreme neighbors it ignores. Further, the author presented conditions under which the consensus is asymptotically almost surely (with probability 1) achieved in random networks and with random nodes’ thresholds. In [21], it was proposed a resilient leader–follower consensus to arbitrary reference values, where each agent ignored a portion of extreme neighbors. This consensus method guarantees a steady-state value lying in the convex hull of initial agent states. In a similar approach, the authors of [22] presented a resilient consensus algorithm for time-varying networks of dynamic agents. In [23], the case of quantized transmissions with communication delays and asynchronous update schemes was studied for update times selected in either a deterministic or random fashion.

In [24], the authors presented a consensus-innovations estimator. If less than half of the agents’ sensors fall under attack, then all of the agents’ estimates converge with polynomial rate to the parameter of interest. In [25], the authors presented a resilient fully distributed averaging algorithm that uses a resilient retrieval procedure, where all non-Byzantine nodes send their initial values and retrieve those of other agents.

Classes of consensus problems: We may classify the problems of consensus based on the domain of the time update as: discrete-time (discrete), as in [26] and [27]; or as continuous-time (continuous), as in [27] and [28]. Also, it can be classified based on the network communication: synchronous, see, for instance, [26]; or asynchronous, see, for example, [1], [9], and [29]. Moreover, the communication between agents may be: deterministic, see, for example, [27]; or stochastic, as in [30], [31], and [32] for instance. Finally, the agents’ network of communication can be categorized as: static, see [16], [27], and [32]; or dynamic, changing in time; see in [22], [26], [33].

Reputation systems: The concept of an entity’s reputation is an opinion about that entity that usually results from an evaluation of the entity based on a set of criteria. Everyday, we implicitly assign an opinion about that entity that usually results from an evaluation of the entity based on a set of criteria. Consider a network of agents \( G = (V, E) \), with initial states \( x^{(0)} \in \mathbb{R}, \) for \( v \in V \). In the nonattacked scenario, agents can reach consensus through the use of a distributed linear iterative algorithm with dynamics given by

\[
x^{(k+1)} = W^{(k)} x^{(k)}
\]

where \( x^{(k)} \) is the vector collecting the \( n \) agents states at time step \( k > 0 \), and the matrix \( W^{(k)} \in \mathbb{R}^{n \times n} \) is such that: 1) \( W^{(k)} \) is 0 if the agents \( u \) and \( v \) do not communicate, and 2) the agents converge to the same quantity, i.e., \( \lim_{k \to \infty} x^{(k)} = x^* \).

Additionally, we consider a set of attacked agents \( A \subset V \). If the agents \( a \in A \) do not follow the update rule of the consensus procedure, then each regular agent, \( v \in V \setminus A \), should be able to identify and discard the attacked agents.

The assumption we made is typical in the state-of-the-art methods to ensure resilient consensus. We remark that the assumption we do make is equivalent to the \( r \)-robustness (\( (r, 1) \)-robust) defined in [19].

The previous assumption is reasonable because each regular agent needs to be able to divide his neighbors into the set of normal nodes and the set of attacked ones, comparing the information that it receives. Hence, if the majority of the information is not legitimate there is no redundancy to allow identifying the attacked neighbors.

A. Preliminaries and Terminology

First, we recall some concepts of graph theory, and we set the notation that we will adopt in this manuscript. A directed graph, or simply a digraph, is an ordered pair \( G = (V, E) \), where \( V \) is a set of \( n \) nodes, and \( E \subseteq V \times V \) is a set of edges. Edges are ordered pairs which represent a relationship of accessibility between nodes. If \( u, v \in V \) and \( (u, v) \in E \) then the node \( v \) directly accesses information of node \( u \). In the scope of consensus algorithms, we also refer to a digraph as a network, and we further say that nodes are agents of the network. A complete digraph or complete network is a digraph such that all nodes can directly access information of every other node.

The neighbors of \( v \) are \( N_v = \{ v \} \cup \{ u : (u, v) \in E \} \). The proper neighbors of \( v \) are \( N_v' = N_v \setminus \{ v \} \). The in-degree of a node \( v \in V \), denoted by \( d_v \), is the number of neighbors of \( v \), i.e., \( d_v = |N_v| \). Likewise, the out-degree of a node \( v \in V \), \( d_v \), is the number of nodes that have \( v \) as neighbor, i.e., \( o_v = |\{ u : v \in N_u \}| \). A convenient way of representing a digraph is by means of its adjacency matrix \( A \in \mathbb{R}^{n \times n} \), where \( A_{u,v} = 1 \) if \( (u, v) \in E \), and \( A_{u,v} = 0 \), otherwise.

A subgraph or a subnetwork \( H = (V', E') \) of a digraph \( G = (V, E) \) is a digraph such that \( V' \subseteq V, E' \subseteq E \). If \( A \) denotes a set of nodes \( A \subset V \), we denote by \( G \setminus A \) the subgraph \( H \) of \( G \) that consists in \( H = (V \setminus A, E') \), where \( E' = \{ e \in E : e = (u, v) \) and \( u, v \notin A \} \).

Given a finite and nonempty array of possibly repeated elements sorted by increasing order \( C = \{ x_1, ..., x_n \} \), with \( x_i \in \mathbb{R} \), with at least one \( j \neq 1 \) and \( x_j \neq x_i \), we define the following element: \( \min_{x \in C} x = y_f \), where, inductively defined as

\[
y_f = \begin{cases} x_i, & \text{if } f = 1 \\ \min \{ x \in C : y_{f-1} < x < x_n \}, & \text{if } \exists y_{f-1} < x < x_n \\ y_{f-1}, & \text{otherwise} \end{cases}
\]

In other words, we are computing the \( f \)th smallest element of the set obtained from the array \( C \) by discarding repeated elements but ensuring that it is not the maximum element. This definition will be paramount to the reputation-based consensus method we propose, as we need to normalize a set of values, dividing them by the difference between the maximum and the \( \min \) element. For instance, consider the sets of sorted elements \( C = \{ 1, 2, 2, 2, 3 \} \). If \( f = 1 \) then \( \min_{x \in C} x = 1 \), and if \( f \geq 2 \) then \( \min_{x \in C} x = 2 \).

II. REPUTATION-BASED CONSENSUS

Consider a network of agents \( G = (V, E(k)) \), with initial states \( x^{(0)} \in \mathbb{R} \), for \( v \in V \). In the nonattacked scenario, agents can reach consensus through the use of a distributed linear iterative algorithm with dynamics given by

\[
x^{(k+1)} = W^{(k)} x^{(k)}
\]

where \( x^{(k)} \) is the vector collecting the \( n \) agents states at time step \( k > 0 \), and the matrix \( W^{(k)} \in \mathbb{R}^{n \times n} \) is such that: 1) \( W^{(k)} \) is 0 if the agents \( u \) and \( v \) do not communicate, and 2) the agents converge to the same quantity, i.e., \( \lim_{k \to \infty} x^{(k)} = x^* \).

Additionally, we consider a set of attacked agents \( A \subset V \). If the agents \( a \in A \) do not follow the update rule of the consensus procedure, then each regular agent, \( v \in V \setminus A \), should be able to identify and discard the attacked agents.

The assumption we made is typical in the state-of-the-art methods to ensure resilient consensus. We remark that the assumption we do make is equivalent to the \( r \)-robustness (\( (r, 1) \)-robust) defined in [19].

The previous assumption is reasonable because each regular agent needs to be able to divide his neighbors into the set of normal nodes and the set of attacked ones, comparing the information that it receives. Hence, if the majority of the information is not legitimate there is no redundancy to allow identifying the attacked neighbors.

A. Attacker Model

Subsequently, we consider an attacker that may corrupt the state of the nodes in the subset \( A \) by adding an unbounded signal. The attacked dynamics are a corrupted version of (1) as follows:

\[
x^{(k+1)} = W^{(k)} x^{(k)} + \Delta^{(k)}
\]

where \( \Delta^{(k)} \in \mathbb{R}^n \), which entails the assumption that the attacker cannot corrupt the communication between nodes to send different messages.
Algorithm I: Synchronous Communication RepC.

1: input: Network of agents $G = (V, E(k))$, agents initial states $x(0)$, number of time steps $T$, and confidence factor $\varepsilon \in [0, 1[$
2: output: agents final states $x(T)$
3: for $k = 1, \ldots, T$ do
4: for $i = 1, \ldots, |V|$ do
5: 

Reputation update:

$$c_{ij}^{(k+1)} = \begin{cases} 1 - \frac{|x_i^{(k)} - x(j)^{(k)}|}{|N_i|}, & j \in N_i \\ 0, & \text{otherwise} \end{cases}$$

Normalized Reputation update:

$$\tilde{c}_{ij}^{(k+1)} = \begin{cases} \frac{c_{ij}^{(k+1)} - \min_{v \in N_i} \tilde{c}_{iv}^{(k+1)}}{\max_{v \in N_i} \tilde{c}_{iv}^{(k+1)} - \min_{v \in N_i} \tilde{c}_{iv}^{(k+1)}}, & i \neq j \\ 1, & \text{otherwise} \end{cases}$$

Normalized Reputation update with confidence $\varepsilon$:

$$c_{ij}^{(k+1)} = \begin{cases} \tilde{c}_{ij}^{(k+1)}, & \text{if } \tilde{c}_{ij}^{(k+1)} > 0, \\ c_{ij}^{(k+1)}, & \text{otherwise} \end{cases}$$

Consensus state update:

$$x_i^{(k+1)} = \frac{\sum_{j \in N_i} c_{ij}^{(k)} x_j^{(k)}}{\sum_{j \in N_i} c_{ij}^{(k)}}$$

6: end for
7: end for

to distinct neighbors. Observe that this assumption allows the attacker to change the state of a subset of agents to (possibly) different values.

Also, the attacker cannot create artificial nodes nor change the network topology, i.e., the structure of $W(k)$, and the dimension $n$ are fixed in (2).

B. Reputation-Based Consensus (RepC)

Next, we propose a reputation-based consensus algorithm (RepC). The idea behind the algorithm is that each time an agent obtains information (states) from its neighbors, the agent measures how discrepant is, on average, the state from one neighbor regarding the states of the remaining ones and its own state. The RepC is composed by two phases: 1) identification of the attacked nodes; 2) computation of the consensus.

Notice that the proposed algorithm is a fully distributed discrete-time consensus algorithm that works for synchronous and asynchronous networks. Also, each agent only needs to have a low computational power to do calculations with the neighbors’ values.

1) Synchronous Communication RepC: Given the maximum number of allowed attacked nodes $f$, the identification of the attacked nodes is performed by the iterative scheme in Algorithm 1.

Note that, in Algorithm 1, $c_{ij}^{(k+1)} = 0$ if $j \notin N_i$, and $c_{ii}^{(k+1)} = 1$, and where $c_{ij}^{(0)} = 1$ for $j \in N_i$ and $c_{ij}^{(0)} = 0$ otherwise, and $x_i^{(0)}$ is the initial value of each agent $i$. Further, $\varepsilon \in [0, 1[$ is a confidence factor which guarantees that each agent does not discard immediately values that are discrepant from its neighbors’ average.

Notice that the selected value for $\varepsilon$ must be small to have a negligible impact on the agents’ consensus states. Also, a large $\varepsilon$ may cause an agent to do not detect an attacked neighbor. This, in turn, makes the asymptotic consensus deviate from the consensus without attacked agents toward a combination of the attacked agents’ asymptotic states.

We illustrate this property in Section III. The proposed method computes a weighted average of the agents’ values. So, the final consensus state is a convex combination of the agents’ initial states.

2) Asynchronous Communication RepC: The asynchronous version of algorithm RepC consists of, at each instance of time, the agents that communicate, $A' \subset A$, follow (3), where $N_i$ is replaced by $N_i \cap A'$.

The iterative scheme (3) may also be used in the scenario where the network of agents evolves with time. The results in Section II-B can be restated for this scenario by considering that the set of neighbors of a node is dynamic, and by verifying, at each time, that each agent has more than two neighbors and more than half of them are regular agents. In Section III-D and E, we illustrate the dynamic network of agents and dynamic network with noisy agents scenarios.

First, we show that RepC converges. To simplify the proof, we assume that we are in the scenario of synchronous communication. The general proof follows the same steps, but it is more complex, and it needs more complex notation to denote the set of neighbors with which a node communicates at each time. Let us define $x_i^{(0)} = (x_i^{(0)} - x_{\text{min}}^{(0)})/(x_{\text{max}}^{(0)} - x_{\text{min}}^{(0)})$, which yields a rescaling of the agents’ states that simplifies the methods’ proof of convergence. This rescaling is to make the proof less extensive.

Further, in the following proofs and for technical reasons, we assume that each attacked agent shares a state converging to some value. Although, in practice, the algorithm is still effective under other circumstances, as we illustrate in Section III.

Moreover, to have guarantees of resilient consensus and derive theoretical result, we make the following additional assumptions:

1) For each regular agent, $v \in V \setminus A$, more than half of the neighbors are regular agents, i.e., $|N_v \cap A| < |N_v|/2$ and the network of normal nodes is connected.

2) Additionally, the attack cannot target the initial state, i.e., $\Delta(0) = 0$ in (2), since this scenario would be undetectable. The sequences of state values for the attacked version would be the same as a normal execution of the algorithm with the attack value as the initial state.

Lemma 1: If for any $i \in V$ we have that $|N_i| > 2$, then each agent that follows the iterative scheme in (3) converges.

Proof: For the proof, suppose that the initial states are rescaled to be in $[0, 1]$. We have that $\|x(k+1) - x(k)\| = \max_i \|x(k+1) - x_i^{(k)}\|$ and, hence, assuming without loss of generality that: (i) $\|c_{i}^{(k)}\|_1 \leq \|c_{i}^{(k+1)}\|_1$,

$$\frac{x_i^{(k+1)} - x_i^{(k)}}{\|c_i^{(k+1)}\|_1} = \frac{c_i^{(k+1)} \cdot x_i^{(k)} - c_i^{(k)} \cdot x_i^{(k)}}{\|c_i^{(k+1)}\|_1} = \frac{c_i^{(k+1)} \cdot x_i^{(k)} - c_i^{(k)} \cdot x_i^{(k)}}{\|c_i^{(k+1)}\|_1}$$

$$\leq \max_{j \in N_i} |c_{ij}^{(k+1)} - c_{ij}^{(k)}|\|x_j^{(k)} - x_i^{(k)}\| \leq \max_{j \in N_i} c_{ij}^{(k+1)} \|x_j^{(k)} - x_i^{(k)}\|$$

because we are assuming that $x_{\text{min}}^{(0)} c_{ij}^{(m)} \in [0, 1]$. Now, we need to compute $\max_{j \in N_i} |c_{ij}^{(k+1)} - c_{ij}^{(k)}|$. First, we notice that we cannot have that $\max_{j \in N_i} |c_{ij}^{(k+1)} - c_{ij}^{(k)}| = |c_{ij}^{(k+1)} - c_{ij}^{(k)}|$, because there is always a $j \in N_i$ such that $c_{ij}^{(k+1)} > c_{ij}^{(k)}$ and all the other $k \neq j \in N_i$ are such that $c_{ij}^{(k)} \leq c_{ij}^{(k+1)}$. Therefore, we need to consider only the following three cases:

1. $c_{ij}^{(k+1)} = c_{ij}^{(k+1)}$, and $c_{ij}^{(k)} = c_{ij}^{(k)}$;
2) \( c_{ij}^{(k+1)} = \frac{c_{ij}^{(k)}}{c_{ij}^{(k)}} \) and \( c_{ij}^{(k)} = e^k; \)

3) \( e_{ij}^{(k+1)} = \frac{e_{ij}^{(k)}}{e_{ij}^{(k)}} \) and \( e_{ij}^{(k)} = e_{ij}^{(k)} \).

For case 1, we have that \( |c_{ij}^{(k+1)} - c_{ij}^{(k)}| = |x_{ij}^{(k+1)} - x_{ij}^{(k)}| < \frac{1}{2} \). Using the same reasoning, for case 2, we have that \( |c_{ij}^{(k+1)} - c_{ij}^{(k)}| = |x_{ij}^{(k+1)} - x_{ij}^{(k)}| < \frac{1}{2} \). We only need to compute 3)

\[
\begin{align*}
|z_{ij}^{(k+1)} - z_{ij}^{(k)}| &= \left| \frac{c_{ij}^{(k+1)} - c_{ij}^{(k)}}{\max_{v \in N_i} (c_{iv}^{(k)})} - \min_{v \in N_i} (c_{iv}^{(k)}) \right| \\
&\leq \left| \frac{z_{ij}^{(k+1)} - z_{ij}^{(k)}}{c_{ij}^{(k+1)} - c_{ij}^{(k)}} \right|
\end{align*}
\]

implying that \( c_{uv}^{(k+1)} = 0 \) or \( c_{uv}^{(k+1)} > 0 \) and \( x_{uv}^{(k+1)} = x_{uv}^{(k)} = x_{uv}^{(k)} \). By transitivity, we can apply the same to each neighbor of all neighbors of \( u \), and so forth. Thus, the result yields for all \( i \in V \setminus A \).

**Lemma 1:** Let \( v \in V \setminus A \) and \( u \in V \). By using the iterative scheme (3), if \( c_{uv}^{(k)} = 0 \) then \( u \in A \).

The proof of Lemma 1 also hints that half of each agent’s neighbors should not be under attack so that each normal node identifies the attacked agents correctly. This property emerges from the following.

**Lemma 2:** Suppose that the iterative scheme (3) converges to a value different from that broadcasted by the attacked agents. If for each agent \( i \in V \setminus A \), less than half of its neighbors are not attacked agents, i.e., \(|N_i \cap A| < |N_i \setminus A|\), then \( c_{uv}^{(k)} = 0 \), for \( a \in A \) and \( c_{uv}^{(k)} = 1 \) for \( v \in N_i \setminus A \).

**Proof:** By Lemma 1, we have that the first regular agent using the iterative scheme in (3) converges to \( x_{uv}^{(k)} \). Let \( y \) denote the value that all the attacked agents in \( A \) share with the neighbors. For a regular agent \( a \in A \), an attacked agent’s reputation \( c_{uv}^{(k)} \) satisfies

\[
c_{uv}^{(k)} = \lim_{k \to \infty} c_{uv}^{(k)} = 1 - \frac{1}{|N_i|} \sum_{v \in N_i} y - \lim_{k \to \infty} x_{uv}^{(k)} = 1 - \frac{|N_i \setminus A|}{|N_i|} |y - x_{uv}^{(k)}|
\]

and the limit of the reputation of a regular user, \( j \notin A \), is given as

\[
c_{ij}^{(k)} = \lim_{k \to \infty} c_{ij}^{(k)} = 1 - \frac{|N_j \cap A|}{|N_j|} |x_{ij}^{(k)} - y|.
\]

Since \(|N_j \cap A| < |N_j \setminus A|\) and \( y \neq x_{uv}^{(k)} \), then \( c_{ij}^{(k)} > c_{ij}^{(k)} \), and because reputations values are normalized to be between 0 and 1, we have that, for all \( i, j \), \( c_{ij}^{(k)} > c_{ij}^{(k)} = 0 \).

Now, we need to show that a regular agent using RepC identifies the attacked neighbors and study the method’s convergence rate.

**Lemma 3:** Let \( \nu \in V \setminus A \) and \( \nu \in \nu \). By using the iterative scheme (3), if \( u \in A \) and \( |A| \leq f \) then \( c_{uv}^{(k)} = 0 \).

**Proof:** Let \( a \in A \) be an attacked node. We want to show that for a regular agent, \( v \in V \setminus A \), the reputation of agent \( a \) strictly decreases with time. Let \( v \in V \setminus A \), we have that \( x_{uv}^{(k+1)} - x_{uv}^{(k)} \leq |x_{uv}^{(k+1)} - x_{uv}^{(k)}| \leq \frac{1}{2} \).

Further, using the induction hypothesis, either (i) or (ii) is true for any set of \( N \) neighbors of \( u \). Hence, for \( j \in N_u \setminus \{v\} \) either \( x_{uj}^{(k)} = x_{uv}^{(k)} \) or \( c_{uv}^{(k)} = 0 \). In any of the cases, we have that

\[
\begin{align*}
x_{uv}^{(k+1)} &= \frac{1}{1 + \sum_{v \in N_u \setminus \{v\}} c_{uv}^{(k)} x_{v}^{(k)}} \left( x_{uv}^{(k)} + \sum_{v \in N_u \setminus \{v\}} c_{uv}^{(k)} x_{v}^{(k)} \right).
\end{align*}
\]

By replacing (7) in (6), it follows that

\[
x_{uv}^{(k+1)} = \frac{1}{1 + \sum_{v \in N_u \setminus \{v\}} c_{uv}^{(k)} x_{v}^{(k)}} \left( x_{uv}^{(k)} + \sum_{v \in N_u \setminus \{v\}} c_{uv}^{(k)} x_{v}^{(k)} \right).
\]

Applying this to all the nodes in the network, we obtain

\[
x_{uv}^{(k+1)} = \frac{1}{1 + \sum_{v \in N_u \setminus \{v\}} c_{uv}^{(k)} x_{v}^{(k)}} \left( x_{uv}^{(k)} + \sum_{v \in N_u \setminus \{v\}} c_{uv}^{(k)} x_{v}^{(k)} \right).
\]
most $\lceil \log_2(\varepsilon) \rceil$ times, we obtain an error between the last two iterations of at most $\varepsilon$.

C. Complexity Analysis

Next, we investigate the complexity analysis of the proposed algorithm $\text{RepC}$ when the network communication is synchronous.

Proposition 2: Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a network of agents, $l = \max_{v \in \mathcal{V}} |\mathcal{N}_v|$, then, for $i$ iterations and for each agent, the iterative scheme (3) has time complexity of $O(l^2 i)$.

Proof: Given a network of agents $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, for time step $k$ and agent $v$, the time complexity of (3) is the sum of the time complexities of computing $c_{u(v)}$, $c_{v(k)}$, $c_{u(v)}$, for each $u \in \mathcal{N}_v$, and $x_{u(k)}$. Computing $c_{u(v)}$ costs $O(|\mathcal{N}_v|^2)$, because there are $O(|\mathcal{N}_v|^2)$ pairs of neighbors values to compute the absolute difference. Each of the remaining steps has time complexity of $O(|\mathcal{N}_v|)$. Hence, the sum of each step time complexity is $O(|\mathcal{N}_v|^2 + 4 \times O(|\mathcal{N}_v|)) = O(|\mathcal{N}_v|^2)$. If $l = \max_{v \in \mathcal{V}} |\mathcal{N}_v|$, then $O(l^2)$ is a bound for the complexity that each incurs. Therefore, for $i$ iterations of the iterative scheme (3), each agent incurs in $O(l^2 i)$ time complexity.

III. ILLUSTRATIVE EXAMPLES

Subsequently, we illustrate the use of $\text{RepC}$ for different kinds of attacks. Further, in the examples, we use $\varepsilon = 0.1$.

A. Same Value for Attacked Nodes

In the following examples, we consider the network of agents depicted in Fig. 1(a).

First, we illustrate algorithm $\text{RepC}$ in the scenario of a network of agents without attacked nodes. The set of agents is $\mathcal{V}_1 = \{1, \ldots, 5\}$ and, thus, the set of attacked agents is $\mathcal{A} = \emptyset$. We set the parameter $f = 1$. Fig. 2 depicts the state evolution of each agent.

Here, we explore the scenario where an attacker targets one agent to share a value close to the consensus depicted in Fig. 1.

The set of agents is $\mathcal{V} = \{1, \ldots, 5\}$ and the set of attacked agents is $\mathcal{A} = \{1\}$. Fig. 3 depicts each agent consensus value. We can see that although the attacker value is very close to the consensus value, the neighbors of the attacked node assign zero to its reputation by using (3). Hence, the attacked node shared values are discarded.

Next, in Figs. 4 and 5, we depict the evolution of the reputations that agents 2 and 3 assign to their neighbors.

B. Different Values for Attacked Nodes

Next, we illustrate the scenario where attacked nodes share different values. For that end, we consider the set of agents $\mathcal{V} = \{1, \ldots, 10\}$, with $\mathcal{A} = \{1, 8\}$, and the network of agents depicted in Fig. 1(b). We explore two scenarios with two attacked agents: 1) both attacked nodes share values (distinct) smaller than the consensus, see Fig. 6; 2) one attacked node shares a value larger than the consensus while the other uses a smaller value than the consensus, see Fig. 7.

C. Asynchronous Communication

We now illustrate the use of algorithm $\text{RepC}$ in the case where the communication between nodes occurs asynchronously. To simulate this scenario, at each time instance, a random subset of agents communicates. The set of agents is $\mathcal{V} = \{1, \ldots, 5\}$, the network of agents is $\mathcal{G}_B$, and the set of attacked agents is $\mathcal{A} = \{1\}$. Fig. 8 depicts the state evolution of each agent when using the asynchronous version.
of algorithm RepC. Each normal node identifies and discards the information of the attacked agent.

D. Dynamic Network

Next, we test the scenario where the network of agents evolves with time and the attacked agents share the same value. We consider two networks composed of 10 agents, as depicted in Fig. 9, with a set of agents $V = \{1, \ldots, 10\}$ and of attacked agents $A = \{1\}$.

We consider that the dynamic network of agents for time instance $k > 0$ is given by $G^{(k)} = \begin{cases} G_C & \text{if } k \leq 10 \\ G_D & \text{otherwise} \end{cases}$. The consensus value of each agent, utilizing the iterative scheme (3), is depicted in Fig. 10.

E. Dynamic Network With Noisy Agents

Finally, we illustrate the scenario where not only the network of agents evolves with time but also the attacked agents share different values, which are drawn from uniform random variables with a fixed mean value. The results are shown in Fig. 11 and 12.

F. Stochastic Communication

When the communication between agents has a stochastic nature, we may still successfully apply RepC. We consider the network $G_E$ in Fig. 13, with $V_1$, and the set of attacked agents $A = \{1\}$. Further, at each time step, only a random subset of agents communicate between them. The described situation is depicted in Fig. 14, where the regular agents could effectively detect the attacked node and achieve the true consensus of the network.

G. RepC vs. State-of-the-Art Approaches

Here, we illustrate how the proposed algorithm competes with the state-of-the-art approaches, based on the idea that each agent discards a set of maximum and minimum neighbor values.

In the first example, consider the set of agents $V_2 = \{1, 2, 3, 4\}$, with the complete network [Fig. 15(a)] and attacked agents $A = \{1\}$. Using the state-of-the-art, i.e., when each agent discards the maximum and minimum neighbors’ values, we obtain the result depicted in Fig. 16. The method cannot deter the attack, and the regular agents converge to the attacker value. Using RepC, as illustrated in Fig. 17, the regular
Fig. 13. Network of agents $G_E$.

Fig. 14. Consensus using a RepC method, with network $G_E$, set of agents $V_1$, attacker agents $A = \{1\}$ and stochastic communication.

Fig. 15. (a) Network of agents $G_F$. (b) Network of agents $G_G$.

Fig. 16. Consensus using a state-of-the-art method, with network $G_F$, set of agents $V_2$, and set of attacked agents $A = \{1\}$. The black line is the true consensus value.

Fig. 17. Consensus using RepC, with network $G_E$, set of agents $V_2$, and attacker agents $A = \{1\}$. The black line is the true consensus.

Fig. 18. Consensus using a state-of-the-art method, with network $G_G$, set of agents $V_3$, and set of attacked agents $A = \{1\}$. The black line is the true consensus value.

Fig. 19. Consensus using RepC, with network $G_F$, set of agents $V_3$, and set of attacked agents $A = \{1\}$. The black line is the true consensus.

Fig. 20. Absolute difference between the consensus resulting from Algorithm RepC when node 1 is under attack to share the Gaussian noise with mean $\mu$ and standard deviation $\sigma$.

agents converge to a value close to the true value, with a small deviation caused by the influence of the $\varepsilon$ parameter.

In the second example, we consider the network of agents depicted in Fig. 15(b), the set of agents $V_3 = \{1, 2, 3, 4, 5\}$ and attacked agents set $A = \{1\}$. The example portrays the scenario where an attacker stubbornly sends to the neighbors the true consensus value. In Fig. 18, we present the consensus states of the agents when using the state-of-the-art approach. We can see that the agents cannot converge to the true consensus value. In Fig. 19, we present the consensus state of the agents when using RepC, and the agents converge to the true consensus.

H. Consensus Final Error

To explore how different is the final consensus value produced by RepC and the consensus value without attacked nodes, we use the complete network of five agents depicted in Fig. 1(a), with agents’ initial states $x(0) = [1 \ 0 \ 3 \ 1.2 \ 2.5]^T$, where agent 1 is under attack and shares values from a Gaussian noise with mean $\mu$ and standard deviation $\sigma$. The consensus value, without attacked nodes, is 1.489. We compute the absolute difference between the consensus value found with RepC in the nonattacked case and the consensus value obtained with RepC when the attacker follows the mentioned strategy. Moreover, we ranged $\mu$ from 0 to 1 in steps of 0.005 and ranged $\sigma$ from 0.1 to 1 in steps.
of 0.005, repeating each attacking scenario 20 times to compute the absolute average error, see Fig. 20.

We can see from Fig. 20 that, on average, we obtain a small final consensus error. When $\mu$ is close to 1, and $\sigma$ is close to 0.1, the attacked node state value is close to what it would be in the nonattacked scenario ($x_i^{(1)} \approx 1$), and it takes more time to be classified as an attacker by its neighbors, yielding a slightly larger (final) consensus error.

IV. CONCLUSION

In this work, we presented a reputation-based consensus algorithm ($\text{RepC}$) for discrete-time synchronous and asynchronous communications in (possibly dynamic) networks of agents. By assigning a reputation value to each neighbor, an agent may discard information from neighbors presenting abnormal behavior.

Algorithm $\text{RepC}$ converges with an exponential rate, and it has polynomial time complexity. Thus, for a network of agents, if we run $i \in \mathbb{N}$ iterations of $\text{RepC}$, we incur in $O(l^{Pi})$ time complexity, where $l$ is the greatest number of neighbors a node has in that network. For attacks with certain properties, we proved that the algorithm does not produce false positives. For other types of attacks, we illustrate the behavior of the proposed algorithm, which also worked as envisaged.

Future work directions include introducing the reputation idea for other types of consensus algorithms. Furthermore, a relevant additional theoretical property to prove (even if only for some type of attacks) is whether or not $\text{RepC}$ may cause an agent to wrongly classify a neighbor as attacked, i.e., if there are false negatives.

REFERENCES

[1] J. Tsitsiklis, D. Bertsekas, and M. Athans, “Distributed asynchronous deterministic and stochastic gradient optimization algorithms,” IEEE Trans. Autom. Control, vol. 31, no. 9, pp. 803–812, Sep. 1986.

[2] B. Johansson, T. Keviczky, M. Johansson, and K. H. Johansson, “Subgradient methods and consensus algorithms for solving convex optimization problems,” in Proc. IEEE 47th Conf. Decis. Control, 2008, pp. 4185–4190.

[3] 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, Jun. 2003.

[4] J. Cortés, S. Martínez, and F. Bullo, “Robust rendezvous for mobile autonomous agents via proximity graphs in arbitrary dimensions,” IEEE Trans. Autom. Control, vol. 51, no. 8, pp. 1289–1298, Aug. 2006.

[5] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle, “Layering as optimization decomposition: A mathematical theory of network architectures,” Proc. IEEE, vol. 95, no. 1, pp. 255–312, Jan. 2007.

[6] D. Silvestre, J. Hespanha, and C. Silvestre, “A pagerank algorithm based on asynchronous Gauss–Seidel iterations,” in Proc. Amer. Control Conf., 2018, pp. 484–489.

[7] R. Olfati-Saber, “Distributed Kalman filtering for sensor networks,” in Proc. IEEE 46th Conf. Decis. Control, 2007, pp. 5492–5498.

[8] P. Ariksson and A. Rantzer, “Experimental evaluation of a distributed Kalman filter algorithm,” in Proc. IEEE 46th Conf. Decis. Control, 2007, pp. 5499–5504.

[9] A. Haseltalab and M. Akar, “Approximate byzantine consensus in faulty asynchronous networks,” in Proc. Amer. Control Conf., 2015, pp. 1591–1596.

[10] A. Haseltalab and M. Akar, “Convergence rate analysis of a fault-tolerant distributed consensus algorithm,” in Proc. IEEE 54th Conf. Decis. Control, 2015, pp. 5111–5116.

[11] D. Silvestre, P. Rosa, R. Cunha, J. P. Hespanha, and C. Silvestre, “Gossip average consensus in a byzantine environment using stochastic set-observers,” in Proc. IEEE 52nd Conf. Decis. Control, 2013, pp. 4373–4378.

[12] D. Silvestre, P. Rosa, J. P. Hespanha, and C. Silvestre, “Finite-time average consensus in a byzantine environment using set-observers,” in Proc. Amer. Control Conf., 2014, pp. 3023–3028.

[13] D. Silvestre, P. Rosa, J. P. Hespanha, and C. Silvestre, “Stochastic and deterministic fault detection for randomized gossip algorithms,” Automatica, vol. 78, pp. 46–60, 2017.

[14] G. Ramos, D. Silvestre, and C. Silvestre, “A general discrete-time method to achieve resilience in consensus algorithms,” in Proc. IEEE 59th Conf. Decis. Control, 2020, pp. 2702–2707.

[15] G. Ramos, D. Silvestre, and C. Silvestre, “General resilient consensus algorithms,” Int. J. Control, vol. 95, no. 6, pp. 1482–1496, 2022.