Reaching Approximate Byzantine Consensus with Multi-hop Communication*

Lili Su, Nitin Vaidya
Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign
{lilisu3, nhv}@illinois.edu

November, 2014

Abstract

We address the problem of reaching consensus in the presence of Byzantine faults. Fault-tolerant consensus algorithms typically assume knowledge of nonlocal information and multi-hop communication; however, this assumption is not suitable for large-scale static/dynamic networks. A handful of iterative algorithms have been proposed recently under the assumption that each node (faulty or fault-free) can only access local information, thus is only capable of sending messages via one-hop communication. In this paper, we unify these two streams of work by assuming that each node knows the topology of up to l^{th} hop neighborhood and can send messages to other nodes via up to l-hop transmission, where $1 \leq l \leq n-1$ and n is the number of nodes. We prove a family of necessary and sufficient conditions for the existence of *iterative* algorithms that achieve *approximate Byzantine consensus* in arbitrary directed graphs. The class of iterative algorithms considered in this paper ensures that, after each iteration of the algorithm, the state of each fault-free node remains in the *convex hull* of the initial states of the fault-free nodes. The following *convergence* requirement is imposed: for any $\epsilon > 0$, after a sufficiently large number of iterations, the states of the fault-free nodes are guaranteed to be within ϵ of each other.

1 Introduction

Consensus is fundamental to diverse applications such as data aggregation [18], distributed estimation [24], distributed optimization [26], distributed classification [13], and flocking [16]. Reaching consensus

*This research is supported in part by National Science Foundation awards NSF 1329681. Any opinions, findings, and conclusions or recommendations expressed here are those of the authors and do not necessarily reflect the views of the funding agencies or the U.S. government.

resiliently in the presence of Byzantine faults has been studied extensively in distributed computing [20, 23, 4, 10, 5], communication networks [17], and mobile robotics [1]. A Byzantine fault is an arbitrary fault that encompasses both omission failures (e.g., crash failures, failing to receive a request, or failing to send a response) and commission failures (e.g., processing a request incorrectly, corrupting local state, and/or sending an incorrect or inconsistent response to a request). Dolev et al. [9] introduced the notion of *approximate Byzantine consensus* by relaxing the requirement of *exact* consensus [22]. The goal in approximate consensus is to allow the fault-free nodes to agree on values that are approximately equal to each other (and not necessarily exactly identical). While *exact* consensus is impossible in *asynchronous* systems [12] in presence of Byzantine faults, approximate consensus is achievable [9]. The notion of approximate consensus is of interest in *synchronous* systems as well, since approximate consensus can be achieved using distributed algorithms that do not require complete knowledge of the network topology [7]. The discussion in this paper applies to synchronous systems. However, analogous results can be obtained for an asychronous system too.

It has been shown that given f Byzantine nodes, if the network node-connectivity is at least $2f+1$, there exist algorithmic solutions for the fault-free nodes to reach consensus over all possible inputs. Conversely, if the network node-connectivity is strictly less than $2f + 1$, then reaching consensus is not guaranteed [11]. However, this stream of work implicitly assumes that each node can send messages to *any* other node via multi-hop transmission. As a result of this communication assumption, the proposed algorithms require fault-free nodes to keep track of the *entire* network topology, leading to huge consumption of both memory resource and computation power. On the contrary, iterative algorithms are typically characterized by local communication (among neighbors, or near-neighbors), simple computations performed repeatedly, and a small amount of state per node. [9, 22] present iterative approximate Byzantine consensus (IABC) algorithms that work correctly in fully connected graphs. Fekete [10] studies the convergence rate of approximate consensus algorithms. There have been attempts at achieving approximate consensus iteratively in *partially* connected graphs. Kieckhafer and Azadmanesh examined the necessary conditions in order to achieve "local" convergence in synchronous [19] and asynchronous [3] systems. [2] presents a specific class of networks in which convergence condition can be satisfied using iterative algorithms. [33, 21] consider a restricted fault model in which the faulty nodes are restricted to sending identical messages to their neighbors.

In this paper, we unify these two streams of work by considering a general communication model that encompasses the $(n-1)$-hop communication and the 1-hop communication, respectively, as two extreme cases. Concretely, we are interested in a class of iterative algorithms for achieving approximate Byzantine consensus in synchronous point-to-point networks that are modeled by arbitrary *directed* graphs. The IABC algorithms of interest have the following properties, which we will soon state more formally:

- *Initial state* of each node is equal to a real-valued *scalar input* provided to that node.
- *Validity* condition: After each iteration of an IABC algorithm, the state of each fault-free node must remain in the *convex hull* of the initial states of the fault-free nodes.
- *Convergence* condition: For any $\epsilon > 0$, after a sufficiently large number of iterations, the states of the fault-free nodes are guaranteed to be within ϵ of each other.

We assume that each node can send messages to nodes that are up to l hops away. We prove a necessary and sufficient condition for the existence of *iterative* algorithms that achieve *approximate Byzantine*

consensus in arbitrary directed graphs for a given l. The aforementioned two streams of work correspond to the two special cases when $l = n - 1$ and $l = 1$, respectively. The proof technique used for proving *sufficiency* in this paper is inspired by the prior work on non-fault-tolerant algorithms [7], as applied in our previous work as well [25, 27, 28].

The rest of the paper is organized as follows. Section 2 presents our system and network models. The family of iterative algorithms of interest are described in Section 3. The necessary condition is demonstrated in Section 4. The sufficiency of the condition obtained in Section 4 is shown constructively in Section 5. Section 6 comments on the connection with the aforementioned two streams of work. Section 7 discusses possible relaxations of our failure model and concludes the paper.

2 Network and Failure Models

In this section, we introduce our *communication* and *failure* models.

Communication Model The system is assumed to be *synchronous*. The communication network is modeled as a simple *directed* graph G, where $\mathcal{V}(G) = \{1, \ldots, n\}$ denotes the set of n nodes, and $\mathcal{E}(G)$ denotes the set of directed edges between nodes in $\mathcal{V}(G)$. We assume that $n \geq 2$, since the consensus problem for $n = 1$ is trivial. Node i can send messages to node j if and only if there exists an i, j–path of length at most l in G, where l is some given integer in $\{1, \ldots, n-1\}$. In addition, we assume each node can send messages to itself as well, i.e., $(i, i) \in \mathcal{E}(G)$ for all $i \in \mathcal{V}(G)$. For each node i, let N_i^{l-} be the set of nodes that can reach node i via at most l hops. Similarly, denote the set of nodes that are reachable from node i via at most l hops by N_i^{l+}. Due to the existence of self-loops, $i \in N_i^{l-}$ and $i \in N_i^{l+}$.

Note that node i may send a message to node j via different i, j–paths. To capture this distinction in transmission routes, we represent a message as a tuple $m = (w, P)$, where $w \in \mathbb{R}$ and P indicates the path via which message m was transmitted. Four functions are defined over m. Let function value be $\mathsf{value}(m) = w$ and let path be $\mathsf{path}(m) = P$, whose images are the first entry and the second entry, respectively, of message m. In addition, functions source and destination are defined by $\mathsf{source}(m) = i$ and $\mathsf{destination}(m) = j$ if P is an i, j–path, i.e., message m is sent from node i to node j.

Failure Model We consider the Byzantine failure model with up to f nodes becoming faulty. A faulty node may *misbehave* arbitrarily. Possible misbehavior includes sending incorrect and mismatching (or inconsistent) messages to different neighbors. In addition, a faulty node k may tamper message m if it is in the transmission path, i.e., $k \in \mathcal{V}(\mathsf{path}(m))$. Recall that $\mathcal{V}(\cdot)$ is the vertex set of a given graph. However, faulty nodes are only able to tamper $\mathsf{value}(m)$, leaving $\mathsf{path}(m)$ unchanged. This assumption is placed for ease of exposition, later in Section 7 we relax this assumption by considering the possibilities that faulty nodes may also tamper messages paths or even fake and transmit non-existing messages.

In addition, faulty nodes may potentially collaborate with each other. Moreover, faulty nodes are assumed to have complete knowledge of the execution of the algorithm, including the states of all nodes, contents of messages the other nodes send to each other, and the algorithm specification.

3 Iterative Approximate Byzantine Consensus (IABC) Algorithms

In this section, we describe the structure of the Iterative Approximate Byzantine Consensus (IABC) algorithms of interest, and state the validity and convergence conditions that they need to satisfy. With a slight abuse of terminology, we will use the terms *node* and *vertex* interchangeably in our presentation.

Each node i maintains state v_i, with $v_i[t]$ denoting the state of node i at the *end* of the t-th iteration of the algorithm. Initial state of node i, $v_i[0]$, is equal to the initial *input* provided to node i. At the *start* of the t-th iteration ($t > 0$), the state of node i is $v_i[t-1]$. The IABC algorithms of interest will require each node i to perform the following three steps in iteration t, where $t > 0$. Note that the faulty nodes may deviate from this specification.

1. *Transmit step:* Transmit current state, namely $v_i[t-1]$, as the message value to nodes in N_i^{l+}, i.e., the nodes that are reachable from node i via at most l hops. If node i is an intermediate node on the route of some message, then node i forwards that message as instructed by the message path.

2. *Receive step:* Receive messages from N_i^{l-}, i.e., the nodes that can reach node i via at most l hops. Denote by $\mathcal{M}_i[t]$ the set of messages that node i received at iteration t.

3. *Update step:* Node i updates its state using a transition function Z_i, where Z_i is a part of the specification of the algorithm, and takes as input the set $\mathcal{M}_i[t]$. Note that $\mathcal{M}_i[t]$ contains $v_i[t-1]$ because $i \in N_i^{l-}$.

$$v_i[t] = Z_i(\mathcal{M}_i[t]). \tag{1}$$

We now define $U[t]$ and $\mu[t]$, assuming that $\mathcal{F}$ is the set of Byzantine faulty nodes, with the nodes in $\mathcal{V} - \mathcal{F}$ being fault-free.

- $U[t] = \max_{i \in \mathcal{V}-\mathcal{F}} v_i[t]$. $U[t]$ is the largest state among the fault-free nodes at the end of the t-th iteration. Since the initial state of each node is equal to its input, $U[0]$ is equal to the maximum value of the initial input at the fault-free nodes.

- $\mu[t] = \min_{i \in \mathcal{V}-\mathcal{F}} v_i[t]$. $\mu[t]$ is the smallest state among the fault-free nodes at the end of the t-th iteration. $\mu[0]$ is equal to the minimum value of the initial input at the fault-free nodes.

The following conditions must be satisfied by an IABC algorithm in presence of up to f Byzantine faulty nodes:

- *Validity:* $\forall\, t > 0,\ \mu[t] \geq \mu[0]$ and $U[t] \leq U[0]$
- *Convergence:* $\lim_{t \to \infty} U[t] - \mu[t] = 0$

The objective of this paper is to identify the necessary and sufficient conditions for the existence of a *correct* IABC algorithm (i.e., an algorithm satisfying the above validity and convergence conditions) for a given G and a given l.

4 Necessary condition

For a correct IABC algorithm to exist, the underlying communication graph G must satisfy the necessary condition proved in this section.

Definition 4.1. *Let W be a set of vertices in G and x be a vertex in G such that $x \notin W$. A W, x–path is a path from some vertex $w \in W$ to vertex x. A set S of vertices such that $x \notin S$ is a W, x–*vertex cut *if every W, x–path contains a vertex in S. The minimum size of a W, x–vertex cut is called the W, x–connectivity and is denoted by $\kappa(W, x)$. Similarly, for any integer $l \geq 2$, a set S_l of vertices is a l–*restricted vertex cut *if the deletion of S_l destroys all W, x–paths of length at most l. Let $\kappa_l(W, x)$ be the minimum size of such restricted vertex cut in G.*

We now define relations $\Rightarrow_l$ and $\not\Rightarrow_l$ that are used frequently in our subsequent discussion.

Definition 4.2. *For non-empty disjoint sets of nodes A and B in G, we say $A \Rightarrow_l B$ if and only if there exists a node $i \in B$ such that $\kappa_l(A, i) \geq f + 1$; $A \not\Rightarrow_l B$ otherwise.*

Let $F \subseteq \mathcal{V}(G)$ be a set of vertices in G, denote the induced subgraph of G induced by vertex set $\mathcal{V} - F$ by G_F. [1]

Condition NC: For any node partition L, C, R, F of G such that $L \neq \emptyset, R \neq \emptyset$ and $|F| \leq f$, in the induced subgraph G_F, at least one of the two conditions below must be true: (i) $R \cup C \Rightarrow_l L$; (ii) $L \cup C \Rightarrow_l R$.

Theorem 4.1. *Suppose that a correct IABC algorithm exists for G. Then G satisfies Condition NC.*

Proof. The proof is by contradiction. Let us assume that a correct IABC exists, and there exists a partition L, C, R, F of $\mathcal{V}(G)$ such that $L \neq \emptyset, R \neq \emptyset$ and $|F| \leq f$, but neither $R \cup C \Rightarrow_l L$ nor $L \cup C \Rightarrow_l R$ holds, i.e., $R \cup C \not\Rightarrow_l L$ and $L \cup C \not\Rightarrow_l R$. Consider the case when all nodes in F, if $F \neq \emptyset$, are faulty, and the other nodes in sets L, C, R are fault-free. Note that the fault-free nodes are not aware of the identities of the faulty nodes. In addition, assume (i) each node in L has initial input μ, (ii) each node in R has initial input U, such that $U > \mu + \epsilon$ for some given constant ϵ, and (iii) each node in C, if $C \neq \emptyset$, has initial input in the interval $[\mu, U]$.

In the *Transmit step* of iteration 1, suppose that each faulty node $k \in F$ sends $w = \mu^- < \mu$ to nodes in $N_k^{l+} \cap L$, sends $w = U^+ > U$ to nodes in $N_k^{l+} \cap R$, and sends some arbitrary value in the interval $[\mu, U]$ to nodes in $N_k^{l+} \cap C$. For messages m such that the faulty node k is in its transmission path, i.e., $k \in \mathcal{V}(\mathsf{path}(m))$, if $\mathsf{destination}(m) \in L$, node k resets $\mathsf{value}(m) = \mu^-$; if $\mathsf{destination}(m) \in R$, node k resets $\mathsf{value}(m) = U^+$; if $\mathsf{destination}(m) \in C$, node k resets $\mathsf{value}(m)$ to be some arbitrary value in $[\mu, U]$.

Consider any node $i \in L$. Since $|F| \leq f$, we know $|N_i^{l-} \cap F| \leq f$. In addition, in G_F $C \cup R \not\Rightarrow_l L$ implies $\kappa_l(C \cup R, i) \leq f$. Let S_l be a minimum restricted $(C \cup R, i)$–cut in G_F. From the perspective of node i, there exist two possible cases:

[1] An induced subgraph of G, induced by vertex set $S \subseteq \mathcal{V}(G)$, is the subgraph H with vertex set S such that $\mathcal{E}(H) = \{(u, v) \in \mathcal{E}(G) : u, v \in S\}$. Recall that $\mathcal{V}(\cdot)$ and $\mathcal{E}(\cdot)$ are the vertex set and edge set, respectively, of a given graph.

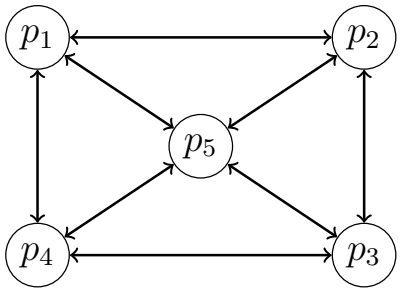

Figure 1: $n = 5$ and $f = 1$.

(a) Both S_l and $N_i^{l-} \cap F$ are non-empty: We know $|N_i^{l-} \cap F| \leq f$ and $|S_l| \leq f$. From node i's perspective, two scenarios are possible: (1) nodes in $N_i^{l-} \cap F$ are faulty, all the messages relayed via them are tampered and the other nodes are fault-free, and (2) nodes in S_l are faulty and the other nodes are fault-free.

In scenario (1), from node i's perspective, the untampered values are in the interval $[\mu, U]$. By validity condition, $v_i[1] \geq \mu$. On the other hand, in scenario (2), the untampered values are μ^- and μ, where $\mu^- < \mu$; so $v_i[1] \leq \mu$, according to validity condition. Since node i does not know whether the correct scenario is (1) or (2), it must update its state to satisfy the validity condition in both cases. Thus, it follows that $v_i[1] = \mu$.

(b) At most one of S_l and $N_i^{l-} \cap F$ is non-empty: Thus, $|S_l \cup (N_i^{l-} \cap F)| \leq f$. From node i's perspective, it is possible that the nodes in $S_l \cup (N_i^{l-} \cap F)$ are all faulty, the messages relayed via nodes in $S_l \cup (N_i^{l-} \cap F)$ are tampered while the rest of the nodes are fault-free. In this situation, the untampered values received by node i (which are all from nodes in $N_i^{l-} \cap L$) are all μ, and therefore, $v_i[1]$ must be set to μ as per the validity condition.

At the end of iteration 1: for each node i in L $v_i[1] = \mu$; similarly, for each node j in R, $v_j[1] = U$; if $C \neq \emptyset$, for each node i in C, $v_i[1] \in [\mu, U]$. All these conditions are identical to the condition when $t = 0$. Then by a repeated application of of above argument, it follows that for any $t \geq 0$, $v_i[t] = \mu$ for all $i \in L$, $v_j[t] = U$ for all $j \in R$ and $v_k[t] \in [\mu, U]$ for all $k \in C$, if $C \neq \emptyset$.

Since L and R both contain fault-free nodes, the convergence requirement is not satisfied. This contradicts the assumption that a correct iterative algorithm exists. □

Note that Condition NC is strictly weaker than the necessary condition under single hop message transmission model (i.e., $l = 1$) [30]. Consider the system depicted in Fig. 1. In this system, there are five processors p_1, p_2, p_3, p_4 and p_5; all communication links are bi-directional; and at most one processor can fail, i.e., $f = 1$. The topology of this system does not satisfy the necessary condition derived in [30]. Since in the node partition $L = \{p_1, p_4\}, R = \{p_2, p_3\}, C = \emptyset$ and $F = \{p_5\}$, neither $L \cup C \Rightarrow R$ nor $R \cup C \Rightarrow L$ holds. However, via enumeration it can be seen that the above graph (depicted in Fig. 1) satisfies Condition NC when $2 \leq l \leq 4 = n - 1$.

Corollary 4.2. *If G satisfies Condition NC, then n must be at least $3f + 1$, and each node must have at least $2f + 1$ incoming neighbors other than itself.*

Proof. The main techniques used here are fairly routine, and are given here largely for both concreteness and completeness.

We first show the claim that $n \geq 3f+1$. For $f = 0$, $n \geq 3f+1$ is trivially true. For $f > 0$, the proof is by contradiction. Suppose that $2 \leq n \leq 3f$. In this case, we can partition $\mathcal{V}(G)$ into sets L, R, C, F such that $0 \leq |L| \leq f$, $0 \leq |R| \leq f$, $0 \leq |F| \leq f$ and $|C| = 0$, i.e., C is empty. Since $0 \leq |L| \leq f$ and $0 \leq |R| \leq f$, we have $L \cup C \not\Rightarrow_l R$ and $R \cup C \not\Rightarrow_l L$, respectively in G_F. This contradicts the assumption that G satisfies Condition NC. Thus, $n \geq 3f+1$.

It remains to show that each node i must have at least $2f+1$ incoming neighbors other than itself. Let $N_i^- = \{j : (j,i) \in \mathcal{E}(G), \text{ and } i \neq j\}$ be the set of incoming neighbors of node i other than node i itself, i.e., $N_i^- = N_i^{1-} - \{i\}$. Suppose that, contrary to our claim, there exists a node i such that $|N_i^-| \leq 2f$. Define set $L = \{i\}$. Partition N_i^- into two sets F and H such that $|H| = \lfloor |N_i^-|/2 \rfloor \leq f$ and $|F| = \lceil |N_i^-|/2 \rceil \leq f$. Note that $H = \emptyset, F = \emptyset$ if and only if $f = 0$. Define $R = \mathcal{V} - F - L = \mathcal{V} - F - \{i\}$ and $C = \emptyset$. Since $|\mathcal{V}| = n \geq \max(2, 3f+1)$, R is non-empty. Now, $N_i^- \cap R = H$, and $|N_i^- \cap R| = |H| \leq f$. Since $L = \{i\}$, $|N_i^- \cap R| \leq f$ and $C = \emptyset$, it follows that $R \cup C \not\Rightarrow_l L$. Also, as $|L| = 1 < f+1$, $L \cup C \not\Rightarrow_l R$. This violates the assumption that G satisfies Condition NC and the proof is complete. □

In Section 5, we prove that Condition NC is also sufficient for the existence of a correct IABC algorithm. Condition NC is not very intuitive. In Theorem 4.3 below, we state another necessary condition that is equivalent to Condition NC, and is somewhat easier to interpret.

Definition 4.3. Meta-graph of SCCs: *Let $K_1, K_2, \ldots, K_k$ be the strongly connected components (i.e., SCCs) of G. The graph of SCCs, G^{SCC}, is defined by*

- *Nodes are $K_1, K_2, \ldots, K_k$;*
- *There is an edge (K_i, K_j) if there is some $u \in K_i$ and $v \in K_j$ such that (u, v) is an edge in G.*

Strongly connected component K_h is said to be a source component *if the corresponding node in G^{SCC} is <u>not</u> reachable from any other node in G^{SCC}.*

It is known that the G^{SCC} is a directed acyclic graph (i.e., DAG) [8], which contains no directed cycles. It can be easily checked that due to the absence of directed cycles and finiteness, there exists one node in G^{SCC} that is not reachable from any other node. That is, a graph G has at least one source component.

Definition 4.4. *The l^{th} power of a graph G, denoted by G^l, is a graph with the same set of vertices as G and a directed edge between two vertices u, v if and only if there is a path of length at most l from u to v in G.*

A path of length one between vertices u and v in G exists if (u, v) is an edge in G. And a path of length two between vertices u and v in G exists for every vertex w such that (u, w) and (w, v) are edges in G. Then for a given graph G with self-loop at each node, the $(u, v)^{th}$ element in the square of the adjacency matrix of G counts the number of paths of length at most two in G. Similarly, the $(u, v)^{th}$ element in the l^{th} power of the adjacency matrix of G gives the number of paths of length at most l between vertices u

and v in G. The power graph G^l is a multigraph[2] and there is a one-to-one correspondence between an edge e in G^l and a path of length at most l in G. Let e be an edge in G^l, and let $P(e)$ be the corresponding path in G, we say an edge e in G^l is covered by node set S, if $\mathcal{V}(P(e)) \cap S \neq \emptyset$, i.e., path $P(e)$ passes through a node in S.

For a given graph G and $F \subseteq \mathcal{V}(G)$, let $E = \{e \in \mathcal{E}(G^l) : \mathcal{V}(P(e)) \cap F \neq \emptyset\}$ be the set of edges in G^l that are covered by node set F. For each node $i \in \mathcal{V}(G) - F$, choose $C_i \subseteq N_i^{l-} - \{i\}$ such that $|C_i| \leq f$. Let

$$E_i = \{e \in \mathcal{E}(G^l) : e \text{ is an incoming edge of node } i \text{ in } G^l \text{ and } \mathcal{V}(P(e)) \cap C_i \neq \emptyset\}$$

be the set of incoming edges of node i in G^l that are covered by node set C_i. With these notations at hand, we are ready to introduce the notion of *reduced graph*.

Definition 4.5. Reduced Graph: *For a given graph G and $F \subseteq \mathcal{V}(G)$, a reduced graph of G^l, denoted by $\widetilde{G^l}_F$, is a graph where the node set and edge set are defined by*
(i) $\mathcal{V}(\widetilde{G^l_F}) = \mathcal{V}(G) - F$, and
(ii) $\mathcal{E}(\widetilde{G^l_F}) = \mathcal{E}(G^l) - E - \cup_{i \in \mathcal{V}(G)-F} E_i$, respectively.

Note that for a given G and a given F, multiple reduced graphs may exist. Let us define set R_F to be the collection of all reduced graph of G^l for a given F, i.e.,

$$R_F \quad = \quad \{\widetilde{G^l}_F : \widetilde{G^l}_F \text{ is a reduced graph of } G^l\}. \tag{2}$$

Since G^l_F, the l^{th} power of the induced subgraph G_F, itself is a reduced graph of G^l, thus R_F is nonempty. In addition, $|R_F|$ is finite since the graph G is finite,

Theorem 4.3. *Suppose that graph G satisfies Condition NC, then for any $F \subseteq \mathcal{V}(G)$ such that $|F| \leq f$, every reduced graph $\widetilde{G^l}_F$ obtained as per Definition 4.5 must contain exactly one* source component.

Proof. For any reduced graph $\widetilde{G^l}_F$, the meta-graph $(\widetilde{G^l}_F)^{SCC}$ is a DAG and finite. Thus, at least one source component must exist in $\widetilde{G^l}_F$. We now prove that $\widetilde{G^l}_F$ cannot contain more than one source component. The proof is by contradiction. Suppose that there exists a set $F \subseteq \mathcal{V}(G)$ with $|F| \leq f$, and a reduced graph $\widetilde{G^l}_F$ corresponding to F, such that $\widetilde{G^l}_F$ contains at least two source components, say K_1 and K_2, respectively. Let $L = K_1$, $R = K_2$, and $C = \mathcal{V} - F - L - R$. Then L, R, C together with the given F form a node partition of $\mathcal{V}(G)$ such that $L \neq \emptyset, R \neq \emptyset$ and $|F| \leq f$.

Since graph G satisfies Condition NC, without loss of generality, assume that $R \cup C \Rightarrow_l L$, i.e., there exists a node $i \in L$ such that $\kappa_l(R \cup C, i) \geq f+1$ in G_F. On the other hand, since L is a source component in $\widetilde{G^l}_F$, by the definition of reduced graph, we know all paths from $R \cup C$ to node i of length at most l in G are covered by $C_i \cup F$, where C_i is defined in Definition 4.5. Thus, C_i is a restricted $(R \cup C, i)$–cut of G_F. However, by construction of $\widetilde{G^l}_F$, the size of C_i is at most f. So we arrive at a contradiction.

□

[2] A multigraph (or pseudograph) is a graph which is permitted to have multiple edges between each vertex pair, that is, edges that have the same end nodes. Thus two vertices may be connected by more than one edge.

Corollary 4.4. *Suppose that graph G satisfies Condition NC. Then it follows that in each reduced graph $\widetilde{G^l}_F \in R_F$, there exists at least one node that has directed paths to all the nodes in $\widetilde{G^l}_F$.*

This corollary follows immediately from Theorem 4.3.

Corollary 4.5. *Suppose that G satisfies Condition NC. Let $|F| = \phi$, for any $\widetilde{G^l}_F \in R_F$ with $\mathbf{H}$ as the adjacency matrix, $\mathbf{H}^{n-\phi}$ has at least one non-zero column.*

Proof. By Corollary 4.4, in graph $\widetilde{G^l}_F$ there exists at least one node, say node k, that has a directed path in $\widetilde{G^l}_F$ to all the remaining nodes in $\mathcal{V}_F$, i.e., $\mathcal{V}(G) - F$. Since the length of the path from k to any other node in $\widetilde{G^l}_F$ can contain at most $n - \phi - 1$ directed edges, the k-th column of matrix $\mathbf{H}^{n-\phi}$ will be non-zero.[3] □

Definition 4.6. *We will say that an entry of a matrix is "non-trivial" if it is lower bounded by β, where β is some constant to be defined later.*

5 Sufficiency: Algorithm 1

We introduce the definition of message cover that will be used frequently in this section. It is closely related to the notion of path cover that we defined before.

Definition 5.1. *For a communication graph G, let $\mathcal{M}$ be a set of messages, and let $\mathcal{P}(\mathcal{M})$ be the set of paths corresponding to all the messages in $\mathcal{M}$, i.e., $\mathcal{P}(\mathcal{M}) = \{\mathsf{path}(m)|m \in \mathcal{M}\}$. A message cover of $\mathcal{M}$ is a set of nodes $\mathcal{T}(\mathcal{M}) \subseteq \mathcal{V}(G)$, such that for each path $P \in \mathcal{P}$, we have $\mathcal{V}(P) \cap \mathcal{T}(\mathcal{M}) \neq \emptyset$. In particular, a minimum message cover is defined by*

$$\mathcal{T}^*(\mathcal{M}) \in \underset{\mathcal{T}(\mathcal{M}) \subseteq \mathcal{V}(G):\ \mathcal{T}(\mathcal{M}) \text{ is a cover of } \mathcal{M}}{\operatorname{argmin}} |\mathcal{T}(\mathcal{M})|.$$

Conversely, given a set of messages $\mathcal{M}_0$ and a set of nodes $\mathcal{T} \subseteq \mathcal{V}(G)$, a maximal set of messages $\mathcal{M} \subseteq \mathcal{M}_0$ that are covered by $\mathcal{T}$ is defined by,

$$\mathcal{M}^* \in \underset{\mathcal{M} \subseteq \mathcal{M}_0:\ \mathcal{T} \text{ is a cover of } \mathcal{M}}{\operatorname{argmax}} |\mathcal{M}|.$$

We further need the following two definitions before we are able to proceed to the description of our algorithm. Recall that $\mathcal{M}_i[t]$ is the collection of messages received by node i at iteration t. Let $\mathcal{M}'_i[t] = \mathcal{M}_i[t] - \{(v_i[t-1], (i,i))\}$. Sort messages in $\mathcal{M}'_i[t]$ in an increasing order, according to their message values, i.e., $\mathsf{value}(m)$ for $m \in \mathcal{M}'_i[t]$. Let $\mathcal{M}_{is}[t] \subseteq \mathcal{M}'_i[t]$ such that (i) for all $m \in \mathcal{M}'_i[t] - \mathcal{M}_{is}[t]$ and $m' \in \mathcal{M}_{is}[t]$ we have $\mathsf{value}(m) \geq \mathsf{value}(m')$; and (ii) the cardinality of a minimum cover of $\mathcal{M}_{is}[t]$ is exactly f, i.e., $|\mathcal{T}^*(\mathcal{M}_{is}[t])| = f$. Similarly, we define $\mathcal{M}_{il}[t] \subseteq \mathcal{M}'_i[t]$ as follows: (i) for all $m \in \mathcal{M}'_i[t] - \mathcal{M}_{il}[t]$ and $m'' \in \mathcal{M}_{il}[t]$ we have $\mathsf{value}(m) \leq \mathsf{value}(m'')$; and (ii) the cardinality of a minimum cover of $\mathcal{M}_{il}[t]$ is exactly f, i.e., $|\mathcal{T}^*(\mathcal{M}_{il}[t])| = f$. In addition, define $\mathcal{M}^*_i[t] = \mathcal{M}'_i[t] - \mathcal{M}_{is}[t] - \mathcal{M}_{il}[t]$.

[3] That is, all the entries of the column will be non-zero (more precisely, positive, since the entries of matrix $\mathbf{H}$ are non-negative). Also, such a non-zero column will exist in $\mathbf{H}^{n-\phi-1}$ too. We use the loose bound of $n-\phi$ to simplify the presentation.

Theorem 5.1. *Suppose that graph G satisfies Condition NC, then the sets of messages $\mathcal{M}_{is}[t]$, $\mathcal{M}_{il}[t]$ are well-defined and $\mathcal{M}_i^*[t]$ is nonempty.*

Proof. For ease of exposition, with a slight abuse of notation, we drop the time indices of $\mathcal{M}_i'[t]$, $\mathcal{M}_{is}[t]$, $\mathcal{M}_{il}[t]$ and $\mathcal{M}_i^*[t]$, respectively. From Corollary 4.2, we know $|N_i^-| \geq 2f+1$. Since $|\mathcal{T}^*(\mathcal{M}_{is})| = f$ and $|\mathcal{T}^*(\mathcal{M}_{il})| = f$, the message from at least one incoming neighbor of node i is not covered by $\mathcal{T}^*(\mathcal{M}_{is}) \cup \mathcal{T}^*(\mathcal{M}_{il})$. So $\mathcal{M}_i^*$ is nonempty.

We prove the existence of $\mathcal{M}_{is}$ and $\mathcal{M}_{il}$ by construction. The set $\mathcal{M}_{is}$ can be constructed using the following algorithm, which can be easily adapted for the construction of set $\mathcal{M}_{il}$. For clarity of proof, we construct $\mathcal{M}_{is}$ and $\mathcal{M}_{il}$ sequentially, although they can be found in parallel.

As before, sort the messages in $\mathcal{M}_i'$ in an increasing order, according to their messages values. Initialize $\mathcal{M}_{is} \leftarrow \emptyset$, $Q \leftarrow \emptyset$ and $\mathcal{M} \leftarrow \mathcal{M}_i'$. At each round, let m_s be a message with the smallest value in $\mathcal{M}$, and update Q, $\mathcal{M}$ as follows,

$$Q \leftarrow Q \cup \{m_s\};$$
$$\mathcal{M} \leftarrow \mathcal{M} - \{m_s\}.$$

If $|\mathcal{T}^*(Q)| \geq f+1$, set $\mathcal{M}_{is} \leftarrow Q - m_s$ and return $\mathcal{M}_{is}$; otherwise, repeat this procedure.

If the algorithm terminates, then by the code, it is easy to see that the returned $\mathcal{M}_{is}$ satisfies the following conditions: For all $m \in \mathcal{M}_i' - \mathcal{M}_{is}$ and $m' \in \mathcal{M}_{is}$ we have $\mathsf{value}(m) \geq \mathsf{value}(m')$; and the cardinality of a minimum cover of $\mathcal{M}_{is}$ is exactly f, i.e., $|\mathcal{T}^*(\mathcal{M}_{is})| = f$. It remains to show this algorithm terminates. Suppose this algorithm does not terminate. The problem of finding a minimum cover of a set of messages, i.e., computing $\mathcal{T}^*(Q)$, can be converted to the problem of finding a minimum cut of a vertex pair, which can be solved in polynomial time. Thus, non-termination implies that $|\mathcal{T}^*(\mathcal{M}_i')| \leq f$, which further implies that the l–restricted $(\mathcal{V}(G) - \{i\}, i)$–connectivity is less than or equal to f. On the other hand, consider the node partition that $L = \{i\}$, $R = \mathcal{V}(G) - \{i\}$, and $C = F = \emptyset$, neither $L \cup C \Rightarrow_l R$ nor $R \cup C \Rightarrow_l L$ holds. This contradicts the assumption that G satisfies Condition NC. So the above algorithm terminates.

We can adapt the above procedure to construct $\mathcal{M}_{il}$ by modifying the initialization step to be $Q \leftarrow \emptyset$, $\mathcal{M} \leftarrow \mathcal{M}_i' - \mathcal{M}_{is}$.

Termination can be shown similarly. Suppose this algorithm does not terminate. Non-termination implies that $|\mathcal{T}^*(\mathcal{M}_i' - \mathcal{M}_{is})| \leq f$, which further implies that in the node partition $L = \{i\}$, $F = \mathcal{T}^*(\mathcal{M}_{is})$, $R = \mathcal{V}(G) - F - L$, $C = \emptyset$, the l–restricted $(R \cup C, \{i\})$–connectivity is no more than f, i.e., $R \cup C \not\Rightarrow_l L$. In addition, since $|L| = 1$, $L \cup C \not\Rightarrow_l R$. This contradicts the assumption that G satisfies Condition NC.

Therefore, $\mathcal{M}_{is}$ and $\mathcal{M}_{il}$ are well-defined.

□

We will prove that there exists an IABC algorithm – particularly *Algorithm 1* below – that satisfies the *validity* and *convergence* conditions provided that the graph G satisfies Condition NC. This implies that

Condition NC is also sufficient. *Algorithm 1* has the three-step structure described in Section 3.

Algorithm 1

---

1. *Transmit step:* Transmit current state, namely $v_i[t-1]$, to nodes in N_i^{l+}. If node i is an intermediate node of some message, then node i forwards that message as instructed by the message path. When node i expects to receive a message from a path but does not receive the message, the message value is assumed to be equal to some default message.

2. *Receive step:* Receive messages from N_i^{l-}.

3. *Update step:*

   Define

$$v_i[t] \;=\; Z_i(\mathcal{M}_i[t]) \;= a_i v_i[t-1] + \sum_{m \in \mathcal{M}_i^*[t]} a_i \, w_m. \tag{3}$$

   where

$$w_m = \mathsf{value}(m),\; a_i = \frac{1}{|\mathcal{M}_i^*[t]| + 1} \text{ and } \mathcal{M}_i^*[t] = \mathcal{M}_i'[t] - \mathcal{M}_{is}[t] - \mathcal{M}_{il}[t].$$

   The "weight" of each term on the right-hand side of (3) is a_i, where $0 < a_i \leq 1$, and these weights add to 1. For future reference, let us define α, which is used in Theorem 5.3, as:

$$\alpha = \min_{i \in \mathcal{V}-\mathcal{F}} a_i. \tag{4}$$

In *Algorithm 1*, each fault-free node i's state, $v_i[t]$, is updated as a convex combination of all the *messages values* collected by node i at round t. In particular, the coefficient of the message value is a_i if the message is in $\mathcal{M}_i^*[t]$ or the message is sent via self-loop of node i; and the coefficient is zero, otherwise. The update step in *Algorithm 1* is a generalization of the update steps proposed in [27, 31, 33], where the update summation is over all the incoming neighbors of node i instead of over message routes. In [27, 31, 33], only single-hop communication is allowed, i.e.,$l = 1$, and the graph G is a simple, thus the fault-free node i can receive only one message from its incoming neighbor. On the contrary, in our model, multi-hop communication is considered and the fault-free node can receive message from a node via multiple routes. The update step in *Algorithm 1* take the multi-routes into account. Actually, the *Algorithm 1* also works when the communication graph G is a multi-graph.

5.1 Matrix Preliminaries

We use boldface upper case letters to denote matrices, rows of matrices, and their entries. For instance, $\mathbf{A}$ denotes a matrix, $\mathbf{A}_i$ denotes the i-th row of matrix $\mathbf{A}$, and $\mathbf{A}_{ij}$ denotes the element at the intersection of the i-th row and the j-th column of matrix $\mathbf{A}$.

Definition 5.2. *A vector is said to be* stochastic *if all the entries of the vector are* non-negative*, and the entries add up to 1. A matrix is said to be row stochastic if each row of the matrix is a stochastic vector.*

For a row stochastic matrix $\mathbf{A}$, coefficients of ergodicity $\delta(\mathbf{A})$ and $\lambda(\mathbf{A})$ are defined as [32]:

$$\delta(\mathbf{A}) := \max_j \max_{i_1,i_2} |\mathbf{A}_{i_1 j} - \mathbf{A}_{i_2 j}|, \tag{5}$$

$$\lambda(\mathbf{A}) := 1 - \min_{i_1,i_2} \sum_j \min(\mathbf{A}_{i_1 j}, \mathbf{A}_{i_2 j}). \tag{6}$$

It is easy to see that $0 \leq \delta(\mathbf{A}) \leq 1$ and $0 \leq \lambda(\mathbf{A}) \leq 1$, and that the rows are all identical if and and only if $\delta(\mathbf{A}) = 0$. Additionally, $\lambda(\mathbf{A}) = 0$ if and only if $\delta(\mathbf{A}) = 0$.

The next result from [14] establishes a relation between the coefficient of ergodicity $\delta(\cdot)$ of a product of row stochastic matrices, and the coefficients of ergodicity $\lambda(\cdot)$ of the individual matrices defining the product.

Claim 5.2. *For any p square row stochastic matrices $\mathbf{Q}(1), \mathbf{Q}(2), \ldots, \mathbf{Q}(p)$,*

$$\delta(\mathbf{Q}(1)\mathbf{Q}(2)\cdots\mathbf{Q}(p)) \ \leq \ \Pi_{i=1}^{p} \lambda(\mathbf{Q}(i)). \tag{7}$$

Claim 5.2 is proved in [14]. It implies that if, for all i, $\lambda(\mathbf{Q}(i)) \leq 1 - \gamma$ for some $\gamma > 0$, then $\delta(\mathbf{Q}(1)\mathbf{Q}(2)\cdots\mathbf{Q}(p))$ will approach zero as p approaches ∞.

Definition 5.3. *A row stochastic matrix $\mathbf{H}$ is said to be a* scrambling *matrix, if $\lambda(\mathbf{H}) < 1$* [14, 32].

In a scrambling matrix $\mathbf{H}$, since $\lambda(\mathbf{H}) < 1$, for each pair of rows i_1 and i_2, there exists a column j (which may depend on i_1 and i_2) such that $\mathbf{H}_{i_1 j} > 0$ and $\mathbf{H}_{i_2 j} > 0$, and vice-versa [14, 32]. As a special case, if any one column of a row stochastic matrix $\mathbf{H}$ contains only non-zero entries that are lower bounded by some constant $\gamma > 0$, then $\mathbf{H}$ must be scrambling, and $\lambda(\mathbf{H}) \leq 1 - \gamma$.

Definition 5.4. *For matrices $\mathbf{A}$ and $\mathbf{B}$ of identical size, and a scalar γ, $\mathbf{A} \leq \gamma\, \mathbf{B}$ provided that $\mathbf{A}_{ij} \leq \gamma\, \mathbf{B}_{ij}$ for all i, j.*

5.2 Matrix Representation of Algorithm 1

Recall that $\mathcal{F}$ is the set of faulty nodes. Let $|\mathcal{F}| = \phi$. Without loss of generality, suppose that nodes 1 through $(n - \phi)$ are fault-free, and if $\phi > 0$, nodes $(n - \phi + 1)$ through n are faulty.

Denote by $\mathbf{v}[0] \in \mathbb{R}^{n-\phi}$ the column vector consisting of the initial states of all the *fault-free* nodes. Denote by $\mathbf{v}[t]$, where $t \geq 1$, the column vector consisting of the states of all the *fault-free* nodes at the end of the t-th iteration, $t \geq 1$, where the i-th element of vector $\mathbf{v}[t]$ is state $v_i[t]$.

Theorem 5.3. *We can express the iterative update of the state of a fault-free node i $(1 \leq i \leq n - \phi)$ performed in (3) using the matrix form in (8) below, where $\mathbf{M}_i[t]$ satisfies the four conditions listed below. In addition to t, the row vector $\mathbf{M}_i[t]$ may depend on the state vector $\mathbf{v}[t-1]$ as well as the behavior of the faulty nodes in $\mathcal{F}$. For simplicity, the notation $\mathbf{M}_i[t]$ does not explicitly represent this dependence.*

$$v_i[t] \ = \ \mathbf{M}_i[t]\ \mathbf{v}[t-1] \tag{8}$$

1. $\mathbf{M}_i[t]$ *is a* stochastic *row vector of size* $(n-\phi)$. *Thus,* $\mathbf{M}_{ij}[t] \geq 0$, *where* $1 \leq j \leq n-\phi$, *and*

$$\sum_{1 \leq j \leq n-\phi} \mathbf{M}_{ij}[t] \ = \ 1$$

2. $\mathbf{M}_{ii}[t] \geq a_i \geq \alpha$.

3. $\mathbf{M}_{ij}[t]$ *is non-zero only if there exists a message* $m \in \mathcal{M}_i[t]$ *such that* $\mathsf{source}(m) = j$ *and* $\mathsf{destination}(m) = i$.

4. *For any* $t \geq 1$, *there exists a reduced graph* $\widetilde{G^l}_F \in R_{\mathcal{F}}$ *with adjacent matrix* $\mathbf{H}[t]$ *such that* $\beta\, \mathbf{H}[t] \leq \mathbf{M}[t]$, *where* β *is some constant* $0 < \beta \leq 1$ *to be specified later.*

From the code of *Algorithm 1*, we know that $v_i[t] = a_i v_i[t-1] + \sum_{m \in \mathcal{M}_i^*[t]} a_i\, w_m$, where $a_i = \frac{1}{|\mathcal{M}_i^*[t]|+1}$. Theorem 5.3 says that we can rewrite $a_i v_i[t-1] + \sum_{m \in \mathcal{M}_i^*[t]} a_i\, w_m$ as

$$\sum_{j \in \mathcal{V}-\mathcal{F}} \mathbf{M}_{ij}[t] v_j[t-1],$$

where $\mathbf{M}_{ij}[t]$s together satisfy the preceding four conditions. The proof of this theorem is presented in Section 5.2.1 below. The last condition above plays an important role in the proof. By "stacking" (8) for different i, $1 \leq i \leq n-\phi$, we can represent the state update for all the fault-free nodes together using (9) below, where $\mathbf{M}[t]$ is a $(n-\phi) \times (n-\phi)$ row stochastic matrix, with its i-th row being equal to $\mathbf{M}_i[t]$ in (8).

$$\mathbf{v}[t] \ = \ \mathbf{M}[t]\ \mathbf{v}[t-1]. \tag{9}$$

By repeated application of (9), we obtain:

$$\mathbf{v}[t] \ = \ \left(\ \Pi_{\tau=1}^{t} \mathbf{M}[\tau]\ \right)\ \mathbf{v}[0].$$

5.2.1 Correctness of Theorem 5.3

We prove the correctness of Theorem 5.3 by constructing $\mathbf{M}_i[t]$ for $1 \leq i \leq n-\phi$ that satisfies the conditions in Theorem 5.3. Recall that nodes 1 through $n-\phi$ are fault-free, and the remaining ϕ nodes ($\phi \leq f$) are faulty. Consider a fault-free node i performing the *update step* in *Algorithm 1*. Recall that $\mathcal{M}_{is}[t]$ and $\mathcal{M}_{il}[t]$ messages are eliminated from $\mathcal{M}_i[t]$. Let $\mathcal{S}_{ig}[t] \subseteq \mathcal{M}_{is}[t]$ and $\mathcal{L}_{ig}[t] \subseteq \mathcal{M}_{il}[t]$, respectively, be the sets of removed messages that are not covered by faulty nodes. Let $\mathcal{P}_i^*[t]$ be the set of paths corresponding to all the messages in $\mathcal{M}_i^*[t]$. *Untampered message representation* of the evolution of v_i and construction of $\mathbf{M}_i[t]$ differ somewhat depending on whether sets $\mathcal{L}_{ig}[t], \mathcal{S}_{ig}[t]$ and $\mathcal{P}_i^*[t] \cap \mathcal{F}$ are empty or not, where $\mathcal{P}_i^*[t] \cap \mathcal{F} = \emptyset$ means that no message in $\mathcal{M}_i^*[t]$ has been tampered by faulty nodes and $\mathcal{P}_i^*[t] \cap \mathcal{F} \neq \emptyset$ means that there exists a message that is tampered by faulty nodes. It is possible that $\mathcal{T}^*(\mathcal{M}_{is}[t]) = \mathcal{T}^*(\mathcal{M}_{il}[t]) = \mathcal{F}$, which means all messages in $\mathcal{M}_{is}[t]$ and $\mathcal{M}_{il}[t]$ are tampered by faulty nodes, i.e., $\mathcal{S}_{ig}[t] = \emptyset$ and $\mathcal{L}_{ig}[t] = \emptyset$. We divide the possibilities into six cases:

1. Case I: $\mathcal{S}_{ig}[t] \neq \emptyset, \mathcal{L}_{ig}[t] \neq \emptyset$ and $\mathcal{P}_i^*[t] \cap \mathcal{F} \neq \emptyset$.

2. Case II: $\mathcal{S}_{ig}[t] \neq \emptyset, \mathcal{L}_{ig}[t] \neq \emptyset$ and $\mathcal{P}_i^*[t] \cap \mathcal{F} = \emptyset$.
3. Case III: one of $\mathcal{S}_{ig}[t], \mathcal{L}_{ig}[t]$ is empty and $\mathcal{P}_i^*[t] \cap \mathcal{F} \neq \emptyset$.
4. Case IV: one of $\mathcal{S}_{ig}[t], \mathcal{L}_{ig}[t]$ is empty and $\mathcal{P}_i^*[t] \cap \mathcal{F} = \emptyset$.
5. Case V: $\mathcal{S}_{ig}[t] = \emptyset, \mathcal{L}_{ig}[t] = \emptyset$ and $\mathcal{P}_i^*[t] \cap \mathcal{F} \neq \emptyset$.
6. Case VI: $\mathcal{S}_{ig}[t] = \emptyset, \mathcal{L}_{ig}[t] = \emptyset$ and $\mathcal{P}_i^*[t] \cap \mathcal{F} = \emptyset$.

We first describe the construction of $\mathbf{M}_i[t]$ in case I, when $\mathcal{S}_{ig}[t] \neq \emptyset, \mathcal{L}_{ig}[t] \neq \emptyset$ and $\mathcal{P}_i^*[t] \cap \mathcal{F} \neq \emptyset$. Let $\bar{w}_{is}[t]$ and $\bar{w}_{il}[t]$ be defined as shown below. Recall that $w_m = \mathsf{value}(m)$.

$$\bar{w}_{is}[t] = \frac{\sum_{m \in \mathcal{S}_{ig}[t]} w_m}{|\mathcal{S}_{ig}[t]|} \text{ and } \quad \bar{w}_{il}[t] = \frac{\sum_{m \in \mathcal{L}_{ig}[t]} w_m}{|\mathcal{L}_{ig}[t]|}. \tag{10}$$

By the definitions of $\mathcal{S}_{ig}[t]$ and $\mathcal{L}_{ig}[t]$, $\bar{w}_{is} \leq w_{m'} \leq \bar{w}_{il}$, for each message $m' \in \mathcal{M}_i^*[t]$. Thus, for each message m', we can find convex coefficient $\gamma_{m'}$, where $0 \leq \gamma_{m'} \leq 1$, such that

$$\begin{aligned} w_{m'} &= \gamma_{m'} \bar{w}_{is} + (1 - \gamma_{m'}) \bar{w}_{il} \\ &= \frac{\gamma_{m'}}{|\mathcal{S}_{ig}[t]|} \sum_{m \in \mathcal{S}_{ig}[t]} w_m + \frac{1 - \gamma_{m'}}{|\mathcal{L}_{ig}[t]|} \sum_{m \in \mathcal{L}_{ig}[t]} w_m. \end{aligned}$$

Recall that in *Algorithm 1*, $v_i[t] = a_i v_i[t-1] + \sum_{m \in \mathcal{M}_i^*[t]} a_i w_m$, where $a_i = \frac{1}{|\mathcal{M}_i^*[t]|+1}$. In case I, since $\mathcal{P}_i^*[t] \cap \mathcal{F} \neq \emptyset$, there exist messages in $\mathcal{M}_i^*[t]$ that are tampered by faulty nodes. We need to replace these "bad messages" by "good messages" in the evolution of v_i. In particular,

$$v_i[t] = a_i v_i[t-1] + \sum_{m \in \mathcal{M}_i^*[t]} a_i w_m \tag{11}$$

$$= a_i v_i[t-1] + \sum_{m \in \mathcal{M}_i^*[t]:\ \mathcal{V}(\mathsf{path}(m)) \cap \mathcal{F} = \emptyset} a_i w_m + \sum_{m \in \mathcal{M}_i^*[t]:\ \mathcal{V}(\mathsf{path}(m)) \cap \mathcal{F} \neq \emptyset} a_i w_m \tag{12}$$

$$= a_i v_i[t-1] + \sum_{m \in \mathcal{M}_i^*[t]:\ \mathcal{V}(\mathsf{path}(m)) \cap \mathcal{F} = \emptyset} a_i w_m \tag{13}$$

$$+ \sum_{m \in \mathcal{M}_i^*[t]:\ \mathcal{V}(\mathsf{path}(m)) \cap \mathcal{F} \neq \emptyset} a_i \Big(\frac{\gamma_m}{|\mathcal{S}_{ig}[t]|} \sum_{m' \in \mathcal{S}_{ig}[t]} w_{m'} + \frac{1 - \gamma_m}{|\mathcal{L}_{ig}[t]|} \sum_{m' \in \mathcal{L}_{ig}[t]} w_{m'} \Big) \tag{14}$$

$$= a_i v_i[t-1] + \sum_{m \in \mathcal{M}_i^*[t]:\ \mathcal{V}(\mathsf{path}(m)) \cap \mathcal{F} = \emptyset} a_i w_m \tag{15}$$

$$+ \sum_{m' \in \mathcal{S}_{ig}[t]} \Big(\sum_{m \in \mathcal{M}_i^*[t]:\ \mathcal{V}(\mathsf{path}(m)) \cap \mathcal{F} \neq \emptyset} \frac{a_i \gamma_m}{|\mathcal{S}_{ig}[t]|} \Big) w_{m'} \tag{16}$$

$$+ \sum_{m' \in \mathcal{L}_{ig}[t]} \Big(\sum_{m \in \mathcal{M}_i^*[t]:\ \mathcal{V}(\mathsf{path}(m)) \cap \mathcal{F} \neq \emptyset} \frac{a_i (1 - \gamma_m)}{|\mathcal{L}_{ig}[t]|} \Big) w_{m'}. \tag{17}$$

That is, $v_i[t]$ can be represented as a convex combination of values of untampered messages collected at iteration t, where $v_i[t-1] = \mathsf{value}(v_i[t-1],(i,i))$. For future reference, we refer to the above convex combination as *untampered message representation of* $v_i[t]$ in case I and the convex coefficient of each message in the untampered message representation as *message weight*.

Note that if m is an untampered message in $\mathcal{M}_i^*[t]$ or $m \in \mathcal{S}_{ig}[t] \cup \mathcal{L}_{ig}[t]$, then $w_m = v_j[t-1]$ holds, where node j is the source of message m, i.e., $\mathsf{source}(m) = j$. $v_i[t]$ can be further rewritten as follows, where $\mathbb{1}\{x\} = 1$ if x is true, and $\mathbb{1}\{x\} = 0$, otherwise.

$$\begin{aligned} v_i[t] = \sum_{j \in \mathcal{V}-\mathcal{F}} v_j[t-1] \Big(& a_i \mathbb{1}\{j=i\} + \sum_{m \in \mathcal{M}_i^*[t]:\ \mathcal{V}(\mathsf{path}(m)) \cap \mathcal{F} = \emptyset} a_i \mathbb{1}\{\mathsf{source}(m) = j\} \\ & + \sum_{m' \in \mathcal{S}_{ig}[t]} \Big(\sum_{m \in \mathcal{M}_i^*[t]:\ \mathcal{V}(\mathsf{path}(m)) \cap \mathcal{F} \neq \emptyset} \frac{a_i \gamma_m}{|\mathcal{S}_{ig}[t]|} \mathbb{1}\{\mathsf{source}(m') = j\} \Big) \\ & + \sum_{m' \in \mathcal{L}_{ig}[t]} \Big(\sum_{m \in \mathcal{M}_i^*[t]:\ \mathcal{V}(\mathsf{path}(m)) \cap \mathcal{F} \neq \emptyset} \frac{a_i (1-\gamma_m)}{|\mathcal{L}_{ig}[t]|} \mathbb{1}\{\mathsf{source}(m') = j\} \Big) \Big), \end{aligned}$$

Thus, for each node $i, j \in \mathcal{V} - \mathcal{F}$, define the entry $\mathbf{M}_{ij}[t]$ as follows,

$$\begin{aligned} \mathbf{M}_{ij}[t] = a_i \mathbb{1}\{j=i\} + & \sum_{m \in \mathcal{M}_i^*[t]:\ \mathcal{V}(\mathsf{path}(m)) \cap \mathcal{F} = \emptyset} a_i \mathbb{1}\{\mathsf{source}(m) = j\} \\ & + \sum_{m' \in \mathcal{S}_{ig}[t]} \Big(\sum_{m \in \mathcal{M}_i^*[t]:\ \mathcal{V}(\mathsf{path}(m)) \cap \mathcal{F} \neq \emptyset} \frac{a_i \gamma_m}{|\mathcal{S}_{ig}[t]|} \mathbb{1}\{\mathsf{source}(m') = j\} \Big) \\ & + \sum_{m' \in \mathcal{L}_{ig}[t]} \Big(\sum_{m \in \mathcal{M}_i^*[t]:\ \mathcal{V}(\mathsf{path}(m)) \cap \mathcal{F} \neq \emptyset} \frac{a_i (1-\gamma_m)}{|\mathcal{L}_{ig}[t]|} \mathbb{1}\{\mathsf{source}(m') = j\} \Big). \end{aligned}$$

The third condition in Theorem 5.3 trivially follows from the above construction. By above definition, $\mathbf{M}_{ij} \geq a_i$, where $\mathbf{M}_{ij} > a_i$ holds when there exists a nontrivial cycle (not a self-loop) of length at most l that contains node i and no faulty nodes. In addition, $a_i \geq \alpha$ by (4). Thus, $\mathbf{M}_{ii}[t] \geq \alpha$. The second condition holds. Now we show that $\mathbf{M}_i[t]$ is a stochastic vector. It is easy to see that $\mathbf{M}_{ij}[t] \geq 0$. In addition, we have

$$
\begin{aligned}
\sum_{j\in\mathcal{V}-\mathcal{F}} \mathbf{M}_{ij}[t] &= \sum_{j\in\mathcal{V}-\mathcal{F}} \Big(a_i \mathbb{1}\{j=i\} + \sum_{m\in\mathcal{M}_i^*[t]:\ \mathcal{V}(\mathsf{path}(m))\cap\mathcal{F}=\emptyset} a_i \mathbb{1}\{\mathsf{source}(m)=j\} \\
&\quad + \sum_{m'\in\mathcal{S}_{ig}[t]} \Big(\sum_{m\in\mathcal{M}_i^*[t]:\ \mathcal{V}(\mathsf{path}(m))\cap\mathcal{F}\neq\emptyset} \frac{a_i\gamma_m}{|\mathcal{S}_{ig}[t]|} \mathbb{1}\{\mathsf{source}(m')=j\}\Big) \\
&\quad + \sum_{m'\in\mathcal{L}_{ig}[t]} \Big(\sum_{m\in\mathcal{M}_i^*[t]:\ \mathcal{V}(\mathsf{path}(m))\cap\mathcal{F}\neq\emptyset} \frac{a_i(1-\gamma_m)}{|\mathcal{L}_{ig}[t]|} \mathbb{1}\{\mathsf{source}(m')=j\}\Big)\Big) \\
&= a_i \sum_{j\in\mathcal{V}-\mathcal{F}} \mathbb{1}\{i=j\} + \sum_{m\in\mathcal{M}_i^*[t]:\ \mathsf{path}(m)\cap\mathcal{F}=\emptyset} a_i \sum_{j\in\mathcal{V}-\mathcal{F}} \mathbb{1}\{\mathsf{source}(m)=j\} \\
&\quad + \sum_{m\in\mathcal{M}_i^*[t]:\ \mathsf{path}(m)\cap\mathcal{F}\neq\emptyset} \Big(\frac{a_i\gamma_m}{|\mathcal{S}_{ig}[t]|} \sum_{m'\in\mathcal{S}_{ig}[t]} \sum_{j\in\mathcal{V}-\mathcal{F}} \mathbb{1}\{\mathsf{source}(m')=j\}\Big) \\
&\quad + \sum_{m\in\mathcal{M}_i^*[t]:\ \mathsf{path}(m)\cap\mathcal{F}\neq\emptyset} \Big(\frac{a_i(1-\gamma_m)}{|\mathcal{L}_{ig}[t]|} \sum_{m'\in\mathcal{L}_{ig}[t]} \sum_{j\in\mathcal{V}-\mathcal{F}} \mathbb{1}\{\mathsf{source}(m')=j\}\Big) \\
&= a_i + \sum_{m\in\mathcal{M}_i^*[t]:\ \mathsf{path}(m)\cap\mathcal{F}=\emptyset} a_i \\
&\quad + \sum_{m\in\mathcal{M}_i^*[t]:\ \mathsf{path}(m)\cap\mathcal{F}\neq\emptyset} \frac{a_i\gamma_m}{|\mathcal{S}_{ig}[t]|} \sum_{m'\in\mathcal{S}_{ig}[t]} 1 \\
&\quad + \sum_{m\in\mathcal{M}_i^*[t]:\ \mathsf{path}(m)\cap\mathcal{F}\neq\emptyset} \frac{a_i(1-\gamma_m)}{|\mathcal{L}_{ig}[t]|} \sum_{m'\in\mathcal{L}_{ig}[t]} 1 \\
&= a_i + \sum_{m\in\mathcal{M}_i^*[t]:\ \mathsf{path}(m)\cap\mathcal{F}=\emptyset} a_i + \sum_{m\in\mathcal{M}_i^*[t]:\ \mathsf{path}(m)\cap\mathcal{F}\neq\emptyset} \frac{a_i\gamma_m}{|\mathcal{S}_{ig}[t]|}|\mathcal{S}_{ig}[t]| \\
&\quad + \sum_{m\in\mathcal{M}_i^*[t]:\ \mathsf{path}(m)\cap\mathcal{F}\neq\emptyset} \frac{a_i(1-\gamma_m)}{|\mathcal{L}_{ig}[t]|}|\mathcal{L}_{ig}[t]| \\
&= a_i + \sum_{m\in\mathcal{M}_i^*[t]:\ \mathsf{path}(m)\cap\mathcal{F}=\emptyset} a_i + \sum_{m\in\mathcal{M}_i^*[t]:\ \mathsf{path}(m)\cap\mathcal{F}\neq\emptyset} a_i \\
&= a_i(|\mathcal{M}_i^*[t]|+1) \\
&= 1.
\end{aligned}
$$

So $\mathbf{M}_i[t]$ is row stochastic.

In case II, since $\mathcal{P}_i^*[t]\cap\mathcal{F}=\emptyset$, all messages in $\mathcal{M}_i^*[t]$ are untampered by faulty nodes. Let m_0 be an arbitrary message in $\mathcal{M}_i^*[t]$, with $\mathsf{source}(m_0)=j^*$. In order to guarantee condition 4) holds, we rewrite

$v_i[t]$ as follows,

$$\begin{aligned}
v_i[t] &= a_i v_i[t-1] + \sum_{m\in\mathcal{M}_i^*[t]} a_i w_m \\
&= a_i v_i[t-1] + a_i w_{m_0} + \sum_{m\in\mathcal{M}_i^*[t]-\{m_0\}} a_i w_m \\
&= a_i v_i[t-1] + \frac{1}{2}a_i w_{m_0} + \frac{1}{2}a_i w_{m_0} + \sum_{m\in\mathcal{M}_i^*[t]-\{m_0\}} a_i w_m \\
&= a_i v_i[t-1] + \frac{1}{2}a_i w_{m_0} + \frac{1}{2}a_i\Big(\frac{\gamma_{m_0}}{|\mathcal{S}_{ig}[t]|}\sum_{m'\in\mathcal{S}_{ig}[t]} w_{m'} + \frac{1-\gamma_{m_0}}{|\mathcal{L}_{ig}[t]|}\sum_{m'\in\mathcal{L}_{ig}[t]} w_{m'}\Big) \\
&\quad + \sum_{m\in\mathcal{M}_i^*[t]-\{m_0\}} a_i w_m \\
&= a_i v_i[t-1] + \frac{1}{2}a_i w_{m_0} + \sum_{m'\in\mathcal{S}_{ig}[t]} \frac{a_i\gamma_{m_0}}{2|\mathcal{S}_{ig}[t]|} w_{m'} + \sum_{m'\in\mathcal{L}_{ig}[t]} \frac{a_i(1-\gamma_{m_0})}{2|\mathcal{L}_{ig}[t]|} w_{m'} \\
&\quad + \sum_{m\in\mathcal{M}_i^*[t]-\{m_0\}} a_i w_m.
\end{aligned}$$

Note that we did not use the above trick in case I. This is because, in case I, by substituting tampered messages in $\mathcal{M}_i^*[t]$ by untampered messages in $\mathcal{S}_{ig}[t]$ and $\mathcal{L}_{ig}[t]$, as will be seen later, condition 4) is automatically guaranteed.

We refer to the above convex combination as the *untampered message representation of* $v_i[t]$ in case II. And the convex coefficient of each message in the above representation as *weight assigned* to that message. Combining the coefficients of messages according to message sources, it is obtained that

$$\begin{aligned}
v_i[t] = \sum_{j\in\mathcal{V}-\mathcal{F}} v_j[t-1]\Big(&a_i\mathbb{1}\{i=j\} + \frac{1}{2}a_i\mathbb{1}\{j=j^*\} + \sum_{m\in\mathcal{M}_i^*[t]-\{m_0\}} a_i\mathbb{1}\{\mathsf{source}(m)=j\} \\
&+ \frac{a_i\gamma_{m_0}}{2|\mathcal{S}_{ig}[t]|}\sum_{m'\in\mathcal{S}_{ig}[t]}\mathbb{1}\{\mathsf{source}(m')=j\} + \frac{a_i(1-\gamma_{m_0})}{2|\mathcal{L}_{ig}[t]|}\sum_{m'\in\mathcal{L}_{ig}[t]}\mathbb{1}\{\mathsf{source}(m')=j\}\Big).
\end{aligned}$$

Thus, define $\mathbf{M}_{ij}$ by

$$\begin{aligned}
\mathbf{M}_{ij} = &a_i\mathbb{1}\{i=j\} + \frac{1}{2}a_i\mathbb{1}\{j=j^*\} + \sum_{m\in\mathcal{M}_i^*[t]-\{m_0\}} a_i\mathbb{1}\{\mathsf{source}(m)=j\} \\
&+ \frac{a_i\gamma_{m_0}}{2|\mathcal{S}_{ig}[t]|}\sum_{m'\in\mathcal{S}_{ig}[t]}\mathbb{1}\{\mathsf{source}(m')=j\} + \frac{a_i(1-\gamma_{m_0})}{2|\mathcal{L}_{ig}[t]|}\sum_{m'\in\mathcal{L}_{ig}[t]}\mathbb{1}\{\mathsf{source}(m')=j\}.
\end{aligned}$$

Follow the same line as in the proof of case I, it can be shown that the above $\mathbf{M}_{ij}$ satisfies conditions 1), 2) and 3).

In case III, case IV, case V and case VI, at least one of $\mathcal{S}_{ig}[t]$ and $\mathcal{L}_{ig}[t]$ is empty, without loss of generality, assume that $\mathcal{S}_{ig}[t]$ is empty. By the definition of $\mathcal{S}_{ig}[t]$, we know that the set $\mathcal{M}_{is}[t]$ is covered

by $\mathcal{F}$. On the other hand, by the definition of $\mathcal{M}_{is}[t]$, a minimum cover of $\mathcal{M}_{is}[t]$ is of size f. Since $|\mathcal{F}| \leq f$, then we know $\mathcal{F}$ is a minimum cover of $\mathcal{M}_{is}[t]$ and $|\mathcal{F}| = f$. From the definition of $\mathcal{M}_{is}[t]$, we know there exists a message with the smallest value in $\mathcal{M}_i^*[t]$, denoted by m_s is not covered by $\mathcal{F}$. So, we can use singleton $\{m_s\}$ to mimic the role of $\mathcal{S}_{ig}[t]$ in cases I and II. Similarly, we can use the same trick when $\mathcal{L}_{ig}[t]$ is empty. The *untampered message representation of* $v_i[t]$ and *message weight* are defined similarly as that in case I and case II.

To show the above constructions satisfy the last condition in Theorem 5.3, we need the following claim.

Claim 5.4. *For node $i \in \mathcal{V} - \mathcal{F}$, in the untampered message representation of $v_i[t]$, at most one of the sets $\mathcal{S}_{ig}[t]$ and $\mathcal{L}_{ig}[t]$ contains messages with assigned weights less than β, where $\beta = \frac{1}{16n^{2l}}$.*

Proof. An untampered message is either in $\mathcal{M}_i^*[t]$ or in $\mathcal{S}_{ig}[t] \cup \mathcal{L}_{ig}[t]$.

For case V and case VI, both $\mathcal{S}_{ig}[t]$ and $\mathcal{L}_{ig}[t]$ are empty, all untampered messages are contained in $\mathcal{M}_i^*[t]$. For each untampered message in $\mathcal{M}_i^*[t]$, its weight in the untampered message representation is $a_i = \frac{1}{|\mathcal{M}_i^*[t]|+1}$. In $\mathcal{M}_i[t]$, there are at most n messages were transmitted via one hop, at most n^2 messages were transmitted via two hops. In general, $\mathcal{M}_i[t]$ contains at most n^d messages that were transmitted via d hops, where d is an integer in $\{1, \ldots, l\}$. Thus,

$$\begin{aligned}
|\mathcal{M}_i^*[t]| + 1 &\leq |\mathcal{M}_i[t]| \\
&\leq n + n^2 + \ldots + n^l \\
&= \frac{n(n^l - 1)}{n - 1} \\
&\overset{(a)}{\leq} \frac{n(n^l - 1)}{\frac{n}{2}} \\
&\leq 2n^l.
\end{aligned}$$

Inequality (a) is true because $n \geq 2$. Thus, $a_i \geq \frac{1}{2n^l}$. In cases V and VI, as both $\mathcal{S}_{ig}[t]$ and $\mathcal{L}_{ig}[t]$ are empty, all untampered messages are with weight no less than $\frac{1}{2n^l}$.

For case III and case IV, WLOG, assume $\mathcal{S}_{ig}[t]$ is empty. An untampered message is either in $\mathcal{M}_i^*[t]$ or in $\mathcal{L}_{ig}[t]$. Since for each untampered message in $\mathcal{M}_i^*[t]$, the weight assigned to it in the untampered message representation of $v_i[t]$ is at least $\frac{1}{2n^l}$. Thus, only $\mathcal{L}_{ig}[t]$ may contain untampered messages with assigned weights less than $\frac{1}{2n^l}$.

For case II, both $\mathcal{S}_{ig}[t]$ and $\mathcal{L}_{ig}[t]$ are nonempty, an untampered message is in one of $\mathcal{M}_i^*[t]$, $\mathcal{S}_{ig}[t]$ and $\mathcal{L}_{ig}[t]$. In the untampered message representation of $v_i[t]$, either $\gamma_{m_0} \geq \frac{1}{2}$ or $1 - \gamma_{m_0} \geq \frac{1}{2}$. WLOG, assume that $\gamma_{m_0} \geq \frac{1}{2}$, which implies that for each message in $\mathcal{S}_{ig}[t]$, the assigned weight is at least $\frac{a_i}{4|\mathcal{S}_{ig}[t]|} \geq \frac{1}{16n^{2l}}$, since $|\mathcal{S}_{ig}[t]| \leq |\mathcal{M}_i[t]| \leq 2n^l$. Let $\beta = \frac{1}{16n^{2l}}$, then we can conclude that only $\mathcal{L}_{ig}[t]$ may contain untampered messages with assigned weights less than β.

It can be shown similarly that the above claim also holds for case I.

□

Now we are ready to show the following property is also true.

Claim 5.5. *For any $t \geq 1$, there exists a reduced graph $\widetilde{G^l}_{\mathcal{F}} \in R_{\mathcal{F}}$ such that $\beta\,\mathbf{H}[t] \;\leq\; \mathbf{M}[t]$.*

Proof. We construct the desired reduced graph $\widetilde{G^l}_{\mathcal{F}}$ as follows. Let

$$E = \{e \in \mathcal{E}(G^l) :\; \mathcal{V}(P(e)) \cap \mathcal{F} \neq \emptyset\}$$

be the set of edges in G^l that are covered by node set $\mathcal{F}$.

For a fault-free node i: (i) if both $\mathcal{S}_{ig}[t]$ and $\mathcal{L}_{ig}[t]$ are empty, then choose $C_i = \emptyset$; (ii) if one of $\mathcal{S}_{ig}[t]$ and $\mathcal{L}_{ig}[t]$ is empty, WLOG, assume that $\mathcal{S}_{ig}[t]$ is empty, then choose $C_i = \mathcal{T}^*(\mathcal{M}_{il}[t])$; (iii) if both $\mathcal{S}_{ig}[t]$ and $\mathcal{L}_{ig}[t]$ are nonempty, WLOG, assume that the weight assigned to every message in $\mathcal{S}_{ig}[t]$ is lower bounded by β, then choose $C_i = \mathcal{T}^*(\mathcal{M}_{il}[t])$. Let

$$E_i = \{e \in \mathcal{E}(G^l) :\; e \text{ is an incoming edge of node } i \text{ in } G^l \text{ and } \mathcal{V}(P(e)) \cap C_i \neq \emptyset\}$$

be the set of incoming edges of node i in G^l that are covered by node set C_i.

Set $\mathcal{V}(\widetilde{G^l}_{\mathcal{F}}) = \mathcal{V}(G) - \mathcal{F}$. And let $\mathcal{E}(\widetilde{G^l}_{\mathcal{F}}) = \mathcal{E}(\widetilde{G^l}) - E - \cup_{i \in \mathcal{V}-\mathcal{F}} E_i$.

From claim 5.4, for node i, at most one of the sets $\mathcal{S}_{ig}[t]$ and $\mathcal{L}_{ig}[t]$ contains messages with assigned weights less than β. Then it is easy to see that the adjacency matrix of the obtained reduced graph, $\mathbf{H}[t]$, has the property that $\beta\,\mathbf{H}[t] \;\leq\; \mathbf{M}[t]$.

□

5.3 Correctness of Algorithm 1

The proof below uses techniques also applied in prior work (e.g., [16, 6, 29, 15]), with some similarities to the arguments used in [29, 15].

Lemma 5.6. *In the product below of $\mathbf{H}[t]$ matrices for consecutive $\tau(n-\phi)$ iterations, at least one column is non-zero.*

$$\Pi_{t=z}^{z+\tau(n-\phi)-1}\,\mathbf{H}[t]$$

Proof. Since the above product consists of $\tau(n-\phi)$ matrices in R_F, at least one of the τ distinct connectivity matrices in $R_{\mathcal{F}}$, say matrix $\mathbf{H}_*$, will appear in the above product at least $n-\phi$ times.

Now observe that: (i) By Lemma 4.5, $\mathbf{H}_*^{n-\phi}$ contains a non-zero column, say the k-th column is non-zero, and (ii) all the $\mathbf{H}[t]$ matrices in the product contain a non-zero diagonal. These two observations together imply that the k-th column in the above product is non-zero. □

Let us now define a sequence of matrices $\mathbf{Q}(i)$ such that each of these matrices is a product of $\tau(n-\phi)$ of the $\mathbf{M}[t]$ matrices. Specifically,

$$\mathbf{Q}(i) \;=\; \Pi_{t=(i-1)\tau(n-\phi)+1}^{i\tau(n-\phi)}\,\mathbf{M}[t]$$

Observe that

$$\mathbf{v}[k\tau(n-\phi)] \;=\; \left(\Pi_{i=1}^{k}\,\mathbf{Q}(i)\right)\;\mathbf{v}[0] \tag{18}$$

Lemma 5.7. *For $i \geq 1$, $\mathbf{Q}(i)$ is a scrambling row stochastic matrix, and $\lambda(\mathbf{Q}(i))$ is bounded from above by a constant smaller than 1.*

Proof. $\mathbf{Q}(i)$ is a product of row stochastic matrices ($\mathbf{M}[t]$), therefore, $\mathbf{Q}(i)$ is row stochastic.

From Lemma 5.5, for each t,

$$\beta\,\mathbf{H}[t] \;\leq\; \mathbf{M}[t]$$

Therefore,

$$\beta^{\tau(n-\phi)}\;\Pi_{t=(i-1)\tau(n-\phi)+1}^{i\tau(n-\phi)}\,\mathbf{H}[t] \;\leq\; \mathbf{Q}(i)$$

By using $z = (i-1)(n-\phi)+1$ in Lemma 5.6, we conclude that the matrix product on the left side of the above inequality contains a non-zero column. Therefore, $\mathbf{Q}(i)$ contains a non-zero column as well. Therefore, $\mathbf{Q}(i)$ is a scrambling matrix.

Observe that $\tau(n-\phi)$ is finite, therefore, $\beta^{\tau(n-\phi)}$ is non-zero. Since the non-zero terms in $\mathbf{H}[t]$ matrices are all 1, the non-zero entries in $\Pi_{t=(i-1)\tau(n-\phi)+1}^{i\tau(n-\phi)}\mathbf{H}[t]$ must each be ≥ 1. Therefore, there exists a non-zero column in $\mathbf{Q}(i)$ with all the entries in the column being $\geq \beta^{\tau(n-\phi)}$. Therefore $\lambda(\mathbf{Q}(i)) \leq 1-\beta^{\tau(n-\phi)}$. □

Theorem 5.8. Algorithm 1 *satisfies the validity and the convergence conditions.*

Proof. Since $\mathbf{v}[t] = \mathbf{M}[t]\,v[t-1]$, and $\mathbf{M}[t]$ is a row stochastic matrix, it follows that *Algorithm 1* satisfies the validity condition.

By Claim 5.2,

$$\lim_{t\to\infty}\delta(\Pi_{i=1}^{t}\mathbf{M}[t]) \;\leq\; \lim_{t\to\infty}\Pi_{i=1}^{t}\lambda(\mathbf{M}[t]) \tag{19}$$

$$\leq\; \lim_{i\to\infty}\Pi_{i=1}^{\lfloor\frac{t}{\tau(n-\phi)}\rfloor}\lambda(\mathbf{Q}(i)) \tag{20}$$

$$=\; 0 \tag{21}$$

The above argument makes use of the facts that $\lambda(\mathbf{M}[t]) \leq 1$ and $\lambda(\mathbf{Q}(i)) \leq (1-\beta^{\tau(n-\phi)}) < 1$. Thus, the rows of $\Pi_{i=1}^{t}\mathbf{M}[t]$ become identical in the limit. This observation, and the fact that $\mathbf{v}[t] = (\Pi_{i=1}^{t}\mathbf{M}[i])\mathbf{v}[t-1]$ together imply that the state of the fault-free nodes satisfies the convergence condition.

Now, the validity and convergence conditions together imply that there exists a positive scalar c such that

$$\lim_{t\to\infty}\mathbf{v}[t] \;=\; \lim_{t\to\infty}\left(\Pi_{i=1}^{t}\mathbf{M}[i]\right)\;\mathbf{v}[0] \;=\; c\,\mathbf{1}$$

where $\mathbf{1}$ denotes a column with all its entries being 1.

□

6 Extension of Above Results

We show that our proposed conditions encompass the conditions in [30] and [11] as special cases.

6.1 When $l = 1$

When $l = 1$, our necessary and sufficient condition coincides with the one provided in [30], which states that: For any node partition L, C, R, F of G such that $L \neq \emptyset$, $R \neq \emptyset$ and $|F| \leq f$, either there exists a node $i \in L$ such that $|N_i^- \cap (R \cup C)| \geq f+1$ or there exists a node $i \in R$ such that $|N_i^- \cap (L \cup C)| \geq f+1$.

6.2 When $l = n - 1$

If G is undirected, it has been shown in [11], that $|\mathcal{V}(G)| \geq 3f + 1$ and node-connectivity $2f + 1$ are both necessary and sufficient for achieving Byzantine approximate consensus. We will show that when $l = n - 1$, our Condition NC is equivalent to the above conditions.

Theorem 6.1. *When $l = n - 1$, if G undirected, then $|\mathcal{V}(G)| \geq 3f + 1$ and the node-connectivity of G is at least $2f + 1$ if and only if G satisfies Condition NC.*

Proof. First we show "Condition NC implies $|\mathcal{V}(G)| \geq 3f + 1$ and node connectivity at least $2f + 1$". It has already been shown in corollary 4.2 that $|\mathcal{V}(G)| \geq 3f + 1$. It remains to show the node connectivity of G is at least $2f + 1$. We prove this by contradiction. Suppose the node-connectivity is no more than $2f$. Let S be a min cut of G, then $|S| \leq 2f$. Let K_1 and K_2 be two connected components in G_S, the subgraph of G induced by node set $\mathcal{V}(G) - S$.

Construct a node partition of G as follows: Let $L = K_1, R = K_2$ and $C = \mathcal{V} - F - L - R$, where (1) if $|S| \geq f + 1$, let $F \subseteq S$ such that $|F| = f$; (2) otherwise, let $F = S$. For the later case, there is no path between $L \cup C$ and R in G_F, then $\kappa(L \cup C, i) \leq f$ for any $i \in R$ in G_F. Similarly, $\kappa(R \cup C, j) \leq f$ for any $j \in L$. On the other hand, we know that G satisfies Condition NC. Thus, we arrive at a contradiction.

For the former case, i.e., $F \subset S$, since G satisfies Condition NC, WLOG, assume $R \cup C \Rightarrow_{n-1} L$ in G_F, i.e., there exists a node $i \in L$ such that there are at least $f + 1$ disjoint paths from set $R \cup C$ to node i in G_F. Add an additional node y and connect node y to all nodes in $R \cup C$. And denote the resulting graph by G'_F. From Menger's Theorem we know that a min y, i-cut in graph G'_F has size at least $f + 1$. On the other hand, since S is a cut of G, then we know $S - F$ is a y, i–cut in G'_F. In addition, we know $|S - F| = |S| - |F| \leq 2f - f \leq f$. Thus we arrive at a contradiction.

Next we show that "$|\mathcal{V}(G)| \geq 3f+1$ and $2f+1$ node-connectivity also imply Condition NC". Consider an arbitrary node partition L, R, C, F such that $L \neq \emptyset, R \neq \emptyset$ and $|F| \leq f$. Since $|\mathcal{V}| \geq 3f + 1$ and $|F| \leq f$, either $|L \cup C| \geq f + 1$ or $|R \cup C| \geq f + 1$. WLOG, assume that $|R \cup C| \geq f + 1$. Add a node y connecting to all nodes in $R \cup C \cup F$ and denote the newly obtained graph by G''. By Expansion Lemma[4],

[4] **Expansion Lemma**: If G is a k-connected graph, and G' is formed from G by adding a vertex y having at least k neighbors in G, then G' is k-connected.

G'' is $|F| + f + 1$ connected. Thus, fix $i \in L$. There are at least $|F| + f + 1$ internally disjoint y, i–paths. So there are at least $f + 1$ internally disjoint y, i–paths in G''_F. Thus $R \cup C \Rightarrow_{n-1} L$ in G_F. Since this holds for all partitions of the form L, R, C, F where $L \neq \emptyset$, $R \neq \emptyset$ and $|F| \leq f$, then we conclude that Condition NC holds. This completes the proof.

□

7 Discussion and Conclusion

Throughout this paper, we assume that faulty nodes are only able to tamper message values, leaving message paths unchanged. However, even when faulty nodes are able to tamper message paths or even fake and transmit non-existing messages, as long as (i) the number of faked messages is finite (each faulty node $k \in \mathcal{F}$ cannot create too many non-existing messages); and (ii) for each message m tampered/faked by the faulty node k, $\mathsf{path}(m)$ must satisfy $k \in \mathcal{V}(\mathsf{path}(m))$, i.e., the faulty node k cannot conceal itself from the message path, using the same line of arguments as in Section 4 and Section 5, it can be shown that the Condition NC described above is the necessary and sufficient condition for the existence of approximate consensus under the relaxed model.

In this paper, we unify two streams of work by assuming that each node knows the topology of up to l^{th} neighborhood and can send message to nodes up to l hops away, where $1 \leq l \leq n - 1$ and n is the number of nodes. We prove a family of necessary and sufficient conditions for the existence of *iterative* algorithms that achieve *approximate Byzantine consensus* in arbitrary directed graphs. The class of iterative algorithms considered in this paper ensures that, after each iteration of the algorithm, the state of each fault-free node remains in the *convex hull* of the states of the fault-free nodes at the end of the previous iteration. The following *convergence* requirement is imposed: for any $\epsilon > 0$, after a sufficiently large number of iterations, the states of the fault-free nodes are guaranteed to be within ϵ of each other.

References

[1] Noa Agmon and David Peleg. Fault-tolerant gathering algorithms for autonomous mobile robots. In *SIAM J. Comput*, pages 1063–1071, 2004.

[2] A.H. Azadmanesh and H. Bajwa. Global convergence in partially fully connected networks (pfcn) with limited relays. In *Industrial Electronics Society, 2001. IECON '01. The 27th Annual Conference of the IEEE*, volume 3, pages 2022 –2025 vol.3, 2001.

[3] M. H. Azadmanesh and R.M. Kieckhafer. Asynchronous approximate agreement in partially connected networks. *International Journal of Parallel and Distributed Systems and Networks*, 5(1):26–34, 2002.

[4] Michael Ben-Or. Another advantage of free choice (extended abstract): Completely asynchronous agreement protocols. In *Proceedings of the Second Annual ACM Symposium on Principles of Distributed Computing*, PODC '83, pages 27–30, New York, NY, USA, 1983. ACM.

[5] Michael Ben-Or, Danny Dolev, and Ezra N. Hoch. Simple gradecast based algorithms. *CoRR*, abs/1007.1049, 2010.

[6] F. Benezit, V. Blondel, P. Thiran, J. Tsitsiklis, and M. Vetterli. Weighted gossip: Distributed averaging using non-doubly stochastic matrices. In *Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on*, pages 1753–1757, June 2010.

[7] Dimitri P. Bertsekas and John N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Optimization and Neural Computation Series. Athena Scientific, 1997.

[8] Sanjoy Dasgupta, Christos Papadimitriou, and Umesh Vazirani. *Algorithms*. McGraw-Hill Higher Education, 2006.

[9] Danny Dolev, Nancy A. Lynch, Shlomit S. Pinter, Eugene W. Stark, and William E. Weihl. Reaching approximate agreement in the presence of faults. *J. ACM*, 33:499–516, May 1986.

[10] A D Fekete. Asymptotically optimal algorithms for approximate agreement. In *Proceedings of the fifth annual ACM symposium on Principles of distributed computing*, PODC '86, pages 73–87, New York, NY, USA, 1986. ACM.

[11] Michael J. Fischer, Nancy A. Lynch, and Michael Merritt. Easy impossibility proofs for distributed consensus problems. In *Proceedings of the fourth annual ACM symposium on Principles of distributed computing*, PODC '85, pages 59–70, New York, NY, USA, 1985. ACM.

[12] Michael J. Fischer, Nancy A. Lynch, and Michael S. Paterson. Impossibility of distributed consensus with one faulty process. *J. ACM*, 32:374–382, April 1985.

[13] Pedro A. Forero, Alfonso Cano, and Georgios B. Giannakis. Consensus-based distributed support vector machines. *J. Mach. Learn. Res.*, 11:1663–1707, August 2010.

[14] John Hajnal and MS Bartlett. Weak ergodicity in non-homogeneous markov chains. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 54, pages 233–246. Cambridge Univ Press, 1958.

[15] Shreyas Sundaram Heath LeBlanc, Haotian Zhang and Xenofon Koutsoukos. Consensus of multi-agent networks in the presence of adversaries using only local information. *HiCoNs*, 2012.

[16] A. Jadbabaie, Jie Lin, and A.S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *Automatic Control, IEEE Transactions on*, 48(6):988–1001, June 2003.

[17] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, and M. Medard. Resilient network coding in the presence of byzantine adversaries. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 616–624, May 2007.

[18] D. Kempe, A. Dobra, and J. Gehrke. Gossip-based computation of aggregate information. In *Foundations of Computer Science, 2003. Proceedings. 44th Annual IEEE Symposium on*, pages 482–491, Oct 2003.

[19] R. M. Kieckhafer and M. H. Azadmanesh. Low cost approximate agreement in partially connected networks. *Journal of Computing and Information*, 3(1):53–85, 1993.

[20] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, July 1982.

[21] Heath J. LeBlanc, Haotian Zhang, Shreyas Sundaram, and Xenofon Koutsoukos. Consensus of multi-agent networks in the presence of adversaries using only local information. In *Proceedings of the 1st International Conference on High Confidence Networked Systems*, HiCoNS '12, pages 1–10, New York, NY, USA, 2012. ACM.

[22] Nancy A. Lynch. *Distributed Algorithms*. Morgan Kaufmann, 1996.

[23] M.O. Rabin. Randomized byzantine generals. In *Foundations of Computer Science, 1983., 24th Annual Symposium on*, pages 403–409, Nov 1983.

[24] I.D. Schizas, G. Mateos, and G.B. Giannakis. Distributed lms for consensus-based in-network adaptive processing. *Signal Processing, IEEE Transactions on*, 57(6):2365–2382, June 2009.

[25] Lewis Tseng and Nitin Vaidya. Iterative approximate consensus in the presence of byzantine link failures. In Guevara Noubir and Michel Raynal, editors, *Networked Systems*, Lecture Notes in Computer Science, pages 84–98. Springer International Publishing, 2014.

[26] J.N. Tsitsiklis, D.P. Bertsekas, and M. Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *Automatic Control, IEEE Transactions on*, 31(9):803–812, Sep 1986.

[27] Nitin H. Vaidya. Matrix representation of iterative approximate byzantine consensus in directed graphs. *CoRR*, pages –1–1, 2012.

[28] Nitin H Vaidya. Iterative byzantine vector consensus in incomplete graphs. In *Distributed Computing and Networking*, pages 14–28. Springer, 2014.

[29] Nitin H. Vaidya, Christoforos N. Hadjicostis, and Alejandro D. Domnguez-Garca. Distributed algorithms for consensus and coordination in the presence of packet-dropping communication links - part ii: Coefficients of ergodicity analysis approach. *CoRR*, pages –1–1, 2011.

[30] Nitin H. Vaidya, Lewis Tseng, and Guanfeng Liang. Iterative approximate byzantine consensus in arbitrary directed graphs. In *Proceedings of the 2012 ACM Symposium on Principles of Distributed Computing*, PODC '12, pages 365–374, New York, NY, USA, 2012. ACM.

[31] Nitin H. Vaidya, Lewis Tseng, and Guanfeng Liang. Iterative approximate byzantine consensus in arbitrary directed graphs: Synchronous and asynchronous systems. Technical report, University of Illinois at Urbana-Champaign, february 2012.

[32] J. Wolfowitz. Products of indecomposable, aperiodic, stochastic matrices. *Proceedings of the American Mathematical Society*, 14(5):pp. 733–737, 1963.

[33] Haotian Zhang and Shreyas Sundaram. Robustness of information diffusion algorithms to locally bounded adversaries. *CoRR*, abs/1110.3843, 2011.