Exact Byzantine Consensus on Arbitrary Directed Graphs Under Local Broadcast Model

Muhammad Samir Khan
Department of Computer Science, University of Illinois at Urbana-Champaign, USA
mskhan6@illinois.edu

Lewis Tseng
Department of Computer Science, Boston College, USA
lewis.tseng@bc.edu

Nitin H. Vaidya
Department of Computer Science, Georgetown University, USA
nitin.vaidya@georgetown.edu

Abstract

We consider Byzantine consensus in a synchronous system where nodes are connected by a network modeled as a *directed graph*, i.e., communication links between neighboring nodes are not necessarily bi-directional. The directed graph model is motivated by wireless networks wherein asymmetric communication links can occur. In the classical point-to-point communication model, a message sent on a communication link is private between the two nodes on the link. This allows a Byzantine faulty node to *equivocate*, i.e., send inconsistent information to its neighbors. This paper considers the *local broadcast* model of communication, wherein transmission by a node is received identically by *all of its outgoing neighbors*, effectively depriving the faulty nodes of the ability to equivocate.

Prior work has obtained sufficient and necessary conditions on *undirected graphs* to be able to achieve Byzantine consensus under the *local broadcast* model. In this paper, we obtain tight conditions on *directed graphs* to be able to achieve Byzantine consensus with binary inputs under the *local broadcast* model. The results obtained in the paper provide insights into the trade-off between directionality of communication links and the ability to achieve consensus.

2012 ACM Subject Classification Theory of computation → Distributed algorithms

Keywords and phrases complexity and impossibility results for distributed computing, fault-tolerance, reliability

Digital Object Identifier 10.4230/LIPIcs.OPODIS.2019.30

Related Version A full version of the paper [14] is available on `arXiv:1911.07298`.

Funding This research is supported in part by the National Science Foundation awards 1409416 and 1733872, and Toyota InfoTechnology Center. 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.

1 Introduction

Byzantine consensus [22] is a classical problem in distributed computing. We consider a synchronous system consisting of n nodes, each with a *binary* input. The objective is for the nodes to reach consensus in the presence of up to f Byzantine faulty nodes. The nodes are connected by a communication network represented by a graph. In the classical point-to-point communication model, a message sent by a node to one of its neighboring nodes is received only by that node. This allows a Byzantine faulty node to send inconsistent messages to its neighbors without the inconsistency being observed by the neighbors. For instance, a faulty node u may report to neighbor v that its input is 0, whereas report to neighbor w that its input is 1. Node v will not hear the message sent by node u to node w. This ability of a faulty

node to send conflicting information on different communication links is called *equivocation* [4]. The problem of Byzantine consensus with equivocation in point-to-point networks is well-studied [2, 7, 17, 20, 22]. In undirected graphs, it is known that $n > 3f$ and connectivity $\geq 2f + 1$ are necessary and sufficient conditions to achieve Byzantine consensus [7].

This paper considers the *local broadcast* model of communication [3, 15]. In this model, a message sent by a node is received identically by all neighbors of that node. This allows the neighbors of a faulty node to detect its attempts to equivocate, effectively depriving the faulty node of the ability to send conflicting information to its neighbors. In the example above, in the local broadcast model, if node u attempts to send different input values to different neighbors, the neighbors will receive all the messages, and can detect the inconsistency. Recent work has shown that this ability to detect equivocation reduces network requirements for Byzantine consensus in *undirected* graphs. In particular, for the local broadcast model, recent work [13, 21] has identified that the following two conditions are both necessary and sufficient for Byzantine consensus in *undirected* graphs: network connectivity $\geq \lfloor 3f/2 \rfloor + 1$ and minimum node degree $\geq 2f$.

In this paper, we study Byzantine consensus in *directed* graphs under the local broadcast model. The directed graph model is motivated by wireless networks wherein asymmetric links may occur. Thus, the communication links between neighboring nodes are not necessarily bi-directional. Under local broadcast, when some node u transmits a message, it is received identically by all of u's *outgoing* neighbors (i.e., by nodes to whom there are outgoing links from node u). The results obtained in the paper provide insights into the trade-off between directionality of communication links and the ability to achieve consensus.

This paper makes two main contributions:

1. **Necessity:** In Section 5, we identify a necessary condition for directed graphs to solve Byzantine consensus. The proof is via a state machine based approach [2, 7, 8]. However, the communication by faulty nodes must follow the local broadcast model which restricts their behavior. The directed nature of the graph also adds to the difficulty. We handle this complexity via non-trivial arguments to show the desired result.
2. **Sufficiency:** In Section 6, we constructively show that the necessary condition is also sufficient by presenting a Byzantine consensus algorithm along with its proof of correctness. The key challenge for directed graphs is that communication may only exist in one direction between some pairs of nodes. Hence, it is not straightforward to adapt the prior algorithm [13] on undirected graphs. More specifically, in undirected graphs, each node has some path(s) to each of the other nodes.[1] Prior algorithm by Khan et. al. [13] utilizes this property to solve Byzantine consensus in undirected graphs under local broadcast. In the directed case, this property is not guaranteed due to the directionality of communication. We prove some non-trivial properties (Section 6.2) implied by the tight condition identified in this paper. This allows us to achieve consensus in a unique "source component" and then propagate that decision to the rest of the graph.

The rest of the paper is organized as follows. We discuss related work in Section 2. Section 3 formalizes the setting and introduces notation. Our main result is presented in Section 4. Necessity of the identified tight condition is shown in Section 5 while sufficiency is shown in Section 6. We summarize in Section 7.

[1] In undirected graphs, consensus is not possible if graph is not connected.

2 Related Work

Byzantine consensus is a well-studied problem [2, 7, 17, 20, 22] with tight conditions known for complete graphs [22], undirected graphs [7], and directed graphs [26] under the point-to-point communication model. For brevity, we focus here on related work that restricts equivocation by faulty nodes.

Rabin and Ben-Or [23] considered complete graphs with global broadcast under synchronous communication, while Clement et. al. [5] looked at non-equivocation in complete graphs under asynchronous communication. Amitanand et. al. [1] restricted equivocation by faulty nodes by partitioning, for each faulty node w, the remaining graph such that a message sent by w to any node is received identically by all nodes in the corresponding partition. However, the underlying graph in [1] is complete while we consider arbitrary directed graphs. Several works [9, 12, 24] have used *undirected* hypergraphs to model partial broadcast for the Byzantine consensus problem. In this model, a message sent on an hyperedge is received identically by all nodes in the hyperedge. The closest work to this paper is by Khan et. al. [13, 21], who obtained that minimum node degree $\geq 2f$ and network connectivity $\geq \lfloor 3f/2 \rfloor + 1$ are tight conditions for Byzantine consensus in *undirected* graphs under the local broadcast model. Here, we consider arbitrary *directed* graphs under the local broadcast model.

Restricted equivocation has also been used to study related problems. [6, 11, 10, 29] looked at reliability and privacy on partial broadcast networks. [3, 15, 16] have investigated the Byzantine *broadcast* problem under local broadcast on both undirected and directed graphs. In Byzantine broadcast, the goal is for a single source to transmit a binary value reliably throughout the network. We consider the Byzantine *consensus* problem, where the goal is for all nodes to agree on a common value.

Another line of work investigates iterative algorithms for *approximate* Byzantine consensus. In this problem, each node starts with a real value (or a vector of real values) and maintains a state variable. The updates are "memory less", i.e., the update rules allow a state update in each round to depend only on the current state and the state values received from neighbors. This problem has been investigated under the classical point-to-point communication model on directed graphs by Tseng and Vaidya [25] and Vaidya et. al. [27, 28], under partial broadcast modeled via directed hypergraphs by Li et. al. [19], and under the local broadcast model on directed graphs by LeBlanc et. al. [18] as well as by Zhang and Sundaram [30]. The network conditions are different than the ones presented in this paper, since the algorithm structure is restricted (as summarized above) in these prior works.

3 System Model and Notation

We consider a synchronous system consisting of n nodes. The communication network connecting the nodes is represented by a *directed* graph $G = (V, E)$, where $|V| = n$. Each of the n nodes is represented by a vertex $u \in V$. We interchangeably use the terms *node* and *vertex*. Every node in the graph knows the communication graph G. Each directed edge $(u, v) \in E$ represents a FIFO link from u to v. When a message m sent by node u on edge (u, v) is received by node v, node v knows that the message m was sent by node u. This assumption is implicit in the previous related work as well. We assume the *local broadcast* model of communication wherein a message sent by any node u is received identically and correctly by each node v such that $(u, v) \in E$.

A *Byzantine* faulty node may exhibit arbitrary behavior; however, its communication is still governed by the local broadcast model. We consider the *Byzantine consensus problem*.

Each node starts with a binary input and must output a binary value. There are at most $f > 0$ Byzantine faulty nodes in the system. The output at each node must satisfy the following conditions.

1. **Agreement:** All non-faulty nodes must output the same value.
2. **Validity:** The output of each non-faulty node must be an input of some non-faulty node.
3. **Termination:** All non-faulty nodes must decide on their output in finite time.

Neighborhood: If $(u, v) \in E$, then u is an *in-neighbor* of v and v is an *out-neighbor* of u. The *in-neighborhood* of a node v is the set of all in-neighbors of v, i.e., $\{u \mid (u, v) \in E\}$. Similarly, the *out-neighborhood* of a node v is the set of all out-neighbors of v, i.e., $\{u \mid (v, u) \in E\}$. In graph G, for node sets A and B, we define in-neighborhood of set B in set A, denoted $\Gamma_G(A, B)$, as the set of in-neighbors of nodes in B that are in set A. That is, $\Gamma_G(A, B) = \{u \in A \mid \exists v \in B \text{ s.t. } (u, v) \in E(G)\}$. Note that $E(G)$ denotes the set of edges in graph G. We will use the above definition for different graphs, hence the subscript G above is important. We may drop the subscript G when it is clear from the context.

We will say that $A \to_G B$ if $|\Gamma_G(A, B)| > f$. Here as well, we may drop the subscript G when it is clear from the context.

Paths in graph G: A path is a sequence of distinct nodes such that if u precedes v in the sequence, then u is an in-neighbor of v in G (i.e., (u, v) is an edge).

- For two nodes u and v, a uv-path P_{uv} is a path from u to v. u is called the *source* and v the *terminal* of P_{uv}. Any other node in the path is called an *internal* node of P_{uv}. Two uv-paths are *node-disjoint* if they do not share a common internal node.
- For a set $U \subsetneq V$ and a node $v \notin U$, a Uv-path is a uv-path for some node $u \in U$. All Uv-paths have v as the terminal. Two Uv-paths are *node-disjoint* if they do not have any nodes in common except terminal node v. In particular, two node-disjoint Uv-paths have different source nodes.

A path is said to *exclude* a set of nodes $X \subset V$ if no internal node of the path belongs to X; however, its source and terminal nodes may potentially belong to X. A path is said to be *fault-free* if none of its internal nodes are faulty. In other words, a path is fault-free if it excludes the set of faulty nodes. Note that a fault-free path may have a faulty node as either source or terminal.

We use the notation $A \overset{X}{\rightsquigarrow}_G B$ if, for every node $u \in B$, there exist at least $f + 1$ node-disjoint Au-paths in G that exclude X, i.e., there exist $f + 1$ node-disjoint Au-paths that have only u in common and none of them contain any internal node from the set X. We may omit the subscript G when it is clear from the context.

With a slight abuse of terminology, we allow a *partition* of a set to have empty parts. That is, $(Z_1, \ldots, Z_k)$ is a partition of a set Y if $\cup_{i=1}^{k} Z_i = Y$ and $Z_i \cap Z_j = \emptyset$ for all $i \neq j$, but some Z_i's can be possibly empty.

For a set of nodes $U \subsetneq V$,

- $G[U]$ is the subgraph induced by the nodes in U.
- G_{-U} is the graph obtained from G by removing all edges (v, u) such that $u \in U$, i.e., by removing all incoming edges to U. Observe that if P is a path in G_{-U}, then P is a path in G that excludes U and terminates in $V - U$. Conversely, if P is a path in G that excludes U and terminates in $V - U$, then P is a path in G_{-U}.

A directed graph G is *strongly connected* if for each pair of nodes u, v, there is both a uv-path and a vu-path in G. A *directed graph decomposition* of G is a partition of G into

non-empty parts $H_1, \ldots, H_k$, where $k > 0$, and each H_i is a maximal strongly connected subgraph of G – each H_i is assumed to be maximal in the sense that adding any nodes to H_i will destroy its strong connectivity. Let $\mathcal{H}$ be the graph obtained from the decomposition by contracting each H_i into a node c_i, so that there is an edge (c_i, c_j) in $\mathcal{H}$ if there is an edge from a node in H_i to a node in H_j in G. Then, graph $\mathcal{H}$ is acyclic. If a node c_i has no in-neighbors, then H_i is called a *source component* of the decomposition. Note that, since $\mathcal{H}$ is acyclic, there is always at least one source component of a directed graph decomposition.

4 Main Results

The main result of this paper is a tight network condition for consensus in directed graphs under the local broadcast model. The following definition presents the condition and the accompanying theorem states the result.

▶ **Definition 1.** *A directed graph G satisfies* condition SC with parameter F *if for every partition (A, B) of V, where both $A - F$ and $B - F$ are non-empty, we have that either $A \overset{F}{\rightsquigarrow} B - F$ or $B \overset{F}{\rightsquigarrow} A - F$. We say that G satisfies* condition SC*, if G satisfies condition SC with parameter F for every set $F \subseteq V$ of cardinality at most f.*

▶ **Theorem 2.** *Under the local broadcast model, Byzantine consensus tolerating at most f Byzantine faulty nodes is achievable on a directed graph G if and only if G satisfies condition SC.*

Proof. The proof follows from Theorems 4, 5, and 11 presented later. ◀

Intuitively, the above condition requires that at least one of the two partitions A and B should have the ability to "propagate" its state to the other partition reliably. For the point-to-point communication model, Tseng and Vaidya [26] obtained an analogous network condition, which is that, for every partition (A, B) of V and a faulty set F, where both $A - F$ and $B - F$ are non-empty, either $A - F \overset{F}{\rightsquigarrow} B - F$ or $B - F \overset{F}{\rightsquigarrow} A - F$. In the point-to-point communication model a faulty node can equivocate. Thus, the condition in [26] does not allow nodes in set F to be source nodes,[2] and requires $A - F$ or $B - F$ to propagate its state to the non-faulty nodes in the other partition. On the other hand, as discussed earlier, local broadcast effectively removes a faulty node's ability to equivocate. Therefore, the condition in Definition 1 allows a node in set F to be a source node in the propagation paths, but does not allow nodes in F to be internal nodes on such paths.

Even though the conditions for point-to-point communication [26] and local broadcast seem similar, the algorithm in [26] is not immediately adaptable to the local broadcast model. One key challenge is that while non-equivocation provided by local broadcast prevents a faulty node from sending conflicting messages in one round, it does not directly stop a faulty node from lying inconsistently across rounds, even to the same neighbor. We discuss this at the end of Section 6.1.

We prove necessity of condition SC via a state machine based approach [2, 7, 8] similar to the proofs of necessity in [13, 26]. However, care must be taken to ensure that we do not break the local broadcast property. The formal proof is given in the full version of the paper [14] – since it is somewhat difficult. In Section 5, we provide an intuitive sketch of the proof. The sufficiency is proved constructively. In Section 6, we present an algorithm to achieve consensus when the communication graph satisfies condition SC, accompanied by a proof of correctness.

[2] Recall that, in a uv-path, u is the source node and v is the terminal.

5 Necessity

In this section, we show that condition SC (Definition 1) is necessary for consensus. We first present another property, condition NC. This condition is equivalent to condition SC, as stated in the next theorem, and we will use it to prove necessity in Theorem 5. Recall that we use $A \rightarrow_G B$ to denote $|\Gamma_G(A, B)| > f$ (subscript G is dropped when clear from context).

▶ **Definition 3.** *A directed graph G satisfies* condition NC with parameter F *if for every partition (L, C, R) of V, where both $L - F$ and $R - F$ are non-empty, we have that either $R \cup C \rightarrow_G L - F$ or $L \cup C \rightarrow_G R - F$. We say that G satisfies* condition NC*, if G satisfies condition NC with parameter F for every set $F \subseteq V$ of cardinality at most f.*

▶ **Theorem 4.** *A directed graph G satisfies condition NC if and only if G satisfies condition SC.*

A formal proof of Theorem 4 appears in the full paper [14]. The following theorem states that condition NC is necessary for consensus. Since condition NC and condition SC are equivalent, as a corollary we get the necessity part of Theorem 2.

▶ **Theorem 5.** *If there exists a Byzantine consensus algorithm under the local broadcast model on a directed graph G tolerating at most f Byzantine faulty nodes, then G satisfies condition NC.*

As mentioned earlier, the formal proof of this theorem is given in the full paper [14]. Here we give a sketch of the proof. Suppose for the sake of contradiction that there exists an algorithm that solves Byzantine consensus under the local broadcast model on a graph G which does not satisfy condition NC. Then there exists a set F of cardinality at most f and a partition (L, C, R) of G, where both $L - F$ and $R - F$ are non-empty, such that $R \cup C \not\rightarrow L - F$ and $L \cup C \not\rightarrow R - F$. We create three executions E_1, E_2, and E_3 using the algorithm as follows.

E_1: $\Gamma(R \cup C, L - F)$, the in-neighborhood of $L - F$ in $R \cup C$, is the faulty set. We partition the faulty set into two parts: the in-neighborhood of $L - F$ in $R - F$ and C, $\Gamma((R-F) \cup C, L-F)$, and the in-neighborhood of $L-F$ in $R \cap F$, $\Gamma(R \cap F, L-F)$. Both these sets have different behavior. In each round, a faulty node in $\Gamma((R-F) \cup C, L-F)$ broadcasts the same messages as the corresponding non-faulty node in E_3, while a faulty node in $\Gamma(R \cap F, L - F)$ broadcasts the same messages as the corresponding non-faulty node in E_2. All non-faulty nodes have input 0. So by validity, all non-faulty nodes decide on output 0 in finite time.

E_2: $\Gamma(L \cup C, R - F)$, the in-neighborhood of $R - F$ in $L \cup C$, is the faulty set. We partition the faulty set into two parts: the in-neighborhood of $R - F$ in $L - F$ and C, $\Gamma((L-F) \cup C, R-F)$, and the in-neighborhood of $R-F$ in $L \cap F$, $\Gamma(L \cap F, R-F)$. Both these sets have different behavior. In each round, a faulty node in $\Gamma((L-F) \cup C, R-F)$ broadcasts the same messages as the corresponding non-faulty node in E_3, while a faulty node in $\Gamma(L \cap F, R - F)$ broadcasts the same messages as the corresponding non-faulty node in E_1. All non-faulty nodes have input 1. So by validity, all non-faulty nodes decide on output 1 in finite time.

E_3: $F \cap (L \cup R)$ is the faulty set. We partition the faulty set into two parts: $F \cap L$ and $F \cap R$. Both these sets have different behavior. In each round, a faulty node in $F \cap L$ broadcasts the same messages as the corresponding non-faulty node in E_1, while a faulty node in $F \cap R$ broadcasts the same messages as the corresponding non-faulty node in

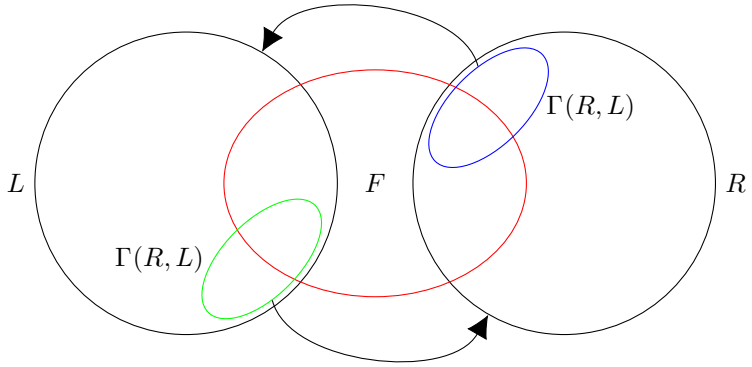

Figure 1 Necessity proof: the simple case when $C = \emptyset$. The nodes in blue, green, and red depict the three faulty sets for executions E_1, E_2, and E_3 respectively.

E_2. All non-faulty nodes in L have input 0 and all non-faulty nodes in $R \cup C$ have input 1. The output of the non-faulty nodes in this execution will be described later.

In the full paper [14] we make the above description of the three executions precise. For the simple case when $C = \emptyset$, Figure 1 depicts the faulty nodes in the three executions.

To see the output of non-faulty nodes in E_3, note that non-faulty nodes in $L - F$ receive the same messages in each round, from their in-neighbors, as the corresponding nodes in E_1. They also have the same input, 0, so they decide on the same output in both the executions, i.e., 0. Similarly, the non-faulty nodes in $R - F$ receive the same messages in each round, from their in-neighbors, as the corresponding nodes in E_2. They also have the same input, 1, so they decide on the same output in both the executions, i.e., 1. Since both $L - F$ and $R - F$ are non-empty, this violates agreement, a contradiction.

6 Sufficiency

In this section, we constructively prove the sufficiency portion of Theorem 2. Together with the necessity result shown in Theorem 5, we have that this result is tight. We present a Byzantine consensus algorithm in Section 6.1. The algorithm utilizes some non-trivial graph properties implied by condition SC. We show these in Section 6.2. In Section 6.3, we give a proof of correctness of the algorithm, assuming that the graph G satisfies condition SC.

6.1 Algorithm

Algorithm 1 presents pseudocode for the proposed algorithm. Each node $v \in V$ maintains a local state variable named γ_v. At the beginning of the algorithm, this is initialized to equal node v's binary input. γ_v is modified during the execution of the algorithm. The output of each node v is the value of its state variable γ_v at the end of the algorithm. The algorithm execution is viewed as being divided into *phases*, each phase consisting of one iteration of the `for` loop in the pseudocode. Each phase has an associated distinct subset $F \subseteq V$ of cardinality at most f.

The algorithm draws inspiration from the strategy used in the algorithms in [13] and [26]. However, the details are significantly different, which we discuss at the end of this section. In particular, each phase (i.e., each iteration of the `for` loop) in the algorithm considers a candidate faulty set F, and the nodes attempt to reach consensus by the end of that phase

assuming that F is indeed the set of all the faulty nodes. Let F^* denote the set of nodes that are actually faulty in a given execution of the algorithm. Then each node updates its state variable in such a manner so that

- **(i)** when $F = F^*$, the state variable at all non-faulty nodes is identical at the end of this phase, i.e., all non-faulty nodes reach consensus in this phase (Lemma 12), and
- **(ii)** when $F \neq F^*$, the value of the state variable at a non-faulty node at the end of the phase equals the state of some non-faulty node at the start of the phase (Lemma 13).

The first objective ensures that the nodes reach agreement. The second inductively implies that this agreement, once achieved, is not lost, as well as that the nodes decide on an input of some non-faulty node, i.e., validity. Termination follows from the fact that there are only a finite number of executions.

We now discuss the steps performed by each node v in a given phase. Some of the steps of the algorithm are based on those in [13] and are explained here again for completeness.

Graph decomposition: In step (a) of a given phase, each node performs a directed graph decomposition on $G - F$. Recall that we assume that each node knows the topology of graph G, so each node can perform this step locally. Since G satisfies condition SC with parameter F, it turns out that $G - F$ has a unique source component S (Lemma 6). The rest of the steps in the phase are aimed at nodes in S attempting to agree on some common value, and then propagating that value to the rest of the graph.

Flooding: In Step (b), nodes in S and their in-neighbors in F, $\Gamma(F, S)$, flood the value of their γ state variables. The "flooding" procedure used here is analogous to that in [13] for undirected graphs. Without much modification, it can be adapted for directed graphs. This procedure is presented in the full version of the paper [14].

Consider a node $u \in S \cup \Gamma(F, S)$. In the flooding procedure, u attempts to transmit its state variable to every node v such that there exists a uv-path in G. At the end of the flooding procedure, for each such node v and a uv-path P_{uv}, v will have received a binary value b along P_{uv}. If P_{uv} is fault-free, then $b = \gamma_u$, the state variable of u. However, if P_{uv} is not fault-free, then an intermediary faulty node may tamper with the messages, so it is possible that $b \neq \gamma_u$.

Consensus in the source component S: Next, using steps (c) and (d), the nodes in the unique source component try to reach consensus. Based on the values flooded by each node, set $S \cup \Gamma(F, S)$ can be partitioned into nodes that flooded 0, namely set Z, and nodes that flooded 1, namely set N. In step (c), node v attempts to estimate the sets Z and N using the values received on paths excluding F, i.e., ignoring paths that have nodes from F as intermediaries. When $F \neq F^*$, faulty nodes may tamper the messages and v may incorrectly categorize some nodes. However, when $F = F^*$, all non-faulty nodes correctly determine Z and N.

In particular, in step (c), each node $v \in S$ partitions $S \cup \Gamma(F, S)$ as follows. Recall that a path is said to exclude set F if none of its *internal* nodes are in F. For each $u \in S \cup \Gamma(F, S)$, node v chooses an arbitrary uv-path P_{uv} that excludes F. It can be shown (Lemma 8) that such a path always exists. For the purpose of step (c), node v is deemed to have received its own γ_v value along path P_{vv} (containing only node v). In step (c), as shown in the pseudo-code, node v partitions $S \cup \Gamma(F, S)$ into sets Z_v and N_v, its estimates of sets Z and N, based on values received along the above paths.

Step (d) specifies the rules for updating γ_v value. γ_v is not necessarily updated in each phase. If $F = F^*$, then all nodes in S will have the same γ_v value after this step (Lemma 12). That is, in the phase in which $F = F^*$, the nodes in S achieve consensus in step (d).

Algorithm 1 Proposed algorithm for Byzantine consensus under the local broadcast model in directed graphs: Steps performed by node v are shown here.

Each node v has a binary input value in $\{0,1\}$ and maintains a binary state $\gamma_v \in \{0,1\}$.
Initialization: $\gamma_v :=$ input value of node v.
For *each* $F \subseteq V$ *such that* $|F| \leq f$ **do**

Step (a): Perform directed graph decomposition on $G - F$. Let S be the unique source component (Lemma 6).

Step (b): If $v \in S \cup \Gamma(F,S)$, then flood value γ_v (the steps taken to achieve flooding are described in the full paper [14]).

Step (c): If $v \in S$, for each node $u \in S \cup \Gamma(F,S)$, identify a single uv-path P_{uv} that excludes F. Let,

$$Z_v := \left\{ u \in S \cup \Gamma(F,S) \mid v \text{ received value 0 from } u \text{ along } P_{uv} \text{ in step (b)} \right\},$$
$$N_v := S \cup \Gamma(F,S) - Z_v.$$

Step (d): If both $Z_v - F$ and $N_v - F$ are non-empty, then

If $Z_v \overset{F}{\rightsquigarrow} N_v - F$,
then set $A_v := Z_v$ and $B_v := N_v - F$,
else set $A_v := N_v$ and $B_v := Z_v - F$.

If $v \in B_v$ and v received value $\delta \in \{0,1\}$, in step (b), identically along any $f+1$ node-disjoint $A_v v$-paths that exclude F, then set $\gamma_v := \delta$.

Step (e): If $v \in S$, then flood value γ_v.

Step (f): If $v \in V - S - F$ and v received value $\delta \in \{0,1\}$, in step (e), identically along any $f+1$ node-disjoint Sv-paths that exclude F, then set $\gamma_v := \delta$.

end
Output γ_v.

Propagating decision to rest of the graph: Since we iterate over all possible faulty sets, in one phase of the algorithm, F is correctly chosen to be exactly F^*, the set of actual faulty nodes. In this phase, nodes in S will reach consensus by step (d). In steps (e) and (f), nodes in S propagate their state to the rest of the graph. In particular, in step (e), nodes in S flood their γ_v value, as in step (b), while step (f) specifies the rules for nodes in $V - S - F$ to update their γ_v state variable.

Output: After all the phases (i.e., all iterations of the `for` loop) are completed, the value of γ_v is chosen as the output of node v.

As mentioned earlier, the proposed algorithm uses the same strategy as in [13] and [26] by iterating over all possible faulty sets. However, the steps performed in each iteration are significantly different than both.

- The algorithm for directed graphs under point-to-point communication in [26] is not immediately adaptable to the local broadcast setting. One key challenge is that it requires nodes to send messages in multiple rounds in a single iteration of the main `for` loop. While non-equivocation provided by local broadcast prevents a faulty node from sending conflicting messages in one round, it does not directly stop a faulty node from lying inconsistently across rounds, even to the same neighbor. In our algorithm, when $F = F^*$, the faulty nodes are only allowed to flood their states once in the entire iteration, preventing them from lying inconsistently across rounds in this iteration.

- On the other hand, the algorithm for undirected graphs under local broadcast [13] does indeed require each faulty node to send a single message for each iteration of the main for loop. However, in directed graphs, in contrast with the undirected setting, communication may only exist in one direction between some pairs of nodes. Our algorithm first achieves consensus in a unique source component and then propagates this to the rest of the graph. The existence of a *unique* source component or its ability to propagate its state to the rest of the graph is not immediately obvious. In the next section, we show that these and other non-trivial properties are guaranteed by condition SC, which our algorithm utilizes to solve consensus.

6.2 Graph Properties

Algorithm 1 relies on some non-trivial properties of graphs that satisfy condition SC. In this section, we prove these properties followed by the proof of correctness of the algorithm in Section 6.3. We first show that there is indeed a unique source component in the directed graph decomposition of $G - F$, performed in step (a) of each phase. Suppose, for the sake of contradiction, that there are two source components S_1 and S_2 of the decomposition. Note that, by construction, there are no edges into S_1 and S_2, except from F. By appropriately selecting sets A and B, the only paths into A (resp. B) are via F and so are limited to at most f, violating condition SC with parameter F, a contradiction.

▶ **Lemma 6.** *For any choice of set F in the algorithm the directed graph decomposition of $G - F$ has a unique source component.*

Proof. Fix an arbitrary set F. Suppose for the sake of contradiction that $G - F$ has two source components S_1 and S_2. Let $C := V - S_1 - S_2$ be the rest of the nodes. We create a partition (A, B) that violates the requirements of condition SC with parameter F (Definition 1). Let $A := S_1$ and $B := S_2 \cup C = V - S_1$. First note that both $A - F = A = S_1$ and $B - F \supseteq S_2$ are non-empty.

Now, by construction, we have that S_2 has no incoming edges except from F. Therefore, F is a cut set that separates S_2 from S_1, i.e., there are no paths from a node in S_1 to a node in S_2 in $G - F$. By Menger's Theorem, there are at most $|F| \leq f$ node-disjoint paths in G from $S_1 = A$ to any node in $S_2 \subseteq B - F$. Therefore, for any node $v \in S_2 \subseteq B - F$, there are at most f node-disjoint Av-paths that exclude F, and so $A \overset{F}{\not\rightarrow}_G B - F$. Similarly, by construction, we have that S_1 has no incoming edges except from F. Therefore, F is a cut set that separates S_1 from $S_2 \cup C - F$. By Menger's Theorem, there are at most $|F| \leq f$ node-disjoint paths in G from $S_2 \cup C = B$ to any node in $S_1 = A - F$. Therefore, for any node $v \in S_1 = A - F$, there are at most f node-disjoint Bv-paths that exclude F, and so $B \overset{F}{\not\rightarrow}_G A - F$. This violates condition SC, a contradiction. ◀

Next, this unique source component S, along with its in-neighborhood in F, satisfies condition SC with parameter F. We follow the same approach as in proof of Lemma 6, assuming for the sake of contradiction that $G[S \cup \Gamma(F, S)]$ does not satisfy condition SC with parameter F, and showing that this implies that G does not satisfy condition SC with parameter F.

▶ **Lemma 7.** *For any choice of set F in the algorithm, let S be the unique source component of $G - F$. Then $G[S \cup \Gamma(F, S)]$ satisfies condition SC with parameter F.*

Proof. Suppose for the sake of contradiction that $H := G[S \cup \Gamma(F, S)]$ does not satisfy condition SC with parameter F. So there exists a partition (A, B) of $S \cup \Gamma(F, S)$, such that

$A - F$ and $B - F$ are non-empty, and $A \overset{F}{\not\rightarrow}_H B - F$ and $B \overset{F}{\not\rightarrow}_H A - F$. Let the rest of the nodes in G be denoted by $C := V - S \cup \Gamma(F, S)$. Let $A' := A \cup C$. Then (A', B) is a partition of V, with $A' - F$ and $B - F$ both non-empty.

We first show that there is no edge from C to S in G. Observe that C is disjoint from $S \cup \Gamma(F, S)$. If a node $u \in C \cap F$ has an edge to a node in S, then $u \in \Gamma(F, S)$, a contradiction since $C \cap \Gamma(F, S) = \emptyset$. On the other hand, if $u \in C - F$ has an edge to a node in S, then the edge exists in $G - F$, which is a contradiction since S is a source component in $G - F$.

Now, since $A \overset{F}{\not\rightarrow}_H B - F$, we have that for some node $v \in B - F$ there are at most f node-disjoint Av-paths in H_{-F}.[3] By Menger's Theorem, there exists a cut set $X \not\ni v$ of cardinality at most f that separates v from $A - X$ in H_{-F}. Since there is no edge from C to S in G, we have that X also separates v from $(A - X) \cup C = A' - X$ in G_{-F}. It follows that there are at most f node-disjoint $A'v$-paths that exclude F in G, and so $A' \overset{F}{\not\rightarrow}_G B - F$.

Similarly, since $B \overset{F}{\not\rightarrow}_H A - F$, we have that for some node $v \in A - F$ there are at most f node-disjoint Bv-paths in H_{-F}. By Menger's Theorem, there exists a cut set $X \not\ni v$ of cardinality at most f that separates v from $B - X$ in H_{-F}. Since there is no edge from C to S in G, we have that X also separates v from $B - X$ in G_{-F}. Note that $v \in A - F \subseteq A' - F$. It follows that there are at most f node-disjoint Bv-paths that exclude F in G, and so $B \overset{F}{\not\rightarrow}_G A' - F$. This violates condition SC, a contradiction. ◀

We now show that the paths identified in steps (c), (d), and (e) of the algorithm do indeed exist. The existence of paths in step (c) follows by construction of S.

▶ **Lemma 8.** *For any choice of set F in the algorithm, let S be the unique source component of $G - F$. Then, for any two nodes $u \in S \cup \Gamma(F, S)$ and $v \in S$, there exists a uv-path that excludes F.*

Proof. If $u \in S$, then there exists a uv-path in $G - F$ since S is strongly connected in $G - F$ by construction. If $u \in \Gamma(F, S)$, then there exists a node $w \in S$ such that (u, w) is an edge in G. Also, there exists a wv-path P_{wv} in $G - F$ since S is strongly connected in $G - F$. So u - P_{wv} is a uv-path in G that excludes F. ◀

The existence of paths in step (d) follows from Lemma 7 and definition of condition SC (Definition 1).

▶ **Lemma 9.** *For any non-faulty node v, and any given phase with the corresponding set F, in step (d), if $v \in B_v$, then there exist $f + 1$ node-disjoint $A_v v$-paths that exclude F.*

Proof. Fix a phase of the algorithm and the corresponding set F. Consider an arbitrary non-faulty node v such that $v \in B_v$ in step (d). By construction, B_v is either $N_v - F$ or $Z_v - F$, both of which are non-empty. In the first case, we have that $A_v = Z_v \overset{F}{\rightsquigarrow} N_v - F = B_v$. In the second case, we have that $Z_v \overset{F}{\not\rightarrow} N_v - F$. By Lemma 7, we have that $G[Z_v \cup N_v] = G[S \cup \Gamma(F, S)]$ satisfies condition SC with parameter F. Therefore $A_v = N_v \overset{F}{\rightsquigarrow} Z_v - F = B_v$. ◀

[3] Recall from Section 3 that G_{-U} is the graph obtained from G by removing all incoming edges to U so that if P is a path in G_{-U}, then P is a path in G that excludes U and terminates in $V - U$, and if P is a path in G that excludes U and terminates in $V - U$, then P is a path in G_{-U}.

For paths in step (e), note that, by construction of S, the in-neighbors of S are contained entirely in F. So there can only be at most f paths into S from $V - S$. Since G satisfies condition SC with parameter F, we get that $S \overset{F}{\rightsquigarrow} V - S - F$.

▶ **Lemma 10.** *For any choice of set F in the algorithm, let S be the unique source component of $G - F$. Then $S \overset{F}{\rightsquigarrow} V - S - F$.*

Proof. Let $A = S$ and $B = F \cup (V - S) = V - A$. Now, since $A = S$ is the unique source component of $G - F$, we have that $\Gamma(B, A) \subseteq F$, i.e., nodes in F are the only ones in B to have an edge into A. So B can have at most f node-disjoint paths to any node in A. Thus $B \overset{F}{\not\rightsquigarrow} A - F$. Since G satisfies condition SC, we have that $S = A \overset{F}{\rightsquigarrow} B - F = V - S - F$, as required. ◀

6.3 Proof of Correctness

We provide a proof of correctness of Algorithm 1 by proving Theorem 11 below.

▶ **Theorem 11.** *Under the local broadcast model, Byzantine consensus tolerating at most f Byzantine faulty nodes is achievable on a directed graph G, if G satisfies condition SC.*

Let us assume that G satisfies condition SC. For convenience, we will refer to the state variable γ_v as the "state of node v". We remind the reader that the algorithm proceeds in phases and each phase has an associated unique set F, with $|F| \leq f$. We use F^* to denote the actual set of faulty nodes in a given execution.

As mentioned earlier, the algorithm attempts to balance two objectives. We formalize them in the two lemmas below. The proofs of Lemmas 12 and 13 are presented later.

▶ **Lemma 12.** *Consider the unique phase of Algorithm 1 where the corresponding set $F = F^*$. At the end of this phase, every pair of non-faulty nodes $u, v \in V$ has an identical state, i.e., $\gamma_u = \gamma_v$.*

▶ **Lemma 13.** *For a non-faulty node v, its state γ_v at the end of any given phase equals the state of some non-faulty node at the start of that phase.*

The correctness of Algorithm 1 follows from these two lemmas, as shown next.

Proof of Theorem 11. To prove the correctness of Algorithm 1, we have to prove the three properties of Agreement, Validity, and Termination, as specified in Section 3.

Termination: The algorithm satisfies the termination property because there are a finite number of phases in the algorithm, and each of them completes in finite time.

Agreement: The total number of faulty nodes is bounded by f. Therefore, in any execution, there exists at least one phase in which the set $F = F^*$. From Lemma 12, all non-faulty nodes have the same state at the end of this phase. Lemma 13 implies that the state of the non-faulty nodes will remain unchanged in the subsequent phases. So all non-faulty nodes will have identical outputs, satisfying the agreement property.

Validity: The state of each non-faulty node is initialized to its own input at the start of the algorithm. So the state of each non-faulty node is its own input at the start of the first phase. By applying Lemma 13 inductively, we have that the state of a non-faulty node always equals the *input* of some non-faulty node. This satisfies the validity property.

This completes the proof of correctness of Algorithm 1. ◀

The rest of the section focuses on proving Lemmas 12 and 13. We assume that the graph G satisfies condition SC, even if it is not explicitly stated. We use F^* to denote the actual faulty set. The following observation follows from the rules used for flooding [14].

▶ **Observation 14.** *For any phase of Algorithm 1, for any two nodes $u, v \in V$ (possibly faulty), if v receives value b along a fault-free uv-path then u broadcast the value b to its neighbors during flooding.*

Fix a phase in the algorithm along with the corresponding set F. For Lemma 12, the correctness relies on the local broadcast property. Suppose, as stated in the statement of the lemma, that $F = F^*$ in a given phase. There are two cases to consider for any non-faulty node v. In the first case $v \in S$. In this case, the paths used by v in step (c) of the phase exclude F. So these paths are fault-free (i.e., none of their internal nodes are faulty). Then, the properties of flooding imply that any two non-faulty nodes u, v will obtain $Z_u = Z_v$ and $N_u = N_v$ in step (c). By a similar argument, all the paths used in step (d) of this phase are also fault-free, and any two non-faulty nodes will end step (d) with an identical state. In the second case $v \in V - S - F$, a repeat of the above argument implies that the paths used in step (f) are both fault-free and have non-faulty source nodes. Since the source nodes are all from S, they have an identical state at the end of step (d). So v correctly updates its state in step (f) to match the identical state in S.

Proof of Lemma 12. Fix a phase of the algorithm and the corresponding set F such that $F = F^*$. We first show that all nodes in S have identical state, at the end of the phase, and then consider nodes in $V - S - F^*$.

Let Z be the set of nodes that flooded 0 in step (b) and let N be the set of nodes that flooded 1 in step (b). Note that Z and N may contain faulty nodes, but due to the broadcast property, a faulty node is in at most one of these sets. Consider any non-faulty node $v \in S$. Then $Z_v = Z$ and $N_v = N$, as follows. Let $w \in S \cup \Gamma(F^*, S)$ be an arbitrary node that flooded 0 (resp. 1) in step (b), i.e., $w \in Z$ (resp. $w \in N$). Now the wv-path P_{wv} identified by v in step (b) excludes F^* and is fault-free. So, by Observation 14, v receives 0 (resp. 1) along P_{wv} and correctly sets $w \in Z_v$ (resp. $w \in N_v$).

Therefore, we have that for any two non-faulty nodes $u, v \in S$, $Z_u = Z_v = Z$ and $N_u = N_v = N$. If $Z - F^*$ (resp. $N - F^*$) is empty, then $S = N - F^*$ (resp. $S = Z - F^*$). So all non-faulty nodes in S have identical state, which is not updated, and the claim is trivially true. So suppose both $Z - F^*$ and $N - F^*$ are non-empty. Since $Z_u = Z_v$ and $N_u = N_v$, we have that $A_u = A_v$ and $B_u = B_v$. Let $A := A_u$ and $B := B_u$. By construction $A \overset{F^*}{\rightsquigarrow} B$. Now all nodes in A flooded identical value in step (b), say α. If $u \in A$, then u's state is α at the beginning of the phase and stays unchanged in step (d), i.e., $\gamma_u = \alpha$. If $u \in B$, then the $f+1$ node-disjoint Au-paths identified by u in step (d) exclude F^* and so are all fault-free. By Observation 14, it follows that u receives α identically along these $f+1$ paths and updates $\gamma_u = \alpha$. Similarly, $\gamma_v = \alpha$.

So we have shown that all non-faulty nodes in S have identical state α at the end of step (d). Since these nodes do not update their state in the rest of the phase, so we have that all nodes in S have state α at the end of the phase. Since all nodes in S are non-faulty, we have that each node in S floods α in step (e). Consider any non-faulty node $v \in V - S - F^*$. The $f+1$ node-disjoint Sv-paths identified by v in step (f) exclude F^* and so are all fault-free. Recall that S contains only nodes in $V - F^*$, i.e., only non-faulty nodes, so the source nodes in these $f+1$ paths are also non-faulty. By Observation 14, it follows that v receives α identically along these $f+1$ paths and updates $\gamma_v = \alpha$. So by the end of the phase, all non-faulty nodes have identical state α, as required. ◀

Lemma 13 follows from the following observation. In steps (d) and (f), if a node v updates its state, then it must have received identical value along $f+1$ node-disjoint $A_v v$-paths. Therefore, at least one of these paths must both be fault-free and have a non-faulty source.

Proof of Lemma 13. Fix a phase of the algorithm and the corresponding set F. We use γ_u^{start} and γ_u^{end} to denote the state γ_u of node u at the beginning and end of the phase, respectively. Let v be an arbitrary non-faulty node. If v does not update its state in this phase, then $\gamma_u^{\text{end}} = \gamma_u^{\text{start}}$ and the claim is trivially true. So suppose that v did update its state in this phase. This implies that $v \in V - F$.

There are now two cases to consider

Case 1: $v \in S$. It follows that v updated its state in step (d). Therefore, $v \in B_v$ and v received identical values along $f+1$ node-disjoint $A_v v$-paths in step (b). Now, at least one of these paths 1) is fault-free, and 2) has a non-faulty source u. By Observation 14, we have that the value received by v along this uv-path in step (b) is the value flooded by u in step (b). So $\gamma_v^{\text{end}} = \gamma_u^{\text{start}}$, where u is a non-faulty node.

Case 2: $v \in V - S - F$. It follows that v updated its state in step (f). Therefore, v received identical values along $f+1$ node-disjoint Sv-paths in step (e). Now, at least one of these paths 1) is fault-free, and 2) has a non-faulty source $w \in S$. By Observation 14, we have that the value received by v along this wv-path in step (e) is the value flooded by w in step (e). Note that $w \in S$ and so the value flooded by w in step (e) is the state of w at the end of this phase, i.e., w flooded γ_w^{end} in step (e). There are further two cases to consider

- **Case i:** w did not update its state in this iteration. Then $\gamma_v^{\text{end}} = \gamma_w^{\text{end}} = \gamma_w^{\text{start}}$. Recall that w is a non-faulty node.
- **Case ii:** w did update its state in this iteration. Then, from Case 1 above, we have that there exists a non-faulty node u such that $\gamma_w^{\text{end}} = \gamma_u^{\text{start}}$. So $\gamma_v^{\text{end}} = \gamma_w^{\text{end}} = \gamma_u^{\text{start}}$, where u is a non-faulty node.

In all cases, we have that γ_v^{end} equals γ_u^{start} for some non-faulty node u. ◀

7 Summary

In this work, we have presented tight conditions for exact binary Byzantine consensus in directed graphs under the local broadcast model. The sufficiency proof in Section 6 is constructive. However, the algorithm has exponential round complexity. We leave finding a more efficient algorithm for future work. The following question is also open: does there exist an efficient algorithm to check if a given directed graph satisfies condition SC?

References

1. S. Amitanand, I. Sanketh, K. Srinathant, V. Vinod, and C. Pandu Rangan. Distributed Consensus in the Presence of Sectional Faults. In *Proceedings of the Twenty-second Annual Symposium on Principles of Distributed Computing*, PODC '03, pages 202–210, New York, NY, USA, 2003. ACM. doi:10.1145/872035.872065.
2. Hagit Attiya and Jennifer Welch. *Distributed Computing: Fundamentals, Simulations and Advanced Topics.* John Wiley & Sons, Inc., USA, 2004.
3. Vartika Bhandari and Nitin H. Vaidya. On Reliable Broadcast in a Radio Network. In *Proceedings of the Twenty-fourth Annual ACM Symposium on Principles of Distributed Computing*, PODC '05, pages 138–147, New York, NY, USA, 2005. ACM. doi:10.1145/1073814.1073841.

4 Byung-Gon Chun, Petros Maniatis, Scott Shenker, and John Kubiatowicz. Attested Append-only Memory: Making Adversaries Stick to Their Word. *SIGOPS Oper. Syst. Rev.*, 41(6):189–204, October 2007. doi:10.1145/1323293.1294280.

5 Allen Clement, Flavio Junqueira, Aniket Kate, and Rodrigo Rodrigues. On the (Limited) Power of Non-equivocation. In *Proceedings of the 2012 ACM Symposium on Principles of Distributed Computing*, PODC '12, pages 301–308, New York, NY, USA, 2012. ACM. doi:10.1145/2332432.2332490.

6 Jeffrey Considine, Matthias Fitzi, Matthew Franklin, Leonid A. Levin, Ueli Maurer, and David Metcalf. Byzantine Agreement Given Partial Broadcast. *Journal of Cryptology*, 18(3):191–217, July 2005. doi:10.1007/s00145-005-0308-x.

7 Danny Dolev. The Byzantine generals strike again. *Journal of Algorithms*, 3(1):14–30, 1982. doi:10.1016/0196-6774(82)90004-9.

8 Michael J. Fischer, Nancy A. Lynch, and Michael Merritt. Easy impossibility proofs for distributed consensus problems. *Distributed Computing*, 1(1):26–39, March 1986. doi:10.1007/BF01843568.

9 Mattias Fitzi and Ueli Maurer. From Partial Consistency to Global Broadcast. In *Proceedings of the Thirty-second Annual ACM Symposium on Theory of Computing*, STOC '00, pages 494–503, New York, NY, USA, 2000. ACM. doi:10.1145/335305.335363.

10 M. Franklin and M. Yung. Secure Hypergraphs: Privacy from Partial Broadcast. *SIAM Journal on Discrete Mathematics*, 18(3):437–450, 2004. doi:10.1137/S0895480198335215.

11 Matthew Franklin and Rebecca N. Wright. Secure Communication in Minimal Connectivity Models. *Journal of Cryptology*, 13(1):9–30, January 2000. doi:10.1007/s001459910002.

12 Alexander Jaffe, Thomas Moscibroda, and Siddhartha Sen. On the Price of Equivocation in Byzantine Agreement. In *Proceedings of the 2012 ACM Symposium on Principles of Distributed Computing*, PODC '12, pages 309–318, New York, NY, USA, 2012. ACM. doi:10.1145/2332432.2332491.

13 Muhammad Samir Khan, Syed Shalan Naqvi, and Nitin H. Vaidya. Exact Byzantine Consensus on Undirected Graphs Under Local Broadcast Model. In *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing*, PODC '19, pages 327–336, New York, NY, USA, 2019. ACM. doi:10.1145/3293611.3331619.

14 Muhammad Samir Khan, Lewis Tseng, and Nitin H. Vaidya. Exact Byzantine Consensus on Arbitrary Directed Graphs under Local Broadcast Model. *CoRR*, 2019. arXiv:1911.07298.

15 Chiu-Yuen Koo. Broadcast in Radio Networks Tolerating Byzantine Adversarial Behavior. In *Proceedings of the Twenty-third Annual ACM Symposium on Principles of Distributed Computing*, PODC '04, pages 275–282, New York, NY, USA, 2004. ACM. doi:10.1145/1011767.1011807.

16 Chiu-Yuen Koo, Vartika Bhandari, Jonathan Katz, and Nitin H. Vaidya. Reliable Broadcast in Radio Networks: The Bounded Collision Case. In *Proceedings of the Twenty-fifth Annual ACM Symposium on Principles of Distributed Computing*, PODC '06, pages 258–264, New York, NY, USA, 2006. ACM. doi:10.1145/1146381.1146420.

17 Leslie Lamport, Robert Shostak, and Marshall Pease. The Byzantine Generals Problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, July 1982. doi:10.1145/357172.357176.

18 H. J. LeBlanc, H. Zhang, X. Koutsoukos, and S. Sundaram. Resilient Asymptotic Consensus in Robust Networks. *IEEE Journal on Selected Areas in Communications*, 31(4):766–781, April 2013. doi:10.1109/JSAC.2013.130413.

19 C. Li, M. Hurfin, Y. Wang, and L. Yu. Towards a Restrained Use of Non-Equivocation for Achieving Iterative Approximate Byzantine Consensus. In *2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, pages 710–719, May 2016. doi:10.1109/IPDPS.2016.62.

20 Nancy A. Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1996.

21 Syed Shalan Naqvi, Muhammad Samir Khan, and Nitin H. Vaidya. Exact Byzantine Consensus Under Local-Broadcast Model. *CoRR*, abs/1811.08535, 2018. arXiv:1811.08535.

22 M. Pease, R. Shostak, and L. Lamport. Reaching Agreement in the Presence of Faults. *J. ACM*, 27(2):228–234, April 1980. doi:10.1145/322186.322188.

23 T. Rabin and M. Ben-Or. Verifiable Secret Sharing and Multiparty Protocols with Honest Majority. In *Proceedings of the Twenty-first Annual ACM Symposium on Theory of Computing*, STOC '89, pages 73–85, New York, NY, USA, 1989. ACM. doi:10.1145/73007.73014.

24 D. V. S. Ravikant, V. Muthuramakrishnan, V. Srikanth, K. Srinathan, and C. Pandu Rangan. On Byzantine Agreement over (2,3)-Uniform Hypergraphs. In Rachid Guerraoui, editor, *Distributed Computing*, pages 450–464, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

25 Lewis Tseng and Nitin Vaidya. Iterative Approximate Byzantine Consensus under a Generalized Fault Model. In Davide Frey, Michel Raynal, Saswati Sarkar, Rudrapatna K. Shyamasundar, and Prasun Sinha, editors, *Distributed Computing and Networking*, pages 72–86, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

26 Lewis Tseng and Nitin H. Vaidya. Fault-Tolerant Consensus in Directed Graphs. In *Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing*, PODC '15, pages 451–460, New York, NY, USA, 2015. ACM. doi:10.1145/2767386.2767399.

27 Nitin H. Vaidya. Iterative Byzantine Vector Consensus in Incomplete Graphs. In Mainak Chatterjee, Jian-nong Cao, Kishore Kothapalli, and Sergio Rajsbaum, editors, *Distributed Computing and Networking*, pages 14–28, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

28 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. doi:10.1145/2332432.2332505.

29 Yongge Wang and Yvo Desmedt. Secure Communication in Multicast Channels: The Answer to Franklin and Wright's Question. *Journal of Cryptology*, 14(2):121–135, March 2001. doi:10.1007/s00145-001-0002-y.

30 H. Zhang and S. Sundaram. Robustness of information diffusion algorithms to locally bounded adversaries. In *2012 American Control Conference (ACC)*, pages 5855–5861, June 2012. doi:10.1109/ACC.2012.6315661.