Crash-tolerant Consensus in Directed Graph Revisited*

Ashish Choudhury† Gayathri Garimella‡ Arpita Patra§ Divya Ravi¶ Pratik Sarkar‖

Abstract

Fault-tolerant distributed consensus is a fundamental problem in secure distributed computing. In this work, we consider the problem of distributed consensus in directed graphs tolerating crash failures. Tseng and Vaidya (PODC'15) presented necessary and sufficient condition for the existence of consensus protocols in directed graphs. We improve the round and communication complexity of their protocol. Moreover, we prove that our protocol requires the optimal number of communication rounds, required by any protocol belonging to a restricted class of crash-tolerant consensus protocols in directed graphs.

Keywords: Directed graph, Consensus, Crash failure, Round complexity.

1 Introduction

Fault-tolerant reliable consensus [11, 6, 7] is a fundamental problem in distributed computing. Informally, a consensus protocol allows a set of n mutually distrusting parties, each with some private input, to agree on a common output. This is ensured even in the presence of a *computationally unbounded* centralized adversary, who may corrupt any f out of the n parties and try to prevent the remaining parties from achieving consensus. Since its inception [11], the problem has been widely studied in the literature and several interesting results have been obtained regarding the possibility, feasibility and optimality of reliable consensus (see [10, 2, 8, 13] and their references for the exhaustive list of work done in this area). However, all these results are derived assuming the underlying communication network to be a *complete undirected graph*, where the parties are assumed to be directly connected with each other by pair-wise private and authentic channels. There are scenarios, where such undirected graphs may not be available. For example, in a typical wireless network, the communication links may support only uni-directional communication. In a radio network, a base station can communicate to the receiving nodes, but communication in the other direction may not be possible. Further, it may be the case that a node is connected to some other node "indirectly" via intermediate nodes. Thus in a practical network like the Internet it is hard to ensure that every user is directly connected with every other user by a dedicated channel. This scenario can be appropriately modelled by a more generic *incomplete directed graph*. We are interested in the consensus problem in such arbitrary directed graphs.

*A preliminary version of this paper got published as a brief announcement in DISC 2017.

†International Institute of Information Technology, Bangalore India. Email: ashish.choudhury@iiitb.ac.in. Financial support from Infosys foundation acknowledged.

‡International Institute of Information Technology, Bangalore India. Email: AnnapurnaGayathri.Garimella@iiitb.org.

§Department of Computer Science and Automation, Indian Institute of Science, Bangalore India. Email: arpita@csa.iisc.ernet.in.

¶Department of Computer Science and Automation, Indian Institute of Science, Bangalore India. Email: divya.ravi@csa.iisc.ernet.in.

‖Department of Computer Science and Automation, Indian Institute of Science, Bangalore India. Email: pratik.sarkar@csa.iisc.ernet.in.

In a series of beautiful work [14, 15, 16], the possibility of consensus protocols in arbitrary directed graphs is studied, where necessary and sufficient conditions are presented for the existence of consensus protocols. Separate conditions are derived for the *fail-stop* and *Byzantine* adversary model. The fail-stop model is a weaker adversary model and assumes that the adversary can crash any f nodes during the execution of a protocol. The more stronger Byzantine adversary model assumes that the adversary has full control over the set of f nodes under its control, which can be forced to behave in any arbitrary fashion during the protocol execution. In this work, we revisit the crash-tolerant version of the consensus problem in arbitrary directed graphs; specifically we look into the round complexity of crash-tolerant consensus protocols in arbitrary directed graphs. We stress that even though the fail-stop model is a weaker adversary model, it is practically motivated. For instance, in a typical distributed system, the chances that some of the "components" of the system stop working are more compared to some of the components behaving erratically. More specifically, it is relatively simpler for an attacker to crash a system and make it stop working completely, compared to taking full control of it and make it behave in an erroneous fashion. Hence studying the round complexity of consensus protocols in arbitrary directed networks against fail-stop corruption is practically motivated.

Existing Results for Crash-tolerant Consensus in Directed Graphs: The necessary (and sufficient) condition for the existence of crash-tolerant consensus protocol in directed graphs is presented in [14] and this is not a straight-forward extension of the necessary condition for the existence of crash-tolerant consensus in undirected[1] graphs. Informally, in directed graphs the necessary condition demands that even if an arbitrary set of f nodes crashes, there should still exist a special node in the graph, called *source*, which should have a directed path to every other node in the remaining graph (see Section 2 for the formal definition of source node and other related terms). The authors in [14] proved the sufficiency of their necessity condition by presenting two consensus protocols, one for the binary and the other for the multi-valued case.[2]

The protocols of [14] are significantly different from the traditional consensus protocols developed for undirected graphs. Specially they belong to a special class of consensus protocols, based on "flooding". In more detail, the protocols consist of several "phases", each consisting of d rounds of "send-receive-update", where d is called the *crash-tolerant diameter* of a directed graph. Informally, d is the maximum distance of any node from a potential source in the graph. Thus any given potential source can propagate its value to all remaining nodes in a single phase within the d rounds of flooding. In a round every node (including the source) broadcasts its value to its neighbours. At the end of the round, each node "updates" its value, by locally applying an update function to the received values. In the subsequent round, nodes broadcast their updated value. Now, two types of update function applied are a min function for a *min phase* and a max function for a *max phase*. The min function requires nodes to update their value by taking the minimum of all the received values (including its own value) and symmetrically in the max function nodes update by taking the maximum of all the received values. In [14], it was also shown that the usage of two different types of update function is necessary to achieve consensus for *anonymous* directed graphs.

The binary consensus protocol of [14] requires $2f + 2$ alternate min-max phases, each with d rounds. The round complexity of the protocol is $(2f + 2) \cdot d$ rounds and the communication complexity is $\mathcal{O}(nfd)$ bits (the number of neighbours of a node is $\mathcal{O}(n)$). In [14] the authors claimed that their binary consensus protocol based on min-max strategy cannot be extended trivially to the multi-valued case. Hence they present a different multi-valued consensus protocol, which in essence runs their binary consensus protocol K times, when the inputs are in the set $\{0, \ldots, K\}$. The idea is to run an instance of the min-max based binary consensus for each candidate $k \in \{0, \ldots, K\}$ to verify if some source node has a value k and if so,

[1] In undirected graphs, $f+1$ node connectivity is both and necessary and sufficient for the existence of crash-tolerant consensus.

[2] In the binary consensus problem, the inputs of each node is a binary value. On the other hand in the multi-valued case, the inputs belong to a publicly known domain.

then try to reach agreement on this value k. The protocol requires $(2f+2) \cdot d \cdot K$ rounds of communication and the communication complexity is $\mathcal{O}(nfdK \log K)$ bits since a node has to communicate $\log K$ bits to every neighbour in a round. Clearly the protocol has exponential round and communication complexity, as $K = 2^{\log K}$.

Our Motivation and Results: In this work, we revisit the consensus protocols of [14] based on min-max strategy. Our main motivation is to improve the round and communication complexity of their protocols because the number of communication rounds and the amount of communication done in each round are crucial resources in a distributed protocol. We consider the binary consensus protocol of [14] and observe that if instead of d, we allow $d+1$ rounds of communication in each of the phases, then it is possible to achieve consensus with just $f+2$ alternate min-max phases, thus making the round complexity $(f+2)(d+1)$. We then show an optimization of our protocol, where we allow *only* d rounds in the *first* and the *last* phase, thus reducing the round complexity to $(f+2)(d+1)-2$. Interestingly, we show that our protocol works even for the multi-valued case, with *no* modifications what so ever. Hence, unlike [14], the round complexity of our multi-valued consensus protocol is *independent* of K. The communication complexity of our protocol is $\mathcal{O}(nfd \log K)$ bits and for significantly large values of K our protocol improves upon the round and communication complexity of the multi-valued consensus protocols of [14]. Moreover, we improve the number of rounds for the binary consensus, for every $f, d \geq 2$.

We also address the problem of lower bound on the minimum number of rounds required by any crash-tolerant consensus protocol in a directed graph, based on min-max strategy and derive three interesting lower bounds. We first consider the case, where only $f+1$ min-max phases are allowed in the protocol and with *no restriction* on the number of communication rounds in each phase. We show that it is impossible to achieve crash-tolerant consensus within $f+1$ phases. Next we consider min-max based consensus protocols with *at least* d rounds in each phase. For such protocols, we show that it is impossible to achieve consensus in general with $(f+2)(d+1)-3$ rounds in total. This further shows that our min-max based protocol with $(f+2)(d+1)-2$ rounds is *round optimal*. Finally we consider min-max based consensus protocols with *exactly* d rounds of communication in each phase. Note that the consensus protocols of [14] belong to this class. For several values of f and d, we show that the minimum number of phases required to achieve consensus in this case is $2f+2$, thus showing that the binary consensus protocol of [14] has the *optimal* number of communication rounds.

All the above lower bounds are derived by presenting non-trivial directed graphs and corresponding adversary strategies, which ensure that consensus is not achieved till sufficient number of min-max phases are allowed in the underlying protocol. We stress that different graphs and adversary strategies are required to derive the lower bound for different cases. Even though the lower bounds are for a restricted class of protocols (namely the one based on min-max strategy), to the best of our knowledge, these are the first (non-trivial) lower bounds on the round complexity of consensus protocols in directed graphs. More importantly, the lower bounds establish that our protocol is the best in terms of the round complexity if one is interested to design consensus protocols based on min-max strategy. Hence to obtain further improvements in the round complexity, a different approach (other than the min-max based strategy) is required.

Informal Discussion on Our Protocol: Our starting point is the binary consensus protocol of [14] with $2f+2$ phases, each with d rounds. The correctness of their protocol is based on the guaranteed occurrence of two *consecutive* crash-free phases, among the $2f+2$ alternate min-max phases, within which consensus is shown to be achieved. We observe that if instead of d rounds, we allow $d+1$ rounds in each phase then consensus can be achieved if we either have two consecutive crash-free phases or a crashed phase followed by a crash-free phase, provided *only one* node crashes during the crashed phase. The base of our observation is the following: if during the crashed phase the single node to be crashed is a non-source node, then it is equivalent to having two consecutive crash-free phases (with source node(s) being unaltered) and so con-

sensus will be achieved within these two phases. On the other hand, if during the crashed phase the single node to be crashed is a source node, then at least one of new source nodes will be at a distance of *one* from the crashed source (this observation lies at the heart of our protocol). If the crashed source node sends its value to one of the new source node before crashing, there will be still d rounds left for this new source node in the crashed phase to further propagate the crashed source node's value in the remaining graph. So in essence, we still get the effect of two consecutive crash-free phases. We further show that with $f + 2$ alternate min-max phases, there always exist either two crash-free phases or a crashed phase with a single crash, followed by a crash-free phase, irrespective of the way adversary crashes the f nodes.

Moving from the binary case to the multi-valued case, we find that the above ideas are applicable even for the multi-valued case. For simplicity, we consider the case when there are two crash-free phases and without loss of generality, let these be a min phase followed by a max phase. Let λ^{min} be the least value among the source nodes at the beginning of crash-free min phase. If the non-source nodes have their value greater than or equal to λ^{min} at the beginning of this phase, then clearly consensus will be achieved at the end of this min phase itself; this is because each node will update their value to λ^{min} at the end of the min phase. On the other hand, if some *non-source* node has a value smaller than λ^{min} at the beginning of the crash-free min phase, then consensus will not be achieved in this phase. However, at the end of this min phase, the modified values of all the nodes (both source as well as non-source) is upper bounded by λ^{min}; moreover all the *source nodes* will have λ^{min} as their modified value. Hence in the next crash-free phase which is a max phase, the value λ^{min} of the source nodes will be the maximum value in the graph and hence consensus will be achieved at the end of the crash-free max phase[3]. The above argument also works for the case when there is a crashed phase followed by a crash-free phase, where it is guaranteed that exactly one node crashes during the crashed phase.

Related Work: In [5], possibility of *approximate* crash-tolerant consensus in dynamic directed graphs is studied; informally in an approximate consensus protocol, the fault-free nodes are supposed to produce outputs within a certain constant ϵ of each other, where $\epsilon > 0$. On contrary, we are interested in the *exact* consensus, where $\epsilon = 0$. As mentioned earlier, most of the literature on consensus considers a complete graph, where parties are connected by pair-wise reliable channels and where the graph is assumed to be *static*. However, there are few works which consider different variations of this model. For example, [3] considers *undirected* graphs and shows that all-pair reliable communication is not necessary to achieve consensus against *Byzantine* adversary, provided nodes can use some authentication mechanism. In [1], Byzantine consensus in *unknown* networks is considered, where the underlying network remains *fully connected*. [4, 9] considers *fault-free*, *approximate* consensus protocols where there are no faults, but the underlying graph is partially connected and *dynamic*. In [12], the authors consider edge corruptions, where edges may get Byzantine corrupted, but nodes remain fault-free. All these variations of consensus is different from the setting considered in this paper and so these results are incomparable to ours. Hence we do not consider these works for further discussion.

Future directions: Our protocol as well as the protocols of [14, 15] are based on min-max strategy. It will be interesting to explore other classes of consensus protocols, based on the combination of different strategies. One could also explore a combination of different update functions in each phase, instead of the same update function being used throughout a phase. We explored only exact consensus in the synchronous communication setting against crash faults. Finding lower bounds on the round complexity of consensus protocols (both exact and approximate) in directed graphs against Byzantine adversary is left as a very interesting and challenging open problem.

[3]This argument also shows that the binary consensus protocol of [14] with $2f + 2$ alternate min-max phases will work for the multi-valued case as well, with no modifications; this is because there will be always two consecutive crash-free phases.

2 Preliminaries, Definitions and Notations

We consider a distributed synchronous network modelled as a simple directed graph $\mathbf{G} = (\mathsf{V}, \mathsf{E})$ where V represents the set of n nodes $\{v_1, v_2, \ldots, v_n\}$ and E represents the set of directed edges between the nodes in V. The communication network is assumed to be *static*; i.e. edges and nodes are not allowed to be inserted or deleted dynamically. Node v_i can communicate to node v_j if and only if the directed edge $(v_i, v_j) \in \mathsf{E}$. Moreover we assume that each node can send messages to itself. For a node $v \in \mathsf{V}$, the set N_v^+ denotes the set of "outgoing neighbours" of v in $\mathbf{G}$. That is, $\mathsf{N}_v^+ = \{v_j | (v, v_j) \in \mathsf{E}\}$. Thus, v can "directly" send messages to the nodes in the set N_v^+. The set N_v^- denotes the set of "incoming neighbours" of v in $\mathbf{G}$. That is, $\mathsf{N}_v^- = \{v_j | (v_j, v) \in \mathsf{E}\}$. Thus, the nodes in N_v^- can "directly" send messages to the node v. The network is assumed to be synchronous where all the nodes are synchronised and there exists a known upper bound on message delay. Any protocol in such a network is assumed to proceed as a sequence of rounds, where in every round, each node sends messages to its outgoing neighbours, receives messages sent by its incoming neighbours in that round, followed by local computation. We assume a *computationally unbounded* adaptive adversary $\mathcal{A}$, which can corrupt any f nodes in $\mathbf{G}$ in a fail-stop fashion, where a corrupted node can crash at any point of time during the execution of a protocol; however till the node crashes, it *honestly* follows the instructions of the underlying protocol. We also assume that if a node crashes during a round, then an arbitrary subset of its outgoing messages for that round are delivered to the corresponding neighbours, as decided by $\mathcal{A}$. We next define the consensus problem.

Definition 2.1 (**Multi-valued Crash-Tolerant Consensus** [14])**.** Let Π be a synchronous protocol for the n nodes in $\mathbf{G}$, where each node $v_i \in \mathsf{V}$ has an input $in_i \in \{0, \ldots, K\}$ and each party has an output $out_i \in \{0, \ldots, K\}$, where K is publicly known. Then Π is called a crash-tolerant consensus protocol tolerating $\mathcal{A}$ if the following holds: **(1) Agreement**: All fault-free nodes should have the same output. That is, for every fault-free nodes $v_i, v_j \in \mathsf{V}$, the condition $out_i = out_j$ holds. **(2) Validity**: the output at any fault-free node must be some node's input. That is $out_i \in \{in_1, \ldots, in_n\}$ should hold. **(3) Termination**: every fault-free node eventually decides on an output.

We next recall few definitions from [14].

Definition 2.2 (**Reduced Graph** [14])**.** Given a directed graph $\mathbf{G} = (\mathsf{V}, \mathsf{E})$ and a subset $\mathsf{F} \subset \mathsf{V}$, the reduced graph induced by F is $\mathbf{G}_\mathsf{F} = (\mathsf{V}_\mathsf{F}, \mathsf{E}_\mathsf{F})$, where $\mathsf{V}_\mathsf{F} = \mathsf{V} - \mathsf{F}$ and $\mathsf{E}_\mathsf{F} = \mathsf{E} \setminus \{(v_i, v_j) | v_i \in \mathsf{F} \text{ or } v_j \in \mathsf{F}\}$.

Definition 2.3 (**Crash-tolerant Node Connectivity** [14])**.** A graph $\mathbf{G} = (\mathsf{V}, \mathsf{E})$ is said to satisfy k crash-tolerant node connectivity if for any $\mathsf{F} \subset \mathsf{V}$ with $|\mathsf{F}| \leq k$, there is at least one node $s \in \mathsf{V} \setminus \mathsf{F}$ that has a directed path to all the nodes in the corresponding reduced graph $\mathbf{G}_\mathsf{F}$.

Definition 2.4 (**Source of a Reduced Graph [14])**. Let $\mathbf{G} = (\mathsf{V}, \mathsf{E})$ be a graph and let $\mathsf{F} \subset \mathsf{V}$, with $\mathbf{G}_\mathsf{F}$ being the corresponding reduced graph. Then a node v_s in $\mathbf{G}_\mathsf{F}$ is called the source of $\mathbf{G}_\mathsf{F}$ if there exists a directed path from v_s to all the nodes in $\mathbf{G}_\mathsf{F}$.

For a reduced graph $\mathbf{G}_\mathsf{F}$, we denote by $\mathsf{S}_{\mathbf{G}_\mathsf{F}}$ the set of source nodes. The necessity condition for the existence of crash-tolerant consensus in a directed graph is given in Theorem 2.5.

Theorem 2.5 (**Necessary Condition for Crash-Tolerant Consensus [14])**.** *Crash-tolerant consensus tolerating $\mathcal{A}$ is possible in a directed graph $\mathbf{G}$ only if $\mathbf{G}$ has f crash-tolerant node connectivity.*

We end this section with the definition of crash-tolerant diameter d of a directed graph. Informally it denotes the maximum number of rounds over all possible reduced graphs induced by various subsets of size atmost f, within which the message of a potential source node can reach all the remaining nodes in a reduced graph.

Definition 2.6 (Crash-tolerant Diameter [14]). A spanning tree in a directed graph $\mathbf{G} = (\mathsf{V}, \mathsf{E})$ is said to be a rooted spanning tree rooted at a node $v_r \in \mathsf{V}$ if v_r has a directed path to all the nodes in V. Let $tree(v_r, \mathbf{G})$ denote the set of rooted spanning trees, rooted at v_r. We define $height(v_r, \mathbf{G})$ as the minimum height of all the trees $T \in tree(v_r, \mathbf{G})$. That is:

$$height(v_r, \mathbf{G}) = \min_{T \in tree(v_r, \mathbf{G})} (\text{height of T}).$$

The crash-tolerant diameter d is defined as follows:

$$d = \max_{\mathsf{F} \subset \mathsf{V},\, |\mathsf{F}| \leq f} \; \max_{v_s \in \mathsf{S}_{\mathbf{G}_\mathsf{F}}} (height(v_s, \mathbf{G}_\mathsf{F})).$$

Note that in a directed graph with n nodes, d is always upper bounded by n.

2.1 Some Properties of Graphs with f Crash-Tolerant Node Connectivity

In this section we state few properties of reduced graphs which will be used in the rest of the paper. In the rest of this section we consider an arbitrary directed graph $\mathbf{G} = (\mathsf{V}, \mathsf{E})$ which has f crash-tolerant node connectivity. Moreover we consider a scenario where during the execution of an arbitrary protocol, $\mathcal{A}$ crashes a subset of nodes $\mathsf{F} \subset \mathsf{V}$, where $|\mathsf{F}| \leq f$. The corresponding reduced graph is denoted as $\mathbf{G}_\mathsf{F} = (\mathsf{V}_\mathsf{F}, \mathsf{E}_\mathsf{F})$. If $\mathcal{A}$ further crashes additional $|\mathsf{T}|$ nodes in $\mathbf{G}_\mathsf{F}$, with $|\mathsf{F} \cup \mathsf{T}| \leq f$, then the corresponding reduced graph is denoted as $\mathbf{G}_{\mathsf{F}'} = (\mathsf{V}_{\mathsf{F}'}, \mathsf{E}_{\mathsf{F}'})$, where $\mathsf{F}' = \mathsf{F} \cup \mathsf{T}$ denotes the set of nodes, crashed by $\mathcal{A}$ so far. We use the notation $\mathbf{G}_\mathsf{F} \rightarrow \mathbf{G}_{\mathsf{F}'}$ to denote the transition when the additional nodes in T get crashed. The set $\mathsf{S}_{\mathbf{G}_\mathsf{F}}$ and $\mathsf{S}_{\mathbf{G}_{\mathsf{F}'}}$ will denote the set of source nodes for $\mathbf{G}_\mathsf{F}$ and $\mathbf{G}_{\mathsf{F}'}$ respectively. Note that both $\mathsf{S}_{\mathbf{G}_\mathsf{F}}$ and $\mathsf{S}_{\mathbf{G}_{\mathsf{F}'}}$ will be non-empty, as we are assuming $\mathbf{G}$ to have f crash-tolerant node connectivity. Due to space constraints, the proofs of the following properties are available in Appendix A.

The following proposition states that if a node has a directed path to a source node in a reduced graph then the node also is a source node of the reduced graph.

Proposition 2.7. *If a node $v_i \in \mathsf{V}_\mathsf{F}$ has a directed path to any node $v_j \in \mathsf{S}_{\mathbf{G}_\mathsf{F}}$ in $\mathbf{G}_\mathsf{F}$, then $v_i \in \mathsf{S}_{\mathbf{G}_\mathsf{F}}$.*

As an immediate corollary of the above we get the following:

Corollary 2.8. *Let $v_i, v_j \in \mathsf{S}_{\mathbf{G}_\mathsf{F}}$ with $v_i \neq v_j$. Then all intermediate nodes along any directed path[4] between v_i and v_j also belong to $\mathsf{S}_{\mathbf{G}_\mathsf{F}}$.*

We next claim that during the execution of a protocol, a non-source node in a reduced graph cannot become a source node in the next reduced graph, as long as there exists at least one source node in the old reduced graph that is not crashed by the adversary.

Claim 2.9. Consider an arbitrary $v_i \in \mathsf{V}_\mathsf{F}$, such that $v_i \notin \mathsf{S}_{\mathbf{G}_\mathsf{F}}$. Moreover let $v_i \in \mathsf{V}_{\mathsf{F}'}$ (i.e. node v_i is not crashed during the transition $\mathbf{G}_\mathsf{F} \rightarrow \mathbf{G}_{\mathsf{F}'}$). Let there exist at least one node, say $v_j \in \mathsf{S}_{\mathbf{G}_\mathsf{F}}$ that is not crashed by the adversary during the transition $\mathbf{G}_\mathsf{F} \rightarrow \mathbf{G}_{\mathsf{F}'}$ (i.e. $v_j \notin \mathsf{T}$). Then $v_i \notin \mathsf{S}_{\mathbf{G}_{\mathsf{F}'}}$.

Based on the above claim, we next claim that during the execution of a protocol, the source set remains intact in reduced graphs, unless some subset of nodes within the source set crashes.

Claim 2.10. If during the transition $\mathbf{G}_\mathsf{F} \rightarrow \mathbf{G}_{\mathsf{F}'}$ $\mathsf{T} \cap \mathsf{S}_{\mathbf{G}_\mathsf{F}} = \emptyset$, then $\mathsf{S}_{\mathbf{G}_\mathsf{F}} = \mathsf{S}_{\mathbf{G}_{\mathsf{F}'}}$.

Finally we claim that if a source node of $\mathbf{G}_\mathsf{F}$ crashes during the transition $\mathbf{G}_\mathsf{F} \rightarrow \mathbf{G}_{\mathsf{F}'}$, then at least one of the outgoing neighbours of this crashed source node will be the source for the next reduced graph.

Claim 2.11. Let $\mathsf{T} = \{v_s\}$, where $v_s \in \mathsf{S}_{\mathbf{G}_\mathsf{F}}$ (i.e. during $\mathbf{G}_\mathsf{F} \rightarrow \mathbf{G}_{\mathsf{F}'}$ the only node to crash is v_s). Then $\mathsf{N}^+_{v_s} \cap \mathsf{S}_{\mathbf{G}_{\mathsf{F}'}} \neq \emptyset$, where $\mathsf{N}^+_{v_s}$ denotes the set of outgoing neighbours of v_s in the reduced graph $\mathbf{G}_\mathsf{F}$.

[4] Note that a directed path will exist from v_i to v_j and from v_j to v_i in $\mathbf{G}_\mathsf{F}$ as both v_i and v_j are source nodes.

3 Multi-valued Consensus Protocol Based on Min-Max Strategy

Let $\mathbf{G} = (\mathsf{V}, \mathsf{E})$ be a directed graph where $|\mathsf{V}| = n$, such that $\mathbf{G}$ has f crash-tolerant node connectivity. We present a multi-valued crash-tolerant consensus protocol called MinMax (see Figure 1) tolerating $\mathcal{A}$. Similar to the consensus protocols of [14], the algorithm is based on min-max strategy, consisting of $f+2$ phases, with even numbered phases being a min phase while odd numbered phases being a max phase. Each phase further consists of $d+1$ rounds, where d denotes the crash-tolerant diameter of $\mathbf{G}$.

Protocol MinMax($\mathbf{G}$)

The input of the i^{th} node is in_i where $in_i \in [0, K]$. For $i = 1, \ldots, n$, each node v_i executes the following code:

- Repeat for phase $p = 1$ to $f + 2$:
  - If $p \bmod 2 = 0$ then repeat the following steps $d+1$ times (**Min Phase**):
    - Send in_i to all the nodes in $\mathsf{N}^+_{v_i}$.
    - Receive values from the nodes in[a] $\mathsf{N}^-_{v_i}$.
    - Set in_i to the minimum of all the values received.
  - Else repeat the following steps $d+1$ times (**Max Phase**):
    - Send in_i to all the nodes in $\mathsf{N}^+_{v_i}$.
    - Receive values from the nodes in $\mathsf{N}^-_{v_i}$.
    - Set in_i to the maximum of all values received.
- Output $out_i := in_i$ and terminate.

[a] We assume that each node can communicate to itself. Hence v_i also sends its value to itself and receives the same.

Figure 1: Multi-valued crash-tolerant consensus based on min-max strategy

We now prove the properties of MinMax. We first claim that irrespective of the strategy followed by $\mathcal{A}$ during the execution of MinMax, there always exist either two consecutive crash-free phases or a crashed phase with a single crash, followed by a crash-free phase. Formally, let k_i denote the total number of crashes that occur during the i^{th} phase of MinMax, where $k_1 + \ldots + k_{f+2} \leq f$ and each $k_i \in \{0, \ldots, f\}$. Then we have the following lemma.

Lemma 3.1. *Irrespective of the strategy followed by $\mathcal{A}$ during* MinMax*, there exists at least one subsequence $k_{i-1}k_i$ such that either $k_{i-1} = 0, k_i = 0$ or $k_{i-1} = 1, k_i = 0$, where $i \in \{2, \ldots f+2\}$.*

Proof. We prove the lemma using strong induction, over the values of f.

1. Consider the base case where, $f = 1$ and the number of phases are $f + 2 = 3$. The set of all possible sequence $k_1k_2k_3$ is $\{000, 100, 010, 001\}$ and each of them has either the subsequence 00 or 10.

2. Assume the lemma is true for all f where $1 \leq f \leq t-1$; i.e. either the subsequence 00 or 10 occurs among all possible sequence $k_1k_2 \ldots k_{f+2}$, where $f \leq t-1$.

3. Now consider $f = t$. We focus on the last term k_{t+2}, where there are two possible cases:

   - If $k_{t+2} \geq 1$ then it implies that at most $t-1$ faults could occur in the first $t+1$ phases. However by induction hypothesis it follows that irrespective of the adversary strategy, the subsequence 00 or 10 will occur among all possible sequence $k_1k_2 \ldots k_{t+1}$ if at most $t-1$ faults are allowed during $t+1$ phases. This automatically implies that the subsequence 00 or 10 will occur among all possible sequence $k_1k_2 \ldots k_{t+2}$.

– If $k_{t+2} = 0$ then we further focus on the term k_{t+1}. If k_{t+1} has value 1 or 0 we meet our subsequence requirement. However if $k_{t+1} \geq 2$ then it implies that at most $t-2$ faults could occur in the first t phases. However by induction hypothesis it follows that irrespective of the adversary strategy, the subsequence 00 or 10 will occur among all possible sequence $k_1 k_2 \ldots k_t$ if at most $t-2$ faults are allowed during t phases. This automatically implies that the subsequence 00 or 10 will occur among all possible sequence $k_1 k_2 \ldots k_{t+2}$.

□

We next claim that if there are two consecutive crash-free phases then protocol MinMax achieves consensus within those two phases. The idea behind the proof has been already discussed earlier (see section 1). The formal proof is available in Appendix B due to space constraints.

Lemma 3.2. *Let $\mathbf{G}$ have f crash-tolerant node connectivity. If during the execution of* MinMax *there are two consecutive phases, say p_t and p_{t+1}, such that no crash occurs in any of these two phases then consensus is achieved by the end of phase p_{t+1}.*

We next show that consensus will be achieved even if there is crashed phase followed by a crash-free phase, provided only one node crashes during the crashed phase.

Lemma 3.3. *Let $\mathbf{G}$ have f crash-tolerant node connectivity. If during the execution of* MinMax *there are two consecutive phases, say p_t and p_{t+1}, such that only one node crashes during p_t and no node crashes during p_{t+1}, then consensus is achieved by the end of phase p_{t+1}.*

Proof: Without loss of generality let p_t be a min phase and p_{t+1} be a max phase. Let F denote the nodes that have crashed before the phase p_t and let $\mathbf{G}_{\mathsf{F}} = (\mathsf{V}_{\mathsf{F}}, \mathsf{E}_{\mathsf{F}})$ be the reduced graph at the beginning of p_t. Let $\mathbf{G}_{\mathsf{F}} \rightarrow \mathbf{G}_{\mathsf{F}'}$ denote the transition, where a single node in $\mathbf{G}_{\mathsf{F}}$, say v_c, crashes during the phase p_t, resulting in the reduced graph $\mathbf{G}_{\mathsf{F}'}$. Let $\mathsf{S}_{\mathbf{G}_{\mathsf{F}}}$ and $\mathsf{S}_{\mathbf{G}_{\mathsf{F}'}}$ denote the set of source nodes for the reduced graphs $\mathbf{G}_{\mathsf{F}}$ and $\mathbf{G}_{\mathsf{F}'}$ respectively. If the crashed node $v_c \notin \mathsf{S}_{\mathbf{G}_{\mathsf{F}}}$ then the proof of the lemma is exactly the same as Lemma 3.2, as in this case $\mathsf{S}_{\mathbf{G}_{\mathsf{F}}} = \mathsf{S}_{\mathbf{G}_{\mathsf{F}'}}$ (follows from Claim 2.10). So we next consider the case when $v_c \in \mathsf{S}_{\mathbf{G}_{\mathsf{F}}}$. We have two further sub-cases:

- *If the node v_c crashes during the first round of p_t and without propagating its value to any node in $\mathsf{S}_{\mathbf{G}_{\mathsf{F}'}}$*: in this case we can effectively ignore the effect of the initial source set $\mathsf{S}_{\mathbf{G}_{\mathsf{F}}}$. Moreover, from the second round onward of p_t, each node in the source set $\mathsf{S}_{\mathbf{G}_{\mathsf{F}'}}$ will get the required d rounds of communication to propagate their value to every other node in $\mathbf{G}_{\mathsf{F}'}$ during p_t. Furthermore, during p_{t+1}, the source set remains the same as $\mathsf{S}_{\mathbf{G}_{\mathsf{F}'}}$. In essence, this is equivalent to as if p_t and p_{t+1} are executed over the reduced graph $\mathbf{G}_{\mathsf{F}'}$ with at least d rounds of communication, with the source set being $\mathsf{S}_{\mathbf{G}_{\mathsf{F}'}}$ and with no crash occurring in these phases. So using exactly the same arguments as in Lemma 3.2, we can conclude that consensus will be achieved by the end of phase p_{t+1}.

- *If the node v_c crashes after propagating its value to at least one node in $\mathsf{S}_{\mathbf{G}_{\mathsf{F}'}}$*: let λ denote the value of v_c at the beginning of p_t. We first note that there exists at least one source node, say $v_{s'} \in \mathsf{S}_{\mathbf{G}_{\mathsf{F}'}}$, such that $v_{s'}$ receives λ from v_c at the end of the first round of p_t. This is because the node $v_{s'}$ will be an outgoing neighbour of the node v_c in $\mathbf{G}_{\mathsf{F}}$ (this follows from Claim 2.11). Moreover, the node $v_{s'}$ can propagate λ to all the remaining nodes in $\mathbf{G}_{\mathsf{F}'}$ during p_t. This follows from the definition of d and the fact that p_t has still d rounds of communication left.

  Let $\lambda^{\mathsf{min}}_{\mathbf{G}'}$ denote the minimum value among the nodes in $\mathsf{S}_{\mathbf{G}_{\mathsf{F}'}}$ at the beginning of p_t and let $\lambda^{\mathsf{min}} = \min(\lambda, \lambda^{\mathsf{min}}_{\mathbf{G}'})$. We claim that at the end of p_t, all the nodes in $\mathsf{S}_{\mathbf{G}_{\mathsf{F}'}}$ will have λ^{min} as their updated value. This is because no node in $\mathsf{S}_{\mathbf{G}_{\mathsf{F}'}}$ will ever see a value smaller than λ^{min} being propagated. Next we claim that all the non-source nodes in $\mathbf{G}_{\mathsf{F}'}$ will have their value updated to λ^{min} or a smaller value

at the end of p_t. More specifically, if the non-source nodes in $\mathbf{G}_{\mathsf{F}'}$ have their value greater than λ^{min} at the beginning of phase p_t, then all these non-source nodes will set λ^{min} as their updated value at the end of p_t and consensus will be achieved at the end of p_t. This is because λ^{min} will be propagated to all these nodes. On the other hand, if some non-source node has a value smaller than λ^{min} at the beginning of p_t, then the node will set a value smaller than λ^{min} as its updated value at the end of p_t. In this case, at the beginning of p_{t+1}, the value λ^{min} will be maximum value of any node. Moreover, all the nodes in $\mathsf{S}_{\mathbf{G}_{\mathsf{F}'}}$ will have λ^{min} as their value. Since p_{t+1} is a max phase and no crash occurs during p_{t+1}, it follows from the definition of d that λ^{min} will be propagated to all the nodes in $\mathbf{G}_{\mathsf{F}'}$ and every node will set λ^{min} as their updated value at the end of p_{t+1}, thus attaining consensus. □

The following theorem follows from Lemma 3.1-3.3 and the fact that every node will terminate the protocol after $(f+2)(d+1)$ rounds. In every round, each node has to send $\log|K|$ bits to all its outgoing neighbours and there are $\mathcal{O}(n)$ outgoing neighbours of every node; this proves the communication complexity.

Theorem 3.4. *Let $\boldsymbol{G} = (\mathsf{V}, \mathsf{E})$ be a directed graph with f crash-tolerant node connectivity and crash-tolerant diameter d, where $|\mathsf{V}| = n$. Then protocol* MinMax *is a $(f+2)(d+1)$ round protocol for multi-valued consensus tolerating $\mathcal{A}$. The protocol has communication complexity $\mathcal{O}(nfd\log|K|)$ bits.*

Further optimization in the round complexity of MinMax**:** In Appendix C we present a modified version of MinMax called MinMax$''$, where we allow the first phase and the last phase to have exactly d rounds; the remaining f phases still consist of $d+1$ rounds of communication. Hence the total round complexity is $(f+2)(d+1)-2$. We show that MinMax$''$ still achieves consensus. The idea is as follows: we first consider a variation MinMax$'$ of MinMax, where only the first phase is restricted to d rounds and show that MinMax$'$ still achieves consensus within $f+2$ phases. This is argued depending upon whether the first phase of MinMax$'$ is crash-free or not. If it is crash-free, then the execution of MinMax$'$ is "equivalent" to that of MinMax, where there are $d+1$ rounds in the first phase and where the first phase is crash-free. On the other hand, if the first phase of MinMax$'$ is not crash-free, then in the remaining $f+1$ phases (each of which has $d+1$ rounds), at most $f-1$ crashes can occur. Now using Lemma 3.1, we can say that in these $f+1$ phases, either there will be at least two consecutive crash-free phases or a crashed phase with a single crash followed by a crash-free phase. So consensus will be achieved by the end of $(f+2)th$ phase. We then consider protocol MinMax$''$, which is a variation of MinMax$'$ in that the last phase is now restricted to d rounds. Now again depending upon whether the last phase of MinMax$''$ is crash-free or not we show that consensus will be achieved by MinMax$''$.

4 Lower Bounds on Round Complexity of Consensus Protocols Based on Min-Max strategy

In this section we consider crash-tolerant consensus protocols based on min-max strategy, consisting of alternate min and max phases and show few impossibility results regarding the minimum number of rounds required by consensus protocols. Based on these results, we conclude that our protocol MinMax requires *optimal* number of communication rounds. We assume protocols which have alternate min-max phases, where the first phase is a min phase (this is without loss of generality).

4.1 Impossibility of Consensus in $f+1$ Phases (Irrespective of the Number of Rounds)

Consider the family of directed graphs with f-crash-tolerant node connectivity such that every graph $\mathbf{G} = (\mathsf{V}, \mathsf{E})$ has the following properties (see Figure 2): $\mathsf{V} = \{v_1, v_2, ..., v_{f+3}\}$. The edge set $\mathsf{E} = \{(v_i, v_j)\}$, where $i < j$ and $1 \leq i \leq f+2$. Clearly $|\mathsf{E}| = \frac{((f+3)^2-(f+3))}{2} - 1$. The graph has exactly two "sink" nodes

v_{f+2}, v_{f+3}, which do not have any outgoing edges. Node v_1 has input value 1, while nodes $v_2, \ldots, v_{f+3}$ have input value 0. The graph has $d = 1$.

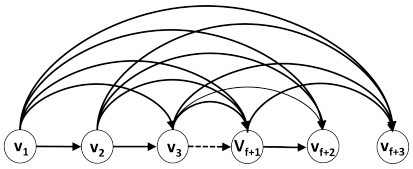


Figure 2: The family of graphs in which it is impossible to achieve consensus in $f + 1$ phases.

Let Π_{MinMax} be an arbitrary protocol for $\mathbf{G}$, consisting of $f + 1$ alternate min-max phases, where the first phase is a min phase. Moreover, each phase has *at least* d rounds of communication[5]. We consider the following adversary strategy $\mathcal{A}_{\mathsf{MinMax}}$ by $\mathcal{A}$ during Π_{MinMax}: No crash occurs during phase p_1. Adversary crashes node v_i during phase p_{i+1} for $i = 1, \ldots, f$ in the following fashion: during the first round of p_{i+1}, the node v_i sends its value to nodes $v_{i+2}, \ldots, v_{f+3}$ and crashes. Hence, except v_{i+1}, all the neighbours of v_i receive v_i's value. The adversarial strategy ensures the following: if p_{i+1} is a min (resp. max) phase, then v_i will be the source node at the beginning of p_{i+1} with value 0 (resp. 1), while all the remaining nodes $v_{i+1}, \ldots, v_{f+3}$ will have value 1 (resp. 0). At the end of p_{i+1}, the node v_{i+1} will be the source node with value 1 (resp. 0), while all the remaining nodes $v_{i+2}, \ldots, v_{f+3}$ will have value 0 (resp. 1). Hence at the end of each min (resp. max) phase, all the nodes in the graph except the source will have value 0 (resp. 1). So at the end of Π_{MinMax}, the reduced graph will have nodes v_{f+1}, v_{f+2} and v_{f+3}, with v_{f+1} being the source and where v_{f+2} and v_{f+3} will have values, different from v_{f+1}. The formal proof is available in Appendix D.

Theorem 4.1. *Consensus will not be achieved in* $\mathbf{G}$ *(Figure 2) by* Π_{MinMax} *against the strategy* $\mathcal{A}_{\mathsf{MinMax}}$.

4.2 Impossibility of Consensus with $(f + 2)(d + 1) - 3$ Rounds in Total

Here we consider min-max protocols with $f+2$ phases and $(f+2)(d+1)-3$ rounds in total, with each phase having at least d rounds. We present a family of directed graphs and a corresponding adversarial strategy against which consensus will not be achieved at the end of $f + 2$ phases. This shows that the minimum number of rounds required is $(f + 2)(d + 1) - 2$, implying that our protocol MinMax''(optimized variant of MinMax) is *round optimal*. Note that the adversarial strategy $\mathcal{A}_{\mathsf{MinMax}}$ and the graph of Fig 2 cannot be used to derive the lower bound. This is because if we allow $f + 2$ phases then consensus will be achieved in the graph of Fig 2 against $\mathcal{A}_{\mathsf{MinMax}}$. Hence we need to modify the graph and also the adversary strategy. The detailed proof can be found in Appendix E.

4.3 Impossibility of Consensus based on Min-Max Strategy in $2f + 1$ Phases with d Rounds

Here we consider protocols based on min-max strategy, consisting of $2f + 1$ alternate min and max phases, with each phase having d rounds of communication. For several values of f and d, we show that there exist graphs for which it is impossible to reach consensus within $2f+1$ phases. The adversary strategy in all these graphs will be the following: without loss of generality, we will assume that the first phase is a min phase. Hence there will be $f + 1$ min phases and f max phases. There will be *no* crash during the min phases and during each max phase, *one new* node will get crashed by the adversary. Moreover, the node to be crashed will be the *source* node of the reduced graph at the beginning of that phase. It will be always ensured that every reduced graph has *only one* source node. It will be ensured that no consensus is achieved during any of the min phases. This is achieved by ensuring that at the beginning of each min phase, the source has value 1 and there exists at least one non-source node with value 0. As a result, at the end of each min phase, the source will retain 1 as its value (as it will never see the value 0 during the min phase because there will be only one source), while there will be at least one node, which retains 0 as its value. During each of the

[5]The number of rounds in each phase need to be finite so that Π_{MinMax} should terminate for each node.

f max phases, the adversary will crash the source node. The crashed node will crash during the first round of a max phase, after sending its value 1 *only* to the new source of the reduced graph. The new source will further get $d-1$ rounds to propagate the value 1 that it received from the crashed source. However, it will be ensured that that there is some node with value 0 which is d distance apart from the new source, such that it never sees the value 1 during the max phase and as a result, it retains its original value 0, thus preventing consensus being achieved during the crashed max phase.

Presenting a generalized graph which maintains the above properties for a general value of f and d is an extremely challenging task. But we believe that indeed it is possible to come up with a graph and corresponding adversary strategy for any arbitrary value of f and d. In Appendix F, we present graphs for several values of f and d, where consensus is achieved only at the end of $2f+2$ phases.

References

[1] E. Alchieri, A. N. Bessani, J. S. Fraga, and F. Greve. Byzantine Consensus with Unknown Participants. In *OPODIS*, pages 22–40, 2008.

[2] H. Attiya and J. Welch. *Distributed Computing: Fundamentals, Simulation and Advanced Topics*. Wiley series on Parallel and Distributed Computing, 2004.

[3] P. Bansal, P. Gopal, A. Gupta, K. Srinathan, and P. K. Vasishta. Byzantine Agreement Using Partial Authentication. In *DISC*, pages 389–403, 2011.

[4] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Optimization and Neural Computation Series. Athena Scientific, 1997.

[5] B. Charron-Bost, M. Függer, and T. Nowak. Approximate Consensus in Highly Dynamic Networks: The Role of Averaging Algorithms. In *ICALP*, pages 528–539, 2015.

[6] D. Dolev. The Byzantine Generals Strike Again. *Journal of Algorithms*, 3(1):14–30, 1982.

[7] M. J. Fischer, N. A. Lynch, and M. Merritt. Easy Impossibility Proofs for Distributed Consensus Problems. In *PODC*, pages 59–70. ACM, 1985.

[8] M. Fitzi. *Generalized Communication and Security Models in Byzantine Agreement*. PhD thesis, ETH Zurich, 2002.

[9] A. Jadbabaie, J. Lin, and A. S. Morse. Coordination of Groups of Mobile Autonomous Agents Using Nearest Neighbor Rules. *IEEE Trans. Automat. Contr.*, 48(6):988–1001, 2003.

[10] N. A. Lynch. *Distributed Algorithms*. Morgan Kaufmann, 1996.

[11] M. Pease, R. E. Shostak, and L. Lamport. Reaching Agreement in the Presence of Faults. *JACM*, 27(2):228–234, 1980.

[12] U. Schmid, B. Weiss, and I. Keidar. Impossibility Results and Lower Bounds for Consensus under Link Failures. *SIAM J. Comput.*, 38(5):1912–1951, 2009.

[13] L. Tseng. Recent Results on Fault-Tolerant Consensus in Message-Passing Networks. In *SIROCCO*, volume 9988 of *Lecture Notes in Computer Science*, pages 92–108, 2016.

[14] L. Tseng and N. H. Vaidya. Crash-Tolerant Consensus in Directed Graphs. *CoRR*, abs/1412.8532, 2014. Full version appeared as [15].

[15] L. Tseng and N. H. Vaidya. Fault-Tolerant Consensus in Directed Graphs. In *PODC*, pages 451–460. ACM, 2015.

[16] L. Tseng and N. H. Vaidya. A Note on Fault-tolerant Consensus in Directed Networks. *SIGACT News*, 47(3):70–91, 2016.

A Proof of the Properties of Reduced Graphs

Proposition 2.7: If a node $v_i \in \mathsf{V_F}$ has a directed path to any node $v_j \in \mathsf{S_{G_F}}$ in $\mathbf{G_F}$, then $v_i \in \mathsf{S_{G_F}}$.

Proof. Since $v_j \in \mathsf{S_{G_F}}$ is a source, it has a directed path to every node in $\mathbf{G_F}$. Hence if v_i has a directed path to v_j, it implies that v_i has a directed path as well to every node in $\mathbf{G_F}$ through v_j. □

Claim 2.9: Consider an arbitrary $v_i \in \mathsf{V_F}$, such that $v_i \notin \mathsf{S_{G_F}}$. Moreover $v_i \in \mathsf{V_{F'}}$ (i.e. node v_i is not crashed during the transition $\mathbf{G_F} \rightarrow \mathbf{G_{F'}}$). Let there exist at least one node, say $v_j \in \mathsf{S_{G_F}}$ that is not crashed by the adversary during the transition $\mathbf{G_F} \rightarrow \mathbf{G_{F'}}$ (i.e. $v_j \notin \mathsf{T}$). Then $v_i \notin \mathsf{S_{G_{F'}}}$.

Proof. We prove the claim by contradiction. Let $v_i \notin \mathsf{S_{G_F}}$ be a non-source node in $\mathbf{G_F}$ and let $v_i \in \mathsf{S_{G_{F'}}}$ be a source node in $\mathbf{G_{F'}}$, while there still exist a source-node $v_j \in \mathsf{S_{G_F}}$ in $\mathbf{G_F}$ that has not crashed during the transition $\mathbf{G_F} \rightarrow \mathbf{G_{F'}}$. Since $v_i \in \mathsf{S_{G_{F'}}}$, node v_i must have a directed path to node v_j in $\mathbf{G_{F'}}$. This path would exist in $\mathbf{G_F}$ as well since during the transition $\mathbf{G_F} \rightarrow \mathbf{G_{F'}}$, no new edges are added. Thus by proposition 2.7, $v_i \in \mathsf{S_{G_F}}$ as well, which is a contradiction. □

Claim 2.10: If during the transition $\mathbf{G_F} \rightarrow \mathbf{G_{F'}}$, $\mathsf{T} \cap \mathsf{S_{G_F}} = \emptyset$, then $\mathsf{S_{G_F}} = \mathsf{S_{G_{F'}}}$.

Proof. We prove the claim by contradiction. Assume $\mathsf{S_{G_F}} \neq \mathsf{S_{G_{F'}}}$. We know by Claim 2.9 that there cannot exist a node v_i such that $v_i \notin \mathsf{S_{G_F}}$ but $v_i \in \mathsf{S_{G_{F'}}}$, unless $\mathsf{S_{G_F}} \subseteq \mathsf{T}$. However, this cannot hold since it is given that $\mathsf{T} \cap \mathsf{S_{G_F}} = \emptyset$. Therefore, for $\mathsf{S_{G_F}} \neq \mathsf{S_{G_{F'}}}$ to hold, there must exist a node, say $v_i \in \mathsf{S_{G_F}}$ such that $v_i \notin \mathsf{S_{G_{F'}}}$, even though $\mathsf{T} \cap \mathsf{S_{G_F}} = \emptyset$. We next show that this is not possible. For this we consider two cases, depending upon the size of $\mathsf{S_{G_F}}$.

- *Case 1: If* $|\mathsf{S_{G_F}}| > 1$: Let $v_j \in \mathsf{S_{G_{F'}}}$ (note that such a v_j exists as we are considering graphs with f crash-tolerant node connectivity). Since $v_i \in \mathsf{S_{G_F}}$ still exists in $\mathbf{G_{F'}}$, it follows from claim 2.9 that $v_j \in \mathsf{S_{G_F}}$ as well. The path from v_i to v_j in $\mathbf{G_F}$ must consist only of nodes in $\mathsf{S_{G_F}}$ (By Corollary 2.8). Since the set of nodes T that crashed during transition is such that $\mathsf{T} \cap \mathsf{S_{G_F}} = \emptyset$, the path from v_i to v_j still exists in $\mathbf{G_{F'}}$ as well. So by Proposition 2.7, $v_i \in \mathsf{S_{G_{F'}}}$, which is a contradiction.
- *Case 2: If* $|\mathsf{S_{G_F}}| = 1$*, comprising of the node* v_i *alone*: By the guarantee of f crash-tolerant node connectivity, there must exist at least one source node, say $v_j \in \mathsf{S_{G_{F'}}}$. Moreover $v_i \neq v_j$ (as we are assuming $v_i \notin \mathsf{S_{G_{F'}}}$). Since $v_i \in \mathsf{S_{G_F}}$ has not crashed during the transition, it follows from claim 2.9 that $v_j \in \mathsf{S_{G_F}}$. Thus both $v_i, v_j \in \mathsf{S_{G_F}}$ which contradicts the assumption that $|\mathsf{S_{G_F}}| = 1$.

□

Claim 2.11: Let $\mathsf{T} = \{v_s\}$, where $v_s \in \mathsf{S_{G_F}}$ (i.e. during the transition $\mathbf{G_F} \rightarrow \mathbf{G_{F'}}$ the only node to crash is a source node v_s). Then $\mathsf{N}^+_{v_s} \cap \mathsf{S_{G_{F'}}} \neq \emptyset$, where $\mathsf{N}^+_{v_s}$ denotes the set of outgoing neighbours of v_s in

the reduced graph $\mathbf{G}_{\mathsf{F}}$.

Proof. By the f crash-tolerant node connectivity guarantee, after the described transition, there exists at least one source node, say $v_{s'}$ in $\mathbf{G}_{\mathsf{F}'}$; i.e. $v_{s'} \in \mathsf{S}_{\mathbf{G}_{\mathsf{F}'}}$. Also, since $v_s \in \mathsf{S}_{\mathbf{G}_{\mathsf{F}}}$ it must have a path to $v_{s'}$ in $\mathbf{G}_{\mathsf{F}}$. If there exist a direct edge from v_s to $v_{s'}$ in $\mathbf{G}_{\mathsf{F}}$ (i.e $v_{s'} \in \mathsf{N}^+_{v_s}$), then $\mathsf{N}^+_{v_s} \cap \mathsf{S}_{\mathbf{G}_{\mathsf{F}'}} \neq \emptyset$. On the other hand, if there exists a directed path $v_s \to v_i \to \ldots \to v_{s'}$ in $\mathbf{G}_{\mathsf{F}}$, where there is a direct edge from v_s to v_i in $\mathbf{G}_{\mathsf{F}}$, then by proposition 2.7 we have $v_i \in \mathsf{S}_{\mathbf{G}_{\mathsf{F}'}}$. Thus in this case also $\mathsf{N}^+_{v_s} \cap \mathsf{S}_{\mathbf{G}_{\mathsf{F}'}} \neq \emptyset$ holds. □

B Properties of the Protocol MinMax

Lemma 3.2: Let $\mathbf{G}$ have f crash-tolerant node connectivity. If during the execution of MinMax there are two consecutive phases, say p_t and p_{t+1}, such that no crash occurs in any of these two phases then consensus is achieved by the end of phase p_{t+1}.

Proof. Without loss of generality let p_t be a min phase and p_{t+1} be a max phase. Let F be the set of nodes that have crashed before phase p_t. So p_t and p_{t+1} are executed over the reduced graph $\mathbf{G}_{\mathsf{F}} = (\mathsf{V}_{\mathsf{F}}, \mathsf{E}_{\mathsf{F}})$ with the set of source nodes $\mathsf{S}_{\mathbf{G}_{\mathsf{F}}}$. Note that the source set does not change during these two phases since no crash occurs. Let $\lambda^{\min}$ denote the minimum value among the nodes in $\mathsf{S}_{\mathbf{G}_{\mathsf{F}}}$ at the beginning of p_t and let $\mathsf{S}^{\min}_{\mathbf{G}_{\mathsf{F}}}$ be the set of source nodes possessing the minimum value $\lambda^{\min}$. Now there are two possible cases:

- *All the nodes in the set $\mathsf{V}_{\mathsf{F}} \setminus \mathsf{S}^{min}_{\mathbf{G}_{\mathsf{F}}}$ have values which are greater that or equal to λ^{min} at the beginning of p_t*: In this case, at the end of p_t, all the nodes in V_{F} will have their value equal to $\lambda^{\min}$. This is because p_t is a min phase and by the definition of crash-tolerant diameter, the value $\lambda^{\min}$ will propagate to each node in V_{F} within d rounds of communication. Hence in this case, consensus is achieved at the end of p_t.
- *At least one node in the set $\mathsf{V}_{\mathsf{F}} \setminus \mathsf{S}_{\mathbf{G}_{\mathsf{F}}}$ has value less than λ^{min} at the beginning of p_t*: In this case, all the nodes in the source set $\mathsf{S}_{\mathbf{G}_{\mathsf{F}}}$ will have their value set to $\lambda^{\min}$ at the end of p_t. This is because no node in the source set $\mathsf{S}_{\mathbf{G}_{\mathsf{F}}}$ will ever see a value smaller than $\lambda^{\min}$ being propagated during p_t, otherwise it contradicts the assumption that $\lambda^{\min}$ is the minimum value among the nodes in $\mathsf{S}_{\mathbf{G}_{\mathsf{F}}}$ at the beginning of p_t. Moreover all the nodes in the set $\mathsf{V}_{\mathsf{F}} \setminus \mathsf{S}_{\mathbf{G}_{\mathsf{F}}}$ will have their values set to a value which is less than or equal to $\lambda^{\min}$ at the end of p_t. This is because p_t is a min phase and the minimum value propagated to any node in $\mathsf{V}_{\mathsf{F}} \setminus \mathsf{S}_{\mathbf{G}_{\mathsf{F}}}$ within d rounds of communication will be either $\lambda^{\min}$ or a value less that it. This implies that at the beginning of the next phase p_{t+1}, the value $\lambda^{\min}$ will be the maximum value of any node. Since all the nodes in $\mathsf{S}_{\mathbf{G}_{\mathsf{F}}}$ have $\lambda^{\min}$ as their value during p_{t+1} and since p_{t+1} is a max phase, no node will ever see a value greater than $\lambda^{\min}$ being propagated during p_{t+1}. Moreover by the definition of d, the value $\lambda^{\min}$ will be propagated to every non-source node during p_{t+1}. Hence at the end of p_{t+1} all the nodes in $\mathbf{G}_{\mathsf{F}}$ will set their values to $\lambda^{\min}$, thus achieving consensus at the end of p_{t+1}.

□

C Optimizing the Round Complexity of MinMax

In this section, we present a slightly modified version of the MinMax protocol, called MinMax$''$ that reduces the round complexity by an additional two rounds. MinMax$''$ is similar to MinMax except with the following difference: The first phase p_1 and the $(f+2)th$ phase p_{f+2} comprises of d rounds as opposed to $(d+1)$ rounds in MinMax. The following set of lemmas prove that MinMax$''$ achieves consensus.

We first consider a variation MinMax$'$ of MinMax, where only the first phase is restricted to d rounds. We show that irrespective of the adversary strategy, consensus will be achieved by MinMax$'$ within $f+2$ phases, for any value of f.

Lemma C.1. *Let G be a directed graph with f' crash-tolerant node connectivity and crash-tolerant diameter d. Let* MinMax$'$ *be a protocol consisting of $f'+2$ alternate min-max phases, where the first phase has d rounds and the remaining phases have $d+1$ rounds. If at most f' crashes occur during* MinMax$'$*, then irrespective of the adversary strategy, consensus will be achieved at the end of* MinMax$'$*.*

Proof. Let the sequence $k'_1 k'_2 \dots k'_{f'+2}$ denote the faults occurred during the $f'+2$ phases of MinMax$'$, where each $k'_i \in \{0, \dots, f'\}$ and $k'_1 + \dots + k'_{f'+2} \leq f'$. Now consider the value of k'_1. Depending upon the behaviour of the adversary, we have the following two cases:

- $k'_1 = 0$: In this case, the execution of MinMax$'$ is equivalent to an execution of MinMax, where the first phase is crash-free. This follows from the definition of crash-tolerant diameter d since d rounds suffice for a source's value to propagate to all nodes in a graph. The remaining execution of MinMax$'$ (namely the remaining $f'+1$ phases) will be exactly the same as in MinMax. Now substituting $f = f'$ in Theorem 3.4, it follows that consensus will be achieved at the end of MinMax$'$.

- $k'_1 \geq 1$: This further implies that at most $(f'-1)$ crashes can occur during the remaining $(f'+1)$ phases, namely $p_2, \dots, p_{f'+2}$. Each of the remaining $(f'+1)$ phases have $d+1$ rounds. Moreover, the difference between the number of remaining phases and the total number of faults that can occur in these remaining phases is atleast two. Hence from Lemma 3.1 there will be either two consecutive crash-free phases or a crashed phase with a single crash followed by a crash-free phase among $p_2, \dots, p_{f'+2}$. More specifically, we consider $f = f'-1$ and rename the remaining $f'+1 = f+2$ phases $\{p_2 \dots p_{f'+2}\}$ as $\{\mathbf{p_1}, \dots, \mathbf{p_{f+2}}\}$, with k_i denoting the number of faults that can occur during $\mathbf{p_i}$, where $k_i \in \{0, \dots, f\}$ and $k_1 + \dots + k_{f+2} \leq f$. It follows via Lemma 3.1 that there exists at least one subsequence $k_{i-1}k_i$ such that either $k_{i-1} = 0, k_i = 0$ or $k_{i-1} = 1, k_i = 0$. Now, it follows directly from lemmas 3.2 and 3.3 that MinMax$'$ achieves consensus.

□

We next consider a variation MinMax$''$ of MinMax$'$, where the last phase is restricted to d rounds (note that the first phase is implicitly restricted to d rounds). We show that irrespective of the adversary strategy, consensus will be achieved by MinMax$''$ within $f+2$ phases, for any value of f.

Lemma C.2. *Let G be a directed graph with f'' crash-tolerant node connectivity and crash-tolerant diameter d. Let* MinMax$''$ *be a protocol consisting of $f''+2$ alternate min-max phases, where the first and the last phase has d rounds and the remaining phases have $d+1$ rounds. If at most f'' crashes occur during* MinMax$''$*, then irrespective of the adversary strategy, consensus will be achieved at the end of* MinMax$''$*.*

Proof. Let the sequence $k''_1 k''_2 \dots k''_{f''+2}$ denote the faults caused by the adversary during the $f''+2$ phases of MinMax$''$, where each $k''_i \in \{0, \dots, f''\}$ and $k''_1 + \dots + k''_{f''+2} \leq f''$. Now consider the value of $k''_{f''+2}$. Depending upon the behaviour of the adversary, we have the following two cases:

- $k''_{f''+2} = 0$: In this case, the execution of MinMax$''$ is equivalent to an execution of MinMax$'$, where the last phase is crash-free. This follows from the definition of crash-tolerant diameter d since d rounds suffice for a source's value to propagate to all nodes in a graph. The remaining execution of MinMax$''$ (namely the first $f''+1$ phases) will be exactly the same as in MinMax$'$. Now substituting $f' = f''$ in Lemma C.1, it follows that consensus will be achieved at the end of MinMax$''$.

- $k_{f''+2} \geq 1$: This further implies that at most $(f''-1)$ crashes could occur during the first $(f''+1)$ phases, namely $p_1, \ldots, p_{f''+1}$. Moreover, p_1 will have d rounds, while $p_2, \ldots, p_{f''+1}$ will have $d+1$ rounds each. This is equivalent to an execution of MinMax$'$ protocol with at most $f' = f''-1$ crashes and $f'+2 = f''+1$ phases $\{p_1, \ldots p_{f''+1}\}$. It now follows from lemma C.1 that MinMax$''$ achieves consensus by the end of $p_{f''+1}$.

□

The following theorem directly follows from Lemma C.2 by substituting $f'' = f$ and Theorem 3.4.

Theorem C.3. *Let $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$ be a directed graph with f crash-tolerant node connectivity and crash-tolerant diameter d, where $|\boldsymbol{V}| = n$. Then protocol* MinMax$''$ *is a $(f+2)(d+1)-2$ round protocol for multi-valued consensus tolerating $\mathcal{A}$. The protocol has communication complexity $\mathcal{O}(nfd \log |K|)$ bits.*

D Proof of Impossibility of Consensus based on Min-Max Strategy in $f+1$ Phases

Theorem 4.1: Consensus will not be achieved in **G** (Figure 2) by Π_{MinMax} against the strategy $\mathcal{A}_{\mathsf{MinMax}}$.

Proof. To prove the lemma, we consider each phase of Π_{MinMax}:

- At the end of p_1, there will no change in the values of the node in **G**. This is because this is a min phase. As node v_1 will only see value 1 throughout p_1, its value will remain 1. While $v_2, \ldots, v_{f+3}$ will retain 0 as their value, even though they see both 0 and 1 during p_1.

- At the end of p_2, the reduced graph $\mathbf{G}_{\mathsf{F}_1}$ will consist of nodes $v_2, \ldots, v_{f+3}$, with v_2 being the source node, where $\mathsf{F}_1 = \{v_1\}$ is the set of nodes crashed till now. Moreover, v_2 will retain 0 as its value, while $v_3, \ldots, v_{f+3}$ will set 1 as their value. This is because, as per the strategy of $\mathcal{A}_{\mathsf{MinMax}}$, the crashed node v_1 sends 1 to all the nodes in $\mathbf{G}_{\mathsf{F}_1}$, except v_2 and since it is a max phase, the nodes $v_3, \ldots, v_{f+3}$ will set 1 as their input. However, v_2 will not see any value other than 0 throughout p_2.

- At the end of p_3, the reduced graph $\mathbf{G}_{\mathsf{F}_2}$ will consist of nodes $v_3, \ldots, v_{f+3}$, with v_3 being the source node, where $\mathsf{F}_1 = \{v_1, v_2\}$ is the set of nodes crashed till now. Moreover, v_3 will retain 1 as its value, while $v_4, \ldots, v_{f+3}$ will set 0 as their value. This is because, as per the strategy of $\mathcal{A}_{\mathsf{MinMax}}$, the crashed node v_2 sends 0 to all the nodes in $\mathbf{G}_{\mathsf{F}_2}$, except v_3 and since it is a min phase, the nodes $v_4, \ldots, v_{f+3}$ will set 0 as their value. However, v_3 will not see any value other than 1 throughout p_3.

- In general, if f is odd, then the last crashed phase p_{f+1} will be a max phase. The reduced graph $\mathbf{G}_{\mathsf{F}}$ will consist of nodes v_{f+1}, v_{f+2} and v_{f+3}, with values 0, 1 and 1 respectively; here $\mathsf{F} = \{v_1, \ldots, v_f\}$ denote the set of crashed nodes till the end of p_{f+1}. Hence no consensus will be achieved among v_{f+1}, v_{f+2} and v_{f+3} at the end of Π_{MinMax}. For a demonstration of the execution Π_{MinMax} against $\mathcal{A}_{\mathsf{MinMax}}$ for odd value of f, see Figure 3.

  On the other hand, if f is even, then the last crashed phase p_{f+1} will be a min phase. The reduced graph $\mathbf{G}_{\mathsf{F}}$ will consist of nodes v_{f+1}, v_{f+2} and v_{f+3}, with values 1, 0 and 0 respectively. Hence no consensus will be achieved among v_{f+1}, v_{f+2} and v_{f+3} at the end of Π_{MinMax}. For a demonstration of the execution Π_{MinMax} against $\mathcal{A}_{\mathsf{MinMax}}$ for even value of f, see Figure 4.

□

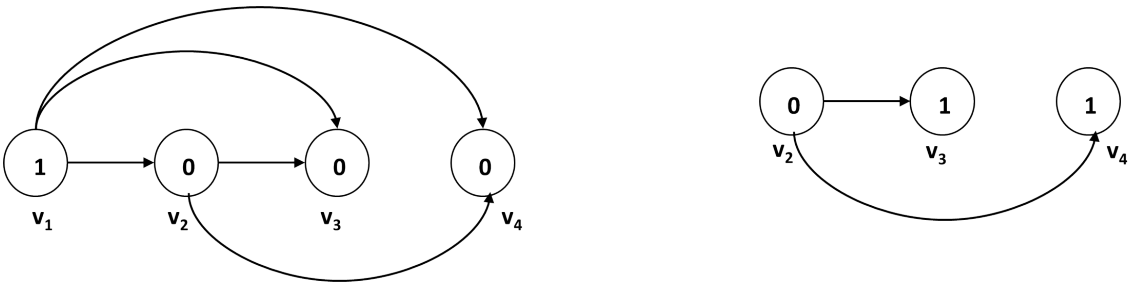


Figure 3: Execution of Π_{MinMax} in $\mathbf{G}$ against $\mathcal{A}_{\mathsf{MinMax}}$ for $f = 1$. The first figure denotes the values of the nodes at the beginning of p_1. No crash occurs and the values of the nodes remain the same at the end of p_1. The second figure denotes the reduced graph due to the crash of v_1 during p_2, with the values of the nodes at the end of p_2.

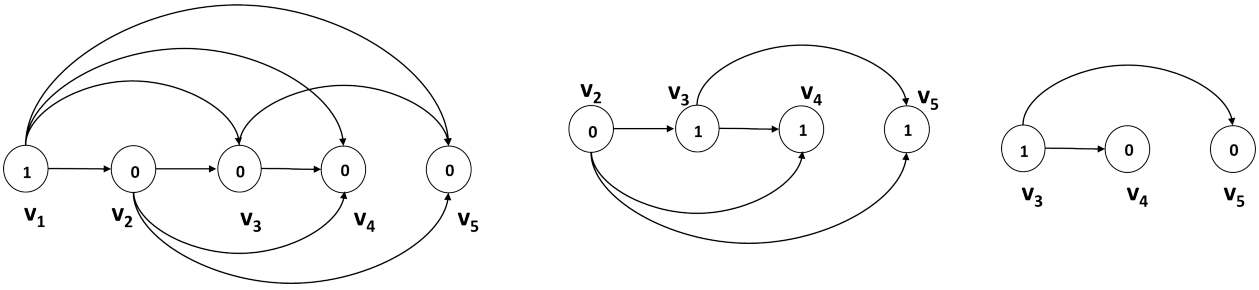


Figure 4: Execution of Π_{MinMax} in $\mathbf{G}$ against $\mathcal{A}_{\mathsf{MinMax}}$ for $f = 2$. The first figure denotes the values of the nodes at the beginning of p_1. No crash occurs and the values of the nodes remain the same at the end of p_1. The second figure denotes the reduced graph due to the crash of v_1 during p_2, with the values of the nodes at the end of p_2. The third figure denotes the reduced graph due to the crash of v_2 during p_3, with the values of the nodes at the end of p_3.

E Impossibility of Consensus with $(f+2)(d+1)-3$ Rounds in Total

Π is an arbitrary min-max based protocol with $f+2$ phases $p_1 \ldots, p_{f+2}$ and $(f+2)(d+1)-3$ rounds, with each phase having at least d rounds. We first derive few properties of Π, based on counting arguments. To begin with we claim that in Π, there exist *at least* three phases, each with *exactly* d rounds of communication.

Lemma E.1. *In protocol* Π *there exist at least three phases consisting of exactly* d *rounds each.*

Proof. We prove the lemma by contradiction. Suppose there are at most two phases with exactly d rounds. Then, the remaining f phases must have at least $d+1$ rounds. This sums upto a total of $2(d)+(f)(d+1) = 2d+fd+f$, which is one more than the total number of rounds of Π i.e $(f+2)(d+1)-3 = fd+f+2d+2-3 = 2d+fd+f-1$. Thus, we have arrived at a contradiction which proves that Π must have at least *three* phases with exactly d rounds each. □

Based on Lemma E.1, we next show that in Π, other than the first and the last phase, there exists at least one "intermediate phase" $p_\ell \in \{p_2, \ldots, p_{f+1}\}$ consisting of *exactly* d rounds; moreover if $p_\ell \neq p_{f+1}$ then it holds that the phase $p_{\ell+2}$ in Π has *at most* $d+1$ rounds[6]. More specifically, let $r_1, \ldots, r_{f+2}$ denote the number of rounds in phase $p_1, \ldots, p_{f+2}$ of Π respectively, where each $r_i \geq d$ and where $r_1 + \ldots + r_{f+2} = (f+2)(d+1)-3$. Then we have the following lemma.

[6]Note that if $p_\ell = p_{f+1}$ then there is no phase $p_{\ell+2}$ in Π. If phase $p_{\ell+2}$ exists in Π then it will have either d or $d+1$ rounds because in Π each phase has at least d rounds.

Lemma E.2. *In protocol* Π*, one of the following holds:*

- *There exists a phase* $p_\ell \in \{p_2, \ldots, p_f\}$*, where* $r_\ell = d$ *and* $r_{\ell+2} \leq d+1$*.*
- $r_{f+1} = d$*.*

Proof. We prove the lemma by contradiction. So assume that none of the two conditions hold in protocol Π. Hence $r_{f+1} \geq d+1$. From Lemma E.1 we know that are at least three phases consisting of d rounds each, which further implies that there exists at least one phase $p_\ell \in \{p_2, \ldots, p_f\}$, with $r_\ell = d$. Note that as per our assumption, we have $r_{\ell+2} \geq d+2$ for every such p_ℓ. Now we have the following two cases, depending upon the total number of phases with d rounds in Π:

- Π *has exactly three phases of* d *rounds*: consider $r_1 = r_{f+2} = d$ and $r_\ell = d$ for exactly one $p_\ell \in \{p_2, \ldots, p_f\}$. Moreover, $r_{\ell+2} \geq d+2$. Furthermore $r_i \geq d+1$, for all $i \in \{2, \ldots f+1\}$ and $i \neq \ell, \ell+2$. This sums upto a total of at least $3(d)+d+2+(f-2)(d+1) = 2d+fd+f$ rounds, which is at least one more than the available number of rounds in Π and so we arrive at a contradiction.
- Π *has more than three phases of* d *rounds*: Let there be $m+2$ phases with d rounds each, where $m > 1$. Consider $r_1 = r_{f+2} = d$. Moreover, as per our assumption, $r_{f+1} \geq d+1$. Furthermore for each of the m phases $p_\ell \in \{p_2, \ldots, p_f\}$ with d rounds, the corresponding $p_{\ell+2}$ phase has at least $d+2$ rounds. This sums upto a total of at least $(m+2)(d)+m(d+2)+(f-2m)(d+1) = 2d+fd+f$ rounds, which is at least one more than the available number of rounds in Π and so we arrive at a contradiction.

□

Now consider the directed graph **G** (see Figure 5) with f crash-tolerant node connectivity and with crash-tolerant diameter d. We next recall the adversary strategy $\mathcal{A}_\Pi$ consisting of sub-strategies $\mathcal{A}^1_\Pi$ and $\mathcal{A}^2_\Pi$ against the protocol Π, executed over **G**:

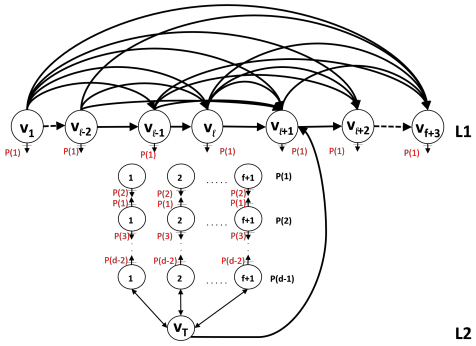


Figure 5: The family of graphs in which it is impossible to achieve consensus in $f+2$ phases with each round having at least d rounds and with total $(f+2)(d+1)-3$ rounds.

- From phase p_1 to $p_{\ell-1}$, the sub-strategy $\mathcal{A}^1_\Pi$ is followed, which is exactly the same as the strategy $\mathcal{A}_{\mathsf{MinMax}}$. Namely p_1 is crash-free. Then in each of the subsequent phase $p_2, \ldots, p_{\ell-1}$, the source node crashes during the first round of the phase after sending its value to all its outgoing neighbours, except the next source in the $L1$ layer.
- Between p_ℓ and $p_{\ell+2}$, the sub-strategy $\mathcal{A}^2_\Pi$ is followed: at the beginning of phase p_ℓ, the node $v_{\ell-1}$ in the $L1$ layer will be the source node. During the first round of p_ℓ, the source $v_{\ell-1}$ crashes after sending its value to the next source v_ℓ. The next phase $p_{\ell+1}$ is a crash-free phase. During $p_{\ell+2}$, the source node v_ℓ crashes in one of the following two ways, depending upon whether $p_{\ell+2}$ has d or $d+1$ rounds:
  - $p_{\ell+2}$ *has* d *rounds*: here v_ℓ crashes after sending its value to all its neighbours (both in the $L1$ layer as well as the $P(1)$ layer), except the next source $v_{\ell+1}$.

– $p_{\ell+2}$ *has* $d+1$ *rounds*: here v_ℓ crashes after sending its value only to its neighbours in the $L1$ layer (and not to the neighbours in $P(1)$ layer), except $v_{\ell+1}$.

- For the remaining phases $p_{\ell+3}, \ldots, p_{f+2}$, sub-strategy $\mathcal{A}^1_\Pi$ is followed, where in each phase, the source node in $L1$ layer crashes after sending its value to all its neighbour (in $L1$ layer and $P(1)$ layer), except the next source in the $L1$ layer.

Note that in the above description, we assumed that $p_\ell \neq p_{f+1}$, so that phase $p_{\ell+2}$ exists in Π. In case $p_\ell = p_{f+1}$, then $\mathcal{A}^1_\Pi$ is followed from p_1 to p_f, while $\mathcal{A}^2_\Pi$ is followed during p_{f+1} and p_{f+2}. We now prove the following theorem.

Theorem E.3. *Let Π be a min-max protocol wth $f+2$ phases with total $(f+2)(d+1)-3$ rounds, where each phase has at least d rounds. Then consensus will not be achieved in* **G** *(Figure 5) by Π against $\mathcal{A}_\Pi$.*

Proof of Theorem E.3: For simplicity, we first assume that $p_\ell \neq p_{f+1}$, so that phase $p_{\ell+2}$ exists in Π. We next analyse the outcome of the adversary strategy $\mathcal{A}_\Pi$ against Π over the graph **G** in Fig 5.

- *Phase* $p_1, \ldots, p_{\ell-1}$: No change occurs in the graph during p_1 as it is a crash-free min phase. At the end of p_1, the source node v_1 has value 1, while all the remaining nodes in the graph has value 0. Due to the adversary strategy $\mathcal{A}^1_\Pi$, the following invariant is maintained at the end of each phase p_i for $i = 2, \ldots, \ell-1$: the source of the reduced graph will be v_i. If p_i is a min phase then v_i has value 1, while all the remaining nodes have value 0. On the other hand if p_i is a max phase then v_i has value 0, while all the remaining nodes have value 1. Hence consensus is not achieved in any of the phases $p_1, \ldots, p_{\ell-1}$.

- *Phase* $p_\ell, p_{\ell+1}$ *and* $p_{\ell+2}$: Without loss of generality, let p_ℓ be a max phase, implying that $p_{\ell+1}$ and $p_{\ell+2}$ are min and max phase respectively. We analyse each of these three phases separately.

  – *Phase* p_ℓ: At the beginning of p_ℓ, node $v_{\ell-1}$ will be the source node with value 1, while all the remaining nodes in the reduced graph will have value 0. As per the adversary strategy, at the end of the first round of p_ℓ, the new source v_ℓ will have value 1, as it is a max phase. This value further gets propagated to all other nodes in the graph, except v_T, within the remaining $d-1$ rounds of p_ℓ. At the end of p_ℓ, all the nodes in the reduced graph have value 1, except the node v_T, which retains the value 0. This is because v_T is at a distance d from the new source v_ℓ. Hence consensus is not achieved in this phase.

  – *Phase* $p_{\ell+1}$: as per the adversary strategy this is a crash-free phase. However, due to the presence of backward edges from v_T to the nodes in $P(d-1)$ and to the node $v_{\ell+1}$, the value 0 of v_T starts propagating backward. So even though there is no crash during $p_{\ell+1}$, by the end of $p_{\ell+1}$, all the nodes in the graph except the source v_ℓ, reset their value to 0. Note that v_ℓ never sees the value 0 and hence it retains the value 1. Hence consensus is not achieved in this phase.

  – *Phase* $p_{\ell+2}$: here there are two cases, depending upon whether $p_{\ell+2}$ has d or $d+1$ rounds.

    * $p_{\ell+2}$ *has* d *rounds*: here the value 1 is passed to all the neighbours of the source v_ℓ, except $v_{\ell+1}$, at the end of the first round after which v_ℓ crashes. As there are still $d-1$ rounds left in $p_{\ell+2}$, the value 1 further propagates to all the remaining nodes in the reduced graph. Hence at the end of this phase, except the node $v_{\ell+1}$, all the nodes have value 1 and so consensus is not achieved in this phase.

* $p_{\ell+2}$ *has* $d+1$ *rounds*: here the value 1 is passed to all the neighbours of the source v_ℓ, except $v_{\ell+1}$ and the nodes in layer $P(1)$, at the end of the first round after which v_ℓ crashes. As there are still d rounds left in $p_{\ell+2}$, the value 1 further propagates to all the remaining nodes in the reduced graph. Hence at the end of this phase, except the node $v_{\ell+1}$, all the nodes have value 1 and so consensus is not achieved in this phase. Note that it is important that this phase has at most $d+1$ rounds, as other wise the node $v_{\ell+1}$ would also receive 1 from v_T if there are $d+2$ rounds in this phase. This is because once v_ℓ crashes, node v_T also becomes a source node, apart from $v_{\ell+1}$.

- *Phase* $p_{\ell+3}, \ldots, p_{f+2}$: Note that the invariant condition that was maintained at the end of $p_2, \ldots, p_{\ell-1}$ is re-instated at the end of $p_{\ell+2}$. Namely, one of the source nodes $v_{\ell+1}$ has a value 0, while all the remaining nodes have value 1. The invariant is going to be retained for phase $p_{\ell+3}, \ldots, p_{f+2}$. More specifically, the following is ensured at the end of each phase p_i for $i = \ell+3, \ldots, f+2$: the source of the reduced graph will be v_{i-1} (and not v_i). If p_i is a min phase then v_{i-1} has value 1, while all the remaining nodes have value 0. On the other hand if p_i is a max phase then v_{i-1} has value 0, while all the remaining nodes have value 1. Hence consensus is not achieved in any of the phases $p_{\ell+3}, \ldots, p_{f+2}$.

  Once p_{f+2} is over the reduced graph consists of nodes v_{f+1}, v_{f+2} and v_{f+3}, along with all the P nodes and v_T. The source of the reduced graph will be v_{f+1} having a value different from all the remaining nodes in the graph, all of which will have the same value.

We next consider the case when $p_\ell = p_{f+1}$. In this case, phase $p_{\ell+2}$ does not exist in Π. Here $\mathcal{A}^1_\Pi$ is followed from p_1 to p_f, while $\mathcal{A}^2_\Pi$ is followed during p_{f+1} and p_{f+2}. We analyse the different phases of Π in this case.

- *Phase* $p_1, \ldots, p_f$: Here no change occurs in the graph during p_1. The following invariant is maintained at the end of each phase p_i for $i = 2, \ldots, f$: the source of the reduced graph will be v_i. If p_i is a min phase then v_i has value 1, while all the remaining nodes have value 0. On the other hand if p_i is a max phase then v_i has value 0, while all the remaining nodes have value 1. Hence consensus is not achieved in any of the phases $p_1, \ldots, p_f$.

- *Phase* p_{f+1}: Without loss of generality, let p_{f+1} be max phase. At the beginning of p_{f+1}, node v_f will be the source node with a value 1, while all the remaining nodes in the reduced graph will have value 0. As per the adversary strategy, at the end of the first round of p_{f+1}, the new source v_{f+1} will have value 1, as it is a max phase. This value further gets propagated to all other nodes in the graph, except v_T, within the remaining $d-1$ rounds of p_{f+1}. At the end of p_{f+1}, all the nodes in the reduced graph have value 1, except the node v_T, which retains the value 0. This is because v_T is at a distance d from the new source v_ℓ. Hence consensus is not achieved in this phase.

- *Phase* p_{f+2}: As per the adversary strategy, this is a crash-free phase. Since it is a min phase, the value 0 of v_T gets propagated back to all the nodes of the graph, except the source node v_{f+1}, which retains its value 1. Hence consensus is not achieved in this phase. □

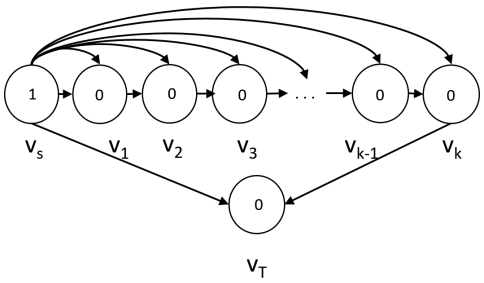
F Impossibility of Consensus in $2f+1$ Min-Max Phases with d Rounds Each

Here we show the impossibility of consensus in $2f+1$ alternate min-max phases for different values of f and d, where each phase has d rounds of communication.

Lemma F.1. *Let $f = 1$ and $d \geq 1$. Then there exist directed graphs with f crash-tolerant node connectivity and crash-tolerant diameter d, and an adversary strategy for $\mathcal{A}$, where consensus can be achieved only if there are $2f + 2$ alternate min and max phases, where each phase has d rounds of communication.*

Proof: Consider the family of graph given in Fig. 6, with the initial values of the nodes given in the figure. The graph has crash-tolerant diameter $d = k$ where $k \geq 1$ and has f crash-tolerant node connectivity. The graph contains $k + 2$ nodes where in the first phase v_s is the initial source. It has a direct edge to all the nodes. In this graph v_i has a direct edge only to v_j, for $i \in \{1, \ldots, k-1\}$ and $j = i + 1$, and v_k has a direct edge to the sink v_T. It is easy to verify that the graph has crash-tolerant node connectivity of 1: if v_s crashes then v_1 becomes the source, else v_s is the source. If v_1 becomes the source then it is at a distance k from the node v_T and so $d = k$ in the graph.

The adversarial strategy in the graph is as follows: the first phase is a crash-free min phase, due to which no change will occur to the value of the nodes in the graph. This is because the source v_s sees only value 1, while the remaining nodes will retain 0 as their value, as this is a min phase. In the next phase which is a max phase, the source v_s crashes during the first round after sending its input *only* to the source of the reduced graph, namely the node v_1. Moreover, this new source v_1 will get $d-1 = k-1$ rounds to propagate the value 1 to the nodes $v_2, \ldots, v_k$. As a result, at the end of second phase, the nodes $v_1, \ldots, v_k$ will set 1 as their value as it is a max phase, However node v_T will still retain 0 as its value, since it never sees the value 1. During the third phase which is a crash-free min phase, no change happens in the graph. Specifically, the nodes $v_1, \ldots, v_k$ will only see the value 1; on the other hand even though v_T sees the value 1 as well as 0, it retains 0 as its value since this is a min phase. Hence no consensus will be achieved between $v_1, \ldots, v_k$ and v_T at the end of the third phase. During the fourth phase there will be no crash (as $f = 1$) and consensus is achieved, where all the nodes have value 1. In Fig. 6, we present the transition table with the values of each node at the end of each round of each of the four phases. In the table, the symbol "X" denotes that the corresponding node has crashed. □



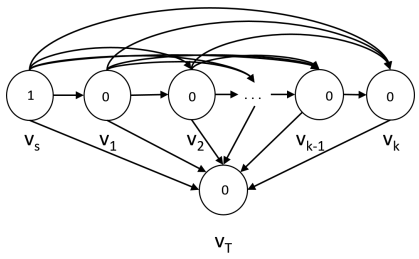
		v_s	v_1	v_2	..	v_k	v_T
Phase 1	Round 1	1	0	0	..	0	0
	:				..		
	Round k	1	0	0	..	0	0
Phase 2	Round 1	X	1	0	..	0	0
	:				..		
	Round k	X	1	1	..	1	0
Phase 3	Round 1	X	1	1	..	1	0
	:				..		
	Round k	X	1	1	..	1	0
Phase 4	Round 1	X	1	1	..	1	1
	:				..		
	Round k	X	1	1	..	1	1

Figure 6: The graph for the case $f = 1$ and $d = k$, with the corresponding transition table.

Lemma F.2. *Let $d = 1$ and $f \geq 1$. Then there exist directed graphs with f crash-tolerant node connectivity and crash-tolerant diameter d, and an adversary strategy for $\mathcal{A}$, where consensus can be achieved only if there are $2f + 2$ alternate min and max phases, where each phase has d rounds of communication.*

Proof: Consider the family of directed graphs given in Fig. 7, with crash-tolerant diameter $d = 1$ and which has f crash-tolerant node connectivity, where $f = k$. The initial value of the nodes are also given in the figure. The graph contains $k + 2$ nodes where in every phase there is always one source. The initial source v_s and the potential sources v_i have a direct edge to all the j^{th} nodes in the graph, for $i \in \{1, \ldots, k-1\}$ and $j > i$. There is one node v_T in the graph which is the sink of the graph and all the other nodes have outgoing edges to this sink.

The adversarial strategy in this graph is as follows: the first phase is a crash-free min phase, due to which no changes occur in the value of the nodes. Then each subsequent min phase is also a crash-free min phase. During each max phase, the adversary crashes only node, namely the source of the current reduced graph, during the first round of the max phase. Moreover, before crashing, the source node sends its value *only* to the source of the next reduced graph. For instance, during the first max phase (which is overall the second phase in the protocol), the source v_s crashes after sending its value 1 only to the source of the next reduced graph, namely v_1. Since $d = 1$, it is ensured that the value of the crashed source node does not propagate further in the graph during a crashed max phase. The manner in which the crashed source node propagates its value during a max phase , it is ensured that in the immediate next min phase (which is crash-free), no node in the reduced graph will change their value. This is because there will be only one source node in the reduced graph, and with value 1, and throughout the min phase it sees only value 1. On the other hand, the non-source nodes will see both 0 and 1 and will retain 0 as their input. As there will be $f = k$ crashed max phases, it follows that at the end of phase number $2f$, the reduced graph will be left with nodes v_k and v_T, with values 1 and 0 respectively and with v_k as the source. During the phase number $2f + 1$, no change will occur in the graph and hence no consensus is achieved between v_k and v_T. The consensus is finally achieved in phase number $2f + 2$. The transition table with the values of each node at the end of each round of each of the $2f + 2$ phases is given in Fig. 7. □



		v_s	v_1	v_2	..	v_k	v_T
Phase 1	Round 1	1	0	0	..	0	0
Phase 2	Round 1	X	1	0	..	0	0
Phase 3	Round 1	X	1	0	..	0	0
Phase 4	Round 1	X	X	1	..	0	0
Phase 5	Round 1	X	X	1	..	0	0
:	:	:	:	:	:	:	:
Phase 2k+1	Round 1	X	X	X	X	1	0
Phase 2k+2	Round 1	X	X	X	X	1	1

Figure 7: The graph for the case $f = k$ and $d = 1$, with the corresponding transition table.

Lemma F.3. *Let $d = 2$ and $f = 2$. Then there exist directed graphs with f crash-tolerant node connectivity and crash-tolerant diameter d, and an adversary strategy for $\mathcal{A}$, where consensus can be achieved only if there are $2f + 2$ alternate min and max phases, where each phase has d rounds of communication.*

Proof: Consider the first graph shown in Fig. 8, along with the initial value of the nodes. There are four "levels" in the graph, where each level has a specific function. The first level (L1) contains the initial source and the source nodes for the subsequent reduced graphs. The second level (L2) contains nodes whose goal is to "pump" the value 0 into the graph during the second min phase; looking ahead, the value 0 will be pumped to all the nodes, except the "current" source node, which will retain the value 1, thus preventing consensus during the second min phase. The fourth level (L4) contains a "sink" node, which will not achieve consensus within $2f + 1$ phases. The third level (L3) makes sure that each source node is at a distance of $d = 2$ from the sink. All the nodes in L1 have edges to all the nodes in L3, denoted by the " $\overset{3}{\nrightarrow}$ " symbol in the graph. The same holds for the nodes in L2. Nodes in L3 have edges to L2 denoted by the " $\overset{2}{\nrightarrow}$ " symbol. It is easy to verify that the graph has a crash-tolerant connectivity of two and has $d = 2$.

The adversary strategy is the following: no crash occurs during the first phase and hence there is no change in the values of the nodes in the graph at the end of the first phase. During the first round of the second phase (second graph in Fig. 8), the source node crashes and passes on its value 1, *only* to the next node, which becomes the new source of the reduced graph. In the second round, the new source further transmits this value to L3 and L1, due to which all of them attain the value 1 at the end of phase two. The nodes in L2 and L4 remain unaffected, since one more round is required to propagate 1 to them. The third phase (third graph in Fig. 8) is a crash-free min phase, and the nodes in L2 reset the value of the rightmost node in L1 and the nodes in L3 by pumping 0. At the end of the third phase, all the nodes have value 0 except the second node in L1, which is the source of the current reduced graph. In the fourth phase (fourth figure in Fig. 8), the second node in L1 crashes during the first round, propagating its value 1 *only* to the next node in L1, which now becomes the new source. The new source further transmits its value 1 to L2 and L3 in the next round, but it fails to propagate 1 to L4, as this further requires one more round. Thus at the end of the fourth phase, all the nodes in the reduced graph have value 1, except the node in L4. During the firth phase (the second last figure in Fig. 8), no change occur in the graph and consensus is not achieved. Finally consensus will be achieved during the sixth phase (the last figure in Fig. 8) which is a crash-free phase. □

Lemma F.4. *Let $d = 2$ and $f = k$, where $k \geq 3$. Then there exist directed graphs with f crash-tolerant node connectivity and crash-tolerant diameter d, and an adversary strategy for $\mathcal{A}$, where consensus can be achieved only if there are $2f + 2$ alternate min and max phases, where each phase has d rounds of communication.*

Proof: We first explain how to extend the graph for $f = 2, d = 2$ (the first figure in Fig. 8) to $f = 3, d = 2$; the extension to $f = k$ and $d = 2$ is done in a natural fashion. The graph for the case $f = 3, d = 2$ is given in Fig. 9. For increasing the value of f we increase the "width" of the graph by adding three extra nodes in L1, L2 and L3, denoted in the graph by blue circles. Correspondingly, new directed edges need to be included which are indicated in blue colour. We include one node in L1, since three nodes will crash and so we need to add one source node too. The extra node in L2 is used to reset the third node in L1 from left during third phase. The node in L3 is used to preserve the f crash-tolerant property, as otherwise the node in L4 gets disconnected from the graph in case if three nodes in L3 get crashed.

The basic intuition for the graph construction is that in every max phase, the source node will crash and transmit its value 1 to the next source node. The new source node will transmit this value further to all the other nodes that are within a distance $d - 1$ apart from it. There will be some nodes which are d distance apart, and they will retain their original value 0 and prevent the graph from attaining consensus. Also there will be some pumping nodes, which will be at d distance apart from the source nodes. In the next min phase, the source will contain 1 and all nodes that were $d - 1$ distance will contain 1. The rest of the nodes will

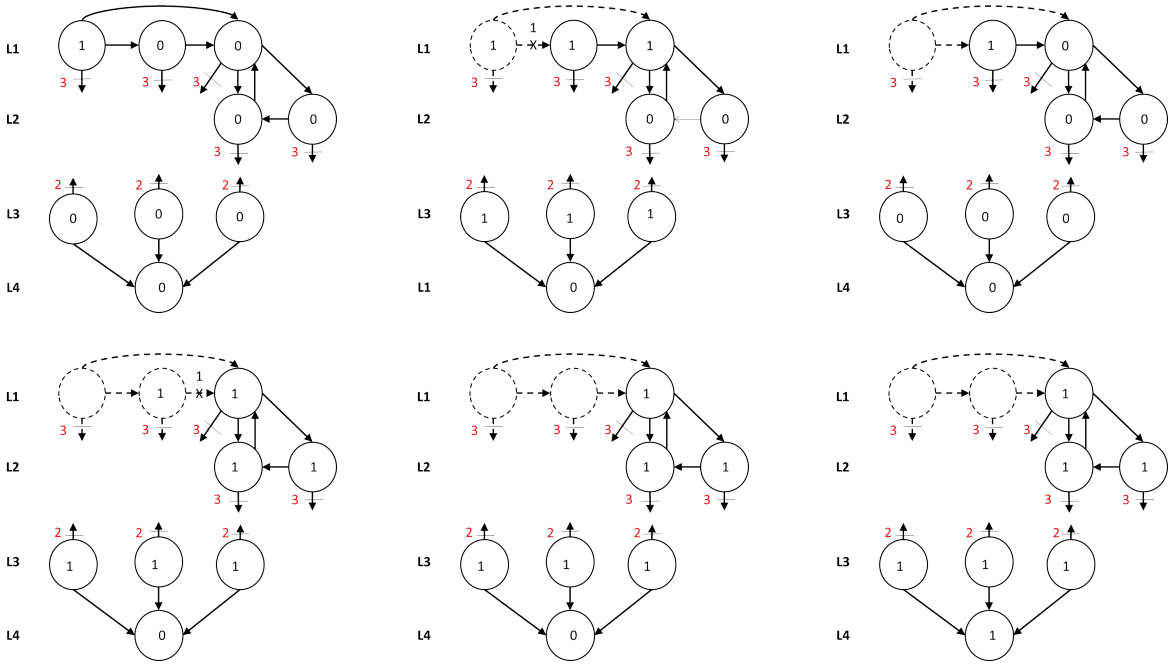


Figure 8: The graph showing the impossibility of achieving consensus within $2f + 1$ phases for $f = 2$ and $d = 2$. The figures from left to right show the value of nodes in each graph at the end of the first, second, third, fourth, fifth and sixth phase respectively.

have 0 from previous min phase. The pumping node (in L2) will provide 0 to the next possible (potential) source and also make whole graph attain the value 0, except the current source node. The pumping nodes will also shift towards the right in L2. Note that different nodes in L2 play the role of pumping node during different min phase. The pumping node in the second min phase (which is overall the third phase in the protocol) is the first node (from left) in L2, whereas in the third min phase (which is overall the fifth phase in the protocol) it is the second node in L2. Note that the third (the rightmost) node in L2 always retain value 0, till the sixth phase, since this node pumps the value 0 to the other nodes in L2 during the min phases. In Fig. 10, we pictorially represent the value of each node at the end of each of the eight phases.

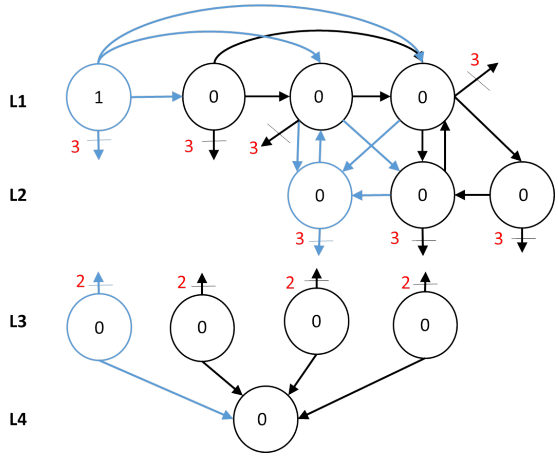


Figure 9: Extending from $f = 2$ to $f = 3$ with $d = 2$. The nodes and edges in blue colour represent the additional nodes and edges which need to be incorporated.

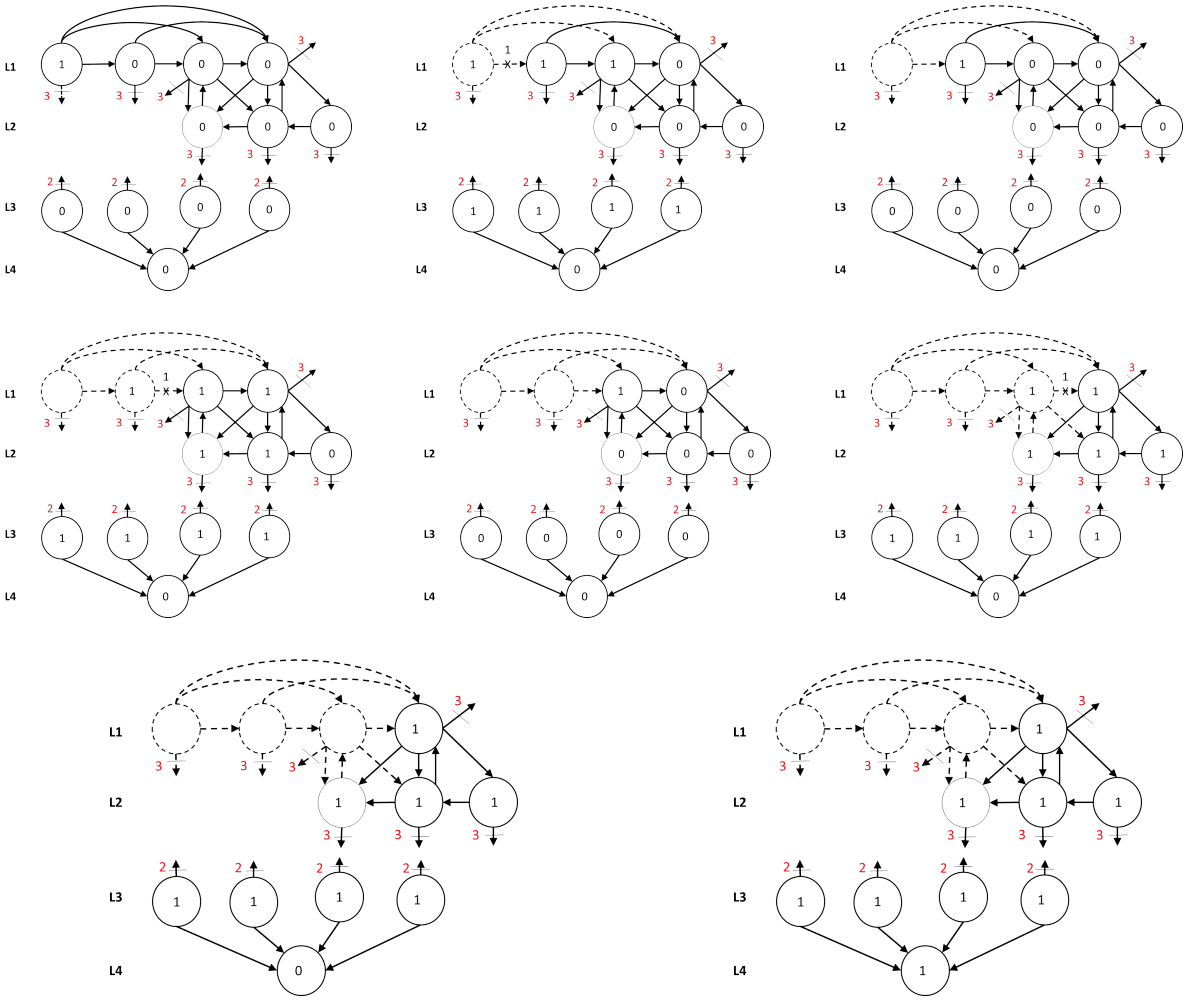


Figure 10: The graph showing the impossibility of achieving consensus within $2f + 1$ phases for $f = 3$ and $d = 2$. The figures from left to right show the value of nodes in each graph at the end of the first, second, third, fourth, fifth, sixth, seventh and eighth phase respectively.