Robustness and efficiency of voting consensus protocols within byzantine infrastructures

Angelo Capossele, Sebastian Müller, Andreas Penzkofer

Version of Record: https://www.sciencedirect.com/science/article/pii/S2096720921000026
Manuscript_7780a3ca2fe8df0460634c5d31d266d2

ROBUSTNESS AND EFFICIENCY OF VOTING CONSENSUS PROTOCOLS WITHIN BYZANTINE INFRASTRUCTURES

Angelo Capossele*, Sebastian Mueller+, Andreas Penzkofer*
* IOTA Foundation, 10405 Berlin, Germany
+ Aix-Marseille Université, CNRS, Centrale Marseille, I2M, UMR 7373, 13453 Marseille, France

January 2, 2021

ABSTRACT

This paper investigates several voting consensus protocols with low computational complexity in noisy Byzantine infrastructures. Using computer simulations, we show that explicit randomization of the consensus protocol can significantly increase the robustness towards faulty and malicious nodes. We identify the optimal amount of randomness for various Byzantine attack strategies on different kinds of network topologies.

Keywords: Distributed systems, consensus protocols, Byzantine infrastructures, simulation studies

1 Introduction

1.1 Context

Distributed consensus algorithms allow networked systems to agree on a required state or opinion in situations where centralized decision making is difficult or even impossible. As distributed computing is inherently unreliable, it is necessary to reach consensus in faulty or Byzantine infrastructure. This problem's importance stems from its omnipresence and fault tolerance is one of the most fundamental aspects of distributed computing. Common examples include clock synchronization, resource allocation, task scheduling, and replicated file systems, see [1].

The consensus problem arises traditionally as well in technical systems for object tracking [2], density classification [3], sensor fusion [4], robot localization [5], and mission planning [6]. Furthermore, collective and distributed decision making is important in social dynamics [7, 8]. The consensus problem is also at the heart of distributed ledger technology, [9], since it allows to reach consensus on whether to add a transaction to a ledger or to agree on a timestamp of a transaction.

Every node in a distributed consensus system follows simple rules, using only local information and node-to-node communication. Algorithms performing such tasks should be robust toward different types of disturbances and Byzantine settings. A landmark study [10] considers the robustness of consensus with faulty nodes. In particular, [10] shows that for synchronous systems, a necessary condition for reaching consensus is that at most $1/3$ of the nodes are faulty. In the significant work [11], this result is strengthened in the sense that in asynchronous situations already one faulty node can prevent an agreement.

1.2 Distributed binary majority consensus

This article focuses on a certain class of consensus, namely, binary majority consensus. Some basic algorithms in this protocol class are simple majority consensus, [8], Gacs-Kurdyumov-Levin (GKL)[12], and random neighbors majority, [13]. These protocols may achieve good performances in noiseless and undisturbed networks. However,

their performances significantly decreases with noise [12, 14] or errors [15] and may completely fail in a Byzantine setting, see Section 2.1. The sensitivity of these protocols to topological changes is studied in [16, 17, 18]. This weak robustness against faulty nodes may explain why simple majority voting was not thoroughly investigated until recently. In [13] several variants of the binary majority consensus were compared in networks with different types of disturbances. Moreover, [13] studied the potential capacity of randomization to improve the performance of the protocol.

1.3 A new contender - fast probabilistic consensus

While the faulty node behavior appears to be the most severe among the above disturbances, Byzantine, or malicious nodes introduce the next challenge for consensus protocols and can be considered the cutting edge in the area. Recently, [19] proposed a new leaderless binary majority consensus protocol, called the fast probabilistic consensus (FPC), and obtained theoretical results on the convergence and safety of the protocol. A special feature of the protocol is that it uses a sequence of random numbers that are either provided by a trusted source [20] or generated by the participants of the protocol themselves using some decentralized random number generator protocol [21, 22, 23]. See Section 5.2 for more details.

Randomized algorithms are used in various fields and are known to offer efficient ways to reduce complexity and increase performances. In our context, a deterministic consensus protocol might fail in various situations, e.g., see [13] and Section 2.1. While detection methods might prevent such situations, these techniques are not always available or not yet developed for Byzantine infrastructure. Simple consensus protocols without randomization nor fault detection can rarely guarantee the eventual agreement of the nodes. However, noisy environment and random topology may lead to better performances, see [24, 25, 15]. In addition to these effects, the randomization in the FPC protocol serves as "fog of war", [26], for potential malicious attackers, making it at best impossible for the attacker to control the honest nodes. However, let us note that the additional randomness may have some negative side effects: it can delay the termination of the protocol and lower the integrity rate; see Section 4.4 for a more detailed discussion.

The FPC contains two additional new features compared to its predecessors. Firstly, nodes follow a local stopping rule on when to stop querying other nodes. Once all nodes stopped querying, the protocol terminates automatically. Secondly, the protocol is asymmetric in the following sense. The two binary choices, 0 and 1, do not need to be of the same importance. For instance, in various applications, it may be appropriate to distinguish between the following two different kinds of errors. If initially, a majority of the nodes prefer 0 but the eventual outcome of the protocol is 1. On the other hand, if a majority of the nodes initially prefer 1, but the eventual outcome of the protocol is 0. This distinction between the different kinds of errors is a standard approach in statistical test theory and the evaluation of binary classifiers. The two different kinds of errors of a binary decision rule or classifier are measured differently in different fields. For example, in medicine, sensitivity and specificity are often used, in statistical test theory, one speaks of errors of type I and type II, and in computer science, precision and recall are preferred.

Finally, let us stress further essential properties of the studied protocols. Nodes use only local information on the network, allowing the protocol to run on permissionless networks. Moreover, there is no central entity or elected leader that "supervises" the network and decides whether consensus was achieved. The lack of such an entity means that each node must decide when to stop using a local rule, i.e., using only the information locally available to it.

1.4 Key contributions

In Section 2, we compare several leaderless binary majority consensus protocols, namely the simple majority consensus (SMC), the random majority consensus (RMC), and the FPC. We show that the former two can be understood as a special case of the FPC. We propose a variation of FPC with reduced complexity by removing the randomness of the initial threshold and omitting the cooling phase. This modified version is easier to implement and contains nevertheless all ingredients that allow the protocol to be robust and of low communicational complexity in Byzantine infrastructures.

In [19], it is assumed that every node holds a complete network view. However, in permissionless systems or in less reliable networks with inevitable churns (nodes join and leave), this assumption is not necessarily verified. We define in Section 3.1 several graph topologies modeling the partial network view for the nodes.

Our focus in this paper lies in the performances of the different consensus protocols in Byzantine infrastructures. The performances of the protocols are measured using integrity rate, agreement rate, and termination rate. Furthermore, we subdivide the integrity rate into 0- and 1- integrity rate to take into account the asymmetric nature of the protocol. To study the Byzantine nature of the infrastructure, we propose in Section 3.2 specific cautious and Berserk adversarial strategies. These strategies serve as a benchmark to compare the different protocols.

Since the theoretical bounds on the performances on FPC obtained in [19] are not optimal, we perform a numerical performance study with various parameter settings in Section 4. Since the RMC and SMC emerge as special cases of the FPC, we explicitly discuss the differences between the protocols where applicable.

Some of the most relevant results are as follows: Figs. 1 and 2 demonstrate that FPC performs well if nodes only have a partial network view. We show that the protocol exhibits good scaling behavior, with a message complexity of essentially $O(n)$ even in Byzantine infrastructure, see Figs. 8 and 9. We show that the random threshold of the FPC is necessary to allow the protocol to withstand a positive proportion of nodes that follow a Berserk strategy, see Fig. 12. Furthermore, we identify the optimal amount of randomness in the common random threshold in various situations; see Section 4.5. We also note that the protocol allows some flexibility on this feature since it is not necessary to provide a common random threshold for every round, see Figs. 17 and 18.

A more detailed summary of the conclusions from the simulation analysis is given in Section 6.

1.5 Further related work

There is a wide range of classical work on (probabilistic) Byzantine consensus protocols [27, 28, 29, 30, 31, 32]. A disadvantage of these papers' approaches is that they typically require that every node communicates with every other node in the network. This fact induces a communication complexity of $O(n^2)$ messages in each round. Moreover, in permissionless, unreliable, or large networks, nodes may only be able to communicate with a subset of the network.

Most of the previous research focuses on failures within a network infrastructure, rather than on malicious agents. The work [33] defines a fast and scalable Byzantine fault-tolerance protocol. A leaderless Byzantine consensus is studied in [34] that provides a robust infrastructure where there is a failure in the leader of the consensus network. Another Byzantine fault-tolerant method that does not require a leader node is Honey Badger [35].

2 Protocols

In order to define the consensus protocols we introduce some notation. We assume the network to have n nodes indexed by $1, 2, \ldots, n$. Every node i has an opinion or state. We note $s_i(t)$ for the opinion of the node i at time t. Opinions take values in $\{0, 1\}$. Note that in some references, e.g., [13], this is chosen to be $\{-1, 1\}$.

At each time step each node chooses k (random) neighbors C_i, queries their opinions and calculates

$$\eta_i(t+1) = \frac{1}{k_i(t)} \sum_{j \in C_i} s_j(t),$$

where $k_i(t)$ is the number of replies at time t.

Any binary decision, as in statistical test theory or binary classification, gives rise to two possible errors. In many applications, the two possible errors are asymmetric in the sense that one error is more severe than the other. For instance, in statistical test theory, one speaks of errors of type I type II, while in computer science, precision and recall are preferred. Choosing the threshold of the first voting round $\tau \in (0.5, 1]$ allows the protocol to include this kind of asymmetry.

It is possible that not all nodes respond and one can set the value for $s_j(t)$ in the case of no response to 0. This will introduce another asymmetry to the protocol, however, this case is similar to a proportion of nodes being faulty and can be modeled through strategies for malicious nodes introduced in Section 3.2.5.

2.1 Simple majority consensus (SMC)

We set $C_i = \mathcal{N}_i$, where $\mathcal{N}_i$ is the set of neighbours of node i, i.e., all neighbors are queried. The first opinion update is defined by

$$s_i(1) = \begin{cases} 1, \text{ if } \eta_i(1) \geq \tau, \\ 0, \text{ otherwise,} \end{cases}$$

and for $t \geq 1$:

$$s_i(t+1) = \begin{cases} 1, \text{ if } \eta_i(t+1) > 0.5, \\ 0, \text{ if } \eta_i(t+1) < 0.5, \\ s_i(t), \text{ otherwise.} \end{cases}$$

This kind of protocol is also known as "cellular automata model", see [36, 37, 38], as majority dynamics, see e.g., [8, 39, 18, 40] and in a more general setting as the Ising model, see [41].

Without any additional security measures, this protocol is not only fragile against malicious attackers but can even produce agreement failures without disturbances, e.g. [18]. A variation, called GKL, was proposed in [12] to avoid certain security features but stays extremely fragile in Byzantine infrastructures.

2.2 Random neighbors majority consensus (RMC)

The updated opinion of a node is no longer computed using information from all its neighbors but using information from some randomly selected neighbors. A key feature is that neighbor selection is dynamic in the way that each node chooses different neighbors in each step t. This feature allows the RMC to be secure in various situation where the SMC would lead to an agreement or termination failure. At each time each node i running uses the opinions from $C_i(t)$ random neighbors to update its own opinion:

$$s_i(1) = \begin{cases} 1, \text{ if } \eta_i(1) \geq \tau, \\ 0, \text{ otherwise,} \end{cases}$$

and for $t \geq 1$:

$$s_i(t+1) = \begin{cases} 1, \text{ if } \eta_i(t+1) > 0.5, \\ 0, \text{ if } \eta_i(t+1) < 0.5, \\ s_i(t), \text{ otherwise.} \end{cases}$$

We assume that the $C_i(t)$ are independent for $i \leq n$ and $t \in \mathbb{N}$. At each time t for each i the neighbors C_i of a node i are selected in a sample from a discrete uniform distribution on $\mathcal{N}_i$ without replacement.

Let us note that in previous works, e.g., [13], neighbors C_i of a node i were chosen using sampling with replacement and hence repetitions are possible. Our choice of using selection without replacement, facilitates the comparison with the following consensus protocol. Moreover, if the number of vertices is much higher than the number of queried neighbors, the difference between these two different choices is negligible.

In [13] also the own state was taken into consideration and the protocol uses a variation for the calculation of the η's:

$$\widetilde{\eta}_i(t+1) = \frac{1}{k_i(t)+1} \left(s_i(t) + \sum_{j \in C_i} s_j(t) \right),$$

where $k_i(t)$ is the number of replies at time t. We note that in preliminary studies we did not detect any significant differences between these two versions for the values of k under consideration.

2.3 Fast probabilistic consensus (FPC)

We consider a special version of the FPC introduced in [19] in choosing some parameters by default; we remove the cooling phase of FPC and the initial threshold's randomness. The cooling phase in the original protocol in [19] may decrease the minimum number of voting rounds required for termination. However, for a large enough required number of consecutive rounds, the cooling phase can be removed without affecting the protocol's performance. The

Algorithm 1: FPC update for $n \geq 2$

```
Input: opinionOld
Output: opinionNew
u := UNIF(β,1 − β)
while (LENGTH(undecided) > 0) AND (iteration ≤ maxIt) do
    for node in undecided do
        c:= SAMPLE(N\node, k, replace=FALSE)
        eta:= MEAN(opinionOld[c])
        if (eta > u) then
            opinionNew[node]:= 1
        else if (eta < u) then
            opinionNew[node]:= 0
        else
            opinionNew[node]:= opinionOld[node]
        if (opinionOld[node]=opinionNew[node]) then
            cnt[node]++
        else
            cnt[node]:= 0
        if (cnt[node]=l) then
            undecided:= undecided\node
    iteration++
```

proof strategy in [19] suggests that theoretical results remain valid in the absence of the cooling phase. However, a self-contained proof of this fact would take a considerable amount of space and is not included in this paper. Note also that a generalization of the theoretical results to different network topologies seems feasible, however not straightforward, and would undoubtedly yield a merit-worthy paper. In the original FPC, the initial threshold is randomly chosen in an interval $[a, b]$. This random threshold gives additional protection against some specific attacks but increases the integrity failure if the initial proportion of 1 opinion is close to b. We also want to emphasize that the theoretical results in [19] remain valid for a deterministic threshold and that the randomness in the subsequent rounds is sufficient for the protocol's robustness. For these reasons, we set $a = b = \tau$. However, the randomness of the subsequent thresholds is again the key ingredient that makes the protocol robust.

Let $U_t, t = 1, 2, \ldots$ be i.i.d. random variables with law $\mathrm{Unif}([\beta, 1-\beta])$ for some parameter $\beta \in [0, 1/2]$. The update rules are now given by

$$s_i(1) = \begin{cases} 1, \text{ if } \eta_i(1) \geq \tau, \\ 0, \text{ otherwise,} \end{cases}$$

and for $t \geq 1$:

$$s_i(t+1) = \begin{cases} 1, \text{ if } \eta_i(t+1) > U_t, \\ 0, \text{ if } \eta_i(t+1) < U_t, \\ s_i(t), \text{ otherwise.} \end{cases}$$

We provide a pseudo-code of FPC in Algorithm 1.

By comparing with Section 2.2 it can be seen that the essential additional ingredient compared to RMC is that the threshold of 0.5 for times $t \geq 2$ becomes random. More specifically results obtained in this paper for $\beta = 0.5$ provide information about the RMC protocol.

Furthermore, as can be seen by comparing to Section 2.1, for $\beta = 0.5$ the protocol is identical to FPC when the value of the variable *replace* in Algorithm 1 is set to true and k is set to be the number of neighbors.

2.4 Termination of the consensus protocols

We introduce local termination rules to reduce the communication complexity of the protocols. Every node keeps a counter variable `cnt` that is incremented by 1 if there is no change in its opinion and that is set to 0 if there is a change of opinion. Once the counter reaches a certain threshold `l`, i.e., $\texttt{cnt} \geq \texttt{l}$, the node considers the current state as final. The node will therefore no longer send any queries but will still answer incoming queries. In case of no autonomous termination the algorithms is halted after `maxIt` iterations.

3 Modelling assumptions

3.1 Network topology

We consider networks with n nodes and where the nodes are enumerated from 1 to n. Some node pairs are connected by a link/edge. For a given node i we denote the set of all neighbors of i by $\mathcal{N}_i$. We consider the following network topologies:

1. Complete graph: every node is connected to every other node
2. Regular ring lattice: In the regular ring lattice, introduced in [12], every node is connected to an even number d of other nodes in the following way. The n nodes are numerated as $\{1, \dots, n\}$ and each node connected to $d/2$ nodes on its "left" and $d/2$ nodes on its "right", i.e., there is an edge between i and j if and only if

   $$0 < |i - j| \bmod (n - d/2) \leq d/2.$$

   We denote the proportion of the network a node is connected to by

   $$\delta = d/N$$

3. Watts-Strogatz graphs: The Watts-Strogatz model, introduced in [42] is a random graph generation model that produces graphs with small-world properties, including short average path lengths and high clustering. A Watts-Strogatz graph with n vertices and mean degree d (even) is constructed as follows. Let $\gamma \in [0, 1]$ be a model parameter and assume that $n \gg k \gg \ln n \gg 1$. The $nd/2$ undirected edges are now defined as follows:
   (a) Construct a regular ring lattice of degree d.
   (b) For every node $i = 1, \dots, n$ take every edge connecting i to its $d/2$ rightmost neighbors and rewire it with probability γ. Rewiring here means that the edge is replaced by an edge between i and a node chosen uniformly from the other nodes that are not yet neighbors of i.

   The underlying ring lattice structure produces a locally clustered network, while the randomly rewiring reduces the diameter of the network. There are about $\gamma \frac{nd}{2}$ "non-lattice" edges. Varying γ allows interpolation between the regular ring lattice ($\gamma = 0$) and a network close to the Erdös-Renyi graph $G(n, p)$, e.g., [43, Chapter 4], with the edge inclusion probability $p = d/(n - 1)$ ($\gamma = 1$). However, note that in contrast to an $G(n, p)$ in the Watts-Strogatz graph every node has at least $d/2$ neighbors. Situations with a high mean degree d allow to model perturbations of the complete graph.

With the above definition (1) becomes a special case of (2), where the number of neighbors is $N - 1$. Furthermore, (2) can be considered as a special case of (3) if $\gamma = 0$.

After the construction of the underlying graph, we (re)-assign the node ids randomly.

3.2 Faulty and malicious nodes

3.2.1 Positions of the faulty and malicious nodes

We assume that there is a proportion of q faulty or malicious nodes, and we call the remainder of the nodes normal or honest, respectively. There are two natural possibilities to choose faulty nodes. In the first, we choose $\lceil qn \rceil$ faulty

or malicious nodes at random. This is done by performing sampling without replacement. In the second, each node becomes faulty with probability q independent of all the other nodes. While the second way seems to be more natural in the modeling of faulty nodes, we choose the first method since it is more appropriate for modeling malicious nodes. In any case, the differences in the simulation outcomes are rather marginal, especially if n is large.

As the assignment of the node ids in the construction of the network topology is random, we can choose the first $n_h = n - \lceil qn \rceil$ nodes as honest (normal) nodes and the remaining nodes as malicious (faulty) nodes.

We want to note that our results rely on the fact that the malicious nodes are chosen at random. An attacker that has means to interfere with the network topology might, for instance, perform eclipse attacks or use specific properties of the network topology.

3.2.2 Initial configuration of opinions

An opinion $s_i(0) \in \{0, 1\}$ is assigned to all normal (honest) nodes at time $t = 0$. These opinions form the initial opinion of the nodes. The mean

$$p_0 = \frac{1}{n_h} \sum_{i=1}^{n_h} s_i(0)$$

is the initial proportion of 1 opinion, where $p_0 \in [0, 1]$.

We initiate all nodes having an index smaller than $p_0 n_h$ with 1, while the remainder of nodes has opinion 0.

Previous studies, e.g., [13], used a flip-coin procedure to assign the initial opinions, that is, each node obtains its opinion by an independent Bernoulli trial in which the opinion 1 is chosen with probability p_0 and the opinion 0 is chosen with probability $1 - p_0$. Due to the reassignment of node ids these two approaches essentially coincide for n large.

3.2.3 Types of failures

In this paper we focus on three types of failures:

1. *Termination failure*: We say the protocol suffers a termination failure if some nodes reach the maximum round `maxIt`.
2. *Agreement failure*: The protocol is said to have an agreement failure if the opinion with which the normal or honest nodes terminate, is not the same for all nodes. In the majority of the cases a termination failure would also lead to an agreement failure.
3. *Integrity failure*: We say that the protocol suffers an integrity failure if the final opinion for all nodes is the opposite of the initial honest majority. Naturally, an agreement failure would also lead to an integrity failure.

The constraint on what counts as a failure may be relaxed by requiring that for at least a proportion ϵ of the honest nodes the above definition of failure occurs. In this paper, we take the more conservative approach and assume that $\epsilon = 0$ for the termination and integrity failure, and $\epsilon = 0.001$ for the agreement failure. Therefore for $N \leq 1000$ an agreement failure is achieved already with a single node.

3.2.4 Faulty nodes and message loss

Faulty nodes are assigned an opinion opposite to the initial majority of the normal nodes. We note s^{n} for the opinions of the normal nodes and s^{f} for the opinions of the faulty nodes. The dynamics are then formally defined by:

$$s_i^{\mathrm{f}}(t) = \begin{cases} 0, \text{ if } \sum_i s_i^{\mathrm{n}}(0) \geq 0.5, \\ 1, \text{ otherwise} \end{cases} \qquad \forall t \geq 1.$$

The update of the opinions $s_i^{\mathrm{n}}(t)$ of the normal nodes is done according to the protocols described in the previous section. All faulty nodes answer queries of normal nodes about their state value but never change their own state. Since the strategy MinVS, defined in Section 3.2.5, has the same effect as the above modeling of faulty nodes, we give simulation results only for the MinVS.

A loss of messages can severely decrease the performance and safety of the studied protocols, since a node's decision is based on less state information. We consider the model, where a query from node i to node j is lost with message loss probability $\mathcal{E}_{i,j} \in [0,1)$:

$$s_{i,j}(t) = \begin{cases} \star, & \text{with probability } \mathcal{E}_{i,j}, \\ s_{i,j}(t), & \text{with probability } 1-\mathcal{E}_{i,j}. \end{cases}$$

Using an indicator function we can write the number of successful queries as

$$k_i(t) = \sum_{j \in C_i(t)} \mathbf{1}\{s_{i,j}(t) \in \{0,1\}\}$$

and the definition of η becomes

$$\eta_i(t+1) = \frac{1}{k_i(t)} \sum_{j \in C_i} s_j(t) \mathbf{1}\{s_{i,j}(t) \in \{0,1\}\}.$$

For sake of brevity and better presentation, we do not present simulation results on message loss. We content ourselves noting that performances of the FPC are in general much better under message loss than in Byzantine infrastructure. We, therefore, focus in this study on the performance of the protocol in Byzantine environment without message loss.

3.2.5 Malicious nodes

We assume that a proportion of $q \in [0,1)$ nodes are malicious and try to interfere with the protocol. As in [19] we distinguish between two different kinds of adversaries:

- Cautious adversary: any adversarial node must maintain the same opinion in the same round, i.e., respond the same value to all the queries it receives in that round.
- Berserk adversary: an adversarial node may respond different opinions to different queries in the same round.

The reason for this distinction is that Berserk adversaries are much more harmful but can easily be detected if honest nodes communicate with each other. Hence, in networks with fault detection, a Berserk strategy might not be feasible for an adversary. An adversary may also choose to answer only to some nodes and to follow a semi-cautious strategy, see [19]. For sake of a better presentation we suppose in this paper that nodes always answer all queries.

Byzantine infrastructures and attack strategies can be very diverse. We assume that adversarial nodes can exchange information freely between themselves and can agree on a common strategy. A single individual or entity may control them all.

Strategy for integrity failure.

By monotonicity of the majority rule, the worst-case strategy for integrity failure is the initial minority vote strategy (**MinVS**): The adversary tries to turn the initial majority in transmitting continuously the opinion of the initial minority. The dynamics of this strategy are exactly the same as in the setting for faulty nodes.

Cautious strategy for agreement and termination failure

We consider the cautious strategy where the adversary transmits at time $t+1$ the opinion of the minority of the honest nodes of step t. We call this strategy the inverse vote strategy (**IVS**).

Berserk strategies for agreement and termination failure

We consider a Berserk strategy, that we dub *maximal variance strategy* (**MVS**). In this approach, the adversary waits until all honest nodes received opinions from all other honest nodes. The adversary then tries to subdivide the honest nodes into two equally sized groups of different opinions while trying to maximize the variance of the η-values.

In order to maximize the variance, the adversary requires knowledge over the η-values of the honest nodes at any given time. The adversary then answers queries of undecided nodes in such a way that the variance of the η's is maximized by keeping the median of the η's close to 0.5, see Algorithm 2 for a formal definition.

Algorithm 2: Adapted Berserk Strategy: the maximal variance strategy (MVS).

Input: opinion
Output: eta

```
for node in nodesHonest do
    c[node]:= SAMPLE(𝒩\node, k, replace=FALSE)
    etaInter[node]:= MEAN(opinion[c[node] ∩ nodesHonest])
    kHonest[node]:= LENGTH(c[node] ∩ nodesHonest)
    completed[node] := (kHonest[node]=k)
while (LENGTH(!completed)<nHonest) do
    if MEDIAN(etaInter)>0.5 then
        node := ARGMIN(etaInter[!completed])
        opinionAdv[node]:=0
    else
        node := ARGMAX(etaInter[!completed])
        opinionAdv[node]:=1
    etaInter[node]:= (etaInter[node] * kHonest[node]
                      + opinionAdv[node]*(k-kHonest[node]))/k
    completed[node] := TRUE
eta := etaInter
```

4 Performance evaluation

4.1 Simulation methodology and parameters

In this section, we analyze the FPC protocol based on the network topologies and malicious actors defined in the previous sections. For most of our analysis, we assume the default parameter set in Table 1 and any deviations from these values are mentioned explicitly.

Table 1: Default simulation parameters

Parameter		Value
n	Number of nodes	1000
τ	Initial threshold	2/3
k	Quorum size	21
β	Lower random threshold bound	0.3
`l`	Final consecutive round	10
`maxIt`	Max termination round	100
q	Proportion of adversarial nodes	0.1

Figures shown in this section are obtained from data for which each data point is the average over a sample of at least 10,000 simulation runs. The high sample size per data point ensures that the standard error of the mean value is less than 1%.

The methodology for our simulation studies is as follows:

In order to assess the performance of the protocol we define several metrics, that evaluate the resilience against failures as well as the message complexity in Section 4.2. A traditional way to measure the performance of the studied protocols is to consider the agreement rate, i.e., the fraction of time the protocol reaches agreement among the non-faulty honest nodes. In order to obtain a more complete picture, we also study termination and integrity rates. We define termination rate as the rate at which the protocol reaches consensus between all nodes within a reasonable time. Furthermore, integrity rate is the rate at which the final opinion equals the initial majority opinion.

In Section 4.3 we analyze the performance of the protocol if nodes do not have a complete network view or if the network is not fully connected. To enable this we employ the Watts-Strogatz graph to model the network topology. This model allows to interpolate the topology between a worst-case scenario, in this case, a ring graph and the complete graph.

Since a Byzantine environment is more severe than faulty nodes or message loss we focus our study on the former. In Section 3.2.5, three different types of adversarial strategies are introduced and applied to investigate the performance of the protocol.

In Section 4.4 we study the integrity of the protocol by applying the initial minority vote strategy, which aims to achieve an integrity failure. Due to the introduced asymmetry, for the integrity of 0- and 1-opinions, we subdivide the discussion on the integrity rate into whether integrity is maintained when the initial honest majority is 1 or 0. Under these conditions, we analyze the impact of the added asymmetry, the scalability of the protocol as well as up to which proportion of adversary nodes the protocol performs well.

In Section 4.5 we compare two strategies that aim for agreement and termination failure. In the inverse voting strategy, we assume the adversary may not send differing opinions to different nodes, while in the maximal variance strategy we allow this capability for the adversary. We call this capability cautious and Berserk, respectively. We compare the severity of these two different approaches, how the randomness of the protocol can overcome these attacks, and whether the protocol still performs well when the randomness is not continuously supplied.

4.2 Performance measures

In the following, we define the metrics that we employ to analyze the performance of our protocol. The first three definitions are rates that are based on the failures defined in Section 3.2.3.

Termination rate

We are interested in the rate at which the protocol terminates for all nodes before the maximum round `maxIt`. For any given simulation run we say the termination is 1 if all nodes conclude before `maxIt`, while it is 0 otherwise. The rate is determined by averaging over the termination values for several simulation runs.

Agreement rate

We study the performance of the consensus protocols in terms of their agreement rate (also known as convergence rate, see Section 1.2) that is the fraction of initial configurations (or simulations) resulting in successful agreement. If agreement is achieved we assign the value 1, or 0 otherwise. Similarly to the termination rate the agreement rate is obtained by averaging over several simulation runs. Note that the protocol cannot terminate when agreement is not achieved for all nodes. Hence the agreement rate is less or equal to the termination rate.

Integrity Rate

We say integrity is achieved, if the opinion of the initial honest majority is preserved after the protocol concludes, i.e., the protocol finishes with 1 if $p_0 \geq 0.5$ or 0 if $p_0 < 0.5$. In the first case, we also speak of 1-integrity and in the second case we speak of 0-integrity. If integrity is achieved we assign the value 1, or 0 otherwise. The integrity rate is obtained by averaging over several simulation runs.

Since FPC utilizes an initial threshold $\tau \geq 0.5$, it is expected that the integrity failure rate is greater than zero when $p_0 < \tau$, even in the presence of only honest actors. This will be noted in the some of the later sections by a convergence of the integrity rate to a value less than 1. An alternative definition for the integrity rate would be if the protocol concludes with 1 if $p_0 \geq \tau$ or 0 if $p_0 < \tau$. However, this would complicate the analysis unnecessary, which is why the former approach is taken.

Integrity cannot be achieved when the protocol does not terminate or no agreement exists. Hence the integrity rate is limited by the termination and agreement rates.

Maximal time to termination $\overline{T}_{max}$

For each simulation, we consider the maximal number of rounds until all nodes terminate the protocol. The average of these maximal rounds overall simulations is called $\overline{T}_{max}$.

Mean time to termination $\overline{T}_{mean}$

For each simulation, we calculate the average number of rounds over all nodes. We denote by $\overline{T}_{mean}$ the average of these numbers overall simulations. Since each node queries k nodes each round, the average total number of messages sent, i.e., the message complexity, is given by

$$C = \overline{T}_{mean} kn.$$

4.3 FPC with a partial network view

We study the FPC-protocol on non-complete graphs and in a setting without an adversary. We utilize the Watts-Strogatz graph and the associated parameters γ and δ that have been described in Section 3.1 to analyze the effects of different network topologies on the performance of FPC.

In Fig. 1 we show the performance of the protocol on a ring lattice, which is equivalent to a Watts-Strogatz graph with rewiring probability $\gamma = 0$. Furthermore, we set the proportion of initial 1-opinions to $p_0 = \tau$. This corresponds to the case where we expect a high integrity failure, see Section 4.2. It can be seen that if the ring lattice nodes do not have a sufficient amount of neighbors, termination and hence agreement cannot be achieved. This is caused by the random distribution of initial opinions which results in locally differing 0- or 1-majorities.

The performance at values of $\delta < 0.5$ can be improved by rewiring some of the connections of the nodes, according to the method described in Section 3.1. Fig. 2 shows the variation of the agreement rate with the rewiring probability γ. We see that even if only a small proportion of the network can be queried, the protocol can come to agreement in an honest setting if a sufficient amount of nodes are rewired. For example, if the queryable proportion of the network is $\delta = 0.1$, it is sufficient if $\approx 30\%$ of the nodes get rewired.

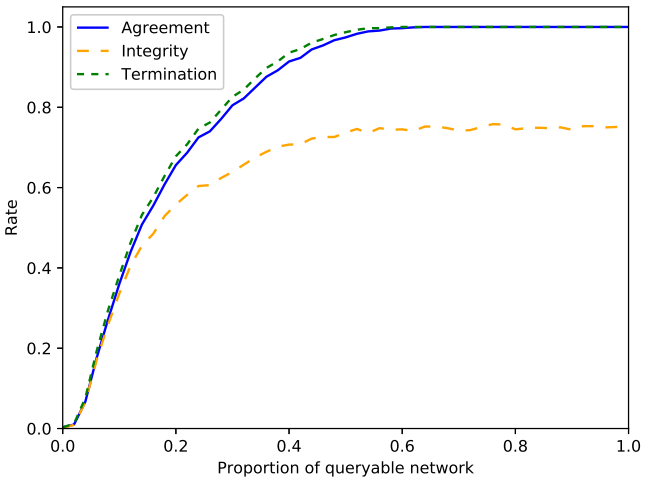

Fig. 1: Termination, agreement and integrity rate as a function of the proportion δ of the network that is queryable by a node, for a ring lattice graph.

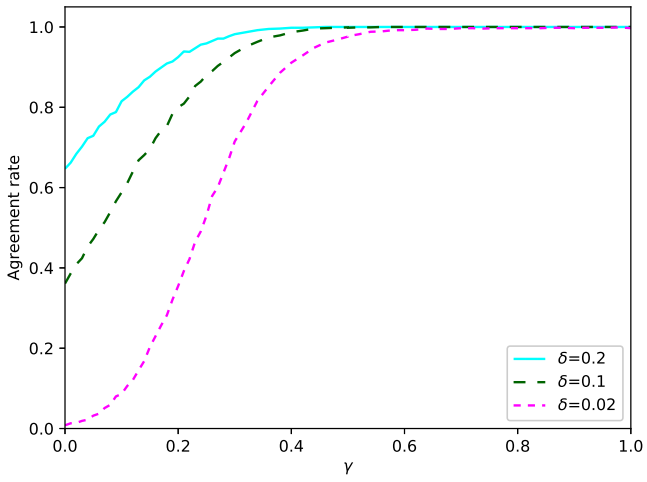

Fig. 2: Agreement rate as a function of the rewiring probability γ for several values of δ.

For sake of simplicity and to enable comparison with theoretical results we assume a complete graph for the network in the following sections. However, as we have shown in this section the partial network view has noticeable effects on the termination and agreement rate. We will, therefore, further investigate on the effects of a partial network view in Section 4.5.

4.4 Integrity failure

An adversary may attempt to change the initial majority-opinion in the network. If successful we say that the protocol suffers an integrity failure. We distinguish the integrity failure into two types. Firstly the adversary may attempt to change the opinion for which initially the majority opinion is 0, and we denote a successful inversion of the opinion as a 0-integrity failure. Secondly, if initially, the majority of nodes would vote 1 an integrity failure would be called 1-integrity failure.

The most effective strategy for the adversary to achieve an integrity failure is to continuously vote the opposite of the initial majority. We will therefore, focus on this strategy in this section.

4.4.1 The 0-integrity failure

In order to assess the capability of the protocol to support an asymmetric behavior as described in the beginning of Section 2, we investigate the likelihood of a 0-integrity failure with the strategy MinVS defined in Section 3.2.5.

As discussed in Section 2 the asymmetry is enabled through the introduction of the initial threshold τ. Since this threshold is introduced to support the asymmetric importance of the two integrity failures, we investigate the protocol in a worst-case scenario, where 51% of the honest nodes have opinion 0, i.e., $p_0 = 0.49$.

Fig. 3 shows the 0-integrity rate with the initial threshold τ for several values of proportions of nodes controlled by the adversary q. The fact that the integrity rate resembles a staircase function can be explained by the fact that η's can only take a finite number of values. As expected the integrity rate increases with τ. However, for the default parameter set (see Table 1) the integrity rate is limited below 1 for $q = 0.2$. On the other hand, by comparing Fig. 3 and Fig. 4 it can be seen that for $q = 0.1$ a range for the initial threshold exists for which both 0-Integrity and 1-integrity is fully achieved.

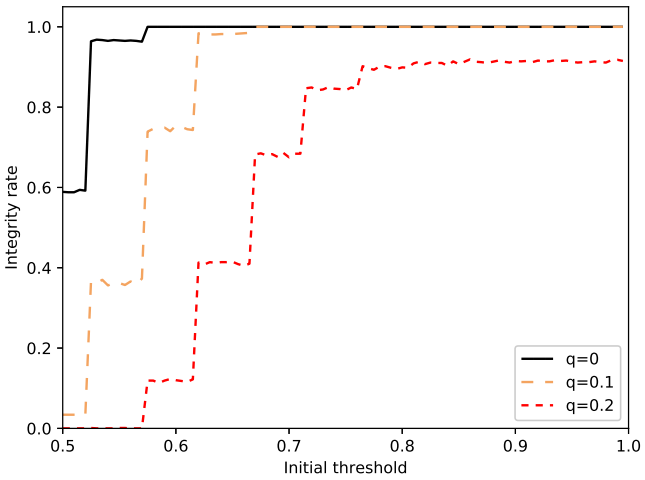

Fig. 3: 0-Integrity rate as a function of the initial threshold τ with $p_0 = 0.49$.

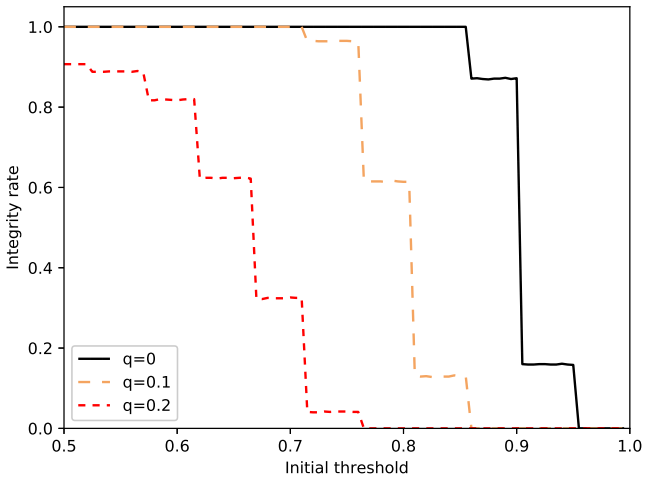

Fig. 4: 1-Integrity rate as a function of the initial threshold τ with $p_0 = 0.9$.

4.4.2 The 1-integrity failure

In order to protect against a 0-integrity failure the threshold has to be above 0.5, i.e., $\tau > 0.5$. Albeit a high value for τ is a good protection mechanism against a 0-integrity failure it can increase the possibility for a 1-integrity failure. The initial threshold should be, therefore, tuned carefully. The integrity rate can be particularly important if the initial 1-opinion is a supermajority, i.e., it is close to 1. Hence in this section we set $p_0 = 0.9$. As in the previous section we employ the MinVS.

Fig. 4 shows the integrity rate as a function of the initial threshold τ. The termination and agreement rate are not presented because it is either equal or close to one. We see that if τ is selected too high, 1-integrity is not guaranteed.

Minimum required consecutive rounds

In order to achieve a low message complexity, the final consecutive round $\mathtt{l}$ should be chosen low for the protocol to terminate quickly. Indeed as can be seen from Fig. 5 the protocol performs well even for a small $\mathtt{l}$ if $q = 0.1$. However if the adversary controls a larger proportion of the network the capacity of the protocol to perform well at lower values decreases, as can be seen for $\mathtt{l} < 6$. Furthermore, if the adversary controls more than a certain amount of nodes a higher $\mathtt{l}$ does not always increase the integrity rate, as can be seen for $q = 0.2$.

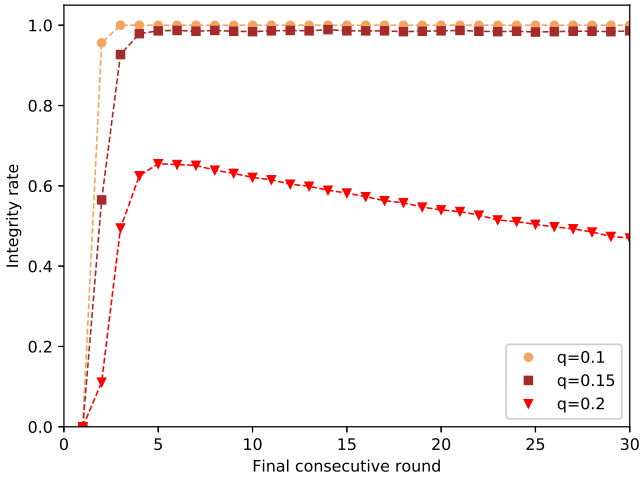

Fig. 5: 1-Integrity rate as a function of the final consecutive round $\mathtt{l}$

Quorum size

The protocol can be improved by increasing the number of queried nodes k, as can be seen from Figs. 6-7. However, this comes at the expense of an increased message complexity. Similarly to the analysis with varied τ, see Fig. 4, we can observe a discrete dependency of the integrity and agreement rate on k. Again this is caused by the discreteness of the possible η-values and their relation to the initial threshold τ.

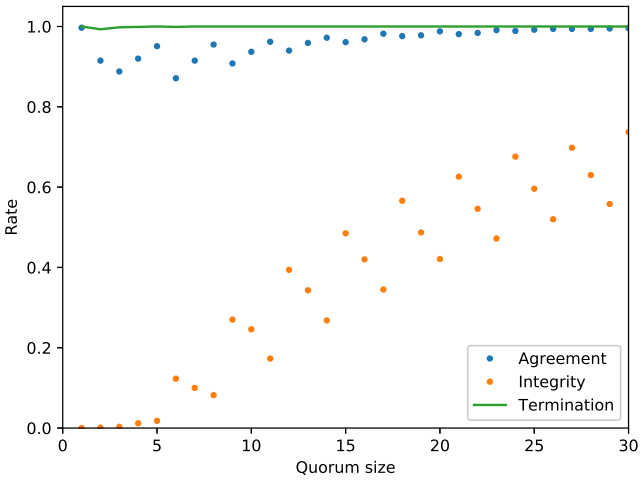

Fig. 6: Termination and integrity rate as a function of the variation of the number of queried nodes k.

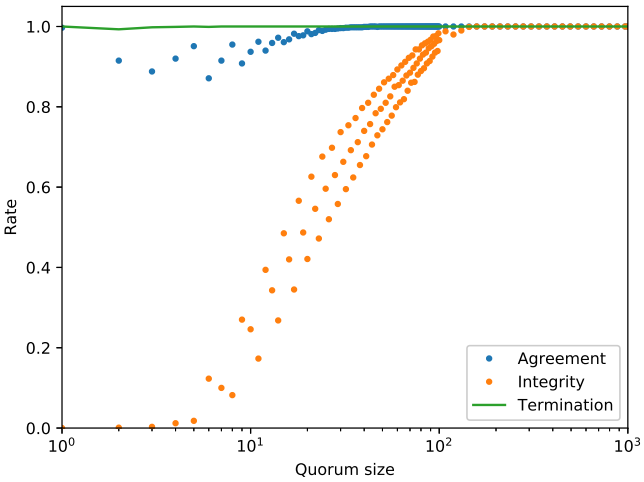

Fig. 7: Termination and integrity rate as a function of the variation of the number of queried nodes k.

Scalability

The performance of the protocol depends little on the overall size of the network for a given parameter setting. In order to display this, we keep the parameter setting to $k = 21$, which leads to an integrity rate below 1. As we increase the size of the nodes we say that an agreement failure occurred if at least 0.1% of the nodes came to a different conclusion. We speak of a termination failure if at least 0.1% of the nodes failed to terminate.

However, as indicated above this can be resolved by increasing k. As can be seen from Fig. 8 the protocol achieves a similar performance as the number of nodes increases. Since $p_0 = \tau$ a large integrity failure is to be expected, see Section 4.2. Fig. 9 shows that although the maximal time to termination $\overline{T}_{max}$ increases, the meantime to termination $\overline{T}_{mean}$ remains almost constant and hence the total message complexity is essentially linear in n, i.e., increases almost $O(n)$.

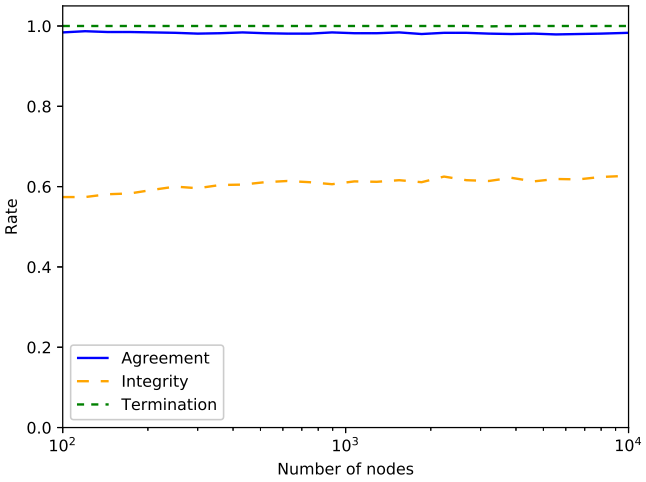

Fig. 8: Termination-, Agreement- and Integrity-Rate as a function of the number of nodes N for $q = 0.2$.

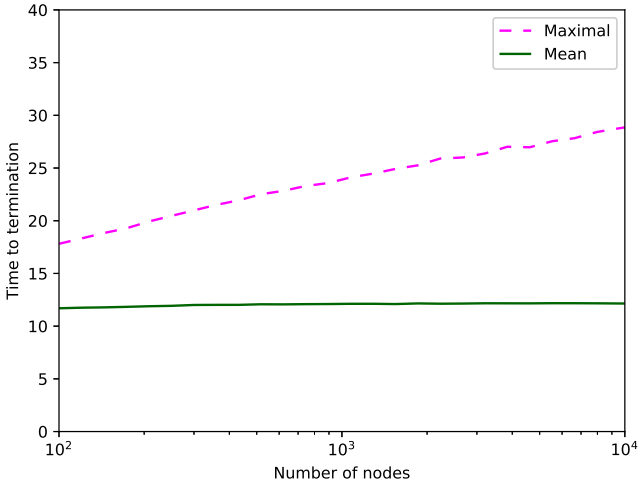

Fig. 9: Time to termination as a function of the number of nodes N for $q = 0.2$.

Initial mean opinion

Generally, the protocol performs best if all of the honest nodes share the same opinion. Fig. 10 shows a "heat map" of the integrity rate with the mean honest initial opinion p_0 and the proportion of adversary nodes q. It can be seen, that if more than 90% of the honest nodes share the same opinion, the protocol can still withstand a 15% attack.

Randomization of the threshold

Fig. 11 shows that the randomness of the thresholds has some impact on the performance of the protocol in terms of integrity rate. However as long as the randomness is not above a certain threshold, i.e., $\beta \geq 0.3$, the protocol can withstand an adversary proportion of about 15%.

We also analyze the scenario where the 0-integrity failure described in Section 4.4.1 is of less concern. This, for example, may be the case if we are certain that the initial opinion is only in one of the two scenarios: the mean initial opinion is either a super-minority or a super-majority, i.e., $p_0 \ll 0.5$ or $p_0 \gg 0.5$, respectively. Under these circumstances it would not be necessary to provide an initial threshold $\tau > 0.5$. Figs. 10-11 show the extend of the region for which the integrity rate is one, if $\tau = 0.666$ and $\tau = 0.5$. From Fig. 10 it can be seen that this region is increased, in particular for lower p_0. Similarly, Fig. 11 shows that a integrity rate of 1 is achieved for values of β close to 0.5.

4.5 Termination and agreement failure

In this section, we investigate the worst-case scenario for termination and agreement failure. An initial average opinion p_0 close to the initial threshold τ enables a potential attacker the most to achieve an agreement or termination failure.

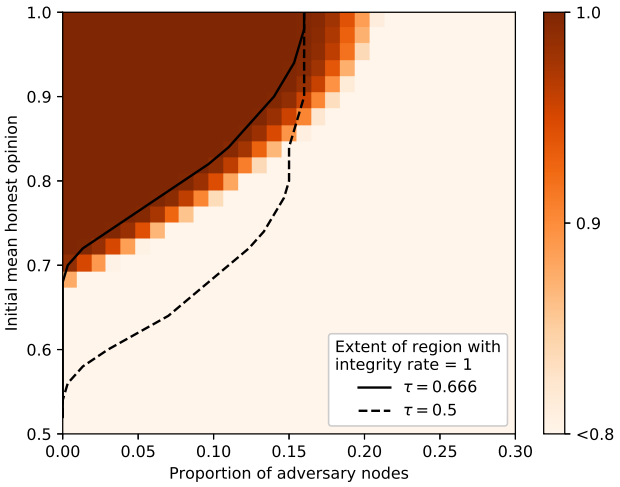

Fig. 10: Integrity rate as a function of q and p_0.

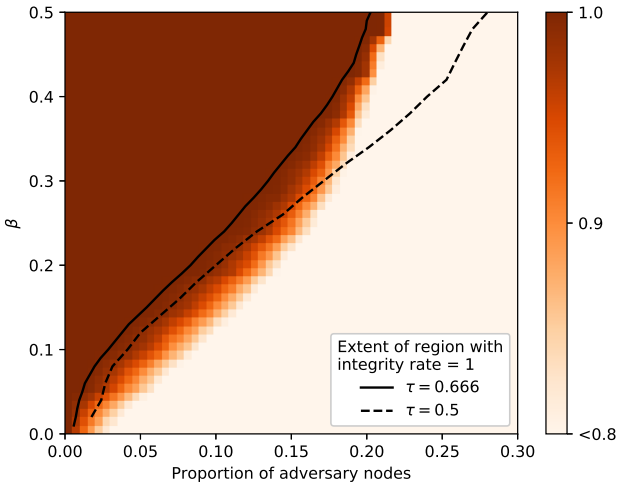

Fig. 11: Integrity rate as a function of q and β.

For instance, the protocol of a simple majority rule, see Section 2.2, may be prevented from terminating already with a small percentage of malicious nodes. For this reason, we set $p_0 = \tau$ in this section.

Intermediate opinion distribution

Fig. 12 shows the development of the discrete η's distribution for the following three scenarios for a single simulation run:

a) Firstly we employ the cautious strategy IVS, see Section 3.2.5, against the protocol without randomness, i.e., $\beta = 0.5$ and $q = 0.3$. It can be seen that the strategy is able to keep the η distribution close to 0.5 for several rounds. Due to the randomness introduced by the quorum selection the adversary ultimately fails to prevent the protocol from terminating. It should be noted that the shown simulation results for this case represent a worst-case scenario, and that for most simulation runs the final termination round remained below $t = 30$.

b) In the second figure we apply the Berserk strategy MVS, see Section 3.2.5 with no random threshold and the much lower adversary proportion $q = 0.1$. We see that the strategy succeeds in splitting the nodes into two groups and that the protocol can be prevented from terminating.

c) In the third scenario randomness is added; i.e., $\beta = 0.3$. Due to the random threshold within a short time (less than round 5) a supermajority of the nodes' opinions is formed, i.e., the opinions converge to one value, in this case 0. After the convergence, the adversary still manages to affect the $\eta's$ of some of the nodes, as can be noticed through the formation of the small group in proximity of 0. However, since β is large enough, the strategy fails to flip a large enough amount of node's opinion. Ultimately at about round $\mathtt{l} = 10$ the nodes start to finalize and the protocol terminates shortly after.

Comparison of adversarial strategies

Fig. 13 shows the termination and agreement rate with the proportion of adversarial nodes for several adversarial strategies. As can be seen with the introduction of the random thresholds the protocol is resilient against the investigated attacks in terms of termination and agreement failure for a large proportion of adversarial nodes. It can also be seen that the Berserk strategy MVS is significantly more severe than IVS.

We assess the resilience against a Berserk adversary given each node only has a partial view of the network, by employing a Watts-Strogatz graph, see Sections 3.1 and 4.3. Fig. 14 shows the termination and agreement rate with δ. It can be seen, that individual nodes do not require a complete network view in order to come to agreement. Analogous to the analysis for Fig. 2, if the network view is partially randomized, i.e., $\gamma > 0$, a smaller network view

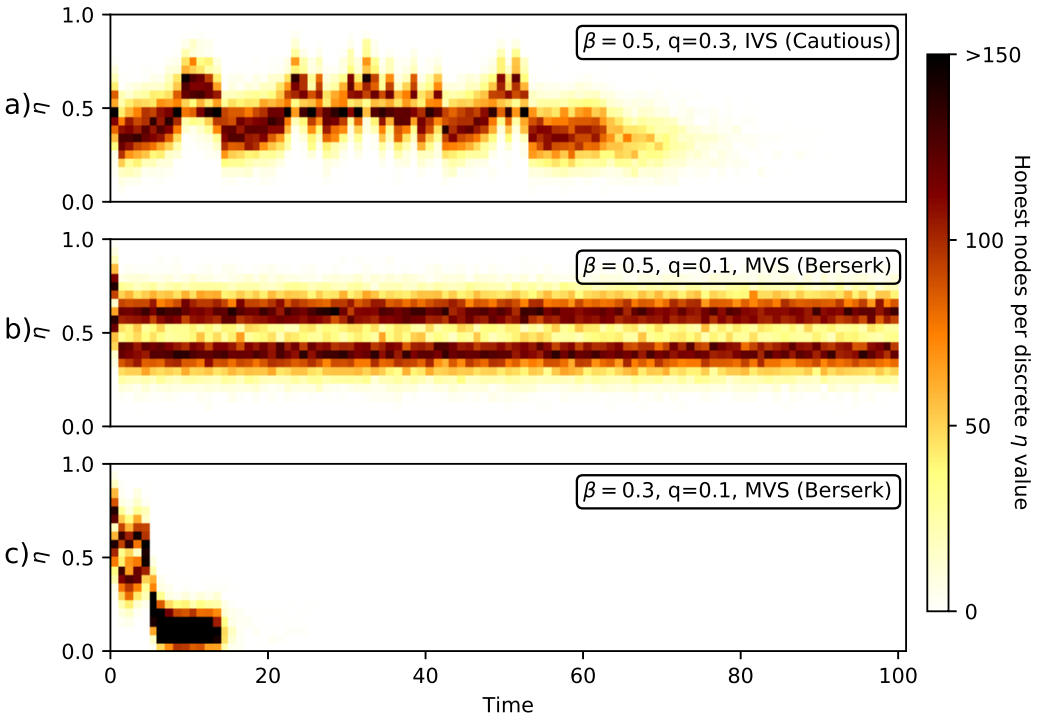

Fig. 12: Evolution of the number of undecided nodes for a given η-value.

may be sufficient. Again MVS appears more severe than IVS. Therefore, for the remainder of this section, we will focus on the more troublesome MVS.

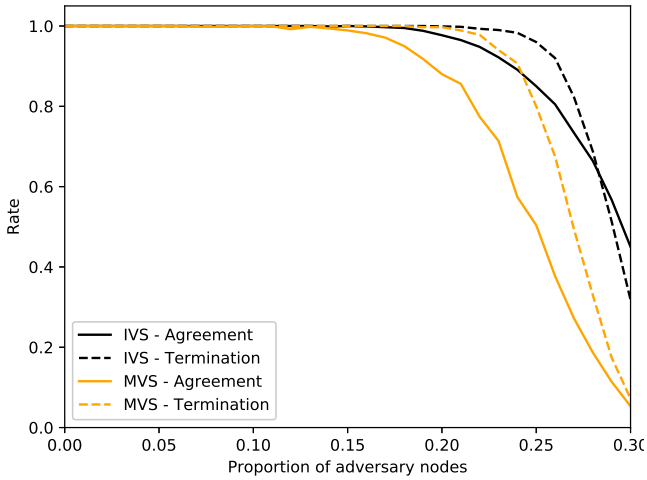

Fig. 13: Termination and Agreement rate as a function of q.

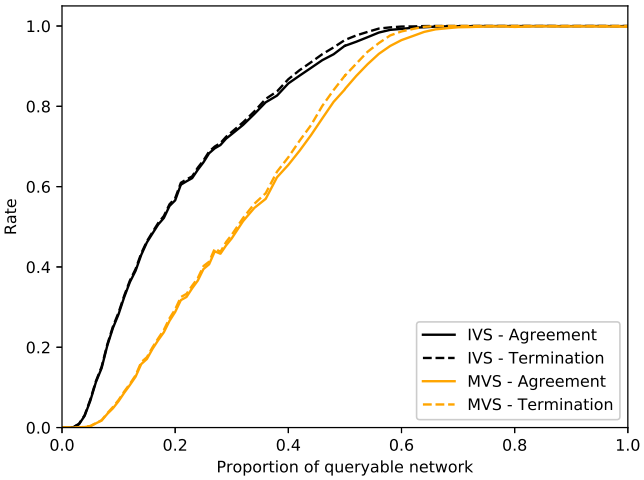

Fig. 14: Termination and agreement rate for a Watts-Strogatz graph as a function of the parameter δ.

Randomization of the threshold

Figs. 15-16 show the variation of the termination and agreement rate with q and β. It can be seen that if no randomness is employed the protocol can be prevented from terminating. Note that this case represents the RMC protocol. Introducing randomness via decreasing β improves the termination as well as the agreement rate, with an optimum at about $\beta = 0.3$. We study the dependency of the protocol on the frequency of a random threshold. More precisely, we

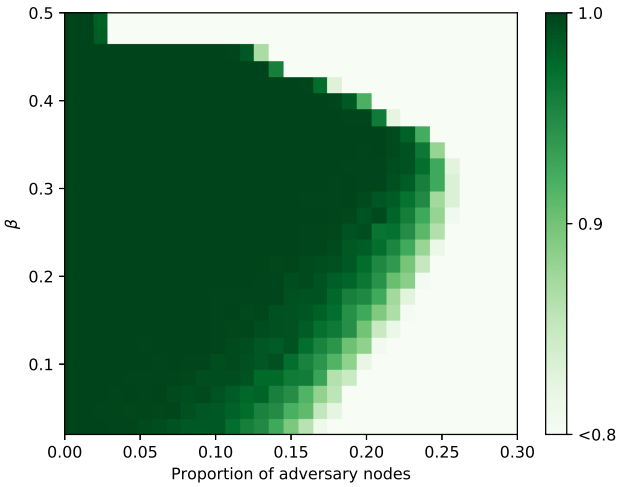

Fig. 15: Termination rate

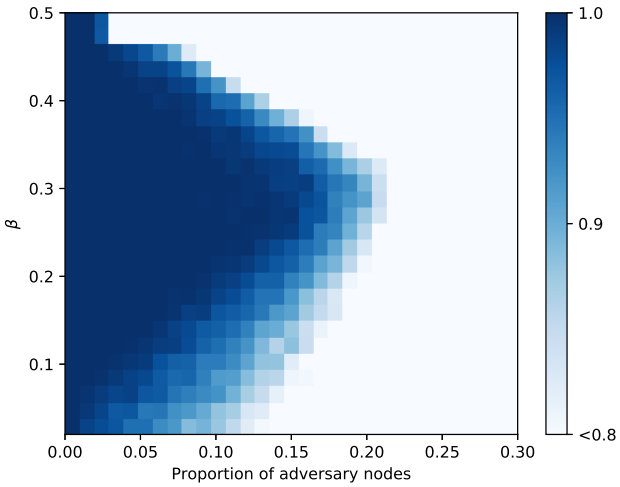

Fig. 16: Agreement rate

assume that the protocol uses in a given round a common random number for the threshold with probability r, or sets the threshold to 0.5 otherwise. From Fig. 17 it can be seen that the agreement and integrity rate increase with r, and that in order to achieve a high agreement rate a random number is not required for every round. Note that for $r = 0$ the protocol simplifies to the RMC protocol. Since $p_0 = \tau$, the integrity rate converges to a value less than 1, see also Section 4.2. Furthermore, even just for a very small r the protocol manages to terminate entirely before `maxIt`. This also reflects in the total message complexity which is significantly reduced with the introduction of the randomness, as can be seen from Fig. 18.

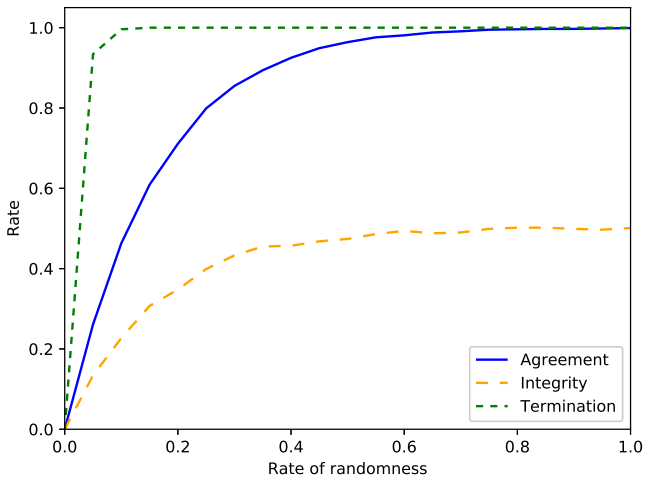

Fig. 17: Termination, agreement and integrity rate as a function of the probability r for a round having a random threshold.

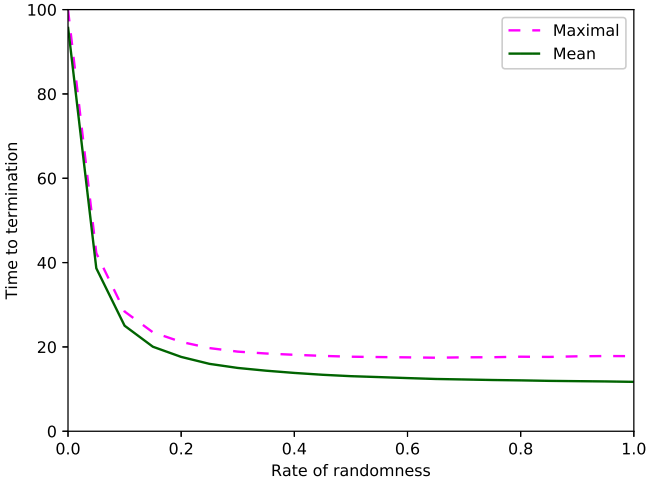

Fig. 18: Time to termination as a function of the probability r for a round having a random threshold.

Impact of the initial mean opinion

We study how the protocol performs if p_0 is different from the first threshold τ. Figs. 19-20 show the termination and agreement rate with p_0 and q. It can be seen, that as expected the agreement rate is affected most for p_0 close to the initial threshold. It can be seen that p_0 has no impact on the performance of the protocol below a certain value of q. This suggests that the addition of the randomness provides good protection for any mean initial honest opinion. Furthermore and as expected, Fig. 20 shows that the agreement rate is affected most by this strategy if p_0 is close to τ.

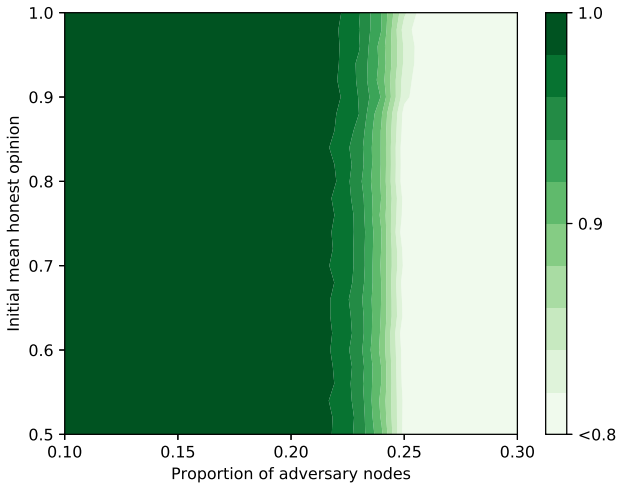

Fig. 19: Termination rate

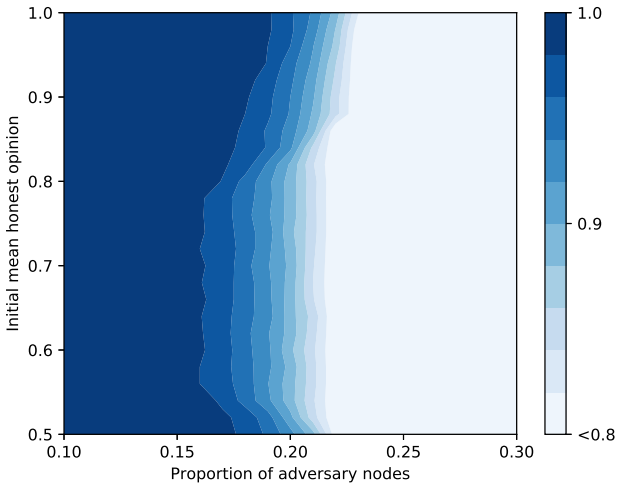

Fig. 20: Agreement rate

5 Discussion

5.1 Synchronicity

In the presence of faults, synchronous and asynchronous systems are fundamentally different in terms of message or time complexity, but also in terms of what type of problems can be solved. In this paper we focused on the synchronous setting; all nodes update their states simultaneously and the common random number serves as a natural clock signal. Due to the local update rules the majority dynamics and FPC may also work in an asynchronous setting. However, the FPC will lose the additional feature of the common random threshold. Figures 17 and 18 indicate that the common random thresholds is only needed at some proportion of the steps. The nature of the protocol also suggests that the performances of the FPC won't suffer if only some positive fraction of the honest nodes share the same random threshold from time to time. The details on the minimal amount of "synchronization" are likely to depend on the precise nature of the system and should be elaborated in additional studies.

5.2 Generating random numbers

The FPC requires the system to generate, from time to time (more precisely, once in each round), a random number available to all the participants (this is very similar to the "global-coin" approach used in many works on Byzantine consensus, see e.g., [27]). In the case of distributed systems these random numbers may be provided by a trusted source [20].

We observe that such random number generation can be done in a decentralized way as well (provided that the proportion of the adversarial nodes is not too large) by leveraging on cryptographic primitives such as verifiable secret sharing, threshold signatures, or verifiable delay functions; see, e.g., [21, 22, 23]. It is important to observe that, as shown in Figures 17 and 18, even if from time to time the adversary can get (total or partial) control of the random number, this can only lead to delayed consensus without agreement or termination failure. Also, not all honest nodes need to agree on the same number. We verified this claim through simulations, but it is plausible since an adversarial

behavior can be considered a worst-case node without access to the random number. Let us note that the theoretical results in [19] can be generalized to non-uniform random variables. Therefore, the protocol is robust towards a bias in the random numbers. We believe that this robustness of the protocol towards a small bias and possibly unreliable source of randomness is one of its main advantages; the protocol can successfully be implemented even in the absence of a "perfect" random number generator. We refer to [44, 45] for an overview of the difficulties in the generation of decentralized random numbers.

5.3 Incentives

In FPC, nodes must take part in the consensus finding. Throughout the paper, we assumed that every honest node has enough incentive to vote and participate in the consensus protocol. FPC can be applied to various situations, and the incentives may differ from case to case. For instance, the protocol may be used in a permissioned network, where participants decided beforehand on how to distribute possible gains of the consensus outcome. This may apply to committees in a smart contract, permissioned distributed ledgers, or other distributed systems. In feeless situations, participants benefit from a functioning infrastructure and may therefore be incentivized to participate and maintain the safety of the system.

6 Conclusions

In this paper, we compare leaderless binary majority consensus protocols in various faulty and Byzantine infrastructures. We focus on the fast probabilistic consensus protocol (FPC) introduced in [19]. In this protocol, randomness is added to a random majority consensus protocol in order to improve performance in the presence of Byzantine actors. We introduce local stopping rules to enable nodes to conclude individually and automatically. Furthermore, the threshold for majority in the first round is applied at a fixed value > 0.5 to support asymmetric importance of the 0- and 1-integrity.

We define several adversarial strategies to analyze the performance of the protocol in Byzantine environment. An adversary can be cautious by answering in the same round with the same opinion to everyone, or Berserk by responding to different queries with different opinions. Although the latter may be detectable, we assume in this paper that no such detection mechanism is implemented in the protocol to weaken or prevent a Berserk attack. As a consequence, the Berserk attack can lead to more frequent agreement or termination failures.

We employ simulation tools to study the performance of the protocol with a high statistical significance. In order to assess the performance of the protocol, we study the rate at which the protocol concludes successfully. This is measured with respect to three types of failures: agreement, termination and integrity failure. For the latter, we distinguish also between a 0- and a 1-integrity failure.

In various real-world situations, nodes may not have a complete network view. We study the protocol when nodes are only capable of querying parts of the network. In order to do so, we employ a Watts-Strogatz model to create a graph that provides a graph with the edges connecting nodes that can query each other. We show that a partial network view of about 50% is sufficient if we consider the conservative case of a ring graph. We also show that if some of the connections on the ring graph are randomly rewired, the protocol performs well with a much smaller network perception per node.

From the defined strategies we apply the most efficient one to evaluate the integrity rate. We assume the most conservative case for the 0-integrity, since we analyze this failure by assuming that the initial honest opinion is 49%. We show that there is a competition between securing the 0- and the 1-integrity rate. Nevertheless, albeit we assume the most conservative case for the 0-integrity the protocol can prevent both integrity failures if the adversary has up to 10% control over the network. As expected the performance of the protocol improves if the quorum size is increased and it is sufficient to query a fraction of the nodes to avoid an integrity failure. We investigate the performance of the protocol when the network is scaled and it is shown that the protocol performs similarly with the same parameter settings per node. More particularly this means that the average message complexity per node remains constant and that, therefore, the total message complexity in the network increases essentially as $O(n)$. We also show that dependent on the application it may be necessary to reduce the range of the random threshold since the protocol can withstand a higher number of byzantine nodes when the threshold remains close to 0.5.

We analyze the termination and agreement rate of the protocol for a cautious and Berserk Byzantine strategy and for a worst-case initial setup. We show that the protocol can be prevented from terminating and coming to agreement for an extensive amount of time and that the introduction of the randomness drastically reduces the effects of this attack. We note that for the same proportion of adversarial nodes the Berserk strategy is significantly more severe than the cautious strategy and can reach agreement failure much more efficiently. We show that similar to a network occupied by only honest actors the protocol performs well if each node only has a view of the network of about 50%. Generally, the protocol appears sensitive to the spread of the random thresholds and an optimum parameter set exists. On the other hand, it is also not necessary for the nodes to receive a random number for the random threshold every round, since the performance of the protocol remains the same even if a simple majority rule is applied occasionally.

References

[1] M. Barborak, A. Dahbura, M. Malek, The consensus problem in fault-tolerant computing, ACM Computing Surveys 25 (2) (1993) 171–220.

[2] A. Gee, R. Cipolla, Fast visual tracking by temporal consensus, Image and Vision Computing 14 (2) (1996) 105–114.

[3] M. Mitchell, P. T. Hraber, J. P. Crutchfield, Revisiting the edge of chaos: Evolving cellular automata to perform computations, Complex Systems 7 (1993).

[4] R. Olfati-Saber, J. S. Shamma, Consensus filters for sensor networks and distributed sensor fusion, in: Proceedings of the 44th IEEE Conference on Decision and Control, 2005, pp. 6698–6703.

[5] A. Simonetto, T. Keviczky, R. Babuška, Distributed nonlinear estimation for robot localization using weighted consensus, in: 2010 IEEE International Conference on Robotics and Automation, 2010, pp. 3026–3031.

[6] M. Alighanbari, J. P. How, An unbiased Kalman consensus algorithm, in: 2006 American Control Conference, 2006, p. 6.

[7] C. Castellano, S. Fortunato, V. Loreto, Statistical physics of social dynamics, Rev. Mod. Phys. 81 (2009) 591–646.

[8] E. Mossel, J. Neeman, O. Tamuz, Majority dynamics and aggregation of information in social networks, Autonomous Agents and Multi-Agent Systems 28 (3) (2013) 408–429.

[9] Z. Zheng, S. Xie, H. Dai, X. Chen, H. Wang, An overview of blockchain technology: Architecture, consensus, and future trends, in: 2017 IEEE International Congress on Big Data (BigData Congress), IEEE, 2017, pp. 557–564.

[10] M. C. Pease, R. E. Shostak, L. Lamport, Reaching agreement in the presence of faults, J. ACM 27 (1980) 228–234.

[11] M. J. Fischer, N. A. Lynch, M. S. Paterson, Impossibility of distributed consensus with one faulty process, Journal of the ACM 32 (2) (1985) 374–382.

[12] P. Gács, G. L. Kurdyumov, L. A. Levin, One-dimensional Uniform Arrays that Wash out Finite Islands, in: Problemy Peredachi Informatsii, 1978.

[13] A. Gogolev, N. Marchenko, L. Marcenaro, C. Bettstetter, Distributed binary consensus in networks with disturbances, ACM Trans. Auton. Adapt. Syst. 10 (3) (2015) 19:1–19:17.

[14] S. Kar, J. M. F. Moura, Distributed average consensus in sensor networks with random link failures, in: 2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP '07, Vol. 2, 2007, pp. II–1013–II–1016.

[15] A. A. Moreira, A. Mathur, D. Diermeier, L. Amaral, Efficient system-wide coordination in noisy environments, Proc. Natl. Acad. Sci. U. S. A. 101 (2004) 12085–12090.

[16] M. Saks, F. Zaharoglou, Wait-free k-set agreement is impossible, Proceedings of the twenty-fifth annual ACM symposium on Theory of computing - STOC '93 (1993).

[17] D. B. Kingston, R. W. Beard, Discrete-time average-consensus under switching network topologies, in: 2006 American Control Conference, 2006, pp. 6 pp.–.

[18] I. Benjamini, S.-O. Chan, R. O'Donnell, O. Tamuz, L.-Y. Tan, Convergence, unanimity and disagreement in majority dynamics on unimodular graphs and random graphs, Stochastic Processes and their Applications 126 (9) (2016) 2719 – 2733.

[19] S. Popov, W. J. Buchanan, FPC-BI: Fast Probabilistic Consensus within Byzantine Infrastructures, Journal of Parallel and Distributed Computing 147 (2021) 77–86.

[20] J. Kelsey, L. T. Brandão, R. Peralta, H. Booth, A reference for randomness beacons: Format and protocol version 2, Tech. rep., National Institute of Standards and Technology (2019).

[21] S. Popov, On a decentralized trustless pseudo-random number generation algorithm, Journal of Mathematical Cryptology 11 (1) (2017) 37–43.

[22] E. Syta, P. Jovanovic, E. K. Kogias, N. Gailly, L. Gasser, I. Khoffi, M. J. Fischer, B. Ford, Scalable bias-resistant distributed randomness, in: 2017 IEEE Symposium on Security and Privacy (SP), Ieee, 2017, pp. 444–460.

[23] D. Boneh, J. Bonneau, B. Bünz, B. Fisch, Verifiable delay functions, in: Annual International Cryptology Conference, Springer, 2018, pp. 757–788.

[24] M. Saks, N. Shavit, H. Woll, Optimal time randomized consensus - making resilient algorithms fast in practice, in: Proceedings of the Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA '91, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1991, pp. 351–362.

[25] J. Aspnes, Fast deterministic consensus in a noisy environment, Journal of Algorithms 45 (1) (2002) 16–39.

[26] C. von Clausewitz, On War, Princeton University Press, 1984.

[27] M. K. Aguilera, S. Toueg, The correctness proof of Ben-Or's randomized consensus algorithm, Distributed Computing 25 (5) (2012) 371–381.

[28] M. Ben-Or, Another advantage of free choice: Completely asynchronous agreement protocols (extended abstract), in: Proceedings of the 2nd ACM Annual Symposium on Principles of Distributed Computing, Montreal, Quebec, 1983, pp. 27–30.

[29] G. Bracha, Asynchronous Byzantine agreement protocols, Information and Computation 75 (2) (1987) 130–143.

[30] P. Feldman, S. Micali, An optimal probabilistic algorithm for synchronous Byzantine agreement, in: International Colloquium on Automata, Languages, and Programming, Springer, 1989, pp. 341–378.

[31] R. Friedman, A. Mostefaoui, M. Raynal, Simple and efficient oracle-based consensus protocols for asynchronous Byzantine systems, IEEE Transactions on Dependable and Secure Computing 2 (1) (2005) 46–56.

[32] M. O. Rabin, Randomized Byzantine generals, in: 24th Annual Symposium on Foundations of Computer Science (sfcs 1983), IEEE, 1983, pp. 403–409.

[33] J. Liu, W. Li, G. O. Karame, N. Asokan, Scalable Byzantine consensus via hardware-assisted secret sharing, IEEE Transactions on Computers 68 (1) (2018) 139–151.

[34] T. Crain, V. Gramoli, M. Larrea, M. Raynal, Dbft: Efficient leaderless Byzantine consensus and its application to blockchains, in: 2018 IEEE 17th International Symposium on Network Computing and Applications (NCA), IEEE, 2018, pp. 1–8.

[35] A. Miller, Y. Xia, K. Croman, E. Shi, D. Song, The honey badger of bft protocols, in: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, ACM, pp. 31–42.

[36] M. Tanaka-Yamawaki, S. Kitamikado, T. Fukuda, Consensus formation and the cellular automata, Robotics and Autonomous Systems 19 (1) (1996) 15 – 22, evolutional Robots.

[37] D. Wolz, P. P. B. de Oliveira, Very effective evolutionary techniques for searching cellular automata rule spaces, J. Cellular Automata 3 (2008) 289–312.

[38] M. Marques-Pita, L. M. Rocha, Conceptual structure in cellular automata - the density classification task, in: ALIFE, 2008.

[39] M. A. Abdullah, M. Draief, Global majority consensus by local majority polling on graphs of a given degree sequence, Discrete Applied Mathematics 180 (2015) 1 – 10.

[40] B. Gärtner, A. N. Zehmakan, Majority model on random regular graphs, Lecture Notes in Computer Science (2018) 572–583.

[41] M. Ostilli, E. Yoneki, I. X. Leung, J. F. Mendes, P. Lió, J. Crowcroft, Statistical mechanics of rumour spreading in network communities, Procedia Computer Science 1 (1) (2010) 2331 – 2339, iCCS 2010.

[42] D. J. Watts, S. H. Strogatz, Collective dynamics of 'small-world'networks, Nature 393 (6684) (1998) 440–442.

[43] R. van der Hofstad, Random graphs and complex networks. Vol. 1, Cambridge Series in Statistical and Probabilistic Mathematics, [43], Cambridge University Press, Cambridge, 2017.

[44] B. Bünz, S. Goldfeder, J. Bonneau, Proofs-of-delay and randomness beacons in Ethereum, 2017.

[45] D. Hurley-Smith, C. Patsakis, J. Hernandez-Castro, On the unbearable lightness of fips 140-2 randomness tests, IEEE Transactions on Information Forensics and Security (2020) 1–1doi:10.1109/TIFS.2020.2988505.