Submitted to *Entropy*. Pages 1 - 18.

OPEN ACCESS

Article

Randomized Binary Consensus with Faulty Agents

Alexander Gogolev [1,2]*, Lucio Marcenaro [2]

[1] Institute of Networked and Embedded Systems, University of Klagenfurt, Lakeside B02a, Klagenfurt, Austria

[2] Department of Naval, Electric, Electronic and Telecommunication Engineering, University of Genoa, Via Opera Pia 11a, Genoa, Italy

* Author to whom correspondence should be addressed; alexander.gogolev@aau.at, +(43)46327003640.

Version May 13, 2014 submitted to Entropy. *Typeset by LaTeX using class file mdpi.cls*

Abstract: This paper investigates self-organizing binary majority consensus disturbed by faulty nodes with random and persistent failure. We study consensus in ordered and random networks with noise, message loss, and delays. Using computer simulations, we show that: (a) explicit randomization by noise, message loss, and topology can increase robustness towards faulty nodes, (b) commonly-used faulty nodes with random failure inhibit consensus less than faulty nodes with persistent failure, and (c) in some cases such randomly failing faulty nodes can even promote agreement.

Keywords: Binary Consensus; Randomized Consensus; Self-Organizing Systems; Faulty Agents; Byzantine Failure; Density Classification;

---

1. Introduction

The use of consensus algorithms is reported in various systems, ranging from distributed database management [1], to detection [2], and mission planning [3].

Networked algorithms for distributed decision making, operating in real-life systems should be robust towards various disturbances. Studies on robustness of consensus algorithms investigate the influence of noise [4,5], message loss [6], random topologies [7], and faulty node behavior [8]. Faulty nodes are often considered as one of the main impediments to consensus [8,9].

Scholars approach the problem of fault tolerance with fault-detection [10,11], increasing system-wide synchrony [12,13], and randomization. Randomization is a technique that utilizes random processes (that are often considered as negative disturbances) to increase fault tolerance [14]. Unlike the fault-detection, randomization does not significantly increase the complexity of the algorithm and does not require system-wide adjustments such as an imposed synchrony. Studies show that randomization can be beneficial for consensus, both in terms of efficiency [14,15] and fault tolerance [14,16]. Recent studies show that such beneficial randomization can sometimes be provided explicitly by noise [17,18] or errors [15].

This motivated us to investigate the impact of faulty nodes on self-organized binary majority consensus. In this article we focus on faulty nodes with persistent and random failure and different layout over the network. We study influence of the faulty nodes in ring lattices and Watts-Strogatz [19] and Waxman [20] networks randomized by message loss, additive noise, and topology randomization.

We show that the decrease in efficiency induced by faulty nodes can be mitigated by randomization of different origin. We show that commonly-used faulty nodes with random failure and faulty nodes with random full failure are less adverse for consensus than faulty nodes with persistent failure. Finally, we show that in some cases randomization by faulty nodes can even promote consensus.

The article is organized as follows. Section 2 gives a short overview of related work. Section 3 describes system modeling. Section 4 presents simulation results and analysis. Finally, Section 5 concludes the article.

2. Related Work

Self-organization is a phenomenon often observed in systems, where simple local interactions of networked agents can produce global coordination [21–23]. Networked control algorithms, inspired by such systems can be efficient and robust [24]. Binary majority consensus exhibits self-organizing features: it is performed by simple rules in a distributed manner, and can show an increase in efficiency with stochastic intrusions [15,25]. Studies on self-organized consensus can provide practical insights on engineering of networked control systems. In this article we focus on simple binary majority consensus algorithms to investigate whether randomization can have positive effect not only on its efficiency but also on its robustness.

2.1. Distributed Binary Majority Consensus

In this article we focus on a *wait-free binary majority consensus* — a sub-class of the *general consensus*. Consensus algorithms are a class of algorithms that aim to provide common decision for all nodes in a networked system and satisfy the following conditions [14]:

1. *Agreement*. All nodes choose the same value.
2. *Termination*. All non-faulty nodes eventually decide.
3. *Validity*. The common output value is an input value of some node.

Let us briefly specify the *wait-free Binary Majority Consensus* (BMC) in this perspective. *Binary majority consensus* is a sub-class of consensus algorithms with specific *agreement* and *termination* conditions. Binary majority consensus algorithms provide that a network agrees on state that is selected out of limited set of binary inputs, generally defined as $\{0, 1\}$ or $\{-1, 1\}$. The agreed state should correspond to the *initial majority* of all states in the network. A *wait-free* requirement specifies a *termination* condition: such algorithms are terminated after a predefined time T, whether agreement was reached or not. Wait-free binary majority consensus can be beneficial in real-life networked systems, where the termination time is important. Time limitation, however, can lead to lower efficiency and higher sensitivity to disturbances [15,26,27].

Strict termination conditions of BMC make it difficult to guarantee the agreement. Due to this efficiency of the binary majority consensus is registered as *convergence rate*, R — a fraction of initial network configurations that result in successful agreement. BMC has been actively studied since Gacs *et al.* [26] introduced the Gacks-Kurdymov-Levin (GKL) consensus that provides $R \simeq 82\%$. This convergence rate was registered in a synchronized ring lattice of $N = 149$ nodes, where each node is connected to its $2K = 6$ neighbors. Since then most scholars adopted this network as a reference case for comparing the convergence rate of BMC algorithms. In the last several decades several solutions slowly advanced the R up to 86% [28]. Land and Belew [29] show that a *deterministic* algorithm cannot solve the consensus with 100% efficiency in a reference setup. This motivated research on *randomized* solutions that advanced the convergence rate up to 90% [15,30]. However, proposed solutions only work in a limited set of synchronous networks, and were not tested for robustness towards faulty node behavior.

2.2. Fault Tolerance of Consensus

Studies on the robustness and the fault tolerance of general consensus algorithms consider faulty nodes as one of the main impediments to consensus. Faulty nodes are generally represented as Byzantine faulty nodes — nodes that can have any arbitrary failure, except full failure. Early study by Pease *et al.* [31] shows that in a *synchronized* networked system of N nodes, M of them being faulty, consensus is possible if $M < \frac{N-1}{3}$. Later, Fischer *et al.* [8] strengthened this condition for *asynchronous* systems, showing that consensus may become impossible with already $M = 1$.

Due to strict termination conditions, BMC can be sensitive to the faulty node behavior. Specific cases of BMC with faulty nodes has been previously studied in [32] and [17]. Thus, [32] reports that Simple Majority (SM) consensus is stronger inhibited by faulty nodes with persistent failure than by faulty nodes with random failure in some networks. Another affect is reported in [17], where it is shown that Gacs-Kurdyumov-Levin (GKL) and SM consensus inhibited by a low number of faulty nodes with persistent failure can restore convergence rate with randomization.

This article complements and extends these works for a wider range of network models, types of disturbances and faulty nodes. We study BMC with faulty nodes in ring lattices, Watts-Strogatz and Waxman networks with various stochastic disturbances. We consider persistently and randomly failing faulty nodes in networks with random and clustered faulty node layout. We show that randomization by topology, noise, and message loss can mitigate the decrease in efficiency induced by faulty nodes of different type. We also show that in some cases faulty nodes with random failure can even promote consensus. Finally we explain and illustrate the mechanisms behind these effects with convergence analysis.

3. Experimental Setup

3.1. Network Model

We investigate BMC in ring lattices and randomized Watts-Strogatz (WS) [19] and Waxman [20] networks. Watts-Strogats graph can produce networks, ranging from ordered grids to Small-World, and fully random networks. Due to this WS networks are widely used to model systems interactions, spanning from technical systems [15] to natural [33], and social networks [34]. Graphs proposed by Waxman [20], on the other hand, are widely used to model human-designed random networks, like internet [35]. These types of systems compliment each other and allow us to compare the efficiency of the algorithms with preceding solutions and cover major network models for areas, where consensus algorithms found their use.

3.1.1. Ring Lattices

To model ring lattices we follow a reference network design introduced in [26]. Such a network is initially created as a one-dimensional cellular automaton of N nodes, connected to their $2K$ closest neighbors. This automaton is then closed in a ring to avoid boundary effects. Such setup is often used to register convergence rate of BMC [30,36,37]. Neighbors of each node $i \in \{0, \dots, N\}$ are split into three sets: set of all neighbors N_i, $||N_i|| = 2K$, $N_i = \{i-K, \dots, i-1, i+1, \dots, i+K\}$, set of left-side neighbors N_l, $||N_l|| = K$, $N_l = \{i-K, \dots, i-1,\}$ and set of right-side neighbors N_r, $||N_r|| = K$, $N_r = \{i+1, \dots, i+K\}$. These sets are further used by consensus algorithms to access neighbors' state information. For comparison purposes, for all algorithms we use $K = 3$, initially defined for GKL in [26]. Here and further we study undirected graphs, and refer to the "link" as the connection between nodes i and j. For ring lattices and WS networks a "link length" between nodes i and j is defined as the difference between their respective indices. For Waxman networks link length is an actual Euclidean distance, randomly chosen in the beginning of simulation. Random and complex networks such as WS and Waxman graphs are often characterized with the path lengths that are composed of multi-hop connections. The simple consensus algorithms studied in this paper only account for the closest, one-hop neighborhood of each node. Due to this we characterize the networks with "link length" and "node degree" rather than a "path length".

3.1.2. Watts-Strogatz Networks

A Watts-Strogats graph can produce networking models ranging from ordered grids to fully random networks. It is initially modeled as one-dimensional ring of N nodes, where each node is connected with the next K nodes. Further, with rewiring probability $P \in [0, 1]$ each link of the node i is substituted with a link to a random node $j \notin \{i-K, \dots, i+K\}$. I.e., at $P = 0$ a network is a $2K$-connected ordered grid of N nodes. At $P = 0.5$ approximately half of the links is substituted with random ones, and the network can be represented as a Small-World graph. Finally, at $P = 1$ all links are random and the network is a fully random graph.

3.1.3. Waxman Networks

A Waxman graph is built as follows. First, for each pair of nodes $i, j \in \{1, 2, \ldots, N\}, i \neq j$, the distance d is randomly uniformly chosen from the interval $(0, 1]$. Next, the nodes are linked with probability

$$\alpha \exp\left(-\frac{d}{\beta}\right) , \tag{1}$$

with parameters $\alpha, \beta \in (0, 1]$. Parameters α and β influence the system as follows. An increasing α yields an increasing link probability, thus increasing the average node degree. An increasing β has an influence similar to that of P in WS networks: it increases the number of long random links compared to short links, thus increasing the average link length in the network. We model sparsely connected Waxman graphs with fixed $\alpha = 0.05$ and $\beta \in [0.01, 0.4]$. Within the given parameter range of β, we limit the average node degree and average link length to match the WS model.

3.2. Consensus Algorithms

At the first time step $t = 0$ every node $i \in \{0, \ldots, N\}$ is randomly assigned with a binary state $\sigma_i \in \{-1, 1\}$. The combination of all N initial states σ_i is called *initial configuration*. The sum of all states in initial configuration $\sum_{i=0}^{i=N} \sigma_i[0]$ is called initial density and denoted as $\rho[0]$.

At every time step $0 \leq t \leq T$ each node updates its state following a given consensus algorithm, based on its own state, and the state information received from neighboring nodes. Within T time steps all nodes are expected to agree on a single state, corresponding to the initial majority (density). I.e., a network is converged if there exist time $t_c \leq T$, so that $\sum_{i=0}^{i=N} \sigma_i[t_c] = N$ for $\rho[0] > 0$, or $\sum_{i=0}^{i=N} \sigma_i[t_c] = -N$ for $\rho[0] < 0$. We use $T = 2N$ as initially defined in [26].

In this article we focus on randomized Gacs-Kurdyumov-Levin and Simple Majority consensus algorithms which we will now briefly describe.

3.3. Simple Majority Consensus

With Simple Majority consensus every node updates its state on a basis of its own state, and the state information received from its neighbors.

$$\sigma_i[t+1] = G\left(\sigma_{i,i}[t] + \sum_{j \in N_i} \sigma_{i,j}[t]\right) . \tag{2}$$

Here, $\sigma_{i,j}[t]$ denotes the state of the node j at the time t received by the node i. The update function $G(x)$ is defined as in [15,25]:

$$G(x) = \begin{cases} -1 & \text{for } x < 0, \\ +1 & \text{for } x > 0. \end{cases} \tag{3}$$

SM consensus is arguably the simplest algorithm for binary majority sorting, and has a balanced design: in ring lattices each node i receives equal number of messages from both sides of the lattice. Due to this SM indicates low convergence rate in ordered and noiseless systems, but in strongly randomized setups it can show high convergence rate [15,17,32], and outperform GKL.

One can see that $G(x)$ is not defined for $x = 0$, which is a valid assumption for undisturbed networks where an odd number of received state messages ensures that their sum is always either negative either positive. However, in noisy networks or in networks with message loss a sum of received state messages can sometimes be equal to 0. For this case we adjust $G(x)$ in a following manner: if a decision cannot be taken (i.e., when the sum of received state messages is equal to 0) the state of the node stays unchanged: $\sigma_i[t+1] = \sigma_i[t]$.

3.4. Gacs-Kurdyumov-Levin Consensus

GKL consensus is known among the best algorithms for binary majority problem [36]. It is simple and efficient, and is often used as a benchmark for new algorithms [30,37,38]. Nodes driven by GKL, update their states as follows. Depending on its own current state, each node chooses which side to receive messages from: if $\sigma_{i,i}[t] < 0$, node i receives state information from the first and the third neighbor to the left, if $\sigma_{i,i}[t] > 0$, it receives information from the first and the third neighbor to the right.

$$\sigma_i[t+1] = \begin{cases} G\Big(\sigma_{i,i}[t] + \sigma_{i,l_1}[t] + \sigma_{i,l_3}[t]\Big) & \text{for } \sigma_{i,i}[t] < 0, \\ G\Big(\sigma_{i,i}[t] + \sigma_{i,r_1}[t] + \sigma_{i,r_3}[t]\Big) & \text{for } \sigma_{i,i}[t] > 0. \end{cases} \tag{4}$$

Here, $l_1,\ l_3$ and $r_1,\ r_3$ are the first and the third neighbors of the node i to the left and to the right, respectively. One can see that essentially GKL is a modification of SM consensus with a built-in state-direction bias. This bias provides for high efficiency of GKL in ring lattices but it can lead to low efficiency if the network structure or the update sequence are disturbed [15,17].

3.5. Update Mode

System-wide synchrony can be crucial for consensus process [12,13]. We simulate systems with synchronous and asynchronous update functions. In the synchronous mode all nodes update their states simultaneously. In the asynchronous mode nodes are updated sequentially, one after each other, according to their indices, i.e., $0 \to N$. To update its state, a node uses the latest available states of its neighbors.

3.6. Initial Configurations

For our simulations we use test sets combined of 10^4 initial configurations. Each initial configuration is composed of N initial states σ_i obtained as a result of a coin-flip operation, returning 1 or -1 with equal probability, as in [15,26].

3.7. Faulty Nodes Modeling

We study faulty nodes with two failure models: faulty nodes with random failure, modeled after Byzantine failure model, and faulty nodes with persistent failure.

We implement faulty nodes as follows. At a starting time $t = 0$, M faulty nodes are added to N non-faulty nodes to avoid bias of the initial configuration. Network topology is then created for all $N + M$ nodes. After adding M faulty nodes to the system they are labeled as faulty and counter consensus according to their failure model.

3.7.1. Faulty Nodes Layout

We use two schemes of faulty nodes layout: clustered and distributed. With clustered layout all faulty nodes are located next to each other. Location of the cluster is randomly chosen at each simulation run. With distributed layout all faulty nodes are randomly placed over the network independent from each other.

3.7.2. Faulty Nodes with Random Failure

We implement faulty nodes with random failure after commonly-used Byzantine random failure with a reduced state space. Such nodes randomly change their broadcasted state, independently from the state information received from their neighbors. We investigate two types of faulty nodes:

- two-state faulty nodes, randomly switching between states $\sigma_M \in \{-1, 1\}$, and
- three-state faulty nodes, switching between $\sigma_M \in \{-1, 0, 1\}$.

The first case presents a faulty node that broadcasts correct and erroneous state information with equal probabilities. The second case additionally implements a state of sending no information, i.e., a full failure.

3.7.3. Faulty Nodes with Persistent Failure

Faulty nodes with persistent failure are modeled as follows. After M faulty nodes are added, they are assigned with a faulty value σ_M, opposite to the initial majority: if $\sum_{i=0}^{i=N} \sigma_i[0] < 0, \sigma_M = 1$, and if $\sum_{i=0}^{i=N} \sigma_i[0] > 0, \sigma_M = -1$. During consensus process such faulty nodes broadcast their state but do not update it. Unlike faulty nodes with random failure, faulty nodes with persistent failure provide enduring inhibition for consensus.

3.8. Additive Noise

To introduce the noise, we modify the system as follows. Recall that in the original system node i receives state information from the node j via state information message $\sigma_{i,j}[t]$. We implement noise added to the received state information by the following transformation:

$$\sigma_{i,j}[t] \to \sigma_{i,j}[t] + \phi_{i,j} \; . \tag{5}$$

Here, a random value $\phi_{i,j}$ is a sample of added noise. We implement two types of noise: Additive White Gaussian Noise (AWGN), where $\phi_{i,j} \sim \mathcal{N}\big(0, (\frac{A}{3})^2\big)$, and Additive White Uniform Noise (AWUN), where $\phi_{i,j} \sim \mathcal{U}\big(-A, A\big)$, with the magnitude $A \in [0, 4]$. Previous studies mostly consider AWGN as the most common noise type in real networks [5,6], and AWUN is generally used to model the response of filters and amplifiers [39]. The range for the noise amplitude A is chosen empirically to account for level of disturbances that not only promote, but also hinder consensus.

3.9. Message Loss

A message loss can inhibit the BMC, since a node decision is based on an odd number of state information messages received from other nodes. If a message is lost, a node can come to a state when the sum of received state messages is equal to zero, and the state of the node stays unchanged. In our model, a state information is lost with the probability $\mathcal{E}_{i,j} \in [0, 1)$, i.e., if a message from node the j to node the i is lost, the received state message $\sigma_{i,j}[t] = 0$:

$$\sigma_{i,j}[t] \rightarrow \begin{cases} \sigma_{i,j}[t], \text{with probability } (1 - \mathcal{E}) \\ 0, \text{ with probability } \mathcal{E} \end{cases} . \quad (6)$$

In our simulations the state information of the node i is also affected by the noise and message loss, i.e., $\sigma_i \neq \sigma_{i,i}$. This scenario corresponds to the problem of distributed detection where nodes with unreliable sensory inputs are expected to agree whether a detected event took place. The other possible scenario assumes influence of noise and message loss only in node-to-node communication, i.e., $\sigma_i = \sigma_{i,i}$. We omit results for this scenario as our simulations only indicate a slight decrease of randomizing influence (both positive and negative), while the character of the influence remains the same.

4. Performance Analysis

As we mention above, the distributed binary majority consensus problem is generally solved in a wait-free manner. Additional restrictions in system connectivity and synchrony make it difficult to guarantee the convergence.

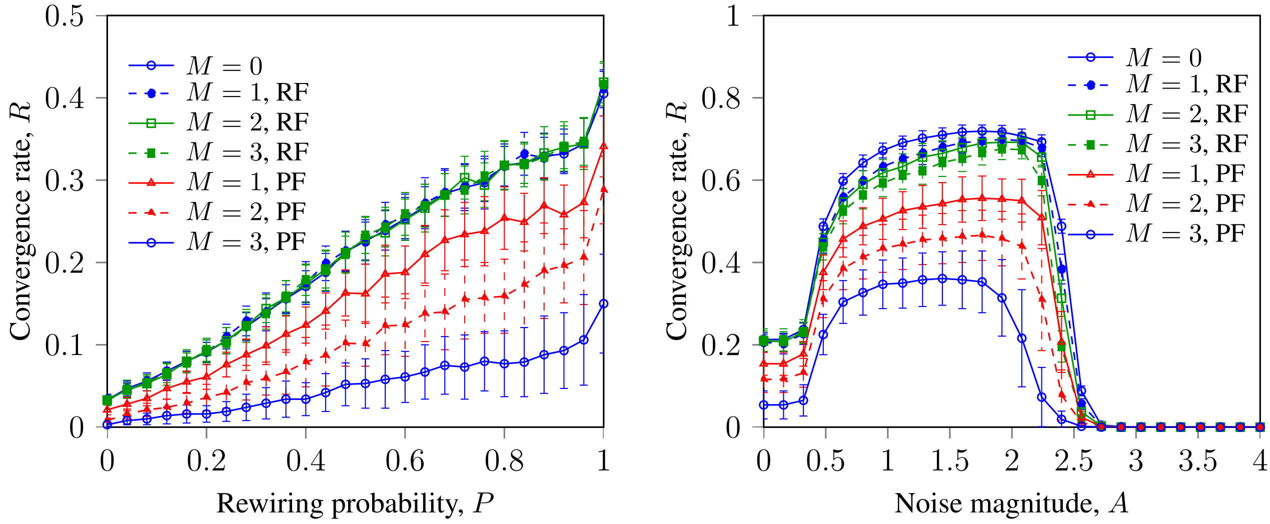
Due to this efficiency of wait-free binary majority consensus is generally measured as convergence rate R - a fraction of initial system configurations that result in a successful agreement. For each set of parameters we generate three random networks which are then simulated over 30 sets of initial configurations. Resulting 90 values of R are then averaged and plotted with 95% confidence intervals.

We investigate impact of faulty nodes on SM and GKL in WS and Waxman networks randomized by noise, message loss and topology. In the following sections we consequently compare the impact of faulty nodes with persistent and random failure in randomized networks with different faulty nodes layout. Next, we investigate the effect of strong consensus promotion by faulty nodes with random failure, observed in [32].

4.1. Faulty Nodes with Random and Persistent Failure

Let us analyze SM and GKL with faulty nodes and randomization by topology, noise and message loss. Figures 1a and 1b show that randomization by topology and noise can promote robustness of SM consensus towards faulty node behavior in asynchronous and synchronized networks respectively. It also shows that noise and topology randomization promote consensus in systems without faulty nodes ($M = 0$). This extends results earlier obtained in [15], where it was shown that topology randomization and low level of errors can promote asynchronous SM. Figure 1 also shows that faulty nodes with persistent failure inhibit consensus stronger than faulty nodes with random failure.

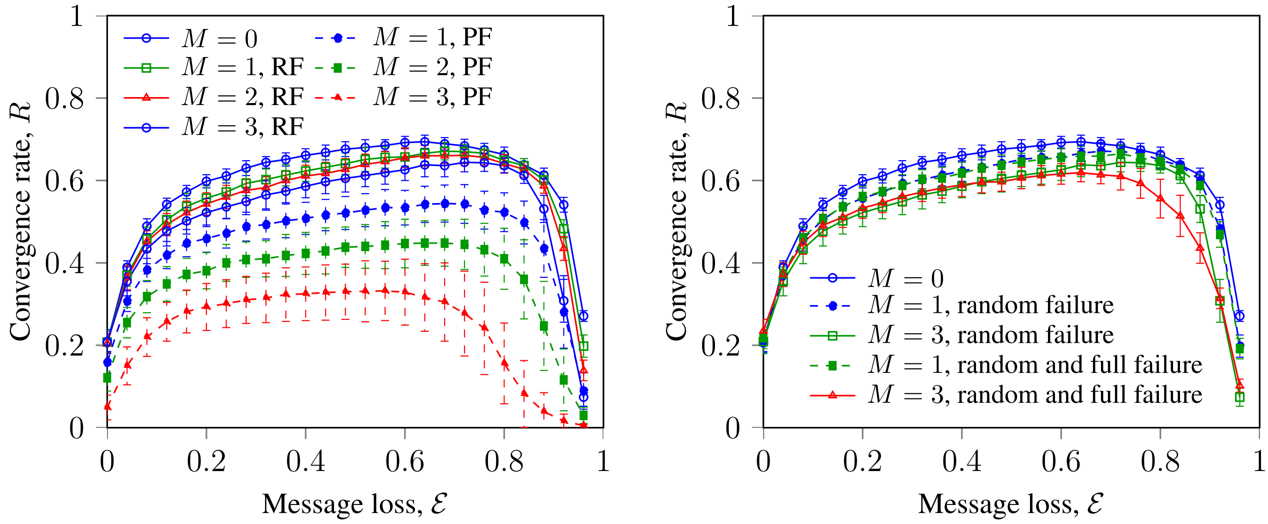
Figure 1. SM with M faulty nodes. Noise (AWUN) and topology randomization in WS networks. Faulty nodes with persistent failure (denoted as PF) inhibit consensus stronger than faulty nodes with random failure (denoted as RF). $K = 3$, $N = 99$.



a) Topology randomization promotes consensus in asynchronous noiseless WS networks, $A = 0$

b) Additive noise promotes consensus in synchronized random WS networks, $P = 1$

Figure 2. Asynchronous SM with M faulty nodes and message loss in random WS networks ($P = 1$). Stochastic message loss increases convergence rate of SM. PF and RF stand for faulty nodes with persistent and 2-state random failure models, respectively. $K = 3$, $N = 99$.



a) Faulty nodes with persistent failure are more adverse than faulty nodes with random failure

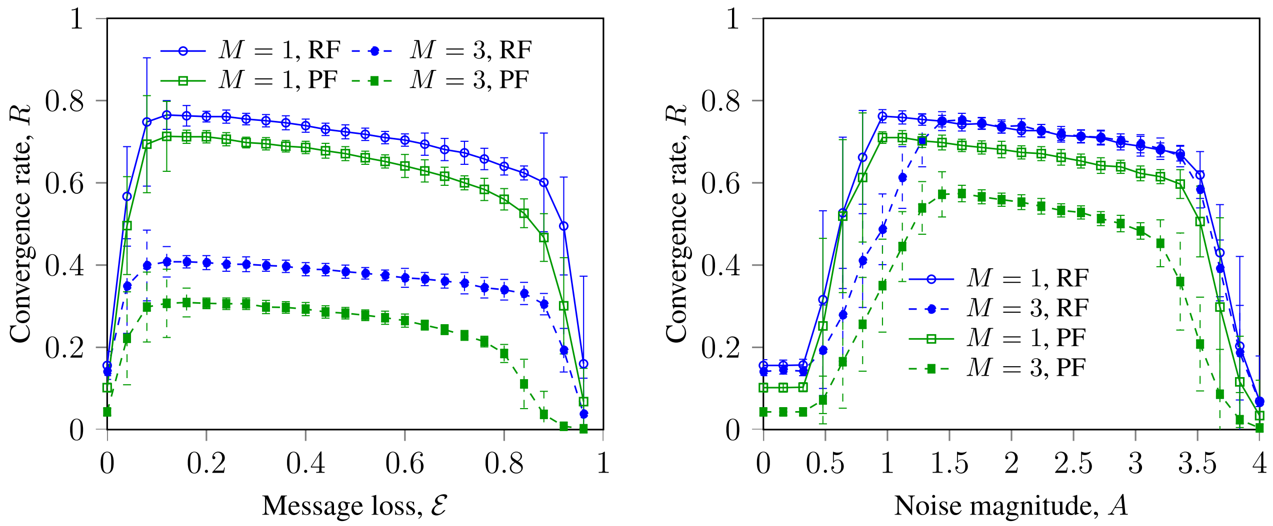
b) Faulty nodes with random and full failure show little difference in impact

This can be explained as follows. Topology randomization in WS networks connects a faulty node with random neighbors, enabling latter to overcome the reduced negative impact. Additive noise washes out the negative impact of the faulty node and promotes consensus in a similar manner. Cluster-breaking impact of randomization also contributes to the convergence rate in the systems without faulty nodes. This effect was earlier observed in [15,18,25], and can be explained as follows. Binary majority consensus is designed to provide a common decision for all nodes in the system, so the stable clusters of nodes sharing a different state inhibit the convergence. Some algorithms, like GKL, explicitly introduce the direction bias to wash out such clusters, for other algorithms the cluster-breaking effect can be provided by stochastic disturbances.

Randomization by message loss can promote consensus with faulty nodes, as shown in Figure 2. Thus, Figure 2a shows that in random WS networks message loss can increase R of SM and GKL with faulty nodes of both types. Figure 2b shows convergence rate of SM in random WS networks, indicating that faulty nodes with random full failure have impact similar to that of faulty nodes with random failure.

Figure 3 presents convergence rate of SM in Waxman networks with randomization by noise and message loss. It indicates that in Waxman networks faulty nodes with persistent failure inhibit consensus stronger than faulty nodes with random failure — effect we earlier observed for WS networks. These

Figure 3. Asynchronous SM with M faulty nodes in loosely connected Waxman networks. Faulty nodes with persistent failure are more adverse than ones with random failure. PF and RF stand for faulty nodes with persistent and random failure models, respectively. $K = 3$, $N = 99$, $\alpha = 0.05$, $\beta = 0.18$.



a) Message loss promotes consensus till randomization is outweighed by information loss, $A = 0$

b) Additive noise (AWGN) promotes consensus while message exchange prevails over stochasticity, $\mathcal{E} = 0$

observations extend results reported in [17,32] for a wider range of network topologies and a larger scope of randomizing disturbances and faulty node types.

It can be explained by the nature of persistently failing faulty nodes: such nodes always send state information that counters consensus process. Faulty nodes with random failure can also send correct information, and thus contribute to the agreement.

Observed small difference in impact between faulty nodes with random failure and faulty nodes with random full failure can be explained as follows. BMC can be promoted by stochastic message loss due to its "de-clustering" effect. Faulty nodes with stochastic full failure produce localized impact similar to that of message loss, and thus can promote consensus. For the same reason such faulty nodes decrease robustness towards high levels of message loss that can be seen in Figure 3b. This can infer the following generalization: although faulty nodes with full failure are often considered as a strong adversary for consensus, their impact on BMC indicates little difference. Moreover, both types of randomly failing faulty nodes are less adverse than persistently failing faulty nodes. Another important observation is that a number of persistent faulty nodes can be more adverse than an equal or even a bigger number of of randomly failing faulty nodes. In other words, BMC systems can be stronger inhibited with, e.g., M persistent faulty nodes with equal number of both faulty states $\sigma_M \in \{-1, 1\}$ than with $2M$ randomly failing faulty nodes.

4.2. Influence of Faulty Nodes Layout

In the previous section we observed that topology randomization can mitigate the negative impact of faulty nodes. This motivated us to determine whether a static random placement of the faulty nodes can produce similar effect in various networks, as it was observed for ring lattices in [32].

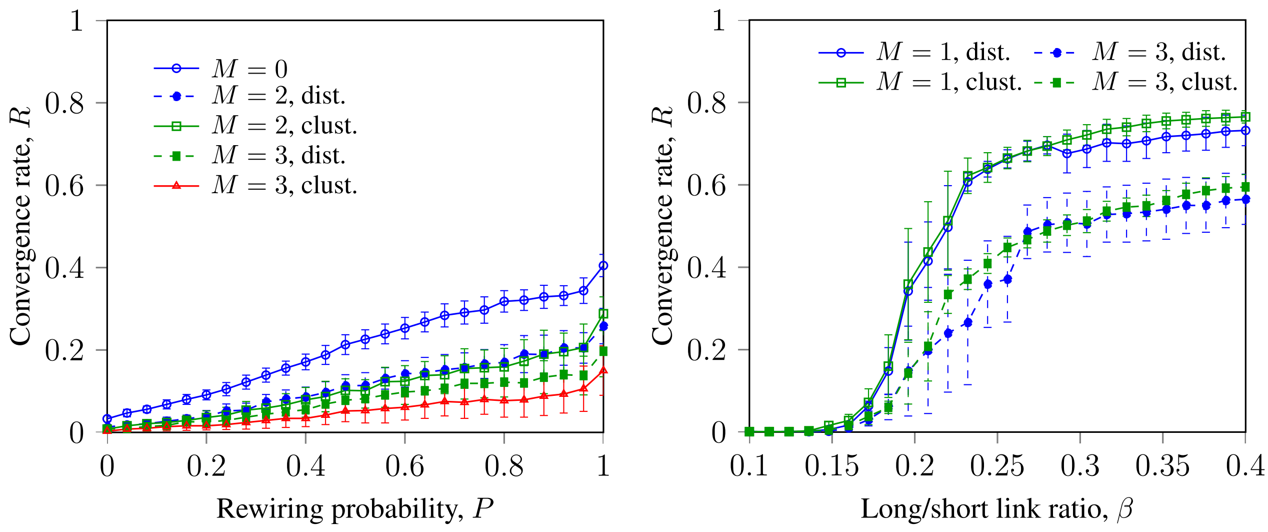
We simulate networks with two types of faulty nodes layout on the network where: (a) all faulty nodes are located in a single cluster, and (b) faulty nodes are randomly placed over the network.

Figures 4 and 5 present R of asynchronous GKL and SM in WS and Waxman networks with persistently failing faulty nodes with random and clustered layouts.

4.2.1. Topology Randomization

Figure 4 shows dynamics of the SM consensus with clustered and randomly placed faulty nodes in Watts-Strogatz and Waxman networks with topology randomization (increasing P and β).

Figure 4. Topology randomization increases efficiency of asynchronous SM in noiseless WS and Waxman networks with M clustered and randomly placed faulty nodes. $K = 3$, $N = 99$, $A = 0$, $\mathcal{E} = 0$.



a) In WS network clustered faulty nodes stronger inhibit SM consensus

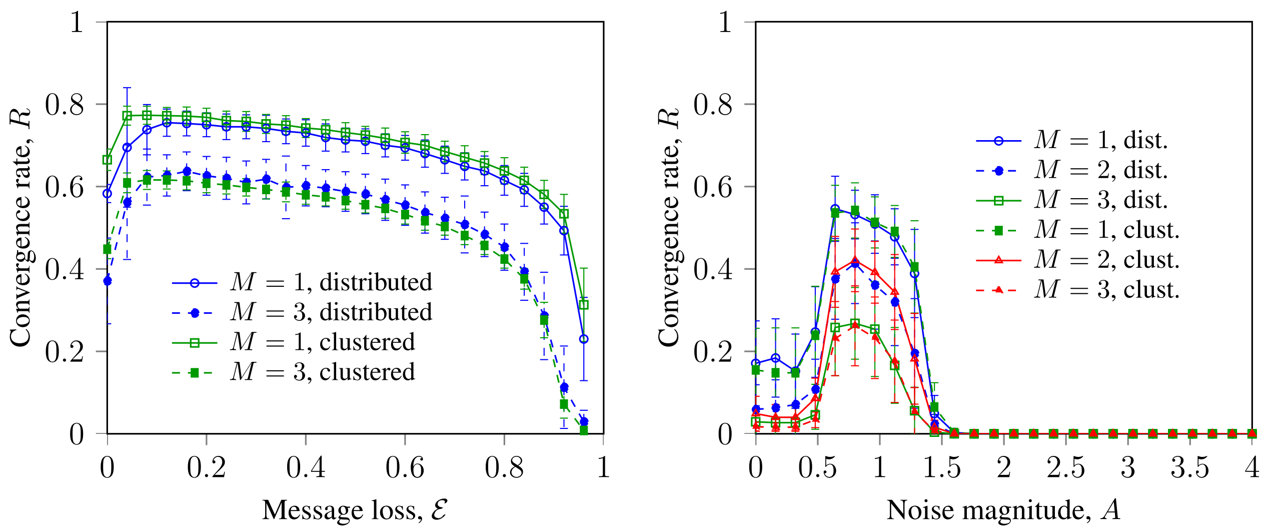
b) In Waxman network, clustered and random layout show little difference in impact

Thus, Figure 4a indicates an effect, similar to that earlier observed in [32]: in WS networks, ranging from a ring lattice ($P = 0$) to a random network ($P = 1$), faulty nodes with clustered layout inhibit asynchronous SM slightly stronger than faulty nodes randomly placed over the network. However, the difference in impact between clustered and randomly placed faulty nodes is low and is not observed in other setups, e.g., with synchronous SM or GKL, or in Waxman networks. The difference in impact is observed with $M \geq 3$ and can be explained by the sensitivity of asynchronous SM to external disturbances.

Further, Figure 4b does not indicate a notable difference in impact between clustered and randomly placed faulty nodes in Waxman network. However, it indicates that increasing topology randomization promotes SM with faulty nodes.

4.2.2. Randomization by Noise and Message Loss in Random Networks

Figure 5. Asynchronous GKL and SM in random WS and Waxman networks ($P = 1$, $\alpha = 0.05$, $\beta = 0.26$). M faulty nodes inhibit GKL stronger than SM. Clustered and randomly placed faulty nodes show little difference in impact. "Clust." and "dist." stand for clustered and random faulty node placement. $K = 3$, $N = 99$.



a) Message loss promotes SM consensus in Waxman networks

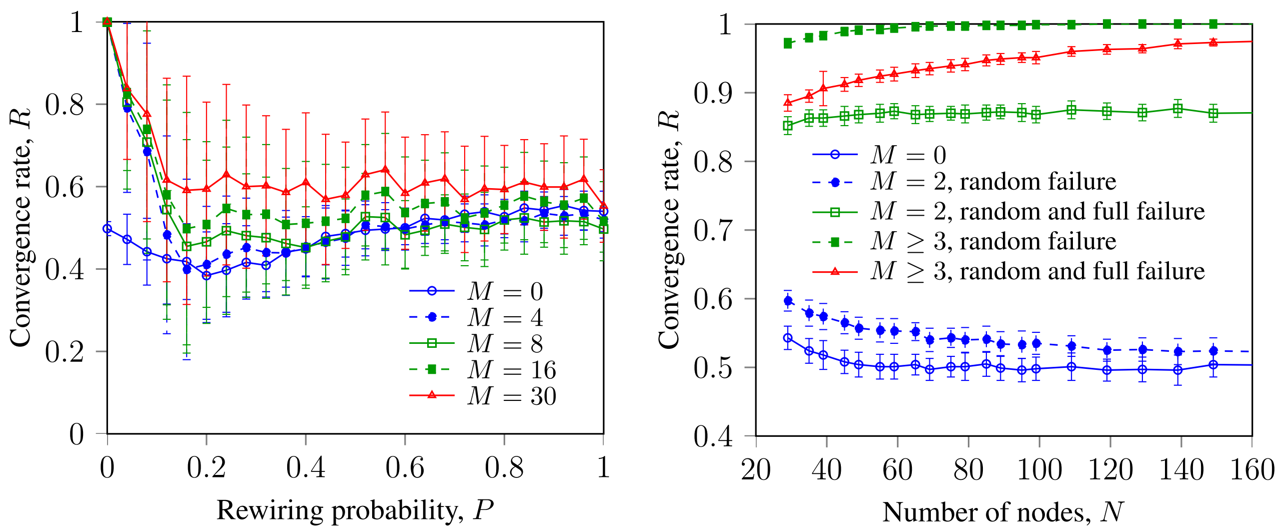
b) Additive noise (AWUN) increases efficiency of GKL in WS networks, while positive impact is outweighed by exceeding stochasticity

Figures 4b, and 5 show that in random networks impact of clustered faulty nodes is similar to that of randomly placed ones. This can be explained by topology randomization that dithers impact of the clustered faulty nodes into a wider set of nodes. This leads to "de-clustering" of the faulty nodes and mitigates the difference in impact with randomly placed faulty nodes. This effect is observed with different types of randomization in both WS and Waxman networks, as can be seen from Figure 5. This can infer that observed difference in the impact of clustered and randomly placed faulty nodes is a feature of the asynchronous SM evident in noiseless ring lattices and WS networks.

4.3. Consensus Promotion with Randomly Failing Faulty Nodes

Figure 6 shows R of asynchronous GKL with clustered randomly failing faulty nodes in WS networks and ring lattices of different size. Figure 6a resembles results similar to that shown in [32], showing that $M \geq K$ faulty nodes located in a single cluster can significantly increase R in ring lattices ($P = 0$). Figure 6b shows that this effect remains with system growth. It also indicates similar consensus promotion with randomly failing faulty nodes with full failure. Impact of randomly failing faulty nodes

Figure 6. Asynchronous GKL in WS networks. $M \geq K$ clustered and randomly failing faulty nodes increase efficiency up to 100%. $N \in \{29 \dots 999\}$, $K = 3$.



a) Topology randomization decreases efficiency in WS networks, $N = 99$

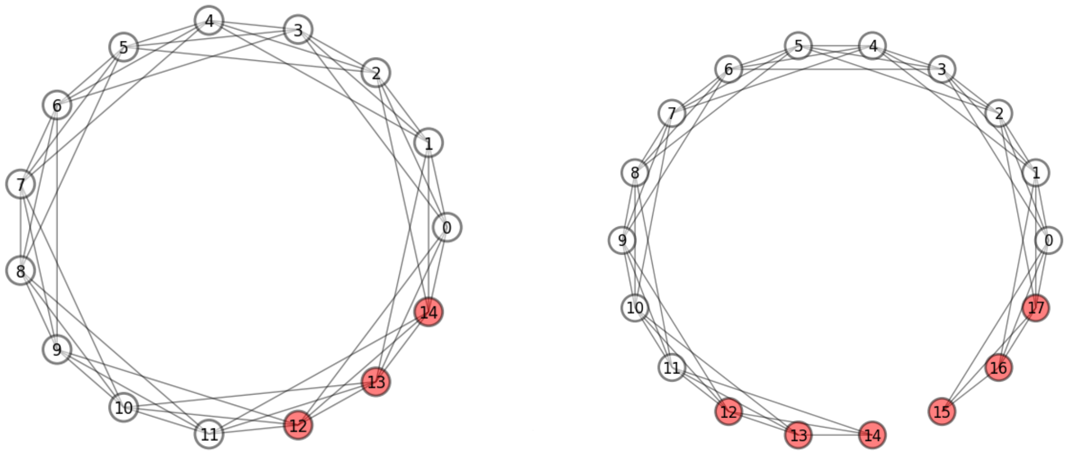
b) System growth promotes consensus in ring lattices ($P = 0$)

is stronger expressed than the impact of randomly failing faulty nodes with full failure, though both types indicate similar dependencies. This can hint to the fact that randomization within the consensus state space can be more efficient [15,17,25]. The positive impact of faulty nodes on GKL consensus can be explained by explicit randomization they impose on the information exchange. It was previously shown that randomization by binary errors can promote consensus [15]. Positive randomization by faulty nodes reaches maximum with $M \geq K$ faulty nodes located in a single cluster (see Fig. 7a). Such setup can be presented as an open one-dimensional lattice with M faulty nodes at both ends (see Fig. 7b). This setup has two important features: it logically "disconnects" the network and thus produces boundary effects that have not been considered previously. These latter features and consensus promotion to $\simeq 100\%$ efficiency motivated us to investigate this case in more detail.

4.3.1. Convergence Dynamics

Let us study how faulty nodes promote consensus in terms of a system evolution. Figures 8 and 9 show state and density evolution of GKL with faulty nodes over time, respectively. Figure 8a shows an agreement process of a synchronous GKL with no faulty nodes. It illustrates that in a connected ring clusters can migrate over the network. Figure 8b shows an example of agreement process of a

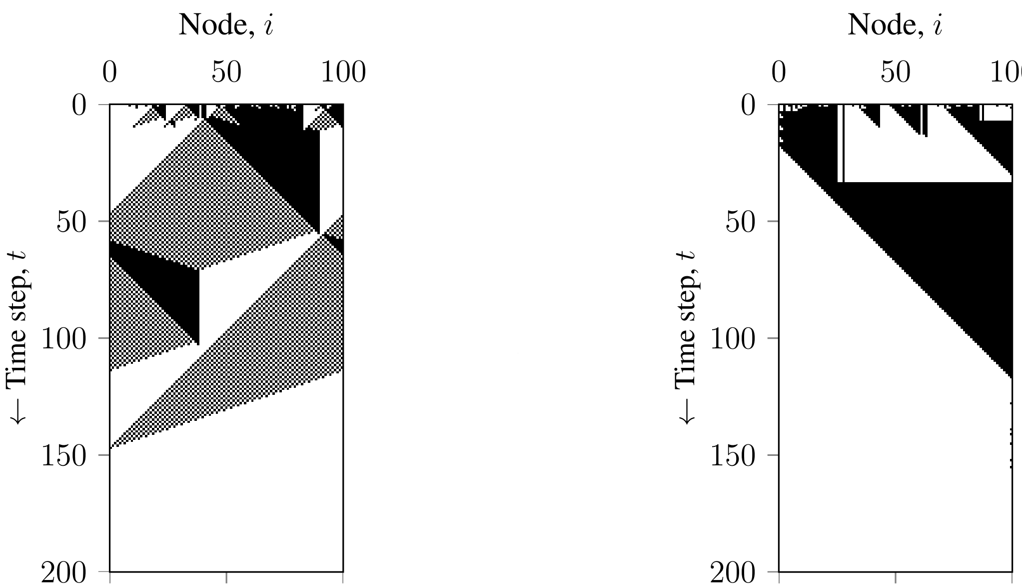
Figure 7. A network with $M = K$ clustered faulty nodes, $N = 15$, $K = 3$.



a) Connected ring with $M = K$ faulty nodes

b) Disconnected ring with $M = K$ faulty nodes at each network border

Figure 8. State evolution of GKL. $K = 3$, $N = 99$.

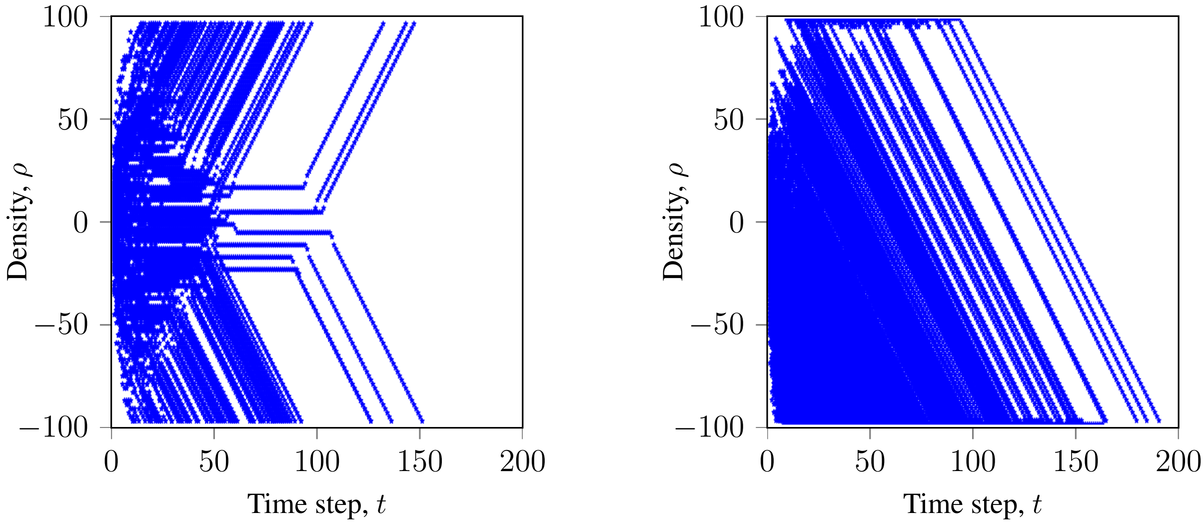


a) Synchronous GKL, $M = 0$, cluster of nodes with the same states "migrates" over the network

b) Asynchronous GKL, $M \geq K$ logically disconnects the ring lattice and restraints cluster to the boarder

synchronous GKL with $M \geq K$ faulty nodes. It shows that a cluster in a logically disconnected ring (see Figure 7) does not migrate and is destroyed faster. Figure 9 shows evolutions of a GKL in ring lattice over 500 initial configurations. Figure 9a shows the density evolution of the successfully converged networks with synchronous GKL. It illustrates that systems (each line represents a network evolving over the unique initial configuration) steadily evolve to correct majorities and terminate. Figure 9b shows density evolution of the asynchronous GKL with $M \geq K$ randomly failing faulty nodes, indicating that state direction bias of the GKL combined with asynchronous updates can steer the system to the expected state. However, it also shows that due to stochastic intrusions systems often evolve closely to

Figure 9. Density evolution of synchronous and asynchronous GKL in ring lattices over 500 initial configurations. $K = 3$, $N = 99$.



a) Synchronous GKL, $\simeq 82\%$ of networks agree on correct majority, $M = 0$

b) Asynchronous GKL, $\simeq 100\%$ of networks agree on correct majority, but often evolve close to incorrect majority, $M \geq K$

the opposite majority, and then get steered to the correct one. This happens due to the steering effect of the asynchronous sequential update and the contribution of the faulty nodes random state messages.

Latter observations can be explained as follows. Even though asynchronous GKL with additional randomization by faulty nodes can reach $R \simeq 100\%$, it can not be considered as a solution to the binary majority consensus problem: the system exhibits significant measure of random dynamics and cannot guarantee a stable correct convergence in the given consensus time T.

5. Conclusions

In this article we study the impact of faulty nodes on randomized binary majority consensus. We simulate two standard algorithms, GKL and SM, in ordered and topologically randomized networks with noise and stochastic message loss. We study faulty nodes with persistent failure in comparison with commonly used faulty nodes with random failure, including nodes with random full failure. We simulate faulty nodes with clustered and random layout, focusing on asynchronous networks. The main contributions of this article can be summarized as follows:

- A number of faulty nodes with persistent failure are more adverse for binary majority consensus than even a larger number of commonly-used faulty nodes with random failure, or faulty nodes with random full failure;
- Simple binary majority consensus algorithms such as Simple Majority do not degrade with randomization, but respond with increase in convergence rate;
- Randomization by noise, message loss and topology can promote such consensus algorithms and mitigate the impact of a low number of faulty nodes.

These new results can be explained by the "de-clustering" influence, provided by explicit stochastic intrusions, such as noise and message loss. Such consensus-promoting "de-clustering" influence, in some cases, can be provided by faulty nodes with random failure. Such nodes can promote BMC in asynchronous networks providing unstable, but efficient convergence.

This can be generalized as follows: due to restrictions in connectivity, synchrony, and time BMC exhibits diverse dynamics. This dynamics can be further exaggerated by disturbances. In particular, randomizing disturbances not only increase the efficiency of BMC but also promote its robustness towards faulty node behavior. Consequently, stochastically failing faulty nodes present a weak adversary for such consensus, and in some cases can even promote it. These observations can infer that aforementioned restrictions and disturbances are not only essential requirement for modeling of distributed consensus systems [15], but an intrinsic feature that helps yield self-organizing behavior from simple networked interactions [24].

This work extends and complements previous investigations on binary majority consensus with stochastic elements [15,17,18,25,30,32] in terms of the robustness towards faulty node behavior.

Acknowledgements

This work was performed in University of Klagenfurt and University of Genoa within the Erasmus Mundus Joint Doctorate in "Interactive and Cognitive Environments", which is funded by the EACEA Agency of the EC under EMJD ICE FPA n 2010-0012. The work of A. Gogolev is supported by Lakeside Labs, Klagenfurt with funding from the ERDF, KWF, and the state of Austria under grant 20214/21530/32606.

Conflict of Interest

The authors declare no conflict of interest.

References

1. Bernstein, P.A.; Goodman, N. An algorithm for concurrency control and recovery in replicated distributed databases. *ACM Transactions on Database Systems* **1984**, *9*, 596–615.
2. Olfati-Saber, R.; Shamma, J.S. Consensus filters for sensor networks and distributed sensor fusion. Proceedings of IEEE Conference on Decision and Control and European Control Conference, 2005, pp. 6698–6703.
3. Alighanbari, M.; How, J.P. An unbiased Kalman consensus algorithm. Proceedings of American Control Conference, 2006, pp. 3519–3524.
4. Aspnes, J. Fast deterministic consensus in a noisy environment. Proceedings of the ACM Symposium on Principles of Distributed Computing, 2000, pp. 299–308.
5. Carli, R.; Fagnani, F.; Frasca, P.; Taylor, T.; Zampieri, R. Average consensus on networks with transmission noise or quantization. Proceedings of European Control Conference, 2007, pp. 4189–4194.

6. Kar, S.; Moura, J.M.F. Distributed average consensus in sensor networks with random link failures and communication channel noise. Proceedings of Asilomar Conference on Signals, Systems and Computers, 2007, pp. 676–680.
7. Kingston, D.B.; Beard, R.W. Discrete-time average consensus under switching network topologies. Proceedings of American Control Conference, 2006, pp. 3551–3556.
8. Fischer, M.J.; Lynch, N.A.; Paterson, M.S. Impossibility of distributed consensus with one faulty process. *Journal of the ACM* **1985**, *32*, 374–382.
9. Fischer, M.J.; Lynch, N.A.; Merritt, M.S. Easy impossibility proofs for distributed consensus problems. Proceedings of the ACM Symposium on Principles of Distributed Computing, 1985, pp. 59–70.
10. Chandra, T.D.; Toueg, S. Unreliable failure detectors for reliable distributed systems. *Journal of the ACM* **1996**, *43*, 225–267.
11. Chandra, T.D.; Hadzilacos, V.; Toueg, S. The weakest failure detector for solving consensus. *Journal of the ACM* **1996**, *43*, 685–722.
12. Dolev, D.; Dwork, C.; Stockmeyer, L. On the minimal synchronism needed for distributed consensus. *Journal of the ACM* **1987**, *34*, 77–97.
13. Dwork, C.; Lynch, N.; Stockmeyer, L. Consensus in the presence of partial synchrony. *Journal of the ACM* **1988**, *35*, 288–323.
14. Aspnes, J. Randomized protocols for asynchronous consensus. *Distributed Computing* **2003**, *16*, 165–175.
15. Moreira, A.A.; Mathur, A.; Diermeier, D.; Amaral, L.A.N. Efficient system-wide coordination in noisy environments. *Proceedings of the National Academy of Sciences of the United States of America* **2004**, *101*, 12085–12090.
16. Ben-Or, M. Another advantage of free choice: Completely asynchronous agreement protocols. Proceedings of the ACM Symposium on Principles of Distributed Computing, 1983, pp. 27–30.
17. Gogolev, A.; Marchenko, N.; Bettstetter, C.; Marcenaro, L. Distributed Binary Consensus in Networks With Faults. under review.
18. Gogolev, A.; Marcenaro, L. Efficient binary consensus in randomized and noisy networks. Proceedings of the IEEE 9th International Conference on Intelligent Sensors, Sensor Networks and Information Processing, 2014.
19. Watts, D.J.; Strogatz, S.H. Collective dynamics of "small-world" networks. *Nature* **1998**, *393*, 440–442.
20. Waxman, B.M. Routing of multipoint connections. *IEEE Journal on Selected Areas in Communications* **1988**, *6*, 1617–1622.
21. Mirollo, R.; Strogatz, S. Synchronization of Pulse-Coupled Biological Oscillators. *SIAM Journal on Applied Mathematics* **1990**, *50*, 1645–1662.
22. Gacs, P. Reliable cellular automata with self-organization. *Journal of Statistical Physics* **2001**, *103*, 45–267.
23. Gogolev, A.; Khakhaev, A.; Pergament, A.; Shtykov, A. Effects of Harmonic Modulation of Current in Glow Discharge Dusty Plasma with Ordered Structures. *Contributions to Plasma Physics* **2011**, *51*, 498–504.

24. Klinglmayr, J.; Kirst, C.; Bettstetter, C.; Timme, M. Guaranteeing global synchronization in networks with stochastic interactions. *New Journal of Physics* **2012**, *14*, 073031.
25. Gogolev, A.; Bettstetter, C. Robustness of self-organizing consensus algorithms: Initial results from a simulation-based study. In *Self-Organizing Systems*; Springer, 2012; Vol. 7166, *LNCS*, pp. 104–108.
26. Gacs, P.; Kurdyumov, G.L.; Levin, L.A. One-dimensional uniform arrays that wash out finite islands. *Problemy Peredachi Informacii* **1978**, *14*, 92–98.
27. Saks, M.; Zaharoglou, F. Wait-free k-set agreement is impossible: The topology of public knowledge. Proceedings of the ACM Symposium on Theory of Computing, 1993, pp. 101–110.
28. Andre, D.; Bennett III, F.H.; Koza, J.R. Discovery by genetic programming of a cellular automata rule that is better than any known rule for the majority classification problem. Proceedings of the First Annual Conference on Genetic Programming, 1996, pp. 3–11.
29. Land, M.; Belew, R.K. No perfect two-state cellular automata for density classification exists. *Physical Review Letters E* **1995**, *74*, 5148–5150.
30. Fates, N. Stochastic cellular automata solutions to the density classification problem. *Theory of Computing Systems* **2013**, *53*, 223–242.
31. Pease, M.; Shostak, R.; Lamport, L. Reaching agreement in the presence of faults. *Journal of the ACM* **1980**, *27*, 228–234.
32. Gogolev, A.; Marcenaro, L. Density classification in asynchronous random networks with faulty nodes. Proceedings of the 22nd International Conference on Parallel, Distributed and Network-Based Processing, 2014.
33. Barabási, A.L.; Reka, A. Emergence of scaling in random networks. *Science* **1999**, *286*, 509–512.
34. Török, J.; Iñiguez, G.; Yasseri, T.; San Miguel, M.; Kaski, K.; Kertész, J. Opinions, conflicts, and consensus: modelling social dynamics in a collaborative environment. *Physical Review Letters* **2013**, *110*, 88701.
35. Van Mieghem, P. Paths in the simple random graph and the Waxman graph. *Probability in the Engineering and Informational Sciences* **2001**, *15*, 535–555.
36. Fukś, H. Solution of the density classification problem with two cellular automata rules. *Physical Review Letters E* **1997**, *55*, R2081–R2084.
37. Mitchell, M.; Hraber, P.T.; Crutchfield, J.P. Revisiting the edge of chaos : Evolving cellular automata to perform computations. *Complex Systems* **1993**, *7*, 89–130.
38. Juille, H.; Pollack, J.B. Coevolving the ”ideal” trainer: Application to the discovery of cellular automata rules. Proceedings of the Annual Conference on Genetic Programming, 1998, pp. 519–527.
39. Vladimirov, I.G.; Diamond, P. A uniform white-noise model for fixed-point roundoff errors in digital systems. *Automation and Remote Control* **2002**, *63*, 753–765.