Resiliency against malicious agents in maximum-based consensus

Masahiro Nakamura\textsuperscript{a}, Hideaki Ishii \textsuperscript{a} \textsuperscript{a} and Seyed Mehran Dibaji \textsuperscript{b}

\textsuperscript{a}Department of Computer Science, Tokyo Institute of Technology, Yokohama, Japan; \textsuperscript{b}Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA, USA

\section*{ABSTRACT}

In this paper, we develop distributed algorithms for achieving resilient consensus via the maximum-value-based approach when adversarial agents may be present in the network. The adversaries intend to prevent the nonfaulty, normal agents from reaching consensus. We extend the class of resilient methods known as the mean subsequence reduced (MSR) algorithms, where the agents make selections on their neighbours’ information at the time of their updates so as to reduce the influence of the malicious agents. In particular, the normal agents try to arrive at the maximum value of their initial states. Due to the malicious agents, the exact maximum may not be reached, the advantage of the approach is the finite-time convergence. We present both synchronous and asynchronous versions of the update schemes and characterize graph theoretic conditions for achieving resilient consensus. A numerical example is provided to illustrate our results.

\section*{1. Introduction}

Recent years have witnessed an increase in the risks of networked control systems becoming subject to serious cyber attacks. While this may not be surprising in view of the wide use of general purpose networks in large-scale control systems, it is important to work towards raising security against malicious intrusions from the systems theoretic perspective \cite{1}. In this context, one fundamental problem is the so-called resilient consensus for multi-agent systems. Here, the agents in a network aim at achieving consensus in their values by interacting with each other, but some of them may be faulty or controlled by attackers trying to disturb the process and even prevent the agents from achieving the task of consensus.

In this paper, we follow the line of research on resilient consensus developed in \cite{2–6}. It is assumed that the malicious agents may behave in a worst-case sense. They may know everything about the nonfaulty, normal agents and their interactions, and they may communicate with each other regardless of the original network structure. On the other hand, to mitigate the influence of such malicious agents, the normal agents take the precaution by ignoring their neighbours whose states exhibit unsafe values and differ the most from their own. Assuming the knowledge of the maximum number $f$ of malicious agents in the entire network, each normal agent neglects up to $f$ largest and up to $f$ smallest values from its neighbours in its updating process. We refer to such update schemes as the mean subsequence reduced (MSR) type algorithms.

This line of research can in fact be seen as a generalization of the distributed algorithms in computer science, see, e.g. \cite{7,8} and the references therein. More specifically, the main feature of the recent results lies in characterizing tight conditions on the network structure to guarantee that the nonfaulty agents reach consensus. By contrast, in conventional results in computer science, the networks have typically been assumed to be complete graphs; this is due to the applications considered there, which are computers and terminals connected over wired networks.

The main focus of this paper is to increase the convergence speed in MSR-type algorithms. Note that most existing results attain consensus only with asymptotic convergence because of the averaging process in the updates. To this end, we employ the maximum-value based approach for realizing finite time convergence though we do not aim at finding the exact maximum value in the network, which is difficult in the presence of adversaries. We will show that in comparison with conventional approaches, not only the convergence becomes faster, but also the necessary computation at each update is less and, moreover, the malicious agents can be identified when they change their values often.

The maximum consensus problem has been studied in, e.g. \cite{9,10}. The idea is relatively simple. As the agents exchange their values, they should update by taking the largest among their neighbours’ values. A disadvantage may be that the information of other agents would be quickly lost in the course. In \cite{11}, the maximum...
consensus algorithm is considered when the updates are asynchronous and communication among agents contains delays while in [12], a more complicated situation is studied where the nodes may dynamically join and leave the network over time.

The usefulness of the maximum-based approach has been demonstrated in the recent literature. For example, in the application of clock synchronization for wireless sensor networks, consensus-based algorithms have been used by [13] and a resilient version by [14]. In a similar context, in [15], it is argued that as long as the clocks of the sensor nodes synchronize, the parameters of the agreed clocks such as their speeds and offsets should be of less importance; rather, fast convergence to consensus is critical in many real-time applications. Another interesting application of the maximum-based consensus is demonstrated in, e.g. [16, 17]. There, a method for estimating the total number of agents in the network in a distributed fashion by the agents is considered; this is done by computing the maximum of sequences of randomly generated numbers. The work [18] considers an efficient use of wireless communication channels for achieving maximum consensus through the idea of superposition of transmitted signals.

In this paper, we study the problem for both synchronous and asynchronous updates for the normal agents where time-varying delays in communication may be present. We characterize the underlying network structure for the agents’ communication and obtain necessary and sufficient conditions. The graph related notion relevant in this setting is called robustness, which is a measure of connectivity of graphs. It was first introduced by [3] for resilient consensus with first-order agent dynamics and then was further explored in [4, 5] for asynchronous and delayed case.

With the maximum-based approach, our goal is to increase the convergence speed in resilient consensus. In fact, we achieve finite time convergence with probability one. This is in contrast to conventional MSR algorithms, whose convergence is asymptotic. We do not, on the other hand, intend to find the exact maximum of agents’ states. In this regard, it is critical to use randomization techniques in the proposed schemes; for general discussions on randomized algorithms, see, e.g. [19, 20]. In terms of network connectivity in consensus, it is well known that the minimum requirement is that the graph contains a directed spanning tree [21]. In such a case, it is not possible to find the true maximum. In our schemes, the agents will randomly take either the maximum or the second maximum among their neighbour values in state updates. It will be shown that deterministic updates are not sufficient; this idea comes from results in quantized consensus mentioned below. As an additional feature of the maximum-based approach, the normal agents can identify the malicious ones especially if they change their values often. In contrast, this function cannot be realized in the conventional MSR algorithms.

We should highlight that the delay model in our study follows those commonly used in the control literature. It is real-time oriented in the sense that agents make updates at any time with the most recent information received from their neighbours. In contrast, in the computer science literature [2, 8], the agents are sometimes required to wait until all neighbours send their new values. We also remark that we study agents taking real-valued states, but the results will be obtained by utilizing proof techniques in quantized consensus in [6]. Such problems have been motivated by concerns on limited capabilities in communications and computations in the agents. Recent studies can be found in, e.g. [22–24].

The paper is outlined as follows: in Section 2, we introduce the problem setting of consensus in the presence of malicious agents. In Section 3, we consider the update scheme when the normal agents are synchronous while Section 4 deals with the asynchronous case. In Section 5, we illustrate the effectiveness of the proposed algorithms through a numerical example. Finally, Section 6 provides concluding remarks. A preliminary version of this paper appeared as [25]; in the current paper, proofs of all results, further simulations and detailed discussions are included.

2. Problem formulation

2.1. Preliminaries on graphs

We review some basic notions of graphs. A directed graph with $n$ agents is given as $G = (\mathcal{V}, \mathcal{E})$ with the set of nodes $\mathcal{V} = \{1, \ldots, n\}$ and the set of edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$. The edge $(j, i) \in \mathcal{E}$ indicates that node $i$ can receive the information of node $j$. Here, we assume that there is no self-loop (i.e. $(i, i) \notin \mathcal{E}$). A graph is said to be complete if there is an edge $(j, i)$ for each $i,j \in \mathcal{V}$, $i \neq j$. For node $i$, its neighbours are the nodes having edges towards $i$; let the set of such nodes be $\mathcal{N}_i = \{j : (j, i) \in \mathcal{E}\}$. The number of neighbours of node $i$ is called its degree and is denoted as $d_i = |\mathcal{N}_i|$.

A path from node $v_1$ to $v_p$ is a sequence of distinct nodes $(v_1, v_2, \ldots, v_p)$, where $(v_i, v_{i+1}) \in \mathcal{E}$ for $i = 1, \ldots, p - 1$. If for all distinct nodes $i$ and $j$, there is a path from $i$ to $j$, the graph is said to be strongly connected. A directed graph is said to contain a directed spanning tree if for some node $i$, there is a path to every node $j \neq i$.

For the distributed algorithms to be presented, we must introduce the measure of connectivity for graphs called robustness [3].

**Definition 2.1:** The graph $G = (\mathcal{V}, \mathcal{E})$ is said to be $(r,s)$-robust ($r, s < n$) if for every pair of nonempty disjoint subsets $\mathcal{V}_1, \mathcal{V}_2 \subseteq \mathcal{V}$ of nodes, at least one of the
The degree of a (r, s)-robust graph is (r + s − 1)-robust.

Moreover, a graph G is (r, s)-robust if it is (r + s − 1)-robust.

In the current problem setting, the property (ii) in the lemma above is of interest because robust graphs may not be strongly connected in general. To check whether a given graph is robust is computationally intensive and must be done following the definition in a combinatorial manner [26].

2.2. Resilient consensus

Consider a network of agents cooperating over the directed graph $G = (V, E)$. Each agent applies a control rule consisting of its neighbours’ state values to make updates by

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

where $x_i[k]$ and $u_i[k]$ represent agent i’s state and control at time k. To achieve consensus means that the agents converge to a globally common value. In the maximum-based approach, the agents seek consensus by updating their states using only the maximum value among the states of itself and its neighbours as

$$x_i[k + 1] = \max \{x_i[k], x_j[k] : j \in N_i \}. \tag{2}$$

Under the update rule (2), if the graph $G$ is strongly connected, the agents’ state will reach the global maximum value in a finite time $K$ as $x_i[k] = \max_{j \in V} x_j[0]$ for $k \geq K$. The time $K$ is upper bounded by the maximum of the shortest path length between any two nodes in the graph $G$; see, e.g. [10].

In this paper, we develop a variant of the maximum-based consensus algorithm with four features. (i) It is a randomized algorithm, which can reach consensus under the weaker assumption that the graph contains a directed spanning tree. This condition is in general necessary and sufficient for attaining consensus [21]. Thus our proposed algorithms function in a more general network setting, but the tradeoff would be that the true maximum may not be attained with some probability even if the graph is strongly connected. (ii) The algorithm is capable to deal with malicious agents which may not follow the given algorithm due to faults or cyber attacks. (iii) The maximum-based approach allows us to obtain resilient algorithms with faster convergence, which is in fact finite time with probability one. In contrast, the algorithms in [3, 4] follow the conventional, average-based approach and reach consensus only asymptotically. (iv) The last feature is interesting: each agent is capable to identify the malicious agents, which is impossible in conventional resilient consensus. It is due to the nature that only finite values can be taken by the normal agents under the max-consensus approach, but malicious agent may violate this.

The objective of the agent network is to achieve global agreement among all agents by applying a local update protocol motivated by (2) in a probabilistic sense. In this work, the network is assumed to have some misbehaving agents that do not follow the given update rule. In the next section, we provide more details for studying this case.
2.3. Resiliency notions

We introduce notions related to malicious agents and consensus in the presence of such agents [3, 4].

Node $i$ is called normal if it updates its state based on the predefined control (to be specified later). Otherwise, it is called malicious and may make arbitrary updates. Malicious agents might deceive normal ones and prevent them from achieving consensus. We study the worst-case behaviours of such agents: They may have the full knowledge of the network and the agents’ states and may even communicate among themselves regardless of the presence of edges. While they will be “stealthy” as long as they remain among the normal agents, their influence on convergence will be very minor. To formally limit the scope of adversaries, we assume that a bound on the number of malicious agents is available. This is a common assumption in the area of distributed algorithms [7].

Definition 2.3: The network is said to be $f$-total malicious if the number $n_M$ of faulty agents is at most $f$: $n_M \leq f$.

The index set of malicious nodes is denoted by $\mathcal{M} \subset \mathcal{V}$. The numbers of normal agents and malicious agents are denoted by $n_N$ and $n_M$, respectively. Also, the states for the normal agents and malicious agents are given in vector forms as $x^N[k] \in \mathbb{R}^{n_N}$ and $x^M[k] \in \mathbb{R}^{n_M}$, respectively.

We now provide the definition of consensus for the network in the presence of misbehaving agents adopted in this paper.

Definition 2.4: If for any initial states, any malicious inputs, and any possible set of malicious agents, the following conditions are met, then the network is said to reach resilient consensus:

1. Safety condition: For each set of initial values of the normal agents, there exists a set $\mathcal{S}$ such that for all normal agents $i \in \mathcal{V} \setminus \mathcal{M}$, it holds $x_i[k] \in \mathcal{S}, \forall k \in \mathbb{Z}_+$.
2. Agreement condition: There is a finite time $K_c \geq 0$ such that $\text{Prob}(x^M[K_c] \in \mathcal{C}_{nn} : x[0]) = 1$, where the consensus set is given as $\mathcal{C}_{nn} := \{x \in \mathbb{R}^{n_N} : x_1 = \cdots = x_{n_N}\}$.

As detailed below, the proposed algorithm involves randomization, and hence the notion of resilient consensus will be used in a probabilistic sense. In the condition (2) of Definition 2.4, the finite time $K_c$ is a random variable and varies depending on various parameters including the initial values of the agents. We must note that deriving a general bound on $K_c$ is a difficult problem.

3. Synchronous update rules

3.1. Max-MSR algorithm

First, we outline the algorithm employed for achieving consensus in the presence of misbehaving agents. The algorithm is the maximum-based version of weighted-MSR (W-MSR) studied in [3, 4] and will be called the Max-MSR algorithm. As we see in our development, this algorithm is closely related to the quantized version of W-MSR studied in [6].

We introduce three parameters for the algorithms:

(i) The probability $p \in (0, 1)$ used in the updates.
(ii) The estimate $\hat{n} \geq n$ of the total number of agents in the network.
(iii) The integer $f_i[k] \in [0, f]$ for agent $i$, representing the maximum number of malicious nodes among its neighbours. Initially, it is set as $f_i[0] = f$. Each time a malicious one is identified, its value is reduced by 1.

Under the maximum-based approach, the normal agents can identify the malicious agents in situations where they change their values often. This can be explained as follows: in our approach, the normal agents take at most $n$ values over time. Hence, here, each normal agent $i$ keeps record of the number of values that each of its neighbour has taken. If this number exceeds $n$ or its estimated value $\hat{n}$ for some neighbour, then agent $i$ can identify that this neighbour is malicious. This function is in fact critical in our development.

Each normal agent $i$ possesses the following parameters: its neighbour set $\mathcal{N}_i$, the bound $f$ on the number of malicious agents, the bound $\hat{n}$ on the total number of agents in the network, and the probability $p$ for updates. The algorithm consists of four steps as follows:

Algorithm 3.1: 0. At time $k = 0$, set $f_i[0] = f$ for $i \in \mathcal{V}$.
1. At each time step $k$, each normal agent $i$ receives the state values of its neighbours in $\mathcal{N}_i$, and sorts them together with its own value in a descending order. The order of its own value is set as $v_i[k] + 1$, that is, there are $v_i[k]$ neighbours whose values are greater than $x_i[k]$.
2. If the maximum of $x_i[k]$ and the $(f_i[k] + 1)$th value equals the minimum of $x_i[k]$ and the $(\hat{n} - f_i[k] + 1)$th value, then no update is made (i.e. $x_i[k + 1] = x_i[k]$).
3. (a) If $v_i[k] < f_i[k]$, there are two cases as follows:
   (i) If $|\mathcal{N}_i| > v_i[k] + f_i[k]$, then with probability $p$, the state is kept unchanged as $x_i[k + 1] = x_i[k]$, and with probability $1 - p$, the state is set to the one smaller than itself, next in the order.
   (ii) Otherwise, the state remains the same $x_i[k + 1] = x_i[k]$. 


(b) If $v_i[k] > f_i[k]$, then with probability $p$, the state is set to the $(f_i[k] + 1)$th value, and with probability $1-p$, it is updated as the value smaller than the $(f_i[k] + 1)$th one, next in the order.

4. If a neighbour $j$ has taken more than $\tilde{n}$ values so far, then this neighbour will be ignored from this time $k$ and we set $f_i[k+1] = f_i[k] - 1$. Otherwise, $f_i[k+1] = f_i[k]$.

To keep the exposition simple, we use common values for $p$ and $\tilde{n}$ though they may be different for each $i$. Also, in the following, for the number of malicious agents, we use the constant $f$ instead of $f_i$ as well.

The basic idea behind this algorithm is that the normal agents make updates by removing the values which may potentially be problematic. Since the number of malicious agents is bounded by $f$, it does not use up to $f$ values greater than itself and, also implicitly, $f$ values smaller than itself; it then updates to the maximum or the second maximum value among the remaining ones.

More specifically, the steps in the Max-MSR can be explained as follows. In step 1, agent $i$ sorts its neighbours’ values. Then, in step 2, it is checked if there are multiple agents taking the same value. In particular, if after removing the $f_i[k]$ largest and the $f_i[k]$ smallest values among the neighbours, all remaining agents including agent $i$ take the same value, then there is no need to update its state. For example, consider the case with $|N_i| = 3$ and $f_i[k] = 1$. Then, if agent $i$ takes the largest among its neighbours, i.e. $v_i[k] = 0$, then the condition is that its value is the same as the third largest (and thus the second largest). Hence, there may be one value smaller than agent $i$’s, but this one can be malicious by $f_i[k] = 1$.

Step 3 (a) corresponds to the situation where there are less than $f_i[k]$ neighbours taking values greater than agent $i$. This means that agent $i$ might have the maximum value among the normal neighbours, so it may keep its current value. Some checking is made in (i) before allowing this agent to probabilistically switch to one value smaller. The condition is that there is at least one neighbour left after removing the values greater than agent $i$’s own and the smallest $f_i[k]$ values. If this condition is not met, there is no candidate value to update to, and as said in (ii), no change will be made in the state. Step 3(b) applies if agent $i$’s value is definitely not the maximum among its normal neighbours and hence it will switch to the maximum after removing the top $f_i[k]$ values.

Compared to the conventional W-MSR, the protocol above is capable to reduce the necessary computation. First, as we have seen above, if malicious agents are identified, their values will not be considered in the updates. Second, in the Max-MSR algorithm, sorting of the neighbours’ values may be simplified. Depending on the collected values, if step 2 does not apply, those smaller than agent $i$’s own need not be sorted, and only the largest and the second largest among them should be found.

The problem considered in this section for the case of synchronous update rule is stated as follows: for the $f$-total malicious model, find a condition on the network topology such that the normal agents using the Max-MSR algorithm outlined above reach resilient consensus with probability one.

### 3.2. Characterization of network structures

The following theorem provides a necessary and sufficient condition to attain resilient consensus for the synchronous updating case. It characterizes robustness in the network structure to be the essential property as it enables the normal agents to maintain sufficient connectivity among themselves for mitigating the influence of the malicious agents.

We introduce the interval $S$ given as follows, which will be shown to be the safety interval:

$$S = \left[ \min x^N[0], \max x^N[0] \right],$$

where the minimum/maximum are taken over all vector entries. We use this convention throughout the paper.

**Theorem 3.2:** Under the $f$-total malicious model, the agent network reaches resilient consensus with probability one through the Max-MSR algorithm if and only if the underlying graph is $(f + 1, f + 1)$-robust.

To establish consensus in this probabilistic setting, we follow the arguments introduced in [6, 22], dealing with consensus with quantized (i.e. integer) state values. Though the agents take real values, this is possible because there will be only finite different values that both normal and malicious agents can take before consensus is reached. This is enabled by step 4 of the Max-MSR algorithm, where each agent keeps track of the maximum number of malicious agents in its neighbour. This is a unique feature of the approach.

The following is a slightly modified version of a result in [22], adapted to the current problem setup.

**Lemma 3.3:** Consider the network of agents interacting over the graph $G$ through the Max-MSR algorithm. Suppose that the following three conditions are satisfied for the normal agents in the network:

(C1) There exists a set $S$ such that for each normal agent $i, x_i[k] \in S$ for all $k \in \mathbb{Z}_+$ and all $x^N[0]$.

(C2) For each state $x[k] = x_0$ at time $k$, there is a finite time $k_x$ such that $\text{Prob}(x^N[k + k_x] \in C_{nx} : x[k] = x_0) > 0$.

(C3) If $x[k] \in C_{nx}$, then $x[k'] \in C_{nx}, \forall k' > k$.

Then, the network reaches consensus with probability one.
The conditions (C1)–(C3) in the lemma intuitively indicate that scenarios for arriving at consensus occur infinitely often in a probabilistic sense. In [22], the result is stated for the integer-valued consensus and moreover limited to the case of undirected graphs. We changed the statement for regular consensus as in [6], which is still valid in the present setting, and our result as well as the algorithm extend the techniques there towards the maximum-based case.

**Proof of Theorem 3.2:** The necessity can be shown in a similar fashion as in [3, 6]. We hence establish sufficiency in the following. It must be shown that by applying the Max-MSR on an $f$-total model, the conditions (C1)–(C3) in Lemma 3.3 hold.

We first prove (C1) related to the safety condition. In particular, we show that for all normal agents, $x_i[k]$ is always in the interval $S$ in (3). Let the minimum and maximum values of the normal agents be denoted by $\underline{x}[k] := \min x[k]$ and $\overline{x}[k] := \max x[k]$, respectively. In the network, at most $f$ malicious agents are present and, at each time, each normal agent ignores the values of at most $f$ neighbours from the above and also below. Hence, malicious agents whose values are outside the range $[\underline{x}[k], \overline{x}[k]]$ will not be taken into account. We thus have that the value of the normal agent $i$ is upper bounded by $x_i[k + 1] \leq \overline{x}[k]$ and, consequently, $\overline{x}[k + 1] \leq \overline{x}[k]$. Therefore, $\overline{x}[k]$ is a monotonically nonincreasing function of time $k$. Similarly, the value of the normal agent $i$ is lower bounded by $x_i[k + 1] \geq \underline{x}[k]$, and thus, $\underline{x}[k]$ is monotonically nondecreasing in $k$. We now conclude that for the normal agent $i$, its state satisfies $x_i[k] \in [\underline{x}[k], \overline{x}[k]] \subset S$ for all $k$. Thus we have shown (C1).

Next, we prove (C2) in Lemma 3.3. Note that $\underline{x}[k]$ and $\overline{x}[k]$ are contained in $S$ and are monotone; moreover, they take only a finite number of values in $S$ by steps 3 and 4 of the algorithm, which is upper bounded by $n_M + nf$. Hence, there is a finite time $K_c$ such that they both reach their final values with probability one. Denote the final values of $\underline{x}[k]$ and $\overline{x}[k]$ by $\underline{x}^*$ and $\overline{x}^*$, respectively. Now, to conduct a proof by contradiction, we assume $\underline{x}^* < \overline{x}^*$. Denote by $X_1[k]$ the set of all agents at time $k \geq K_c$ with state values equal to $\overline{x}^*$ or larger. Likewise, denote by $X_2[k]$ the set of agents that have the minimum value $\underline{x}^*$ or smaller. That is,

$$
X_1[k] := \{i \in \mathcal{V} : x_i[k] \geq \overline{x}^*\},
$$

$$
X_2[k] := \{i \in \mathcal{V} : x_i[k] \leq \underline{x}^*\}.
$$

We show that with positive probability, the normal agents in $X_1[k]$ decrease their values, and the normal agents in $X_2[k]$ increase their values at the next time step. These two sets are nonempty and disjoint by assumption. Moreover, the underlying graph is $(f + 1, f + 1)$-robust. Thus one of the conditions in Definition 2.1 must be fulfilled. In particular, there always exists a normal agent $i$ either in $X_1[k]$ or $X_2[k]$ with $f + 1$ edges from $\mathcal{V} \setminus X_1[k]$ or $\mathcal{V} \setminus X_2[k]$, respectively. We consider the two cases separately.

First, we suppose that the normal agent $i$ is in $X_1[k]$. Its value clearly is $x_i[k] = \overline{x}^*$. We now consider (a) and (b) in step 3 of the algorithm, depending on the value of agent $i$ relative to others. By assumption, agent $i$ has neighbours from $X_1[k]$ and also from more than or equal to $f + 1$ agents in $\mathcal{V} \setminus X_1[k]$. Hence, in total, it satisfies $|N_i| \geq v_i[k] + f + 1$. This indicates that (a)–(ii) does not apply in step 3.

Hence, for the case (a), we should consider (i). Since agent $i$ has more than or equal to $f + 1$ neighbours in $\mathcal{V} \setminus X_1[k]$, and their values are smaller than $\overline{x}^*$, with probability $1 - p$, it holds

$$
x_i[k + 1] \leq \overline{x}^* - \epsilon,
$$

where $\epsilon$ is the smallest, nonzero difference between any two values of the agents given by

$$
\epsilon := \min \{|x_i[k] - x_m[k]| : 0 < j, m \in \mathcal{V}, k \in [0, K_c + n_N]\}.
$$

Note that the possible values that the agents can take over the time interval of $[0, K_c + n_N]$ are bounded by $n_N + (K_c + n_N) n_M$. This is because in Max-MSR, any normal agent will only take either the initial values of the normal agents or the values of the malicious agents, which may vary at each time step. It is enough to know that $\epsilon$ is well defined and $\epsilon > 0$ because of the assumption $\underline{x}^* < \overline{x}^*$.

In the case of (b) of step 3, the number of agents taking values greater than $\overline{x}^*$ is at most $f$, and thus, the value next to the $(f + 1)$th belongs to an agent in $\mathcal{V} \setminus X_1[k]$. Thus, with probability $1 - p$, the update is made as

$$
x_i[k + 1] \leq \overline{x}^* - \epsilon.
$$

By (5) and (7), with positive probability, the normal agent $i$ taking the maximum value $\overline{x}^*$ decreases its value by $\epsilon$.

Next, we consider the case where agent $i$ is in $X_2[k]$. Then, obviously, $x_i[k] = \underline{x}^*$. In step 3 of Max-MSR, the number of neighbours of this agent $i$ is greater than or equal to $f + 1$ and their values are larger than $\underline{x}^*$. This means that agent $i$ is of order $f + 2$ or more, and the procedure in (a) does not apply. In (b) of step 3, the value of the $(f + 1)$th order is in $\mathcal{V} \setminus X_2[k]$ and is greater than $\underline{x}^*$. Thus, with probability $1 - p$,

$$
x_i[k + 1] \geq \underline{x}^* + \epsilon.
$$

Hence, with positive probability, the normal agent $i$ taking the minimum value $\underline{x}^*$ increases its value by $\epsilon$.

We must further show that, with positive probability, the agents in $\mathcal{V} \setminus X_1[k]$ will not update their values to come into $X_1[k + 1]$. Notice that if at time $k$, the normal agent $i$ is in $\mathcal{V} \setminus X_1[k]$, then $x_i[k + 1] \leq \overline{x}^* - \epsilon$. In
step 3 of the algorithm, if (a)-ii) applies, then agent $i$ does not update, and if (a)-i) applies, then with probability $p$, no update is made for agent $i$ either. If (b) applies, with probability $1-p$, it updates to the value next to the $(f+1)$th one; however, the number of agents taking values greater than $\bar{x}$ is at most $f$, its value is smaller than $\bar{x}$. Hence, with positive probability, agent $i$ is not in $\mathcal{X}_1[k+1]$. Similarly, agents in $\mathcal{V} \setminus \mathcal{X}_2[k]$ will not go into $\mathcal{X}_3[k+1]$.

In conclusion, for $k \geq K_c + r_N$, the number of normal agents in either $\mathcal{X}_1[k]$ and $\mathcal{X}_2[k]$ becomes zero with positive probability. This is a contradiction since by (4), these sets must contain normal agents taking the maximum/minimum values in the network. Hence, we have proved (C2).

Finally, we show (C3) in Lemma 3.3. Suppose that the normal agents reached consensus at a common value $x^*$. Because the number of malicious agents is upper bounded by $f$, the normal agents will ignore all such agents $j$ whose values satisfy $x_j \neq x^*$. Therefore, from the update rule, it holds that $x_i[k+1] = x^*, \forall i \in \mathcal{V} \setminus \mathcal{M}$. Thus we arrive at (C3).

The result of this theorem in terms of $(f+1, f+1)$-robust graphs is consistent with the real-valued agent probability, agent $\mathcal{G}$ with condition for reaching consensus in normal networks. In Theorem 3.2, the necessary and sufficient condition for reaching consensus in normal networks is $(f+1)$-robustness. By Lemma 2.2 (iii), a graph is $(f+1)$-robust, or 1-robust, if and only if it has a directed spanning tree.

3.3. Discussion

Some discussions on the proposed algorithm are in order.

As a special case of Theorem 3.2, we can consider the system without any malicious agents. Then, the Max-MSR algorithm reduces to the update rule in (2). The corollary below shows that in this case, consensus can be achieved if and only if the network consists of a directed spanning tree; this is the well-known condition for consensus [21]. To the best of our knowledge, our result is the first to deal with directed graphs and to present this condition in the context of the maximum-based consensus.

**Corollary 3.4:** When no malicious agent is present, the network of agents based on the maximum-based update rule reaches consensus with probability one if and only if the underlying graph has a directed spanning tree.

**Proof:** In Theorem 3.2, the necessary and sufficient condition for reaching consensus in normal networks with $f = 0$ is $(1, 1)$-robustness. By Lemma 2.2 (iii), a graph is $(1, 1)$-robust, or 1-robust, if and only if it has a directed spanning tree.

![Figure 2](image-url) A line graph with no malicious agent fails to reach consensus if $p = 1$ or $p = 0$.

It is emphasized that randomization is essential in the proposed algorithm. Related discussions can be found in, e.g. [6, 22, 23], but are focused on average consensus over undirected graphs. When no malicious agent is present (i.e. $f = 0$), consider the line graph in Figure 2, which clearly contains a spanning tree. Consensus is impossible to reach even in simple cases with a deterministic approach. For example, let the probability be $p = 1$ in the update rule; in this case, the Max-MSR algorithm coincides with the scheme in (2). Take the initial values as $x_i[0] = i$ for $i$. Then, it is easy to see that the agents remain at their initial values at all times. By contrast, under the proposed algorithm, the agents converge at the minimum value 1 almost surely. This, however, might take time because first agent 2 must update to the second largest value 1, and then agent 3 updates to 1, and so on.

Concerning the convergence time of the proposed algorithm, it is finite with probability one. This is in contrast to conventional MSR algorithms, where convergence is only asymptotic. However, further analysis can be involved and difficult in general. As noted above, our approach is based on techniques from quantized consensus. Because of the discrete state values and the probabilistic updates, the overall system can be represented as a Markov chain. The expected hitting times for consensus states can be analysed. Though convergence for quantized consensus has been studied in, e.g. [22, 27, 28], they are mostly limited to undirected graphs or graphs with simple structures such as line and star graphs. This type of analysis is left for future research.

4. Asynchronous update rule with delays

Next, we extend the resilient algorithm to an asynchronous updating scheme with time delays resulting from, e.g. congestion in the communication among agents and the asynchronous updates.

The updates are made asynchronously, meaning that each normal agent $i$ follows the steps in the Max-MSR algorithm of Section 3.1 if $k$ corresponds to its updating time. Otherwise, it keeps its value unchanged: $x_i[k + 1] = x_i[k]$. It is assumed that each normal agent $i$ makes an update at least once in each $K$ consecutive time steps with $K \geq 1$.

The time delays are assumed to be nonuniform, time varying, and upper bounded in length. In particular, at its update time $k$, the normal agent $i$ uses the value of its neighbour $j$ from time $k - \tau_{ij}[k]$. The bound is denoted
\[ \tau \geq 0 \] and thus it holds

\[ 0 \leq \tau_{ij}[k] \leq \tau \text{ for all } i, j \text{ and } k. \]

The specific value of this bound \( \tau \) need not be known in the algorithm. In contrast, the malicious agents may arbitrarily update their values at any time and may use the most recent information (with no delay) of all agents in the network.

Observe that in this model, the agents make updates based on possibly old, but the most recent information received from neighbours. The advantage in real-time aspects is obvious over some distributed algorithms in computer science [2, 8]. There, agents communicate and update based on rounds, where they all wait to make the \( \ell \)-th update until they receive the \( \ell \)-th messages from all neighbours.

We are ready to state the main result for the asynchronous case. It shows that for resilient consensus, a more restrictive condition on the graph robustness is required compared to that in Theorem 3.2. For the safety condition, let the interval \( S \) be given by

\[ S := [\min z^N[0], \max z^N[0]], \text{ where } z[k] = [x^T[k] x^T[k-1] \cdots x^T[k-\tau]]' \in \mathbb{R}^{(\tau+1)n}, \text{ and } z^N[k] \text{ consists of only the normal agents' values from } z[k]. \]

**Theorem 4.1:** Under the \( f \)-total malicious model, the network of asynchronous agents with delayed communication reaches resilient consensus with probability one through the Max-MSR algorithm if the underlying graph is \((2f+1)\)-robust and only if the graph is \((f+1,f+1)\)-robust.

**Proof:** As in the proof of Theorem 3.2, we apply the algorithm to the \( f \)-total model and show that the conditions (C1)–(C3) in Lemma 3.3 are satisfied.

First, we show (C1). Let \( \Xi[k] \) and \( \zeta[k] \), respectively, denote the maximum value and the minimum value of the normal agents from time \( k - \tau \) to \( k \) as

\[ \Xi[k] := \max(x^N[k], x^N[k-1], \ldots, x^N[k-\tau]), \]

\[ \zeta[k] := \min(x^N[k], x^N[k-1], \ldots, x^N[k-\tau]). \]  

By assumption, in the network, there are at most \( f \) malicious agents, and, moreover, in the update rule, each normal agent ignores at most \( f \) large values and at most \( f \) small values received from its neighbour states. Hence, the normal agents use only values within the interval \([\zeta[k], \Xi[k]]\). This implies that the value of each normal agent \( i \) can be bounded by

\[ x_i[k + 1] \leq \Xi[k]. \]  

Furthermore, by (8), for any normal agent \( i \), it holds that

\[ x_i[k] \leq \Xi[k], \ x_i[k-1] \leq \Xi[k], \ldots, \]

\[ x_i[k - (\tau - 1)] = x_i[k + 1 - \tau] \leq \Xi[k]. \]

Hence, by (9) and (10), we obtain the upper bound as

\[ \Xi[k + 1] = \max(x^N[k + 1], \ldots, x^N[k + 1 - \tau]) \leq \Xi[k]. \]

By a similar argument, we can show the lower bound as \( \zeta[k + 1] \geq \zeta[k] \). Thus \( \Xi[k] \) is monotonically nonincreasing and \( \zeta[k] \) is monotonically nondecreasing in time. That is, for each time \( k \), the state value of each normal agent \( i \) satisfies \( x_i[k] \in [\zeta[k], \Xi[k]] \subseteq S^T \). Hence, (C1) is established.

Next, we proceed to showing (C2). Since \( \Xi[k] \) and \( \zeta[k] \) are contained in \( S \) and are monotonically increasing in time, they have limit values which they reach with probability one. In fact, the limits are reached in finite time \( K_\tau > 0 \) with probability one. This is because they take only \( n \) discrete values determined by the initial values of the normal agents and also the values that the malicious agents may take over time, which is no greater than \( nt \) by step 4. Denote the limit values of \( \Xi[k] \) and \( \zeta[k] \) by \( \Xi^* \) and \( \zeta^* \), respectively.

Now, to prove by contradiction, assume that the limits are different: \( \zeta^* < \Xi^* \). Then, let \( X_{1}[k] \) and \( X_{2}[k] \) be the sets of normal agents whose values are equal to \( \Xi^* \) and \( \zeta^* \), respectively, at time \( k \geq K_\tau \). That is,

\[ X_{1}[k] := \{ i \in V \setminus M : x_i[k] = \Xi^* \}, \]

\[ X_{2}[k] := \{ i \in V \setminus M : x_i[k] = \zeta^* \}. \]

As \( \Xi[k] \) converges to \( \Xi^* \), at least one of the sets \( X_{1}[k] \) or \( X_{2}[k] \) must contain a normal agent whose value is \( \Xi^* \). Hence, with probability one, we have

\[ \bigcup_{k=K_\tau}^{K_\tau + \tau} X_{1}[k] \neq \emptyset. \]  

A similar argument holds for \( X_{2}[k] \).

Here, we claim that the set \( X_{1}[K_\tau + l] \) is nonempty. Suppose that this set is empty. We show that, with a positive probability, it holds \( \bigcup_{k=K_\tau}^{K_\tau + \tau} X_{1}[k] = \emptyset \), which contradicts with (11). To do so, it suffices to show if \( X_{1}[k] \) is empty at \( k \geq K_\tau \), then the probability that \( X_{1}[k + 1] \) is empty is nonzero.

First, we show that any normal agent \( i \) in \( V \setminus X_{1}[k] \) at time \( k \geq K_\tau \) will not be in the set \( X_{1}[k + 1] \) at the next time step. Clearly, if the normal agent \( i \) does not make an update at this time, it cannot go inside \( X_{1}[k + 1] \). Hence, suppose that the normal agent \( i \) makes an update. Recall that each normal agent updates at least once in the duration of \( K_\tau \). Due to the assumption that \( X_{1}[k] \) is empty, the value of agent \( i \) can be upper bounded by \( \Xi^* - \epsilon' \), where \( \epsilon' > 0 \) is defined as the maximum difference among agents’ values over the
time interval of \([0, K_c + \bar{k} \cdot n_N]\): 
\[
\epsilon' := \min \left\{ |x[j] - x_m[k_m]| \neq 0 : \right. \\
j, m \in \mathcal{V}, k_j, k_m \in [0, K_c + \bar{k} \cdot n_N] \}
\]

Similarly to the proof for Theorem 3.2, by the assumption \(2\epsilon^* < \mathcal{E}^*\), it follows that \(\epsilon'\) is well defined and \(\epsilon' > 0\). Since the number of malicious agents is upper bounded by \(f\), the update rule of Max-MSR suggests that whether (a) or (b) applies in step 3, it holds \(x_i[k + 1] \leq \mathcal{E}^* - \epsilon'\). Similarly, we can show that \(\mathcal{X}_T[K_c]\) is nonempty.

We next must show that, at the next time step, the values of agents in \(\mathcal{X}_T[k]\) decrease and the values of those in \(\mathcal{X}_R[k]\) increase with probability one. Since the two sets \(\mathcal{X}_T[K_c]\) and \(\mathcal{X}_R[K_c]\) are nonempty and do not intersect, by \((2f + 1)\)-robustness, at least one of the sets contains a normal agent having 2\(f + 1\) edges from outside the set to which it belongs.

Assume that such a normal agent \(i\) is in \(\mathcal{X}_T[k]\). Then, its value clearly is \(x_i[k] = \mathcal{E}^*\). Agent \(i\) has \(2f + 1\) neighbours in \(\mathcal{V} \setminus \mathcal{X}_T[k]\), whose values are smaller than \(\mathcal{E}^*\). Hence, as in the proof of Theorem 3.2, by the Max-MSR, it follows that \(x_i[k + 1] \leq \mathcal{E}^* - \epsilon'\) with probability \(1 - p\). This indicates that agent \(i\) taking the maximum value \(\mathcal{E}^*\) reduces its value by \(\epsilon'\) with positive probability. A similar argument holds for the case of \(\mathcal{X}_R[k]\).

From the above, we have proven that at each time \(k \geq K_c\), with a positive probability, the agents in \(\mathcal{X}_T[k]\) decrease their values and those in \(\mathcal{X}_R[k]\) increase their values. Since the total number of normal agents is \(n_N\) and each agent updates at least once in \(\bar{k}\) time steps, for any \(k \geq K_c + \bar{k} \cdot n_N\), with positive probability, the number of agents in either \(\mathcal{X}_T[k]\) or \(\mathcal{X}_R[k]\) becomes zero. This is in contradiction because these sets are nonempty. Thus (C2) follows.

Finally, for showing (C3), assume that all normal agents have reached the common value \(\mathcal{E}^*\). As we consider the \(f\)-total model for the malicious agents, those taking values different from \(\mathcal{E}^*\) are all ignored by the normal agents. According to the update rule, it follows that \(x_i[k + 1] = \mathcal{E}^*\) for \(i \in \mathcal{V} \setminus \mathcal{M}\). Thus (C3) has been proven. This completes the proof.

The proof of Theorem 4.1 follows along similar lines as that of Theorem 3.2. One subtle technical difference is that the two sets \(\mathcal{X}_T[k]\) and \(\mathcal{X}_R[k]\) must include only normal agents whereas in the proof of Theorem 3.2, the two sets \(\mathcal{X}_T[k]\) and \(\mathcal{X}_R[k]\) may contain malicious agents as well. This is due to the differences in the robustness properties assumed in these results. Under \((2f + 1)\)-robustness, one of the sets \(\mathcal{X}_T[k]\) and \(\mathcal{X}_R[k]\) always contains at least one agent, which is normal by definition, having a sufficient number of neighbours. On the other hand, in Theorem 3.2, for a network with \((f + 1, f + 1)\)-robust structure, the two sets, even if malicious agents exist, will contain at least one normal agent with enough neighbours for the purpose, though it is only \(f + 1\) neighbours this time.

It is noted that a \((2f + 1)\)-robust graph is also \((f + 1, f + 1)\)-robust by Lemma 2.2. This indicates that there is a gap between Theorem 3.2 and Theorem 4.1 for the synchronous scheme and the asynchronous scheme, respectively. This gap is due to the fact that asynchrony in the updating times creates more ways for the malicious agents to deceive the normal agents. This point will be shown through simulations in the next section. For further discussions, see also [6].

5. Numerical example

Here, we present a numerical example to illustrate the effectiveness of the proposed resilient consensus algorithms.

First, we examine the synchronous update rules. Consider the network with seven nodes in Figure 1(a). The normal agents take their initial values from \(\{1, 2, \ldots, 6\}\). Agent 2 is chosen to be malicious and its value remains unchanged in each run as \(x_2[k] = 8\) for all \(k\). We compare the performance of the proposed Max-MSR algorithm with the conventional consensus algorithm and the W-MSR, where the weighted average is taken in the updates with equal weights for all values. The bound \(f\) on the number of malicious agents is set as \(f = 1\). For the Max-MSR algorithm, we use the probability \(p = 0.5\) in updates.

The time responses of the conventional consensus, the W-MSR, and the Max-MSR are shown in Figures 3–5, respectively. With the conventional approach, consensus is reached, but at 8, which is the value of the malicious agent 2 (in the dotted line). For both MSR-based cases, the normal agents come to consensus within the safety interval. However, there are differences in the speed of convergence. In W-MSR, it takes over 15 steps for the values to become sufficiently close to each other. In contrast, for the Max-MSR, the final value of 6 is reached in only eight steps.

To further compare the convergence speed of the three algorithms, we made Monte Carlo simulations
over different initial values for 100 runs. For the proposed Max-MSR, we also verified the effect of the probability $p$. To demonstrate the differences in the methods, the choices of the values were carefully made. The malicious agent 2 takes a fixed value at 7 while the normal agents take random initial values from $\{0, \ldots, 8\}$. Agent 4 is an exception and was always given the fixed initial value at 8, which is the maximum.

Table 1 shows the results for (i) the rate to successfully reach consensus (at error level 0.01) in 10,000 time steps, (ii) the average time for reaching consensus, and (iii) the average value of consensus among the normal agents, where the averages are taken over the successful cases. For the conventional consensus algorithm, it is evident that consensus takes place very close at the malicious value 7 (though this is in the safety interval). The W-MSR is more free from the malicious agent and converges at 6.32 on average. For these two methods, the average convergence times are about 30–40 steps.

For the Max-MSR, the probability $p$ has a significant effect. In particular, taking large or small values in the interval $(0, 1)$ may be harmful. As shown for the cases $p = 0.1, 0.9$, reaching consensus may take a very long time. Furthermore, in the limit case of $p = 1$, the success rate is as low as 0.31. This is because the algorithm becomes deterministic and the agents’ values may stay at their local maxima. That is, agent 4 taking the maximum value will remain unchanged, but other normal agents may be influenced by the malicious agent and gather at its value 7, failing to reach consensus. From Table 1, it seems reasonable to take $p$ in the range $[0.5, 0.7]$ from the viewpoint of consensus time and value.

Next, we looked at the case where the communication among agents is asynchronous and is subject to time delays. We apply the Max-MSR to the same network in Figure 1(a). This network does not meet the robustness condition in Theorem 4.1. Indeed, there are cases where consensus cannot be realized. Again, we set $p = 0.5$ and $f = 1$ with agent 2 being the malicious one, which periodically switches its value as $x_1[k] = 5$ at even time steps and $x_1[k] = 4$ at odd time steps. This choice is made so that the malicious agent stays within the safety interval and the normal agents will not ignore it. The initial state is set as $x(0) = [5 \ 3 \ 1 \ 3 \ 8 \ 0 \ 6]^T$. Also, the normal agents make updates at different times: Agents 1, 4, and 6 at odd time steps, and agents 3, 5 and 7 at even time steps, where the agents in the latter

| $p$  | Conventional | W-MSR | Max-MSR |
|------|--------------|-------|--------|
|      | Success rate | Ave. time | Ave. value |
| 0.1  | 1            | 32.05 | 6.98   |
| 0.3  | 1            | 41.22 | 6.32   |
| 0.5  | 1            | 1997.77 | 5.80   |
| 0.7  | 1            | 24.71 | 6.46   |
| 0.9  | 1            | 9.13  | 6.92   |
| 1    | 1            | 8.15  | 7.12   |
|      | 1            | 86.04 | 7.27   |
|      | 0.31         | 3.51  | 8.00   |
group make use of delayed information by 2 time steps (i.e. $\tau_{ij}[k] = 2$ for $i = 3, 5, 7$ and all $j$). Others can send data without any delay. The time responses are shown in Figure 6. In this case, the normal agents converge at values 4 and 5, divided into two groups. By using the graph in Figure 1(b), which is 3-robust, we confirmed that consensus is achieved as depicted in Figure 7.

6. Conclusion

In this paper, we studied the maximum-based approach to the resilient consensus problem. In the proposed MSR-type algorithm, each agent updates its state by taking the maximum among the neighbours’ values after removing suspicious ones. It was demonstrated that the algorithms probabilistically achieve resilient consensus in finite time and tight conditions on the network structure are obtained for the synchronous and asynchronous cases. Through simulations, the advantage over conventional methods especially in convergence speeds has been demonstrated. In future work, we will study the rate of convergence of the proposed algorithms.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was supported in part by JSPS under Grant-in-Aid for Scientific Research Grants No. 15H04020 and No. 18H01460.

Notes on contributors

**Masahiro Nakamura** received the B.Eng. degree in engineering systems from the University of Tsukuba, Japan, in 2014 and the M.Eng. degree in computer science from the Tokyo Institute of Technology, Japan, in 2016. He is currently with Canon Inc., Tokyo, Japan. His research interests include systems control and distributed algorithms.

**Hideaki Ishii** received the M.Eng. degree from Kyoto University, Japan, in 1998, and the Ph.D. degree from the University of Toronto, Canada, in 2002. He was a Postdoctoral Research Associate at the University of Illinois at Urbana-Champaign, USA, from 2001 to 2004, and a Research Associate at The University of Tokyo, Japan, from 2004 to 2007. Currently, he is a Professor in the Department of Computer Science, Tokyo Institute of Technology, Japan. His research interests are in networked control systems, multi-agent systems, distributed algorithms, and cyber security of control systems.

**Seyed Mehran Dibaji** received the B.Sc. degrees in electrical engineering and in applied mathematics, and the M.Sc. degree in electrical and control engineering, all from Amirkabir University of Technology (Tehran Polytechnic), Iran, in 2009, 2012, and 2012, respectively. He obtained his Ph.D. degree in computational intelligence and systems science from Tokyo Institute of Technology, Japan, in 2016. He is currently a Postdoctoral Associate with the Department of Mechanical Engineering, Massachusetts Institute of Technology, USA. His research interests include resilient distributed control of networks, smart grid control, and fuzzy logic systems.

**ORCID**

Hideaki Ishii [http://orcid.org/0000-0003-0367-824X](http://orcid.org/0000-0003-0367-824X)

Seyed Mehran Dibaji [http://orcid.org/0000-0002-1109-3092](http://orcid.org/0000-0002-1109-3092)

References

[1] Sandberg H, Amin S, Johansson KH. Special issue on cyberphysical security in networked control systems. IEEE Control Syst Mag. 2015;35(1).

[2] Vaidya N, Tseng L, Liang G. Iterative approximate Byzantine consensus in arbitrary directed graphs. In: Proceedings ACM Symposium Principles of Distributed Computing; 2012. p. 365–374.

[3] LeBlanc H, Zhang H, Koutsoukos X, et al. Resilient asymptotic consensus in robust networks. IEEE J Selected Areas Commun. 2013;31:766–781.

[4] Dibaji SM, Ishii H. Resilient multi-agent consensus with asynchrony and delayed information. In: Proceedings 5th IFAC Workshop on Distributed Estimation and Control in Networked Systems (NecSys’15); 2015. p. 28–33.

[5] Dibaji SM, Ishii H. Resilient consensus of second-order agent networks: asynchronous update rules with delays. Automatica. 2017;81:123–132.

[6] Dibaji SM, Ishii H, Tempo R. Resilient randomized quantized consensus. IEEE Trans Autom Contr. 2018;63(8):2508–2522.

[7] Lynch N. Distributed algorithms. San Francisco, CA: Morgan Kaufmann; 1997.

[8] Azadmanesh MH, Kiechaer RM. Asynchronous approximate agreement in partially connected networks. Int J Parallel Distrib Netw. 2002;5:26–34.

[9] Iutzeler F, Ciblat P, Jakubowicz J. Analysis of max-consensus algorithms in wireless channels. IEEE Trans Signal Proc. 2012;60:6103–6107.

[10] Nejad B, Attia S, Raisch J. Max-consensus in a max-plus algebraic setting: the case of fixed communication topologies. In: Proceedings International Symposium Information, Communication and Automation Technologies; 2009. p. 1–7.

[11] Giannini S, Petitti A, Di Paola D, et al. Asynchronous max-consensus protocol with time delays: convergence results and applications. IEEE Trans Circuits Systems I. 2016;63:256–264.

[12] Abdelrahim M, Hendrickx JM, Heemels M. Max-consensus in open multi-agent systems with gossip interactions. In: Proceedings 56th IEEE Conference on Decision and Control; 2017. p. 4753–4758.
[13] Schenato L, Fiorentin F. Average timeSynch: a consensus-based protocol for clock synchronization in wireless sensor networks. Automatica. 2011;47:1878–1886.

[14] Kikuya Y, Dibaji SM, Ishii H. Fault tolerant clock synchronization over unreliable channels. IEEE Trans Control Netw Syst. 2018;5:1551–1562.

[15] He J, Chen J, Cheng P, et al. Secure time synchronization in wireless sensor networks: a maximum consensus-based approach. IEEE Trans Parallel Distrib Syst. 2014;25:1055–1065.

[16] Varagnolo D, Pillonetto G, Schenato L. Distributed cardinality estimation in anonymous networks. IEEE Trans Autom Contr. 2014;59:645–659.

[17] Lucchese R, Varagnolo D, Delvenne JC, et al. Network cardinality estimation using max consensus: the case of Bernoulli trials. In: Proceedings 54th IEEE Conference on Decision and Control; 2015. p. 895–901.

[18] Molinari F, Stańczak S, Raisch J. Exploiting the superposition property of wireless communication for max-consensus problems in multi-agent systems. In: Proceedings 7th IFAC Workshop NecSys; 2018. p. 176–181.

[19] Motwani R, Raghavan P. Randomized algorithms. Cambridge, UK: Cambridge University Press; 1995.

[20] Tempo R, Ishii H. Monte carlo and las Vegas randomized algorithms for systems and control: an introduction. Eur J Control. 2007;13:189–203.

[21] Mesbahi M, Egerstedt M. Graph theoretic methods in multiagent networks. Princeton, NJ: Princeton University Press; 2010.

[22] Kashyap A, Başar T, Srikant R. Quantized consensus. Automatica. 2007;43:1192–1203.

[23] Carli R, Fagnani F, Frasca P, et al. Gossip consensus algorithms via quantized communication. Automatica. 2010;46:70–80.

[24] Cai K, Ishii H. Quantized consensus and averaging on gossip digraphs. IEEE Trans Autom Contr. 2011;56:2087–2100.

[25] Nakamura M, Ishii H, Dibaji SM. Maximum-based consensus and its resiliency. In: Proceedings 7th IFAC Workshop NecSys; 2018. p. 283–288.

[26] Zhang H, Fata E, Sundaram S. A notion of robustness in complex networks. IEEE Trans Control Netw Syst. 2015;2:310–320.

[27] Cai K, Ishii H. Average consensus on general strongly connected digraphs. Automatica. 2012;48:2750–2761.

[28] Etesami SR, Başar T. Convergence time for unbiased quantized consensus over static and dynamic networks. IEEE Trans Autom Contr. 2016;61:443–455.