DISTRIBUTED CONSENSUS IN FAULTY NETWORKS

by

Halil Yiğit Öksüz

B.S., Electrical and Electronics Engineering, Dokuz Eylül University, 2015

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Electrical and Electronics Engineering
Boğaziçi University

2017

DISTRIBUTED CONSENSUS IN FAULTY NETWORKS

APPROVED BY:

Prof. Mehmet Akar .....
(Thesis Supervisor)

Prof. Kemal Cılız .....

Assist. Prof. Onur Cihan .....

DATE OF APPROVAL: 06.09.2017

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my thesis supervisor Prof. Mehmet Akar for his scientific wisdom and his guidance throughout this study. Not only his insight and motivation but also his life-changing advice encouraged me to work harder in order to accomplish the objective of this thesis.

I also would like to express my special thanks to Prof. Kemal Cılız for his support and being a member of my thesis jury. Moreover, I am also grateful to Assist. Prof. Onur Cihan for his guidance and also being a part of my thesis jury.

I am sincerely grateful to my friends and colleagues, Özlem Feyza Erkan, Ümit Develer and Günay Turan for all their support and cooperation. Futhermore, I owe many thanks to my other colleagues and friends at Boğaziçi University for helping me in all aspects of this study.

I cannot find any words to express my strong feelings about my family who encouraged me to choose academic life. I am really grateful to my parents Seyfettin Öksüz, Kamile Öksüz and my dearest friend Burçak Bal for supporting me not only for this research, but also throughout my life.

This thesis is supported by the Scientific and Technological Research Council of Turkey (TUBITAK) under Grant 114E613.

ABSTRACT

DISTRIBUTED CONSENSUS IN FAULTY NETWORKS

The objective of this thesis is to develop fault tolerant consensus algorithms in the presence of two types of fault models: (i) *Structured Byzantine (StrBYZ)* faults that send different structured information to its in-neighbors, (ii) *Unstructured Byzantine (uStrBYZ)* faults which send different random erroneous information to its in-neighbors.

For the StrBYZ fault model, it is shown that the network of non-faulty nodes can achieve approximate Byzantine consensus for synchronous and asynchronous networks without using a fault tolerant algorithm. Furthermore, the analysis is extended to the study of approximate Byzantine *group* consensus in which the non-faulty nodes reach more than one equilibrium.

As opposed to the StrBYZ fault model, it is impossible to guarantee approximate Byzantine consensus under the uStrBYZ fault model without employing a fault tolerant algorithm. To remedy this situation, two fault tolerant algorithms, so called Layered Mean-Select-Reduced (L-MSR) and Rooted Mean-Select-Reduced (R-MSR) are proposed. Necessary and sufficient conditions for the success of these algorithms for single equilibrium are presented. Moreover, it is shown that L-MSR and R-MSR algorithms can also be used to solve the approximate Byzantine *group* consensus problem.

Another contribution of this thesis is to introduce a parameter independent fault tolerant algorithm that can be used to guarantee approximate Byzantine consensus. Using the proposed distributed fault detection scheme, the requirement on the knowledge of the number of faulty agents is relaxed. Convergence analysis for the proposed algorithm is carried out.

ÖZET

HATALI AĞLARDA DAĞITIK ONAYLAŞIM

Bu tez çalışmasında amaç, (i) tüm komşularına belli bir yapıda farklı bilgiler gönderen *Yapısal Bizans*, (ii) tüm komşularına rastgele ve farklı bilgiler gönderen *Yapısal Olmayan Bizans* hata modelleri varlığında hataya dayanıklı onaylaşım algoritmaları geliştirmektir.

Yapısal Bizans hata modeli için, hatalı olmayan düğümlerin hataya dayanıklı bir algoritma kullanmadan senkron ve asenkron ağlarda yaklaşık Bizans onaylaşımına ulaşacağı gösterilmiştir. Dahası, yapılan analizler hatalı olmayan düğümlerin birden fazla denge noktasına ulaştığı yaklaşık Bizans *grup* onaylaşımı için genişletilmiştir.

Yapısal Bizans hata modelinin aksine, yaklaşık Bizans onaylaşımının Yapısal olmayan Bizans hata modeli varlığında, hataya dayanıklı algoritma kullanmadan garantilenmesinin imkansız olduğu gösterilmiştir. Bu nedenle, Katmanlı Ortalamalı-İndirgenmiş-Altdizi (Kat-OIA) ve Köklü Ortalamalı-İndirgenmiş-Altdizi (Kök-OIA) algoritmaları önerilmiştir. Bu algoritmaların başarısı için gerekli ve yeterli koşullar sunulmuştur. Ayrıca, Kat-OIA ve Kök-OIA algoritmalarının yaklaşık Bizans grup onaylaşım problemini çözmek için de kullanılabileceği gösterilmiştir.

Bu tez çalışmasında mevcut diğer bir yenilik, yaklaşık Bizans onaylaşımını garantilemek için kullanılabilecek parametreden bağımsız hataya dayanıklı algoritmadır. Önerilen hata tespit düzeni ile, hatalı düğüm sayısı bilgisine olan gereklilik ortadan kaldırılmıştır. Önerilen hataya dayanıklı algoritmanın yakınsama analizi yapılmıştır.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS iii
ABSTRACT iv
ÖZET v
LIST OF FIGURES ix
LIST OF SYMBOLS xii
LIST OF ACRONYMS/ABBREVIATIONS xv
1. INTRODUCTION 1
1.1. Fault Classification 2
1.2. Contributions of the Thesis 5
1.3. Organization of the Thesis 6
2. DISTRIBUTED CONSENSUS REVIEW 7
2.1. Graph Terminology 7
2.2. Distributed Consensus Algorithm 8
2.3. Iterative Byzantine Consensus 9
2.4. Distributed Resilient Consensus Algorithms 10
2.4.1. Weighted-Mean-Subsequence-Reduced (W-MSR) Algorithm . . 10
2.4.2. The Asynchronous MSR Algorithm 12
2.4.3. Middle Algorithm 13
2.5. Summary of the Chapter 14
3. RESILIENT CONSENSUS UNDER STRUCTURED BYZANTINE FAULTS 15
3.1. Adversary Behaviour and Fault Model 15
3.2. Consensus Model in the Presence of StrBYZ 15
3.3. Analysis Using An Expanded Network Representation 16
3.3.1. Expanded Network Representation for StrBYZ Fault Model . . 17
3.3.2. Results on the Expanded Network 19
3.4. Analysis On The Original Network 21
3.5. On Approximate Byzantine Consensus 24
3.5.1. Approximate Byzantine Consensus On The Expanded Network 25
3.5.2. Approximate Byzantine Consensus On The Original Network . . 26

3.6. Extension to Asynchronous Networks 28
3.7. Summary of the Chapter 30
4. RESILIENT CONSENSUS UNDER UNSTRUCTURED BYZANTINE FAULT MODEL 31
4.1. Adversary Behaviour and Fault Model 31
4.2. Consensus Computation in the Presence of uStrBYZ 31
4.3. Expanded Network Representation for uStrBYZ Fault Model 32
4.4. L-MSR Algorithm and Its Convergence Analysis 34
4.4.1. L-MSR Algorithm 34
4.4.2. Convergence Analysis of the L-MSR Algorithm 36
4.5. R-MSR Algorithm And Its Convergence Analysis 39
4.5.1. The R-MSR Algorithm 39
4.5.2. Convergence Analysis of R-MSR Algorithm 40
4.6. Comparison Of Resilient Distributed Algorithms 42
4.6.1. Comparison of L-MSR and R-MSR Algorithms 42
4.6.2. Comparison of the Proposed Algorithms with Resilient Algorithms in the Literature 43
4.7. Numerical Anaylsis 44
4.8. Summary of the Chapter 47
5. FAULT TOLERANT GROUP CONSENSUS 49
5.1. Group Consensus Preliminaries 49
5.2. Linear Group Consensus Under StrBYZ Fault Model 49
5.3. Approximate Byzantine Group Consensus Under StrBYZ Fault Model . 53
5.4. Approximate Byzantine Group Consensus under uStrBYZ Fault Model 57
5.4.1. Approximate Byzantine Group Consensus using the L-MSR Algorithm 57
5.4.2. Approximate Byzantine Group Consensus using the R-MSR Algorithm 59
5.5. Summary of the Chapter 61
6. A PARAMETER-INDEPENDENT FAULT TOLERANT ALGORITHM . . 62
6.1. Fault Detection Scheme 62

6.1.1. Fault Detection Filter 62
6.2. Parameter-Independent MSR Algorithm (PIMSR) 69
6.3. Summary of the Chapter 71
7. CONCLUSIONS 73
REFERENCES 75

LIST OF FIGURES

Figure 1.1. Fault Models and Classifications 2

Figure 3.1. Original and Expanded Networks where node 1 is StrBYZ. 18

Figure 3.2. Original and Expanded Networks where node 4 is StrBYZ. 24

Figure 3.3. The simulation result of the network given in Figure 3.2(b) 24

Figure 3.4. Original and Expanded Networks where node 6 is StrBYZ. 27

Figure 3.5. The simulation result of the network given in Figure 3.4(b) 28

Figure 3.6. The simulation result of the asynchronous network given in Figure 3.2(b) 30

Figure 4.1. Original and Expanded Networks where node 1 is uStrBYZ. 33

Figure 4.2. Robust networks for numerical analysis. 44

Figure 4.3. The simulation result of the network given in Figure 4.2(a) when non-faulty nodes use the L-MSR algorithm. 45

Figure 4.4. The simulation result of the network given in Figure 4.2(b) when non-faulty nodes use the R-MSR algorithm. 46

Figure 4.5. A network with 5 nodes where node 5 is uStrBYZ and the network of non-faulty nodes in $G_{2,1}$ is 2-robust. 46

Figure 4.6. The simulation result of the network given in Figure 4.5 when non-faulty nodes use R-MSR algorithm. 47

Figure 5.1. Original and Expanded Networks where node 1 and node 9 are StrBYZ. 52

Figure 5.2. The simulation result of the network given in Figure 5.1(a). 52

Figure 5.3. Original and Expanded Networks where node 7 and node 13 are StrBYZ. 56

Figure 5.4. The simulation result of the networks given in Figure 5.3. 56

Figure 5.5. A network with 12 nodes where node 5 and node 12 are uStrBYZ. 58

Figure 5.6. The simulation result of the network given in Figure 5.5. 59

Figure 5.7. A network with 15 nodes where nodes 5, 11 and 15 are uStrBYZ. . 60

Figure 5.8. The simulation result of the network given in Figure 5.7. 61

Figure 6.1. The network with 4 nodes where node 4 is Byzantine. 66

Figure 6.2. Error terms of nodes in Example 6.1. 67

Figure 6.3. Consensus of nodes in Example 6.1. 67

Figure 6.4. Error terms of non-faulty in Example 6.2. 68

Figure 6.5. Consensus of non-faulty nodes in Example 6.2. 68

Figure 6.6. The network with 14 nodes given in [1] where node 14 is Byzantine. 71

Figure 6.7. The simulation result of the network given in the Figure 6.6 using the PIMSR algorithm. 71

LIST OF SYMBOLS

d_i	Number of non-faulty information that each non-faulty node i receives
E	The edge set of graph G
E_{nf}	The edge set of graph G_{nf}
$E_{m,s}$	The edge set of the s-th primary or secondary layer $G_{m,s}$ in the original network G
$\bar{E}_{m,s}$	The edge set of the s-th primary or secondary layer $G_{m,s}$ in the expanded network $\bar{G}$
$E^{nf}_{m,s}$	The edge set of non-faulty vertices of the s-th primary or secondary layer $G_{m,s}$ in the original network G
$\bar{E}^{nf}_{m,s}$	The edge set of non-faulty vertices of the s-th primary or secondary layer $G_{m,s}$ in the expanded network $\bar{G}$
e	The maximum difference between the state values of non-faulty vertices
$e_{m,s}$	The maximum difference between the state values of non-faulty vertices in the s-th primary or secondary layer $G_{m,s}$
F	The set of faulty vertices in the network
$F_{m,s}$	The set of faulty vertices in the s-th primary or secondary layer $G_{m,s}$
f	The maximum number of faulty nodes in a network
$f_{m,s}$	The maximum number of faulty nodes in the s-th primary or secondary layer $G_{m,s}$
G	The underlying directed graph of a network
$\bar{G}$	The underlying directed graph of an expanded network
G_{nf}	The underlying directed graph of a network of non-faulty nodes
$G_{m,s}$	The s-th primary or secondary layer of the network G
$\bar{G}_{m,s}$	The s-th primary or secondary layer of the expanded network $\bar{G}$
$G^{nf}_{m,s}$	The network of non-faulty vertices of the s-th primary or secondary layer $G_{m,s}$ in the original network G

$\bar{G}^{nf}_{m,s}$	The network of non-faulty vertices of the s-th primary or secondary layer $G_{m,s}$ in the expanded network $\bar{G}$
M	The maximum state value of the set of non-faulty vertices
$M_{m,s}$	The maximum state value of the set of non-faulty vertices in the s-th primary or secondary layer $G_{m,s}$
m	The minimum state value of the set of non-faulty vertices
$m_{m,s}$	The minimum state value of the set of non-faulty vertices in the s-th primary or secondary layer $G_{m,s}$
N_i	The set of inclusive neighbors of node i
N_i^+	The set of in-neighbors of node i
N_i^-	The set of out-neighbors of node i
n	Total number of nodes in a graph
$\mathbb{R}$	The set of real values
R_i	The set of eliminated values by node i in the proposed fault tolerant algorithms
r	The number of update round
k	Time index
k_r^i	The update time of node i at round r
k_{M_r}	The maximum sequence of time in a network
k_{m_r}	The minimum sequence of time in a network
V	The set of vertices of graph G
$V_{m,s}$	The set of vertices of the s-th primary or secondary layer $G_{m,s}$ in the original network G.
$\bar{V}_{m,s}$	The set of vertices of the s-th primary or secondary layer $G_{m,s}$ in the expanded network $\bar{G}$.
V^{sp}	The set of vertices in a normal spanning component
$V^{sp}_{m,s}$	The set of vertices in a normal spanning component of s-th primary or secondary layer $G_{m,s}$
V_{nf}	The set of non-faulty vertices of graph G
$V^{nf}_{m,s}$	The set of non-faulty vertices of the s-th primary or secondary layer $G_{m,s}$ in the original network G
$\bar{V}^{nf}_{m,s}$	The set of non-faulty vertices of the s-th primary or secondary layer $\bar{G}_{m,s}$ in the expanded network G.

x	The vector of state values
x_i	The state value of a non-faulty node i
x_{ij}	The state value of node j received by node i
w_{ij}	The averaging coefficients used by node i
$\tilde{w}_{ij,l}$	The averaging coefficients used by an StrBYZ agent j
Δk_r^i	The time interval between two successive update in state of node i
Δk_M	Maximum update time interval in a network
Δk_m	Minimum update time interval in a network
ϵ	A chosen consensus accuracy level

LIST OF ACRONYMS/ABBREVIATIONS

MSR	Mean-Select-Reduced
W-MSR	Weighted Mean-Select-Reduced
StrBYZ	Structured Byzantine
uStrBYZ	Unstructured Byzantine
R-MSR	Rooted Mean-Select-Reduced
L-MSR	Layered Mean-Select-Reduced
PIMSR	Parameter Independent Mean-Select-Reduced

1. INTRODUCTION

Fault-tolerant distributed consensus is an important topic that has been studied over the years. The objective is to make sure that all non-faulty agents achieve consensus on a single value in the presence of misbehaving agents. Recently, Leblanc and coauthors have proposed Mean-Subsequence-Reduced (MSR) based fault tolerant algorithms for synchronous [1] and asynchronous [2] networks and have examined their convergence analysis using the topological property *network robustness* they have introduced. Such MSR algorithms were first presented by *Dolev et al* [3]. In [4], a fault tolerant algorithm which solves the problem of reaching consensus for non-faulty nodes in both synchronous and asynchronous networks independent of the delay on the communication links has been proposed. Convergence rate analysis of this fault-tolerant algorithm has been investigated in [5]. In [6], an iterative distributed function calculation method in the presence of malicious agents has been presented. A method of overcoming these malicious agents has been discussed in [7]. Another approach presented in [8] is to detect the fault and eliminate the faulty node from the network. In [9], non-colluding faults have been investigated in addition to malicious behaviour under the assumption that all faulty agents including malicious ones send the same information to its neighbors as in [1] and [10]. Note that the term *non-colluding* implies that the faulty agents do not collaborate with the other faulty nodes, nor do they try to learn and disrupt the fault-tolerant algorithm the non-faulty nodes employ.

In most of the aforementioned studies, the consensus problem for non-faulty nodes has been studied using the initial knowledge on the number of faulty nodes (or an upper bound on the number of faulty nodes) in the network. However, it is quite restrictive to assume that the number of faulty nodes is available to the non-faulty nodes of the network. Furthermore, fault tolerant algorithms that rely on an upper bound on the number of faulty nodes require very tight network conditions. Although the use of fault-tolerant algorithms is inevitable for some fault models (malicious, Byzantine, etc.), one can want to design the network so that it becomes resilient to certain type of faults in order to reduce the complexity of the design.

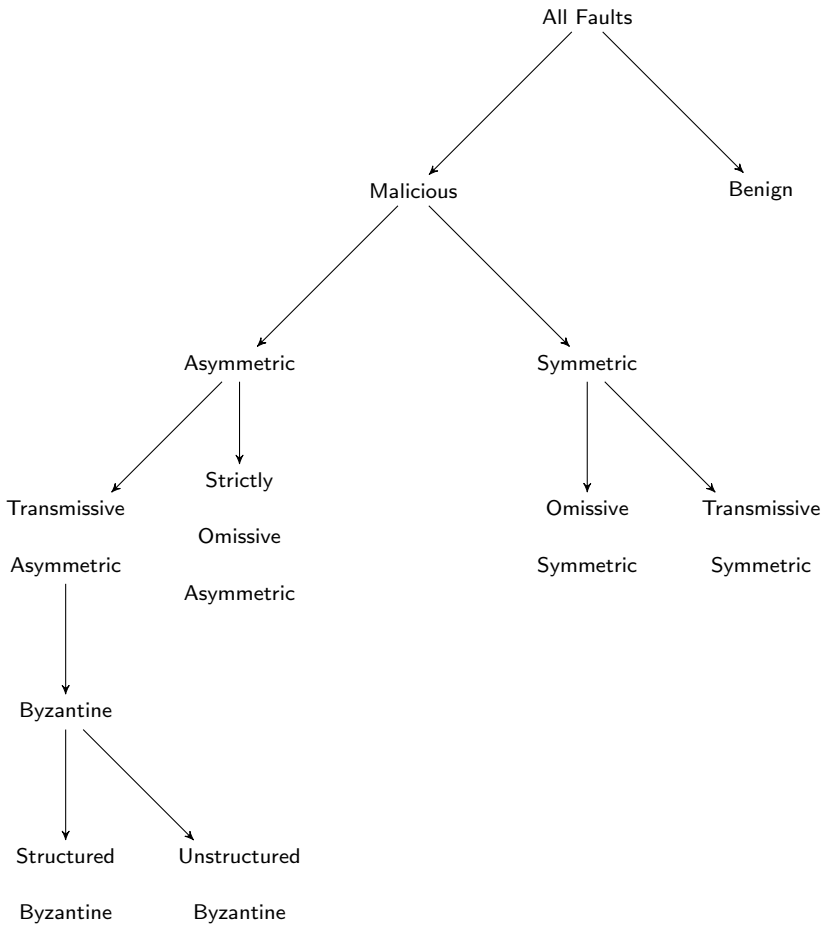


Figure 1.1. Fault Models and Classifications

1.1. Fault Classification

Fault definitions and classifications in the literature might differ and therefore the fault model to be used should be well-defined in order to eliminate the risk of confusion. In [11], fault models for wireless networks are discussed and a fault model classification is given based on the behaviour of each individual fault model. Hybrid fault models are discussed in [12] and [13]. In contrary to the fault classification in [12], some of the studies such as [1] and [4] suggest that the malicious fault model is enclosed by the *Byzantine* fault model. The term Byzantine was first introduced by *Lamport et al* [14], in which a consensus problem was presented in terms of Byzantine generals, and has widely been studied in the past as Byzantine consensus [15–18]. In order to clarify the Byzantine fault model used in this thesis, we extend the fault models given in [12] by dividing the Byzantine fault model into two categories: *structured Byzantine* and

unstructured Byzantine. Following [12], our fault model classification is summarized in Figure 1.1 which is detailed below.

Definition 1.1. (Transmissive Fault) *[12] A transmissive fault is one that sends erroneous information to its neighbors.*

Definition 1.2. (Omissive Fault) *[12] An omissive fault is one that can not send erroneous information to its neighbors but might fail to send to one or more of them.*

Brief explanations of the fault models in Figure 1.1 are given below:

- Benign: a benign fault is obvious to the non-faulty agents which simply ignore the information coming from such nodes.
- Malicious: a fault whose behaviour is not obvious to the non-faulty agents. Malicious faults might disrupt the operational structure of the network, either themselves or by collaborating with other malicious agents.
- Asymmetric Fault: a fault that has transmissive or omissive behaviour with respect to different agents.
- Transmissive Asymmetric Fault (Byzantine): an agent which belongs to this fault model can send any form of erroneous information to its neighbors.
  (i) Unstructured Byzantine (uStrBYZ): uStrBYZ agent sends different unstructured (i.e., random) erroneous information to different agents.
  (ii) Structured Byzantine (StrBYZ): StrBYZ is an agent that sends different structured (i.e., not random) information to different agents.
- Strictly Omissive Asymmetric Fault: a fault that can send correct messages to only one of its neighbors and no information to others.
- Symmetric Fault: an agent which belongs to this fault model has either transmissive or omissive behaviour.
  (i) Transmissive Symmetric Fault: a fault that sends the same erroneous information to all of its neighbors.
  (ii) Omissive Symmetric Fault: a fault that cannot send any information to any of its neighbors.

Figure 1.1, which is also presented in [19], adds to the the classification given in [12] in the sense that we introduce the *Structured Byzantine (StrBYZ)* and *Unstructured Byzantine (uStrBYZ)* fault models. In [1], a Byzantine agent is defined to be one that either sends different information to each of its neighbors or sends an arbitrary value to its neighbors. In this thesis, we distinguish between these two Byzantine behaviours and investigate in detail the case where the faulty agent can send some structured information that is not exactly arbitrary but different for each of its neighbors.

The StrBYZ fault model may arise in real world applications of wireless networks [20]. Similar malicious behaviour called *node replication attack* considered in [21] where adversary insert some nodes to the network that have exactly the same ID as some other nodes. In this thesis, it is shown that network of non-faulty nodes can achieve consensus and guarantee the approximate Byzantine consensus for synchronous and asynchronous networks without using a fault tolerant algorithm. Furthermore, conditions are presented for the non-faulty nodes to guarantee the approximate Byzantine group consensus in which non-faulty nodes reach more than one equilibrium point.

For the uStrBYZ fault model, it is shown that it is impossible to guarantee approximate Byzantine consensus without employing a fault tolerant algorithm. Therefore, two fault tolerant algorithms, so called Layered Mean-Select-Reduced (L-MSR) and Rooted Mean-Select-Reduced (R-MSR) are proposed. Necessary and sufficient conditions for the success of these algorithms are presented. Moreover, it is shown that L-MSR and R-MSR algorithms can be used to solve approximate Byzantine group consensus problem under the uStrBYZ fault model.

For the fault tolerant algorithms considered in [1, 2, 10], the maximum number of faulty agents are assumed to be initially known. However, the requirement of a priori knowledge on the scope of threads (i.e., the number of faulty agents) is quite restrictive. Another contribution of the thesis is to introduce a parameter independent fault tolerant algorithm that can be used to guarantee approximate Byzantine consensus. Using a fully distributed fault detection scheme, the requirement on the knowledge of the number of faulty agents is relaxed.

1.2. Contributions of the Thesis

The contributions of this thesis are as follows:

- For the first time, *structured Byzantine (StrBYZ)* fault model is introduced as a thread model. For StrBYZ, necessary and sufficient conditions for non-faulty nodes to achieve consensus are presented. These results are also extended for asynchronous networks.
- Non-faulty nodes are influenced by the misbehaviour of StrBYZ agents during consensus computation even if they reach consensus. Therefore, we consider approximate Byzantine consensus which includes a validity condition. For the first time, necessary and sufficient conditions for approximate Byzantine consensus to be guaranteed under the StrBYZ fault model are presented.
- Group consensus problem is investigated under the StrBYZ thread model. First, the conditions for non-faulty nodes to achieve group consensus are presented. Then, *Approximate Byzantine Group Consensus* problem is introduced. It is shown that the network of non-faulty nodes must be be resilient enough to guarantee approximate Byzantine group consensus.
- Another fault model that is considered in the thesis is *unstructured Byzantine (uStrBYZ)* fault model. Most/all of the fault tolerant algorithms given in the literature impose some restrictive conditions for the success of these algorithms. In this thesis, 2 resilient algorithms are proposed: *Layered MSR (L-MSR)* and *Rooted MSR (R-MSR)*. Conditions that must be satisfied for the fault tolerant algorithms in the literature are relaxed for the L-MSR and R-MSR algorithms.
- Group consensus problem is also investigated under the uStrBYZ thread model. Necessary and sufficient conditions for approximate Byzantine group consensus to be guaranteed using either R-MSR or L-MSR algorithm are presented.
- Using a fully distributed fault detection scheme, a parameter independent MSR algorithm (PIMSR) is proposed. This algorithm does not require the initial knowledge on the maximum number of faulty (Byzantine, malicious etc.) agents, i.e., it does not use the parameter f.

1.3. Organization of the Thesis

The thesis is organized as follows. In Chapter 2, mathematical notation and graph theoretic information that will be used throughout the thesis are given. The notions of distributed consensus and iterative Byzantine consensus are defined. Some conventional fault tolerant algorithms that exist in the literature are presented.

In Chapter 3, the structured Byzantine fault model is considered as a thread model. Using the expanded network representation, necessary and sufficient conditions for non-faulty nodes to achieve consensus and to guarantee approximate Byzantine consensus are presented.

In Chapter 4, the unstructured Byzantine fault model is considered. Two resilient consensus algorithms, L-MSR and R-MSR, are presented. Necessary and sufficient conditions for the success of these algorithms are given.

In Chapter 5, the concept of Approximate Byzantine Group Consensus is defined. It is shown that L-MSR and R-MSR algorithms are able to solve the approximate Byzantine group consensus problem if the thread model is uStrBYZ. For the StrBYZ fault model, non-faulty nodes of the network do not have to employ any fault tolerant algorithm to guarantee approximate Byzantine group consensus.

In Chapter 6, a parameter-independent fault tolerant algorithm is proposed using a fully distributed fault detection scheme. Finally, in Chapter 7, some comments are provided to conclude the topics discussed in the thesis.

2. DISTRIBUTED CONSENSUS REVIEW

In this chapter, we review the related graph terminology which includes the definitions of subgraph, primary and secondary layer subgraphs, reachability, robustness, regular graph and normal spanning component. After the linear consensus algorithm is presented, the notions of exact and approximate Byzantine consensus are defined. Three fault tolerant algorithms presented in the literature, so called the W-MSR algorithm, the asynchronous MSR algorithm and the Middle Algorithm, are reviewed.

2.1. Graph Terminology

Consider a directed graph denoted by $G = (V, E)$ where V is the node set indexed by the set $I = \{1, 2, \cdots, n\}$ and $E \subset V \times V$ is the directed edge set. It is assumed throughout the thesis that a directed edge $(i, j) \in E$ implies node j provides information to node i (exactly the opposite of graph terminology). Let $x_i[k]$ be the state value of node i and $x_{ij}[k]$ be the value of j received by node i. The set of in-neighbors of node i is defined as $N_i^+ = \{j \in V : (i, j) \in E\}$. Likewise, the set of out-neighbors of node i is defined as $N_i^- = \{j \in V : (j, i) \in E\}$. Since each node has access to its own state, we also consider the inclusive neighbors of node i, denoted by $N_i = N_i^+ \cup \{i\}$. The set of non-faulty (or normal) nodes in this network is denoted by $V_{nf} = V \setminus F$ where F is the set of faulty nodes. The cardinality of a set F is denoted by $|F|$. To avoid confusion, the terms agent and node will be used in the same sense.

Definition 2.1. (Subgraph) *Given two graphs $G = (V, E)$ and $\tilde{G} = (\tilde{V}, \tilde{E})$, $\tilde{G}$ is said to be a subgraph of the graph G if $\tilde{V} \subseteq V$ and $\tilde{E} \subseteq E \cap (\tilde{V} \times \tilde{V})$ hold for $\tilde{G}$.*

Definition 2.2. (Primary layer subgraphs) *[22, 23] Let $G = (V, E)$ be the digraph. There exists l_1 ($l_1 \geq 1$) subsets in the vertex set V such that each subset $V_{1,s}$, $s = 1, ..., l_1$, is the largest possible subset that has a spanning tree for its subgraph $G_{1,s}$ and for all $i \in V_{1,s}$ and $j \notin V_{1,s}$, we have $(i, j) \notin E$. We say $G_{1,s}$ are the primary layer subgraphs of G where the number of primary layer subgraphs is denoted by l_1.*

Definition 2.3. (Secondary layer subgraphs) *[22,23] Let $\bar{V}$ be the set which consists of the vertices that are not in the primary layer subgraphs, i.e., $\bar{V} = V \setminus \bigcup_{s=1}^{l_1} V_{1,s}$. Then there exists l_2 subsets in $\bar{V}$ such that each subset $V_{2,s}$, $s = 1, ..., l_2$, has a spanning tree for its subgraph and there exists exactly one node $i \in V_{2,s}$ which satisfies the following:*

- *For all $j \in V_{2,s} \setminus i$ and $m \in V \setminus V_{2,s}$, we have $(j, m) \notin E$.*
- *There exist at least two vertices in two different subgraphs (either primary or secondary layer) p and v such that $(i, p) \in E$ and $(i, v) \in E$.*
- *i is the root of a spanning tree in $V_{2,s}$.*

We define the subsets $V_{2,s}$, $s = 1, ..., l_2$ as the secondary layer subgraphs of the network G.

Definition 2.4. (Reachability) *[4] Given a digraph $G = (V, E)$ and a nonempty subset S of nodes ($S \subset V$) of G, we say that S is an r-reachable subgraph if there exists an $i \in S$ such that $|N_i \setminus S| \geq r$, where $r \in \mathbb{N}$.*

Definition 2.5. (Robustness) *[4] A digraph $G = (V, E)$ ($n \geq 2$) is r-robust with $r \in \mathbb{N}$, if for every pair of nonempty and disjoint subsets of V, at least one of the subsets is r-reachable.*

Definition 2.6. (Regular Graph) *[1] A digraph $G = (V, E)$ is regular if for all $i, j \in V$, $|N_i^+| = |N_j^+|$.*

Definition 2.7. (Normal Spanning Component) *Given a digraph $G = (V, E)$, a normal spanning component $G^{sp} = (V^{sp}, E^{sp})$ is defined as a component of G such that it has a spanning tree and there are no edges from $V \setminus V^{sp}$ to V^{sp} where $V^{sp} \subseteq V_{nf}$.*

2.2. Distributed Consensus Algorithm

Consider a network of n agents. If there is no faulty agent in the network, the i-th node uses the update rule

$$x_i[k+1] = \sum_{j \in N_i} w_{ij}[k] x_j[k] \tag{2.1}$$

where $w_{ij}[k]$ is the averaging coefficient assigned to node j's value by node i at time step k. For the update rule (2.1), the averaging coefficients are assumed to satisfy the following conditions:

- $w_{ij}[k] = 0$ whenever $j \notin N_i$, $i \in V$, $k \in \mathbb{Z}_{\geq 0}$;
- $0 < \alpha \leq w_{ij}[k]$, $\forall j \in N_i, i \in V$, $k \in \mathbb{Z}_{\geq 0}$ for some parameter α;
- $\sum_{j=1}^{n} w_{ij}[k] = 1$, $\forall i \in V$, $k \in \mathbb{Z}_{\geq 0}$.

The update rule (2.1) can be expressed in matrix form as

$$x[k+1] = W[k]x[k] \tag{2.2}$$

where $x[k] = [x_1[k], x_2[k], \cdots, x_n[k]]^T$ is the state vector of n agents and $W[k] = \big[w_{ij}[k]\big] \in R^{n \times n}$ is a row-stochastic matrix. It is shown in [24] that if the given network has a spanning tree, then the agents of this network achieve consensus using the update rule given in (2.1). When there exists a fault in the network, the dynamics of the network or even the topology of the network might change due to the faulty behaviour. For such cases, we need to take the effect of faults into account and develop a model on consensus dynamics.

2.3. Iterative Byzantine Consensus

In this section, the notions of exact and approximate Byzantine consensus are discussed in detail.

Definition 2.8. (Exact Byzantine Consensus) *[25] A network of non-faulty agents is said to achieve Exact Byzantine Consensus if each agent starts from an arbitrary initial value and all non-faulty agents achieve consensus with the following conditions:*

- *Agreement: No two non-faulty nodes reach on different state values.*
- *Validity: If all non-faulty nodes start with the same initial value x_0, then x_0 is the only possible state value on which a non-faulty node reaches consensus.*

For the case of exact Byzantine consensus, non-faulty nodes reach exactly the same value. However, this is almost impossible for real life applications. To remedy this situation, approximate Byzantine consensus is defined as follows:

Definition 2.9. (Approximate Byzantine Consensus) *[3] A network of non-faulty agents is said to achieve Approximate Byzantine Consensus if each agent starts from an arbitrary initial value and all non-faulty agents achieve consensus with the following conditions:*

- *Agreement: For any pre-assigned $\epsilon > 0$, all non-faulty nodes reach on state values that are within ϵ of each other.*
- *Validity: The state value of each non-faulty node must be in the range of initial values of the non-faulty nodes.*

In approximate Byzantine consensus, even if the steady state values of non-faulty nodes are not exactly equal, they still can reach on state values that are within ϵ of each other. There are many algorithms in the literature that have been proposed to solve the problem of reaching consensus in the presence of Byzantine or other fault models such as malicious, crashing, stopping, etc. In the next section, some of these algorithms will be introduced and their mathematical formulations will be explained.

2.4. Distributed Resilient Consensus Algorithms

In this section, the Weighted-Mean-Subsequence-Reduced (W-MSR) algorithm [1], the Middle Algorithm [26] and the asynchronous MSR algorithm [4] are presented.

2.4.1. Weighted-Mean-Subsequence-Reduced (W-MSR) Algorithm

The fault tolerant W-MSR algorithm is introduced by Leblanc *et al.* [1]. In the W-MSR algorithm, each non-faulty node i receives the state values of its neighbors at time step k. It is assumed that at most f of node i's neighbors might be misbehaving.

Since node i can not know which neighbor is faulty, it removes extreme values in order to update its value without using faulty data. The W-MSR algorithm is given as follows:

(i) At each time k, each non-faulty node i receives the values of its neighbors, and creates a sorted list.

(ii) If there are less than f values strictly larger than its own value, $x_i[k]$, then normal node i removes all values that are strictly larger than its own. Otherwise, it removes precisely the largest f values in the sorted list (breaking ties arbitrarily). Likewise, if there are less than f values strictly smaller than its own value, then node i removes all values that are strictly smaller than its own. Otherwise, it removes precisely the smallest f values.

(iii) Let $R_i[k]$ denote the set of nodes whose values are removed by normal node i in step (ii) at time-step k. Each normal node i applies the update

$$x_i[k+1] = \sum_{j \in N_i[k] \setminus R_i[k]} w_{ij}[k] x_j[k] \tag{2.3}$$

where the weights $w_{ij}[t]$ satisfy the following conditions:

- $w_{ij}[k] = 0$ whenever $j \notin N_i$, $i \in V$, $k \in \mathbb{Z}_{\geq 0}$;
- $0 < \alpha \leq w_{ij}[k]$, $\forall j \in N_i, i \in V$, $k \in \mathbb{Z}_{\geq 0}$;
- $\sum_{j=1}^{n} w_{ij}[k] = 1$, $\forall i \in V$, $k \in \mathbb{Z}_{\geq 0}$.

The W-MSR algorithm is a light-weight algorithm. Each agent only uses the values that it receives. Non-faulty nodes that are implementing W-MSR do not have to remove all extreme values. In this way, removal of the information coming from another non-faulty node can be prevented.

2.4.2. The Asynchronous MSR Algorithm

The asynchronous MSR algorithm is introduced by Haseltalab and Akar (2015) [4]. An important aspect of this algorithm is that it can solve the approximate Byzantine consensus problem for both synchronous and asynchronous networks independent of bounded time delays. Let the set $\{k_0^i, k_1^i, k_2^i, ..., k_r^i\}$, $i \in V$ represent the iteration instants of node i, r be the number of the iterations and f be the maximum number of Byzantine agents in the network. The asynchronous MSR algorithm goes as follows:

(i) Each non-faulty node i sends its state value $x_i[k_r^i]$ at the time step k_r^i.

(ii) Each non-faulty node i saves the values that it has received in the time interval Δk_r^i. If it has received more than one state value from a node, it considers the most recent value that it has received. If it has not received any value from its neighbor node in its update time interval Δk_r^i, it uses the old received values. Then, it sorts the received values in the increasing order.

(iii) Node i eliminates f_a ($f_a > f$) number of maximum values in the sorted list that are greater than its current state value. If the number of greater values is less than f_a, it eliminates all of them. Similarly, it deletes the f_a number of minimum values in the list that are less than its state value. Let $E_i[k_r^i]$ be the set of nodes with deleted state values.

(iv) Node i updates its state value as follows.

$$x_i[k_{r+1}^i] = \sum_{j \in N_i[k_r^i] \setminus E_i[k_r^i]} w_{ij}[k_r^i] x_{ij}[k_r^i] \tag{2.4}$$

The averaging coefficients $w_{ij}[k_r^i]$ are supposed to satisfy the following conditions:

- $w_{ij}[k_r^i] = 0$ whenever $j \notin N_i[k_r^i] \setminus E_i[k_r^i]$;
- $0 < \alpha \leq w_{ij}[k_r^i] \leq 1$, $\forall j \in N_i[k_r^i] \setminus E_i[k_r^i]$;
- $\sum_{j=1}^{n} w_{ij}[k_r^i] = 1$, $\forall i \in V$, $\forall j \in N_i[k_r^i] \setminus E_i[k_r^i]$;

It can be noted that there is no difference between the family of MSR algorithms and the linear consensus algorithm if f_a is chosen as $f_a = 0$.

2.4.3. Middle Algorithm

The Middle algorithm is an iterative approximate Byzantine consensus (IABC) algorithm introduced by Tseng and Vaidya (2012) [26]. Each node i keeps its own state at the time step k denoted with $x_i[k]$ where $k \geq 0$. Initial state of node i, $x_i[0]$ is also expressed as the initial input of node i. At time step k ($k > 0$), the state of node i is $x_i[k-1]$ and the Middle algorithm is given as follows:

(i) *Transmit:* Each node transmits its current state to its neighbors.

(ii) *Receive:* Each node receives state values from its neighbors and forms a list $r_i[k]$ of size $\left|N_i^-\right|$ where N_i^- represents the out-neighbors of node i.

(iii) *Update:*

- Sort the values in $r_i[k]$ in an increasing order and use the sorted order of values to form a partition of nodes in N_i^- into sets B, M, T as follows: (a) set B contains nodes from whom the smallest $\lfloor |N_i^-|/3 \rfloor$ values in the sorted list $r_i[k]$, (b) set T contains nodes from whom the largest $\lfloor |N_i^-|/3 \rfloor$ values in the sorted list $r_i[k]$, (c) set M contains the remaining nodes from whom the values in the "middle" of the sorted list $r_i[k]$. That is, $M = N_i^- - B - T$. Thus, it can be inferred that $|M| = |N_i^-| - 2\lfloor |N_i^-|/3 \rfloor$.
- Define update rule

$$x_i[k] = \sum_{j \in \{i\} \bigcup M} a_i[k] x_j[k-1] \tag{2.5}$$

where

$$a_i[k] = \frac{1}{|M|+1} = \frac{1}{|N_i^-| - 2\lfloor |N_i^-|/3 \rfloor + 1}$$

The Middle algorithm is a parameter independent algorithm that can guarantee approximate Byzantine consensus for non-faulty nodes. Even though it does not require the initial knowledge on the maximum number of faulty agents in the network, there are some differences between the Middle algorithm and the MSR algorithms in terms of topological conditions of the network. Please refer to [26] for details.

2.5. Summary of the Chapter

In this chapter, we have introduced the graph theoretic concepts that will be used throughout the thesis. Then, the linear consensus algorithm has been presented. The concepts of exact and approximate Byzantine consensus have been explained.

Some of the resilient consensus algorithms presented in the literature such as W-MSR algorithm [1], the asynchronous MSR algorithm [4] and the Middle algorithm [26] have been reviewed. Operational structures of these algorithms have been briefly discussed.

3. RESILIENT CONSENSUS UNDER STRUCTURED BYZANTINE FAULTS

In this chapter, the structured Byzantine fault (StrBYZ) is defined and considered as the thread model. For synchronous networks, necessary and sufficient conditions for non-faulty nodes to achieve approximate Byzantine consensus are given. Then, the extension of these results for asynchronous networks are presented.

3.1. Adversary Behaviour and Fault Model

Consider a network $G = (V, E)$ of n agents. Let $G_{nf} = (V_{nf}, E_{nf})$ denote the network of the normal agents in the network G.

Definition 3.1. (Structured Byzantine) *A node $i \in F$ is said to be StrBYZ if it sends different information to its out-neighbors while employing $|N_i^-|$ different update rule (2.1) starting with $|N_i^-|$ different initial conditions.*

The definition of StrBYZ fits the definition of Byzantine faults given in [1] where the StrBYZ agent sends different information to each of its neighbors. However, an StrBYZ agent does not update its state value arbitrarily. That is why we represent this type of Byzantine faults as structured Byzantine which may be encountered in wireless networks [20] or sensor networks [21].

3.2. Consensus Model in the Presence of StrBYZ

Consider a network of n agents where f of the agents are StrBYZ. Due to the misbehaviour of a StrBYZ agent, we have $x_{ji}[k] \neq x_{mi}[k]$ for $i \in N_j \cap N_m$. Therefore, we can represent the update rule for the non-faulty i-th node as

$$x_i[k+1] = \sum_{j \in N_i \cap V_{nf}} w_{ij} x_j[k] + \sum_{j \in N_i \cap F} w_{ij} x_{ij}[k], \;\; i \in V_{nf} \tag{3.1}$$

where $x_{ij}[k]$ represents the information sent by the faulty node j to the non-faulty node i at the time step k. Note that the averaging coefficients w_{ij} satisfy the same conditions with the coefficients in the update rule (2.1). On the other hand, a faulty node i uses $|N_i^-|$ different updates of (2.1) by creating $|N_i^-|$ different copies. These copies update their values by choosing different weights and starting with different initial conditions. Mathematically, for each $i \in N_j^-$, a faulty agent j uses the following update rule

$$x_{ij}[k+1] = \sum_{l \in N_j \cap V_{nf}} \tilde{w}_{ij,l} x_l[k] + \sum_{l \in N_j \cap F} \tilde{w}_{ij,l} x_{jl}[k], j \in F \tag{3.2}$$

where $\tilde{w}_{ij,l}$ are the coefficients that satisfy the same conditions with the coefficients in the update rule (2.1).

3.3. Analysis Using An Expanded Network Representation

In this section, the behaviour of non-faulty nodes is investigated by using an expanded network representation. Since the number of out-neighbors of StrBYZ agents is important for our fault model described in Section 3.1, the following three cases are considered:

Case 1: For all $i \in F$, $|N_i^-| \leq 1$. $\big($From Definition 3.2 we have $G = G'$, $V = V'$ and $E = E'\big)$.

Case 2: For some $i \in F$, $|N_i^-| = 1$ and for some $i \in F$, $|N_i^-| > 1$. $\big($Each of $i \in F$, $|N_i^-| = 1$ still behaves like a normal node$\big)$.

Case 3: For all $i \in F$, $|N_i^-| \geq 1$.

Note that a faulty node i does not create a copy of itself to generate a state value when it does not have any out-neighbors as stated in Case 1. This behaviour of the faulty agent i is the same for Case 2 in which a faulty node has only 1 out-neighbour, i.e., faulty node i does not create a copy of itself to produce a state value different from its own state value. Therefore, it behaves exactly the same as a normal node.

However, when the faulty node i has more than one out-neighbor, it creates copies to produce different state values for each of its out-neighbors. Therefore, we need to consider Case 3 only, as Case 1 and Case 2 do not lead to faulty behaviour.

3.3.1. Expanded Network Representation for StrBYZ Fault Model

Since $x_{ji}[k] \neq x_{mi}[k]$, $i \in N_j \cap N_m$, $j, m \in V$ holds for all $i \in F$, faulty agents can be modelled by making appropriate changes on the network topology as follows:

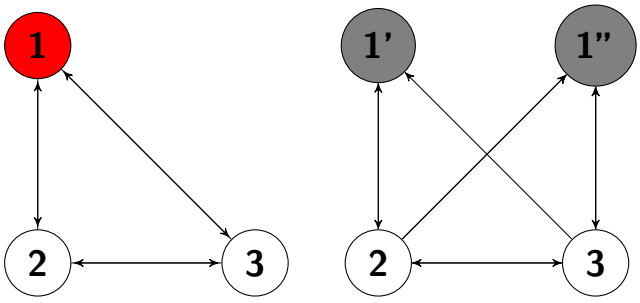
- All existing connections between non-faulty nodes are preserved.
- Each faulty node creates one copy of itself for each of its out-neighbors by preserving its in-neighbours and these copies are disconnected if they are generated from the same faulty node.

Definition 3.2. (Expanded Network of StrBYZ) *Consider a network with associated graph $G = (V, E)$ where E is the edge set; V_{nf} and F denote the sets of non-faulty and StrBYZ faulty agents, respectively. Let*

- $V_i' = \{v_i^1, v_i^2, ..., v_i^{|N_i^-|}\}$, *be represent the vertex set of the copies for each $i \in F$,*
- $E_1' = \{(j, q) : (j, i) \in E, j \in N_i^-, i \in F, q \in V_i'\}$,
- $E_2' = \{(q, j) : (i, j) \in E, j \in N_i^+, i \in F, q \in V_i'\}$,
- $V' = \bigcup_{i \in F} V_i'$,
- $E' = E_1' \cup E_2'$.

Then, the resulting "expanded" network can be represented with associated graph $\bar{G} = (\bar{V}, \bar{E})$ where $\bar{V} = V_{nf} \cup V'$ and $\bar{E} = E_{nf} \cup E'$.

The neighborhood definitions for the expanded network are exactly the same as in the original network. In-neighbor, out-neighbor and inclusive neighbor of node i in the expanded network $\bar{G}$ are denoted by $\bar{N}_i^+$, $\bar{N}_i^-$ $\bar{N}_i$, respectively. The StrBYZ behaviour is illustrated by the expanded network representation in the following example.



(a) The original network (b) The expanded network

Figure 3.1. Original and Expanded Networks where node 1 is StrBYZ.

Example 3.1. *Consider the network* $G = (V, E)$ *with* $n = 3$ *given in Figure 3.1(a) where* $V = \{1, 2, 3\}$ *and* $E = \{(2,1), (3,1), (1,2), (3,2), (1,3), (2,3)\}$. *Suppose that node* 1 *is faulty. Since we have* $N_1^- = \{2, 3\}$ *where* $|N_1^-| = 2$, *there should be 2 copies of node 1 whose in-neighbors are exactly the same with node 1. Hence, we have* $V' = \{1', 1''\}$ *where* $N_1^+ = \bar{N}_{1'}^+ = \bar{N}_{1''}^+ \neq \emptyset$. *The resulting expanded network* $\bar{G} = (\bar{V}, \bar{E})$ *is given in Figure 3.1(b). The edge set of the expanded network is* $\bar{E} = E_{nf} \cup E'$ *where* $E_{nf} = \{(3,2), (2,3)\}$ *and* $E' = \{(2,1'), (3,1''), (1',2), (1'',2), (1',3), (1'',3)\}$. *Since* $E' = E_1' \cup E_2'$ *holds, we have* $E_1' = \{(2,1'), (3,1'')\}$ *and* $E_2' = \{(1',2), (1'',2), (1',3), (1'',3)\}$.

Using Definition 3.2, we can model the misbehaviour of StrBYZ agents. To this end, let the state value for each $j \in V_i'$ be represented in (3.2) by $x_{ij}[k]$. Let $x[k] = \big[x_1[k], x_2[k], \cdots, x_{n_1}[k]\big]^T$ and $\tilde{x}[k] = \big[\tilde{x}_1[k], \tilde{x}_2[k], \cdots, \tilde{x}_f[k]\big]^T$ be the state vectors of the non-faulty and faulty nodes where $\tilde{x}_j[k] = \big[x_{ij}[k]\big]_{i \in N_i^-}$ for each $j \in F$. Then, we can combine (3.1) and (3.2) to obtain

$$\bar{x}[k+1] = \bar{W}\bar{x}[k] \tag{3.3}$$

where $\bar{x}[k] = \big[x^T[k], \tilde{x}^T[k]\big]^T$ and $\bar{W} = [\bar{w}_{ij}]$ is the row-stochastic system matrix, whose elements $\bar{w}_{ij}$ satisfy the same conditions with the coefficients in the update rule (2.1), in the form

$$\bar{W} = \begin{bmatrix} W_{11} & W_{12} & \cdots & W_{1(f+1)} \\ \tilde{W}_{21} & \cdots & \cdots & \tilde{W}_{2(f+1)} \\ \vdots & \vdots & \vdots & \vdots \\ \tilde{W}_{(f+1)1} & \cdots & \cdots & \tilde{W}_{(f+1)(f+1)} \end{bmatrix} \tag{3.4}$$

In (3.4), the matrices W_{ij} and $\tilde{W}_{ij}$ consist of the coefficients in (3.1) and (3.2), respectively. Note that the dimension of $\bar{W}$ depends on the number of faulty nodes and the out-neighbor of each faulty node.

Equation (3.3) is the mathematical representation of the expanded network defined in Section 3.3.1. Using this representation, we first investigate the conditions so that the non-faulty nodes achieve consensus in the expanded network. Then, we seek the necessary and sufficient conditions on the original network for the system given in Equation (3.3) to reach asymptotic consensus.

3.3.2. Results on the Expanded Network

In this part, we investigate the topological conditions on the expanded network for non-faulty nodes to achieve consensus. We first recall the following results from [10].

Lemma 3.1. *[10] Let $G = \{V, E\}$ be a digraph. The network G has a spanning tree if and only if it is 1-robust.*

Lemma 3.2. *[10] Let $G = \{V, E\}$ be an r-robust graph. Then, the graph $G' = \{\{V, j_{new}\}, \{E, E_{new}\}\}$, where j_{new} is a new vertex added to G and E_{new} is the edge set related to j_{new}, is r-robust if j_{new} has at least r in-neighbors in the existing graph G'.*

Lemma 3.3. *Let the network G have a spanning tree. A spanning tree exists in the expanded network $\bar{G}$ if and only if each faulty node in the network G has at least one in-neighbor, i.e., $|N_i^+| \geq 1$, for all $i \in F$.*

Proof. (Sufficiency) Suppose $|N_i^+| \geq 1$, for all $i \in F$. This implies that each of the generated copies of the faulty agents also has at least 1 in-neighbour in $\bar{G}$, i.e., $|N_i^+| \geq 1$, for each $i \in V'$. Since the original network G is 1-robust, the expanded network $\bar{G}$ is also 1-robust. By Lemma 3.1 and Lemma 3.2, the expanded network $\bar{G}$ has a spanning tree.

(Necessity) Suppose that $|N_i^+| < 1$ or simply $|N_i^+| = 0$ for some $i \in F$. This implies that each of the generated copies of the faulty agents does not have any in-neighbours in $\bar{G}$, i.e., $|N_i^+| = 0$, for each $i \in V'$. Even if the original network G is 1-robust, from Lemma 3.3 the expanded network $\bar{G}$ is not 1-robust. Hence, it does not have a spanning tree. □

Lemma 3.3 states a condition under which faulty behaviour does not disrupt consensus. Next, we relate this condition to the network topology by using the expanded network model.

Theorem 3.1. *Non-faulty nodes of the network G achieve consensus in the presence of faulty nodes if and only if they are all in the same primary or secondary layer in the expanded network $\bar{G}$.*

Proof. (Sufficiency) Suppose that non-faulty nodes are all in the same primary or secondary layer in the expanded network $\bar{G}$. In [22], it is shown that the agents reach the same consensus equilibria if and only if they are in the same primary or secondary layer. Using the main theorem of [22], we can conclude that non-faulty nodes of the network G achieve consensus.

(Necessity) Suppose that non-faulty nodes are not all in the same primary or secondary layer in $\bar{G}$. Then, the necessary and sufficient condition given in [22] is not satisfied and non-faulty nodes cannot achieve consensus. □

Since the expanded network $\bar{G}$ is created due to faulty behaviour, it is important for non-faulty nodes to achieve consensus in the expanded network $\bar{G}$. Theorem 3.1 states the condition on $\bar{G}$ that must be satisfied for non-faulty nodes to achieve consensus in the presence of the StrBYZ agents.

3.4. Analysis On The Original Network

In this section, we investigate the topological conditions on the original network for non-faulty nodes to achieve consensus. More specifically, we seek the conditions on the original network G that are necessary and sufficient to satisfy the conditions given in Theorem 3.1 on the expanded network $\bar{G}$.

Lemma 3.4. *Non-faulty nodes of the network G are all in the same primary layer in the expanded network $\bar{G}$ if and only if the following conditions hold:*

- (i) *Non-faulty nodes are all in the same primary layer ($V_{1,s}$) in the original network G, i.e., $V_{nf} \subseteq V_{1,s}$ where $s \in \{1, 2, \cdots, l_1\}$, $l_1 \geq 1$.*
- (ii) *Each faulty node in this primary layer $V_{1,s}$ has at least one in-neighbor, i.e., for each $i \in V_{1,s} \cap F$, $|N_i^+| \geq 1$.*

Proof. (Sufficiency) Suppose that conditions (i) and (ii) hold. By Lemma 3.3, a spanning tree exists in the expanded network $\bar{G}$. For any $i \in V_{nf}$, $i \in V_{1,s}$ where $V_{1,s}$ is the set of nodes in a primary layer $G_{1,s}$ of G, it follows that $i \in \bar{V}_{1,s}$ where $\bar{V}_{1,s}$ is the set of nodes in a primary layer $\bar{G}_{1,s}$ of $\bar{G}$.

(Necessity) Suppose that condition (i) does not hold. Then, $V_{nf} \nsubseteq V_{1,s}$ for an arbitrary s value. This immediately implies that $V_{nf} \nsubseteq \bar{V}_{1,s}$ from Definition 3.2 regardless of condition (ii).

Suppose that condition (ii) does not hold. Then, from Lemma 3.3, existence of a spanning tree in the expanded network $\bar{G}$ is not guaranteed which implies that the non-faulty nodes are no longer in the same primary layer in $\bar{G}$, i.e., $V_{nf} \nsubseteq \bar{V}_{1,s}$. $\square$

Lemma 3.4 states conditions on the original network G for all non-faulty nodes to be in the same primary layer in the expanded network $\bar{G}$. In the following, we present conditions on G for all non-faulty nodes to be in the same secondary layer in $\bar{G}$.

Lemma 3.5. *Non-faulty nodes of the network G are all in the same secondary layer in the expanded network $\bar{G}$ if and only if the following conditions hold:*

- (i) *Non-faulty nodes are all in the same secondary layer $V_{2,s}$ having a non-faulty root node in the network G, i.e., $V_{nf} \subseteq V_{2,s}$ where $s \in \{1, 2, \cdots, l_2\}$, $l_2 \geq 0$.*
- (ii) *Each faulty node in the secondary layer $V_{2,s}$ has at least one in-neighbor, i.e., for each $i \in V_{2,s} \cap F$, $|N_i^+| \geq 1$.*

Proof. (Sufficiency) Suppose that conditions (i) and (ii) hold. By using Lemma 3.3 and the fact that the root node of the secondary layer is non-faulty, a spanning tree exists among the non-faulty nodes in the expanded network $\bar{G}$. For any $i \in V_{nf}$, $i \in V_{2,s}$ where $V_{2,s}$ is the set of nodes in a secondary layer $G_{2,s}$ of G, it follows that $i \in \bar{V}_{2,s}$ where $\bar{V}_{2,s}$ is the set of nodes in a secondary layer $\bar{G}_{2,s}$ of $\bar{G}$.

(Necessity) Suppose that condition (i) does not hold. Then, $V_{nf} \nsubseteq V_{2,s}$ for an arbitrary s value. This immediately implies $V_{nf} \nsubseteq \bar{V}_{2,s}$ from Definition 3.2 regardless of condition (ii).

Suppose that condition (ii) does not hold. Then, by using Lemma 3.3 we conclude that a spanning tree no longer exists among the non-faulty nodes in the expanded network $\bar{G}$. This is further implies that the non-faulty nodes are not in the same secondary layer in the expanded network $\bar{G}$, i.e., $V_{nf} \nsubseteq \bar{V}_{2,s}$. □

Lemma 3.4 and Lemma 3.5 state necessary and sufficient conditions on the original network G for non-faulty nodes to be in the same primary or secondary layer in the expanded network $\bar{G}$, respectively. Using these conditions, we present the main result on the original network G.

Theorem 3.2. *Non-faulty nodes of the network G achieve consensus in the presence of faulty nodes if and only if the following conditions hold:*

(*i*) *Non-faulty nodes are either all in the same primary layer ($V_{1,s}$) or in the same secondary layer ($V_{2,s}$) having a non-faulty root node, i.e., $V_{nf} \subseteq V_{m,s}$ where $s \in \{1, 2, \cdots, l_m\}$ and $m \in \{1, 2\}$.*

(*ii*) *Each faulty node in the same layer that contains all non-faulty nodes in the network G has at least one in-neighbor, i.e., for each $i \in V_{m,s} \cap F$, $|N_i^+| \geq 1$.*

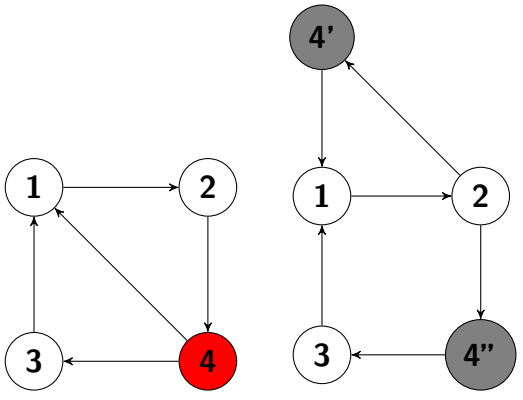
Proof. (Sufficiency) Sufficiency part of the proof follows from Lemma 3.4 and Lemma 3.5 directly.

(Necessity) Suppose that condition (*i*) does not hold. From Lemma 3.4 and Lemma 3.5, we have $V_{nf} \nsubseteq \bar{V}_{l,p}$ where $l \in \{1, 2\}$. This contradicts the condition in Theorem 3.1. Therefore, non-faulty agents cannot reach consensus regardless of condition (*ii*).

Suppose that condition (*ii*) does not does not hold. By Lemma 3.3, spanning tree of G is not preserved which implies that $V_{nf} \nsubseteq \bar{V}_{1,s}$. □

Remark 3.1. *Note that it is impossible for faulty nodes to constitute 2 primary layers in the expanded network $\bar{G}$ when $f < 6$ in the network G under Case 3 ($|N_i^-| > 1$). Therefore, non-faulty nodes achieve consensus when $f < 6$ if and only if the topological conditions in Lemma 3.4 are satisfied.*

Example 3.2. *Consider the network given in Figure 3.2(a) where node 4 is assumed to be faulty. As it can be seen from Figure 3.2(a), the original network graph has a spanning tree. Thus, the original system is itself a primary layer. Since node 4 (faulty) node has 1 in-neighbor, i.e., $N_4^+ = \{2\}$, it follows from Lemma 3.4 that non-faulty nodes are all in the same primary layer in the expanded network given in Figure 3.2(b). Therefore, consensus is achieved in the presence of the StrBYZ agent. Simulation result shown in Figure 3.3 demonstrate that non-faulty nodes achieve consensus.*



(a) The original network (b) The expanded network

Figure 3.2. Original and Expanded Networks where node 4 is StrBYZ.

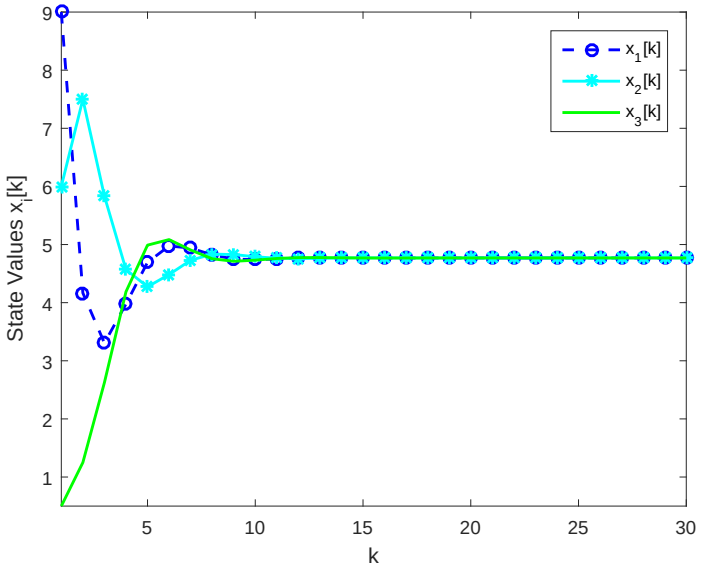


Figure 3.3. The simulation result of the network given in Figure 3.2(b)

3.5. On Approximate Byzantine Consensus

For the given directed network G, let $M[k]$ and $m[k]$ denote the maximum and minimum state values of the set of non-faulty nodes at time step k. In [4], the maximum difference between the state values of non-faulty nodes is given as:

$$e[k] = M[k] - m[k] \tag{3.5}$$

where $e[k] \geq 0$ for all $k \geq 0$ and $e[k] = 0$ if and only if $M[k] = m[k]$. Therefore, approximate Byzantine consensus is guaranteed for the network of non-faulty nodes if the following conditions hold:

- *Agreement*: There exists a $k_0 \geq 0$ such that we have $e[k] \leq \epsilon$ for all $t \geq k_0$.
- *Validity*: For all $k \geq 0$, $m(0) \leq m[k] \leq M[k] \leq M(0)$.

It is important to notice that the results presented in Section 3.3 and Section 3.4 are valid for the case in which non-faulty nodes achieve consensus, i.e., the agreement condition of the approximate Byzantine consensus holds for the network of non-faulty nodes. However, consensus equilibrium is not expected to be in the range of the initial values of non-faulty nodes due to the faulty behavior of StrBYZ agents, i.e., the validity condition of the approximate Byzantine consensus does not hold. In the following, necessary and sufficient conditions for non-faulty nodes to guarantee the approximate Byzantine consensus are presented.

3.5.1. Approximate Byzantine Consensus On The Expanded Network

In this part, we investigate the topological conditions on the expanded network for non-faulty nodes to guarantee approximate Byzantine consensus. In Theorem 3.2, the conditions for non-faulty nodes to achieve consensus, i.e., agreement conditions of the approximate Byzantine consensus are presented. However, these results are not sufficient for the validity condition. Using the notion of *normal spanning component*, we present the main result on the expanded network $\bar{G}$.

Theorem 3.3. *For a given directed network G, approximate Byzantine consensus is guaranteed for non-faulty nodes in the presence of StrBYZ agents if and only if they are all in the same primary or secondary layer having at least one normal spanning component in the expanded network $\bar{G}$, i.e., $\{V_{nf} \cup V^{sp}\} \subseteq \bar{V}_{m,s}$ where $s \in \{1, 2, \cdots, l_m\}$ and $m \in \{1, 2\}$.*

Proof. (Sufficiency) Suppose that non-faulty nodes are all in the same primary or secondary layer having at least one normal spanning component in the expanded network $\bar{G}$. In [22], it is shown that the agents reach the same consensus equilibria if and only if they are in the same primary or secondary layer. Using the main theorem of [22], we can conclude that non-faulty nodes of the network G achieve consensus (agreement). Furthermore, it is easy to see that the normal spanning component acts like a leader in the layer and all other non-faulty nodes reach the consensus equilibrium determined by normal spanning component (validity). In that way, approximate Byzantine consensus is guaranteed for non-faulty nodes of the network G.

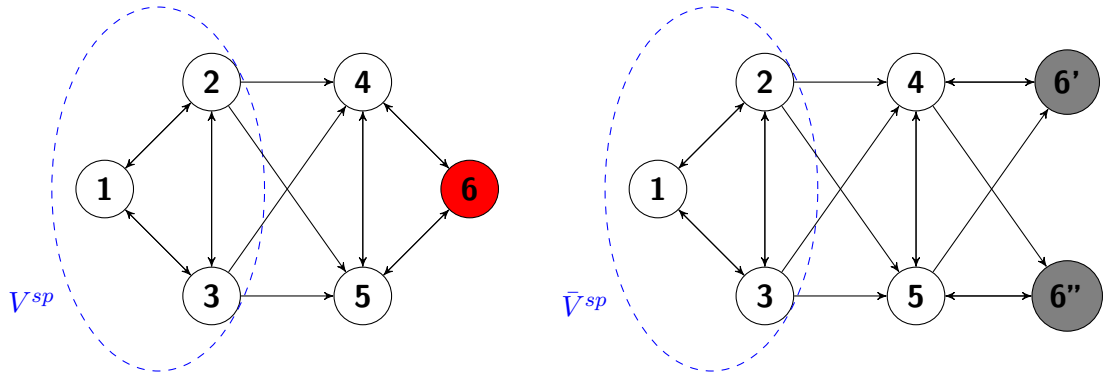
(Necessity) Suppose that non-faulty nodes are not all in the same primary or secondary layer in $\bar{G}$. Then, the necessary and sufficient condition given in [22] is not satisfied and non-faulty nodes cannot achieve consensus. Therefore, approximate Byzantine consensus is impossible. □

Since the expanded network $\bar{G}$ is created due to faulty behaviour, it is important for non-faulty nodes to guarantee the approximate Byzantine consensus in the expanded network $\bar{G}$. Theorem 3.3 states the condition on $\bar{G}$ that must be satisfied for non-faulty nodes to achieve consensus in the presence of the StrBYZ agents.

3.5.2. Approximate Byzantine Consensus On The Original Network

In this section, we investigate the topological conditions on the original network for non-faulty nodes to guarantee approximate Byzantine consensus. More specifically, we seek the conditions on the original network G that are necessary and sufficient to satisfy the conditions on the expanded network $\bar{G}$ given in Theorem 3.3.

Theorem 3.4. *For a given directed network G, approximate Byzantine consensus is guaranteed for non-faulty nodes in the presence of StrBYZ nodes if and only if the following conditions hold:*



(a) The original network (b) The expanded network

Figure 3.4. Original and Expanded Networks where node 6 is StrBYZ.

(*i*) *Non-faulty nodes are either all in the same primary layer* ($V_{1,s}$) *or in the same secondary layer* ($V_{2,s}$) *having a non-faulty root node. Furthermore, the primary or secondary layer that contains non-faulty nodes has a normal spanning component i.e.,* $\{V_{nf} \cup V^{sp}\} \subseteq V_{m,s}$ *where* $s \in \{1, 2, \cdots, l_m\}$ *and* $m \in \{1, 2\}$.

(*ii*) *Each faulty node in the same layer that contains all non-faulty nodes in the network* G *has at least one in-neighbor, i.e., for each* $i \in V_{m,s} \cap F$, $|N_i^+| \geq 1$.

Proof. (Sufficiency) Sufficiency part of the proof follows from Lemma 3.4, Lemma 3.5 and Theorem 3.1 directly.

(Necessity) Suppose that condition (*i*) does not hold. From Lemma 3.4 and Lemma 3.5, we have $V_{nf} \nsubseteq \bar{V}_{m,s}$ where $m \in \{1, 2\}$. This contradicts the condition in Theorem 3.3. Therefore, approximate Byzantine consensus is not possible for non-faulty agents regardless of condition (*ii*).

Suppose that condition (*ii*) does not hold. By Lemma 3.3, spanning tree of G is not preserved which implies that $V_{nf} \nsubseteq \bar{V}_{m,s}$. □

Example 3.3. *Consider the network given in Figure 3.4(a) where node 6 is StrBYZ. As it can be seen from Figure 3.4(a), the original network has a spanning tree and therefore it is itself a primary layer. Initial state values of non-faulty nodes are chosen as* $x[0] = [0.5, 1, 1.5, 2, 2.5, 3]^T$. *Since node 6 (faulty) node has 2 in-neighbor, i.e.,* $N_6^+ = \{4, 5\}$, *it follows from Lemma 3.4 that non-faulty nodes are all in the same*

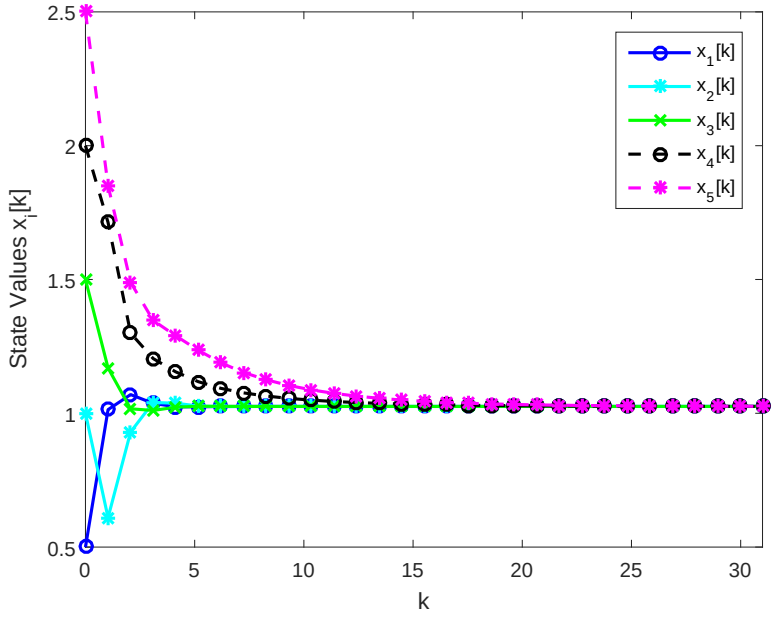


Figure 3.5. The simulation result of the network given in Figure 3.4(b)

primary layer in the expanded network given in Figure 3.2(b). Moreover, the network has a normal spanning component which contains the non-faulty nodes 1, 2 and 3, i.e., $V^{sp} = \{1, 2, 3\}$. *From Theorem 3.4, it can be concluded that approximate Byzantine consensus is guaranteed for non-faulty nodes of the network. As it can be seen from Figure 3.5, agreement condition is satisfied on the state value* $x_{final} = 1.0256$ *which is in the range of the initial state values. Therefore, validity condition is also satisfied.*

3.6. Extension to Asynchronous Networks

In this section, consensus computation in the presence of StrBYZ agents is investigated for asynchronous networks. Let the set $\{k_0^i, k_1^i, k_2^i, ..., k_r^i\}$ where $i \in V$ represents the iteration instants of node i and $r \geq 0$ be the number of the iterations. Then, the update rules given in (3.1) and (3.2) can be rewritten as follows:

$$x_i[k_{r+1}^i] = \sum_{j \in N_i[k_r^i] \cap V_{nf}} w_{ij}[k_r^i] x_j[k_r^i] + \sum_{j \in N_i[k_r^i] \cap F} w_{ij}[k_r^i] x_{ij}[k_r^i], \;\; i \in V_{nf} \qquad (3.6)$$

$$x_{ij}[k^i_{r+1}] = \sum_{l \in N_j[k^i_r] \cap V_{nf}} \tilde{w}_{ij,l}[k^i_r] x_l[k^i_r] + \sum_{l \in N_j[k^i_r] \cap F} \tilde{w}_{ij,l}[k^i_r] x_{jl}[k^i_r], \;\; j \in F \tag{3.7}$$

where $x_{ij}[k]$ represents the information sent by the faulty node j to the non-faulty node i at the time step k. Note that the averaging coefficients w_{ij} and $\tilde{w}_{ij,l}$ satisfy the same conditions with the coefficients in the update rule (2.1).

Since we have $k^i_r \neq k^j_r$ for each $i, j \in V_{nf}$ for asynchronous networks, non-faulty nodes update their state values at different time instants. However, the number of time steps between two successive update time instants of each non-faulty node i must be bounded, i.e., $\Delta k^i_r = k^i_{r+1} - k^i_r$. In [4] and [5], it has been discussed that there exist a minimum update time interval Δk_m and a maximum update time interval Δk_M in the network with or without bounded time delays. Let $\{0, k_{M_1}, \cdots, k_{M_r}, k_{M_{r+1}}, \cdots\}$ be the set of time sequences with $k_{M_{r+1}} = k_{M_r} + \Delta k_M$ and $k_{m_{r+1}} = k_{m_r} + \Delta k_m$. As a result, it can be deduced that each non-faulty node receives at least one state value from its in-neighbors in the time interval Δk_M.

In [27] and [28], it is shown that consensus is achieved in an asynchronous network without time delays if and only if the network has a spanning tree. For the StrBYZ fault model, we seek the conditions on graph topology so that the network has a spanning tree in the expanded network created by faulty behaviour. Furthermore, the assumption made in [4] and [5] guarantees for each non-faulty node to receive at least one state value from in-neighbors in a bounded time interval. Therefore, it follows that the conditions stated Theorem 3.2 and Theorem 3.3 are valid for asynchronous networks with or without time delays under this assumption made in [4] and [5].

Example 3.4. *Consider the network given in Figure 3.2(a) which has asynchronous communication among its agents. Suppose that nodes 1, 2 and 3 update their state values every 5, 2 and 3 seconds, respectively. Let StrBYZ agent update its both state values (copies) every 4 seconds. As it can be seen from Figure 3.6, non-faulty nodes achieve consensus.*

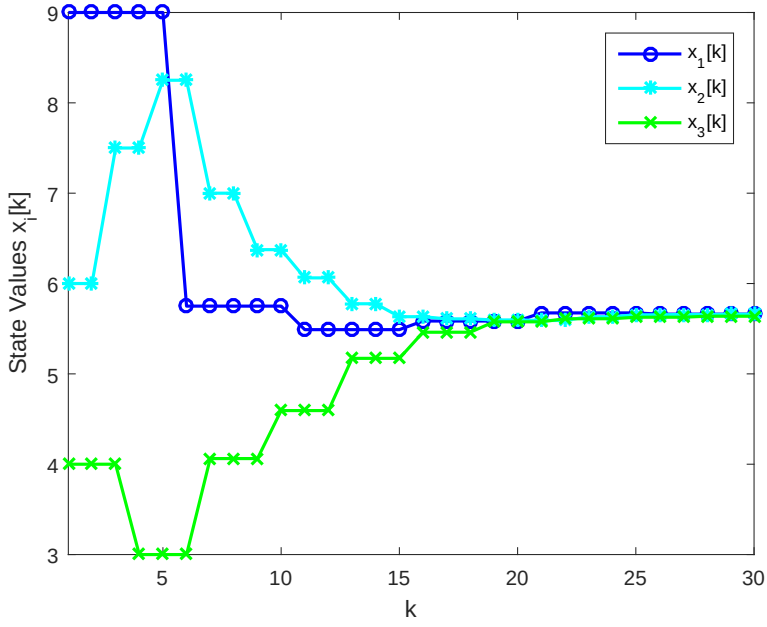


Figure 3.6. The simulation result of the asynchronous network given in Figure 3.2(b)

3.7. Summary of the Chapter

In this chapter, consensus computation has been discussed for synchronous and asynchronous networks under the StrBYZ fault model. The misbehaviour of the StrBYZ agents has been modelled using an expanded network representation. It has been shown that non-faulty nodes must achieve consensus in the expanded network in order to be able to achieve consensus in the original network.

Since the faulty behaviour has some effect on the consensus equilibrium, approximate Byzantine consensus has been considered. We have shown that in addition to the conditions required for non-faulty nodes to achieve consensus, a normal spanning component must exist in the primary or secondary layer so that approximate Byzantine consensus can be guaranteed.

Results on both linear consensus and approximate Byzantine consensus are extended for asynchronous networks. It has been discussed that the conditions for both linear consensus and approximate Byzantine consensus are also valid for asynchronous networks.

4. RESILIENT CONSENSUS UNDER UNSTRUCTURED BYZANTINE FAULT MODEL

In this chapter, we first define the unstructured Byzantine fault (uStrBYZ) and explain its misbehaviour. Then, two fault tolerant algorithms, L-MSR and R-MSR, are proposed. Convergence analysis of these algorithms is carried out. At the end, theoretical results are illustrated by numerical examples.

4.1. Adversary Behaviour and Fault Model

Consider the network $G = (V, E)$ of n agents and let $G_{nf} = (V_{nf}, E_{nf})$ denote the network of the normal agents in the network G.

Definition 4.1. (Unstructured Byzantine) *A node $i \in F$ is said to be uStrBYZ if it does not send the same information to its out-neighbors at the same time step k, i.e., $x_{ji}[k] \neq x_{ki}[k]$, $i \in N_j \cap N_k$ and it applies arbitrary function to update its state value.*

The definition of uStrBYZ fits the definition of the Byzantine fault given in [1] where the uStrBYZ agent sends different faulty information to each of its neighbors. Therefore, uStrBYZ fault model is the one that is hardest to tolerate compared to other fault models presented in Figure 1.1.

4.2. Consensus Computation in the Presence of uStrBYZ

Consider a network of n agents where f of the agents are uStrBYZ. Due to the misbehaviour of uStrBYZ agent i, we have $x_{ji}[k] \neq x_{mi}[k]$ for $i \in N_j \cap N_m$. Therefore, we can represent the update rule for non-faulty i-th node as follows:

$$x_i[k+1] = \sum_{j \in N_i \cap V_{nf}} w_{ij} x_j[k] + \sum_{j \in N_i \cap F} w_{ij} x_{ij}[k], \;\; i \in V_{nf} \tag{4.1}$$

where $x_{ij}[k]$ represents the information sent by the uStrBYZ node j to the non-faulty node i at time step k. Note that the averaging coefficients w_{ij} satisfy the same conditions with the coefficients in the update rule (2.1). Since uStrBYZ agents apply an arbitrary function to update their state values, for each $i \in N_j^-$, an uStrBYZ agent j uses the following update rule:

$$x_{ij}[k+1] = f_{ij}[k], \;\; j \in F \tag{4.2}$$

where $f_{ij}[k]$ is an arbitrary time-varying function. Then, the update rule (4.1) can be rewritten as

$$x_i[k+1] = \sum_{j \in N_i \cap V_{nf}} w_{ij} x_j[k] + v_{ij}[k], \;\; i \in V_{nf} \tag{4.3}$$

where

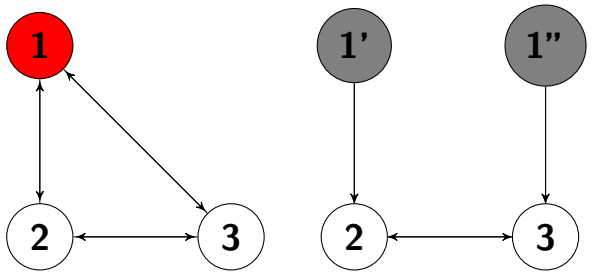
$$v_{ij}[k] = \sum_{j \in N_i \cap F} w_{ij} x_{ij}[k]$$

is a time-varying function that represents the uStrBYZ behaviour on the non-faulty agent i. In this case, non-faulty agents cannot achieve consensus because of the values generated by the function $v_{ij}[k]$. Therefore, use of a fault tolerant algorithm is inevitable to tolerate the misbehaviour of uStrBYZ agents in the network.

4.3. Expanded Network Representation for uStrBYZ Fault Model

In this section, we investigate the topological effect of uStrBYZ behaviour described in Definition 4.1. Since uStrBYZ agents do not use the information that they receive to update their state values and $x_{ji}[k] \neq x_{mi}[k]$, $i \in N_j \cap N_m$, $j, m \in V$ holds for all $i \in F$, faulty agents can be modelled by making appropriate changes on the network topology as follows:

- All existing connections between non-faulty nodes are preserved.



(a) The original network (b) The expanded network

Figure 4.1. Original and Expanded Networks where node 1 is uStrBYZ.

- Each uStrBYZ agent creates one copy of itself for the each of its out-neighbors by disconnecting themselves from the in-neighbors and these copies are disconnected if they are generated from the same faulty node.

Definition 4.2. (Expanded Network of uStrBYZ) *Consider a network with associated graph $G = (V, E)$ where E is the edge set; V_{nf} and F denote the sets of non-faulty and uStrBYZ faulty agents, respectively. Let*

- $V_i' = \{v_i^1, v_i^2, ..., v_i^{|N_i^-|}\}$, *be represent the vertex set of the copies for each* $i \in F$,
- $E_1' = \{(j, q) : (j, i) \in E, j \in N_i^-, i \in F, q \in V_i'\}$,
- $E_2' = \emptyset$,
- $V' = \bigcup_{i \in F} V_i'$,
- $E' = E_1' \cup E_2'$.

Then, the resulting "expanded" network can be represented with associated graph $\bar{G} = (\bar{V}, \bar{E})$ where $\bar{V} = V_{nf} \cup V'$ and $\bar{E} = E_{nf} \cup E'$.

Note that the neighborhood definitions for the expanded network are exactly the same as in the original network. In-neighbor, out-neighbor and inclusive neighbor of node i in the expanded network $\bar{G}$ are denoted by $\bar{N}_i^+$, $\bar{N}_i^-$ $\bar{N}_i$, respectively. The uStrBYZ behaviour is illustrated with the expanded network representation in the following example.

Example 4.1. *Consider the network* $G = (V, E)$ *with* $n = 3$ *given in Figure 4.1(a) where* $V = \{1, 2, 3\}$ *and* $E = \{(2, 1), (3, 1), (1, 2), (3, 2), (1, 3), (2, 3)\}$. *Suppose that node* 1 *is uStrBYZ. Since we have* $N_1^- = \{2, 3\}$ *where* $|N_1^-| = 2$, *there will be 2 copies of the node 1. Then, we can write* $V' = \{1', 1''\}$ *where* $\bar{N}_{1'}^+ = \bar{N}_{1''}^+ = \emptyset$. *The resulting expanded network* $\bar{G} = (\bar{V}, \bar{E})$ *is given in Figure 4.1(b). We can also write the edge set of the expanded network as* $\bar{E} = E_{nf} \cup E'$ *where* $E_{nf} = \{(3, 2), (2, 3)\}$ *and* $E' = \{(2, 1'), (3, 1'')\}$. *Since* $E' = E'_1 \cup E'_2$ *holds, we have* $E'_1 = \{(2, 1'), (3, 1'')\}$ *and* $E'_2 = \emptyset$.

Note that the uStrBYZ agent i having $|N_i^+| \geq 1$ generates $|N_i^+|$ leader node in the expanded network $\bar{G}$. In Example 4.1, although a spanning tree exists in the network given in Figure 4.1(a), the expanded network given in 4.1(b) does not have a spanning tree due to the misbehaviour of the uStrBYZ agent $i = 1$. Therefore, use of a fault tolerant algorithm is necessary when the fault model is uStrBYZ.

4.4. L-MSR Algorithm and Its Convergence Analysis

In this section, Layered MSR (L-MSR) algorithm is proposed to deal with the misbehaviour of uStrBYZ agents. The L-MSR algorithm guarantees approximate Byzantine consensus for non-faulty nodes in either synchronous or asynchronous networks with or without time delay on communication links using the notions presented in Section 3.5 and Section 3.6.

4.4.1. L-MSR Algorithm

Let the set $\{k_0^i, k_1^i, k_2^i, ..., k_r^i\}$, $i \in V$ represent the iteration instants of node i and $r \geq 0$ be the number of the iterations. Suppose that the set of uStrBYZ agents in the same layer that contains non-faulty nodes are denoted by $F_{m,s}$ where m represents the primary or secondary layer, s is the layer index and $|F_{m,s}| = f_{m,s}$.

The L-MSR algorithm:

(i) Each non-faulty $i \in V_{nf}$ sends its state value $x_i[k_r^i]$ at time instant k_r^i.

(ii) Each node $i \in V_{nf}$ saves the values that it has received in time interval Δk_r^i. If it has received more than one state value from a node, it considers the most recent value that it has received. If it has not received any value from its neighbor node in its update time interval Δk_r^i, it uses the old received values. Then, it sorts the received values in increasing order.

(iii) Each node $i \in V_{nf}$ checks the number of in-neighbors. If $|N_i^+| \geq f_{m,s} + 3$ holds, then let $\tilde{f}_i = |N_i^+| - 1$. Otherwise, let $\tilde{f}_i = f_{m,s}$.

(iv) Node i eliminates $\tilde{f}_i$ number of values in the sorted list that are greater than its current state value. If the number of greater values is less than $\tilde{f}_i$, it eliminates all of them. Similarly, it deletes the $\tilde{f}_i$ number of values in the list that are less than its state value. Let $R_i[k]$ be the set of nodes with deleted state values.

(v) Node i updates its state value as follows:

$$x_i[k_{r+1}^i] = \sum_{j \in N_i[k_r^i] \setminus R_i[k_r^i]} w_{ij}[k_r^i] x_{ij}[k_r^i] \tag{4.4}$$

The averaging coefficients $w_{ij}[k_r^i]$ are supposed to satisfy the following conditions:

- $w_{ij}[k_r^i] = 0$ whenever $j \notin N_i[k_r^i] \setminus E_i[k_r^i]$;
- $0 < \alpha \leq w_{ij}[k_r^i] \leq 1$, $\forall j \in N_i[k_r^i] \setminus E_i[k_r^i]$;
- $\sum_{j=1}^{n} w_{ij}[k_r^i] = 1$, $\forall i \in V$, $\forall j \in N_i[k_r^i] \setminus E_i[k_r^i]$.

Note that the above algorithm with $\tilde{f}_i = 0$, $i \in V_{nf}$ is equivalent to the linear consensus algorithm. It is also important to notice that the parameter $\tilde{f}_i$ can be different for each non-faulty node i.

4.4.2. Convergence Analysis of the L-MSR Algorithm

In this section, necessary and sufficient conditions for the success of the L-MSR algorithm are investigated on the original network G. Since non-faulty nodes must be either in the same primary or secondary layer in order to guarantee approximate Byzantine consensus from Theorem 3.3, we need to consider two cases:

Case 1: Non-faulty nodes are in the primary layer having $f_{1,s}$ uStrBYZ agents in the original network G.

Case 2: Non-faulty nodes are in the secondary layer having $f_{2,s}$ uStrBYZ agents in the original network G.

In Case 1, all non-faulty nodes are in the same primary layer. Since a network having a spanning tree is itself a primary layer, this is the basic extension of the case where a non-faulty agent knows the parameter $f_{1,s}$ initially for the entire network as in [1], [2], [4], [5]. However, different from [1], [2], [4], [5], there may exist a secondary layer of uStrBYZ agents in addition to the primary layer of non-faulty nodes. In this case, non-faulty nodes using the L-MSR algorithm need to know only the number of uStrBYZ agents in the primary layer to which they belong. On the other hand, non-faulty nodes are in the same secondary layer in Case 2 where at least 2 primary layers exist constituted by uStrBYZ agents. Although there may be a large number of uStrBYZ agents in the network, the L-MSR algorithm can overcome the destructive behaviour of these faulty agents.

For both Case 1 and Case 2, the network G needs to exhibit robustness against the misbehavior of uStrBYZ agents. Suppose that there exist l_1 primary and l_2 secondary layers in the network G which may be either an synchronous or a asynchronous network with or without a bounded time delay on its communication links. In the following, robustness of the network is related to its minimum in-degree.

Lemma 4.1. *[1] Given an r-robust digraph $G = (V, E)$, with $0 \leq r \leq \lceil n/2 \rceil$, the minimum in-degree of G, $\delta^{in}(G)$, is at least*

$$\delta^{in}(G) \geq \begin{cases} r, & \text{if } r > 1 \\ 2r - 2, & \text{if } r \leq 1 \end{cases} \tag{4.5}$$

Lemma 4.2. *Given a regular and r-robust digraph $G = (V, E)$, with $r > 1$, the in-degree of each node is given by $|N_i^+| = r$ for all $i \in V$.*

Proof. The proof directly follows from Lemma 4.1 and Definition 2.6. □

In Lemma 4.1, it is stated that each node has at least r number of in-neighbors in an r-robust network. Furthermore, each node in the regular and r-robust digraph satisfies the equality given in Lemma 4.2. Extending this notion to the primary or secondary layer having $f_{m,s}$ uStrBYZ agents, we have the following result for the success of the L-MSR algorithm.

Theorem 4.1. *Given a directed network G in which there are $f_{m,s}$ number of uStrBYZ agents in the primary or secondary layer that contains non-faulty nodes. Approximate Byzantine consensus is guaranteed using the L-MSR algorithm if and only if the following conditions hold:*

- *(i) Non-faulty nodes are all in the same primary or secondary layer $G_{m,s}$ in the original network G, i.e., $V_{nf} \subseteq V_{m,s}$ where $s \in \{1, 2, \cdots, l_m\}$ and $m \in \{1, 2\}$.*
- *(ii) The network of non-faulty nodes in the primary or secondary layer $G_{m,s}$ is regular and $(f_{m,s} + 1)$-robust.*

Proof. (Sufficiency) Suppose that conditions (i) and (ii) hold. From Lemma 4.2, $|N_i^+ \cap V_{nf}| = f_{m,s} + 1$ holds for all $i \in V_{nf}$ since the network of non-faulty nodes in the primary or secondary layer $G_{m,s}$ is regular and $(f_{m,s} + 1)$-robust. According to (S3) of the L-MSR algorithm, each non-faulty node checks the condition $|N_i^+| \geq f_{1,s}+3$.

Since this condition is not satisfied, each non-faulty node uses $\tilde{f}_i = f_{m,s}$ except for the roots of the secondary layers. If node i is the root of the secondary layer $G_{2,s}$, then it uses the parameter $\tilde{f}_i = |N_i^+| - 1$. Let $F_{m,s}$ denote the set of uStrBYZ agents in the primary or secondary layer $G_{m,s}$. Since we have $V_{nf} \subseteq V_{m,s}$ in the original network, $F \setminus F_{m,s} \subseteq V \setminus V_{m,s}$ holds. Furthermore, uStrBYZ agents do not use the information that they receive from their in-neighbors to update their state values as $E_2' = \emptyset$ holds. Therefore, the layer $G_{m,s}$ is no longer connected with the rest of the network G. Hence, we have a disconnected subgraph $G_{m,s}$ having $f_{m,s}$ number of uStrBYZ agents. The rest of the proof is the same as in [4] in which the number of uStrBYZ agents is known for the network.

(Necessity) Suppose that condition (i) does not hold. Then, from Theorem 3.1, non-faulty nodes cannot reach consensus.

Suppose that condition (ii) does not hold. Then, the network of non-faulty nodes in the primary or secondary layer $G_{m,s}$ is not regular and $(f_{m,s}+1)$-robust. This implies that non-faulty nodes having more than $f_{m,s}+3$ in-neighbours use the parameter $\tilde{f}_i = |N_i^+| - 1$. This contradicts the necessity proof of Theorem 1 given in [4] as the network of non-faulty nodes in $G_{m,s}$ is not $\tilde{f}_i + 1$-robust. Therefore, approximate Byzantine consensus is impossible. □

Remark 4.1. *It is important to note that $(f_{m,s}+1)$-robustness is not sufficient for the success of the L-MSR algorithm since there may be non-faulty nodes in the primary layer that satisfy $|N_i^+| \geq f_{m,s}+3$ and the network of the non-faulty nodes may still be $(f_{m,s}+1)$-robust.*

Theorem 4.1 requires similar conditions stated in [1], [2], [4] and [5] for the success of the MSR algorithm except for the fact that we do not need the exact number of the uStrBYZ agents for the success of the L-MSR algorithm.

4.5. R-MSR Algorithm And Its Convergence Analysis

In this section, the Rooted MSR (R-MSR) algorithm is presented. Using graph theoretic concepts given in Section 2.1, the necessary and sufficient conditions for the success of the R-MSR algoritm are derived.

4.5.1. The R-MSR Algorithm

In this part, the R-MSR algorithm is proposed to tolerate the misbehaviour of uStrBYZ agents in the network. It is important to note that the R-MSR algorithm does not depend on the exact number of uStrBYZ agents due to the layered structure of the network. It is also reasonable to assume that the root node of the secondary layer $G_{2,s}$ is aware of being the root of the secondary layer $G_{2,s}$. Suppose that the set of uStrBYZ agents in the same layer that contains non-faulty nodes are denoted by $F_{m,s}$ where m represents the primary or secondary layer, s is the layer index and $|F_{m,s}| = f_{m,s}$.

The R-MSR algorithm:

(i) Each non-faulty $i \in V_{nf}$ sends its state value $x_i[k_r^i]$ at time instant k_r^i.

(ii) Each non-faulty node $i \in V_{nf}$ saves the values that it has received in the time interval Δk_r^i. If it has received more than one state value from a node, it considers the most recent value that it has received. If it has not received any value from its neighbor node in its update time interval Δk_r^i, it uses the old received values. Then, it sorts the received values in the increasing order.

(iii) Each non-faulty node i uses the following parameter:

$$\tilde{f}_i = \begin{cases} |N_i^+| - 1, & \text{if } i \text{ is the root of secondary layer } G_{2,s} \\ f_{m,s}, & \text{otherwise.} \end{cases} \tag{4.6}$$

(iv) Node i eliminates $\tilde{f}_i$ number of values in the sorted list that are greater than its current state value. If the number of greater values is less than $\tilde{f}_i$, it eliminates all of them. Similarly, it deletes the $\tilde{f}_i$ number of values in the list that are less than its state value. Let $R_i[k_r^i]$ be the set of nodes with the deleted state values.

(v) Node i updates its state value as follows:

$$x_i[k_{r+1}^i] = \sum_{j \in N_i[k_r^i] \setminus R_i[k_r^i]} w_{ij}[k_r^i] x_{ij}[k_r^i] \tag{4.7}$$

The averaging coefficients $w_{ij}[k_r^i]$ are supposed to satisfy the following conditions:

- $w_{ij}[k_r^i] = 0$ whenever $j \notin N_i[k_r^i] \setminus R_i[k_r^i]$;
- $0 < \alpha \leq w_{ij}[k_r^i] \leq 1$, $\forall j \in N_i[k_r^i] \setminus R_i[k_r^i]$;
- $\sum_{j=1}^{n} w_{ij}[k_r^i] = 1$, $\forall i \in V$, $\forall j \in N_i[k_r^i] \setminus R_i[k_r^i]$.

The R-MSR algorithm captures most of the fault tolerant algorithms presented in the literature. For instance, if the network consists of only one primary layer, then there will be no root node in the network and the R-MSR algorithm will work exactly the same as in [1], [2], [4].

Recall that each secondary layer $G_{2,s}$ in the network has a root node acting like a leader. In order to solve the approximate Byzantine consensus problem for the most general form, the root node of each secondary layer $G_{2,s}$ is allowed to know that they receive faulty information from other primary or secondary layers. Therefore, they remove more information than the other non-faulty nodes.

4.5.2. Convergence Analysis of R-MSR Algorithm

In the following, necessary and sufficient conditions for the success of the R-MSR algorithm are presented.

Theorem 4.2. *Given a directed network G in which there are $f_{m,s}$ number of uStrBYZ agents in the primary or secondary layer that contains non-faulty nodes. Approximate Byzantine consensus is guaranteed using the R-MSR algorithm if and only if the following conditions hold:*

- (i) *Non-faulty nodes are all in the same primary or secondary layer $G_{m,s}$ in the original network G, i.e., $V_{nf} \subseteq V_{m,s}$ where $s \in \{1, 2, \cdots, l_m\}$ and $m \in \{1, 2\}$.*
- (ii) *The network of non-faulty nodes in the primary or secondary layer $G_{m,s}$ is at least $(f_{m,s}+1)$-robust.*

Proof. (Sufficiency) Suppose that conditions (i) and (ii) hold. Then, the network of non-faulty nodes in the primary or secondary layer $G_{m,s}$ is at least $(f_{m,s}+1)$-robust. According to (S3) of the R-MSR algorithm, all non-faulty nodes use $\tilde{f}_i = f_{m,s}$ except for the roots of secondary layers. Root nodes of all secondary layers use $\tilde{f}_i = |N_i^+| - 1$. Let $F_{m,s}$ denote the set of uStrBYZ agents in the primary or secondary layer $G_{m,s}$. Since we have $V_{nf} \subseteq V_{m,s}$ in the original network, $F \setminus F_{m,s} \subseteq V \setminus V_{m,s}$ holds. Furthermore, uStrBYZ agents do not use the information that they receive from their in-neighbors to update their state values as we have $E_2' = \emptyset$. Thus, the layer $G_{m,s}$ is no longer connected with the rest of the network G and we have a disconnected subgraph $G_{m,s}$ having $f_{m,s}$ number of uStrBYZ agents. The rest of the proof is the same as in [4] in which the number of uStrBYZ agents is known for the network.

(Necessity) Suppose that condition (i) does not hold. Then, from Theorem 3.1, non-faulty nodes cannot reach consensus.

Suppose that condition (ii) does not hold. Then, the network of non-faulty nodes in the primary or secondary layer $G_{m,s}$ is not $(f_{m,s}+1)$-robust which contradicts the necessity proof of Theorem 1 given in [4]. Therefore, approximate Byzantine consensus is impossible.

□

Remark 4.2. *In the most of the studies [1], [2], [4], [5] in the literature, the following assumptions have been made:*

(i) The given network has a spanning tree.

(ii) The condition $n > 3f$, where n is number of nodes in the network, must be satisfied.

However, it is clear that approximate Byzantine consensus is guaranteed for non-faulty nodes using either the proposed L-MSR or the R-MSR algorithm even if the above condition (i) or (ii) does not hold. Let $n_{m,s}$ be the number of nodes in the s-th primary or secondary layer $G_{m,s}$. Due to the layered structure of the network, $n_{m,s} > 3f_{m,s}$ is enough for the success of both the L-MSR and R-MSR algorithms.

4.6. Comparison Of Resilient Distributed Algorithms

In this section, the proposed R-MSR and L-MSR algorithms are compared not only with each other but also with algorithms presented in the literature.

4.6.1. Comparison of L-MSR and R-MSR Algorithms

Two resilient distributed algorithms are proposed so far: R-MSR and L-MSR. Although these two algorithms look like each other, there are several differences which can be stated as follows:

(i) In the R-MSR algorithm, root node i of the secondary layer $G_{2,s}$ acts like a leader and uses the parameter $\tilde{f}_i = |N_i^+| - 1$. However, there is no leader behaviour in the L-MSR algorithm. Due to the regular and $(f_{m,s}+1)$-robustness, only the root nodes of the network use the parameter $\tilde{f}_i = |N_i^+| - 1$. For instance, R-MSR algorithm is more suitable for applications having a leader-follower mechanism. On the other hand, in networks where a leader-follower mechanism does not exist, the L-MSR algorithm can be used to guarantee approximate Byzantine consensus.

(ii) In the L-MSR algorithm, the only information that non-faulty nodes have is the number of uStrBYZ agents ($f_{m,s}$) in the primary or secondary layer that contains the non-faulty nodes. That is why more restrictive network conditions are required for the success of the L-MSR algorithm. However, conditions for the success of the R-MSR algorithm are less restrictive since a regular and $(f_{m,s}+1)$-robust graph is also $(f_{m,s}+1)$-robust but not vice versa.

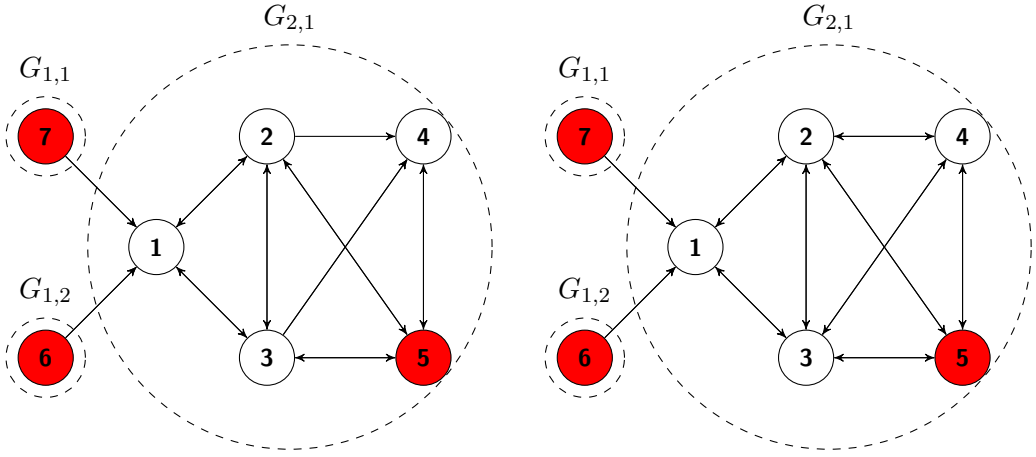
There are also common properties of L-MSR and R-MSR algorithms which can be stated as follows:

(i) For Case 1, in which all non-faulty nodes are in the same primary layer, all non-faulty nodes use the parameter $\tilde{f}_i = f_{m,s}$ in both algorithms.
(ii) Both algorithms can solve the approximate Byzantine consensus problem for synchronous and asynchronous networks with or without bounded time delays.
(iii) Both algorithms can solve the approximate Byzantine group consensus problem to be discussed in Chapter 5 for synchronous and asynchronous networks.

4.6.2. Comparison of the Proposed Algorithms with Resilient Algorithms in the Literature

As the proposed L-MSR and R-MSR algorithms consider a layer by layer network structure instead of the whole network, there exist major differences between the proposed algorithms and those given in the literature. These differences can be stated as follows:

(i) The L-MSR and R-MSR algorithms do not require the overall network to have a spanning tree.
(ii) The condition $n > 3f$, where f is the total number of uStrBYZ agents and n is the total number of agents in the network, is not required for the success of the L-MSR and R-MSR algorithms. An uStrBYZ agent far away from the network of non-faulty nodes cannot effect the overall system behaviour. The only requirement is $|V_{m,s}| > 3f_{m,s}$ for both L-MSR and R-MSR algorithms.



(a) A network with 7 nodes where node 5, node 6 and node 7 are uStrBYZ and the network non-faulty nodes in $G_{2,1}$ is regular and 2-robust.

(b) A network with 7 nodes where node 5, node 6 and node 7 are uStrBYZ and the network non-faulty nodes in $G_{2,1}$ is 2-robust.

Figure 4.2. Robust networks for numerical analysis.

(iii) The L-MSR and R-MSR algorithms do not necessarily delete $2\tilde{f}_i$ number of received state values at each iteration.

(iv) None of the non-faulty nodes waits to receive a certain number of state values in order to update its state value.

4.7. Numerical Anaylsis

In this section, we present some examples to illustrate our theoretical results.

Example 4.2. *Consider the network given in Figure 4.2(a) where nodes 5,6 and 7 are uStrBYZ. The faulty nodes 6 and 7 can be represented as two primary layers, i.e.,* $\{6\} \in V_{1,1}$ *and* $\{7\} \in V_{1,2}$ *and the remaining nodes are in the secondary layer* $G_{2,1}$*, i.e.,* $\{1,2,3,4,5\} \in V_{2,1}$*. Since* $f_{2,1} = 1$ *and* $|N_i^+| \geq f_{2,1} + 3$ *holds only for* $i = 1$*, only the root node* 1 *uses the parameter* $\tilde{f}_1 = |N_i^+| - 1 = 3$*. Other non-faulty nodes uses the parameter* $\tilde{f}_i = f_{2,1} = 1$*, where* $i \in \{2,3,4\}$*. Since all non-faulty nodes are in the secondary layer* $G_{2,1}$ *and the network of non-faulty nodes is regular and 2-robust, it can be concluded from Theorem 4.1 that approximate Byzantine consensus is guaranteed using the L-MSR algorithm as it can be seen from Figure 4.3.*

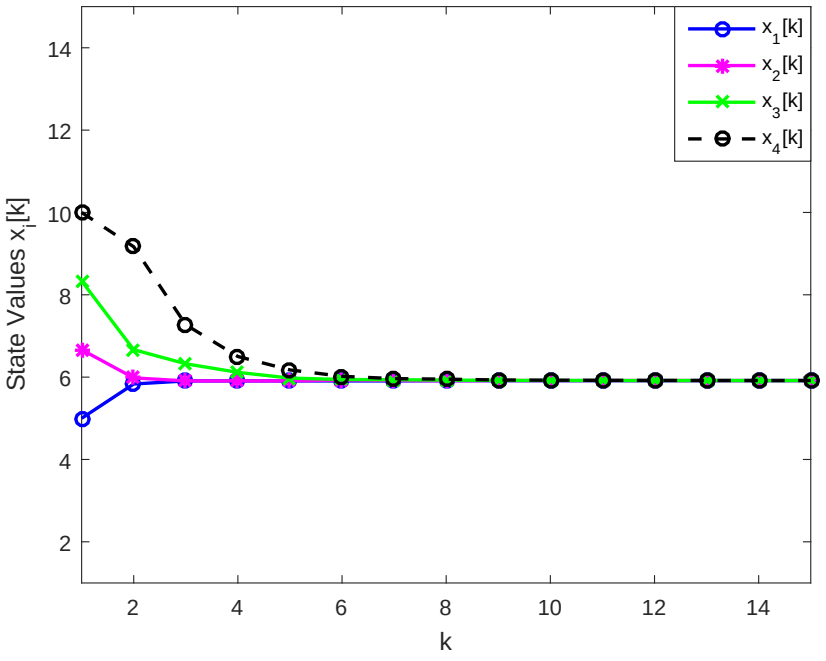


Figure 4.3. The simulation result of the network given in Figure 4.2(a) when non-faulty nodes use the L-MSR algorithm.

Example 4.3. *Consider the network given in Figure 4.2(b) where nodes 5,6 and 7 are uStrBYZ. Suppose that node* 1 *is aware of being the root node of the secondary layer* $G_{2,1}$ *which contains all non-faulty nodes. Then, root node* 1 *uses the parameter* $\tilde{f}_1 = |N_i^+| - 1 = 3$ *and the other non-faulty nodes use* $\tilde{f}_i = f_{2,1} = 1$*, where* $i \in \{2, 3, 4\}$*. Since non-faulty nodes are all in the same secondary layer* $G_{2,1}$ *and the network of non-faulty nodes is 2-robust, it can be concluded from Theorem 4.2 that approximate Byzantine consensus is guaranteed using the R-MSR algorithm as it can be seen from Figure 4.4.*

In Example 4.3, the network given in Figure 4.2(b) does not have a spanning tree. Furthermore, the condition $n > 3f$ is not satisfied as we have $n = 7$ and $f = 3$. However, the second layer that contains all non-faulty nodes has a spanning tree. Moreover, the condition $|V_{m,s}| > 3f_{m,s}$, where $|V_{2,1}| = 4$, is satisfied for the secondary layer $G_{2,1}$.

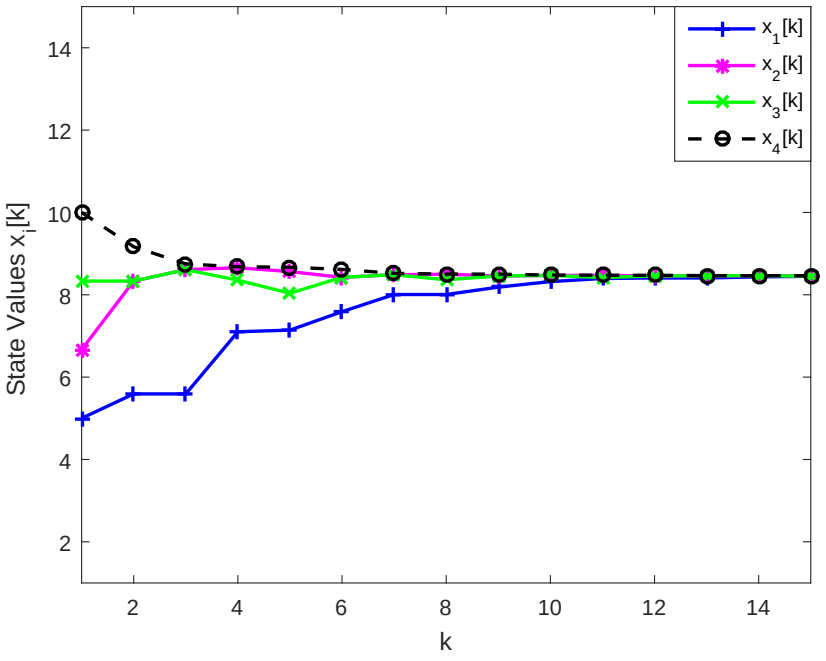


Figure 4.4. The simulation result of the network given in Figure 4.2(b) when non-faulty nodes use the R-MSR algorithm.

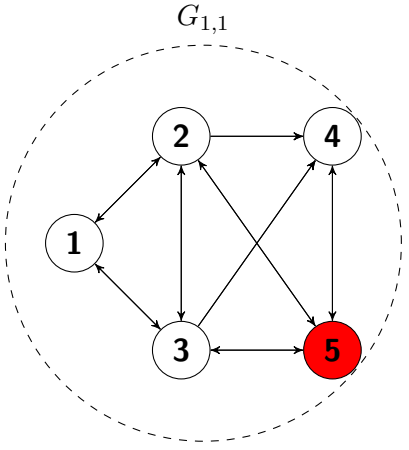


Figure 4.5. A network with 5 nodes where node 5 is uStrBYZ and the network of non-faulty nodes in $G_{2,1}$ is 2-robust.

Example 4.4. *Consider the network given in Figure 4.5 where it is itself a primary layer. Let node 5 be uStrBYZ. Each non-faulty node uses the parameter* $\tilde{f}_i = f_{1,1} = 1$, *where* $i \in \{1, 2, 3, 4\}$, *as there is no root node in* $G = G_{1,1}$. *Since the network of non-faulty nodes is 2-robust, it can be concluded from Theorem 4.2 that approximate Byzantine consensus is guaranteed using the R-MSR algorithm as it can be seen from Figure 4.6.*

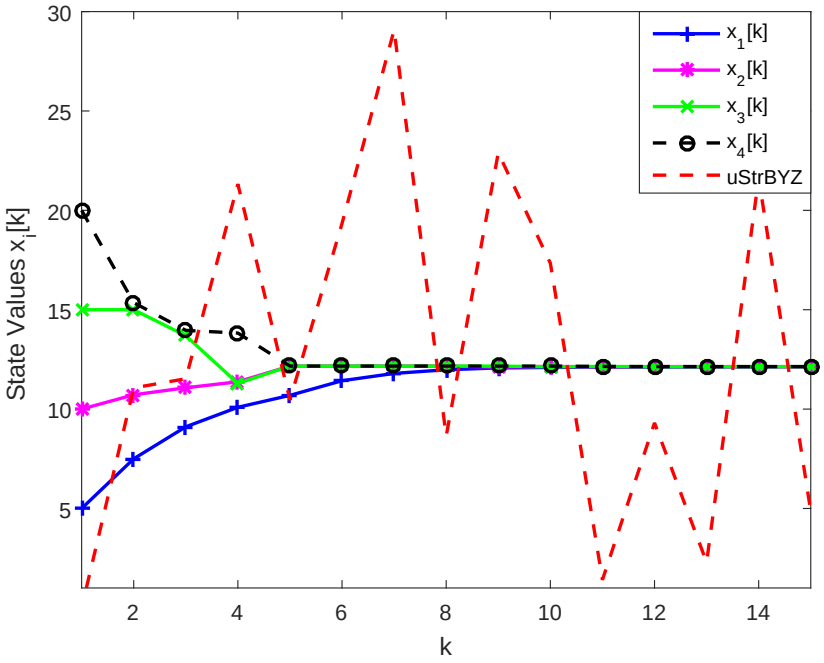


Figure 4.6. The simulation result of the network given in Figure 4.5 when non-faulty nodes use R-MSR algorithm.

When the given network G is assumed to have a spanning tree, i.e., the network G is itself a primary layer, Example 4.4 shows that the R-MSR algorithm is exactly the same as the MSR algorithm given in [4]. In that sense, the R-MSR algorithm can be viewed as generalized form of the family of MSR algorithms.

4.8. Summary of the Chapter

In this chapter, we have first defined the uStrBYZ fault and explained its misbehaviour. It has been shown that the network having an uStrBYZ agent does not have a spanning tree in the expanded network due to the faulty behaviour of the uStrBYZ agent. Therefore, it can be concluded that approximate Byzantine consensus cannot be guaranteed without implementing a fault tolerant algorithm.

To this end, the L-MSR and R-MSR algorithms have been proposed. It is shown that classical conditions presented in the literature do not have to be satisfied if non-faulty nodes use either of these fault tolerant algorithms. Then, convergence analysis

of these algorithms has been discussed. Finally, simulation studies are presented to illustrate our results.

In the next chapter, we introduce the approximate Byzantine group consensus problem with which the group consensus problem is intended to be discussed under the StrBYZ and uStrBYZ fault models.

5. FAULT TOLERANT GROUP CONSENSUS

In this chapter, fault tolerant group consensus computation is investigated. First, group consensus is discussed when the thread model is StrBYZ. Then, approximate Byzantine group consensus problem is introduced for both StrBYZ and uStrBYZ fault models. It is shown that fault tolerant L-MSR and R-MSR algorithms discussed in Chapter 4 can be used to guarantee approximate Byzantine group consensus. Theoretical results are illustrated by simulation studies.

5.1. Group Consensus Preliminaries

In group consensus, the network reaches $K \geq 2$ equilibrium points depending on the network structure. In [22] and [23], topological conditions are presented for K group of the nodes to reach K different equilibrium states.

Lemma 5.1. *[22, 23] For a directed network of n nodes having l_1 primary layers and l_2 secondary layers, the number of equilibrium states when the agents of the network use the update rule (2.1) is computed as*

$$K = l_1 + l_2 \tag{5.1}$$

In other words, Lemma 5.1 states that agents in the same primary or secondary layer reach the same equilibrium state. Therefore, the number of primary and secondary layers in the network determines the number of equilibrium states.

5.2. Linear Group Consensus Under StrBYZ Fault Model

In [22] and [23], results are obtained for the case in which the given network does not contain a faulty behaviour. In this section, group consensus problem is investigated in the presence of StrBYZ agents.

Lemma 5.2. *A group of non-faulty nodes of the network G reaches on the same equilibrium state using the update rule (2.1) in the presence of StrBYZ agents if and only if they are in the same primary or secondary layer in the expanded network $\bar{G}$.*

Proof. Proof directly follows from Lemma 5.1 and Theorem 3.1. □

Since the expanded network is generated due to the faulty behaviour of StrBYZ agents, Lemma 5.2 is an extension of Lemma 5.1 for the case in which non-faulty nodes in the same primary or secondary layer reach the same equilibrium point in the expanded network $\bar{G}$ in the presence of StrBYZ agents. Let $G^{nf}_{m,s} = (V^{nf}_{m,s}, E^{nf}_{m,s})$ and $\bar{G}^{nf}_{m,s} = (\bar{V}^{nf}_{m,s}, \bar{E}^{nf}_{m,s})$ denote the network of non-faulty nodes in the s-th primary or secondary layer in the original network G and the expanded network $\bar{G}$, respectively. Using Lemmas 5.1 and 5.2, we present the following result on the expanded network $\bar{G}$ for non-faulty nodes to reach K number of equilibria.

Lemma 5.3. *Consider a network of l_1 primary layers and l_2 secondary layers where non-faulty nodes use the update rule (2.1) in the presence of StrBYZ agents. Then, the total number of equilibrium states for the non-faulty nodes is given by*

$$K = l_1 + l_2 \tag{5.2}$$

if and only if non-faulty nodes in the same primary and secondary layers of the original network G remain in the same primary and secondary layers in the expanded network $\bar{G}$, i.e., $V^{nf}_{m,s} = \bar{V}^{nf}_{m,s}$.

Proof. (Sufficiency) Suppose that $V^{nf}_{m,s} = \bar{V}^{nf}_{m,s}$ holds. Then, there will be l_1 primary layers and l_2 secondary layers in the expanded network $\bar{G}$. From Lemma 5.1, the number of equilibrium states for non-faulty nodes using the update rule (2.1) in the expanded network $\bar{G}$ is computed as $K = l_1 + l_2$.

(Necessity) Suppose that $V_{m,s}^{nf} \neq \bar{V}_{m,s}^{nf}$ due to the misbehaviour of StrBYZ agents. Then, non-faulty nodes in the same primary or secondary layer of the original network G are not in the same primary or secondary layer of the expanded network $\bar{G}$. From Lemmas 3.4 and 3.5, a primary or secondary layer is preserved in the expanded network if and only if $|N_i^+| \geq 1$ for each $i \in G_{m,s} \cap F$. Otherwise, each StrBYZ agent creates 2 copies, each of which is itself a primary layer. Therefore, non-faulty nodes cannot reach $K = l_1 + l_2$ number of equilibrium states even if we have $K = l_1 + l_2$ number of primary and secondary layers in the expanded network $\bar{G}$. □

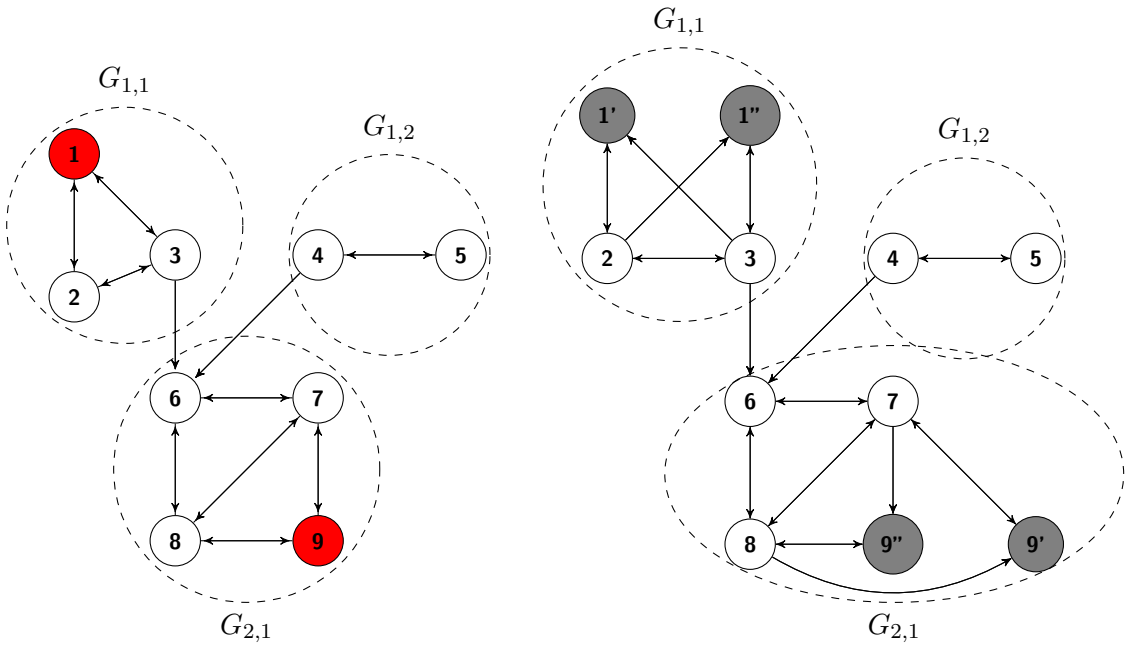
Note that non-faulty nodes must reach K number of equilibrium points in the expanded network $\bar{G}$ in order to reach K number of equilibrium points in the original network G.

Theorem 5.1. *Non-faulty agents of the network having l_1 primary layers and l_2 secondary layers reach $K = l_1 + l_2$ equilibrium states using the update rule (2.1) in the presence of StrBYZ agents if and only if the following conditions are satisfied:*

- (i) *Each StrBYZ agent has at least one in-neighbor from the primary or secondary layer that they belong, i.e., for each $i \in V_{m,s} \cap F$, we have $|N_i^+| \geq 1$ where $N_i^+ \subseteq V_{m,s}$, $s \in \{1, 2, \cdots, l_m\}$ and $m \in \{1, 2\}$.*
- (ii) *Root nodes of all secondary layers in the network G must be non-faulty.*

Proof. (Sufficiency) Suppose that conditions (i) and (ii) hold. From Lemmas 3.3, 3.4 and 3.5, the spanning tree existing in $G_{1,s}$ or in $G_{2,s}$, where $s \in \{1, 2, \cdots, l_m\}$ and $m \in \{1, 2\}$, is preserved in the expanded network $\bar{G}$ and we have $V_{m,s}^{nf} = \bar{V}_{m,s}^{nf}$. From Lemma 5.3, it can be concluded that non-faulty nodes reach $K = l_1 + l_2$ equilibrium states using the update rule (2.1) in the presence of StrBYZ agents.

(Necessity) Suppose that condition (i) does not hold. Then, from Lemmas 3.4 and 3.5, we have $V_{m,s}^{nf} \neq \bar{V}_{m,s}^{nf}$ which implies $K \neq l_1 + l_2$ according to Lemma 5.3.



(a) The original network (b) The expanded network

Figure 5.1. Original and Expanded Networks where node 1 and node 9 are StrBYZ.

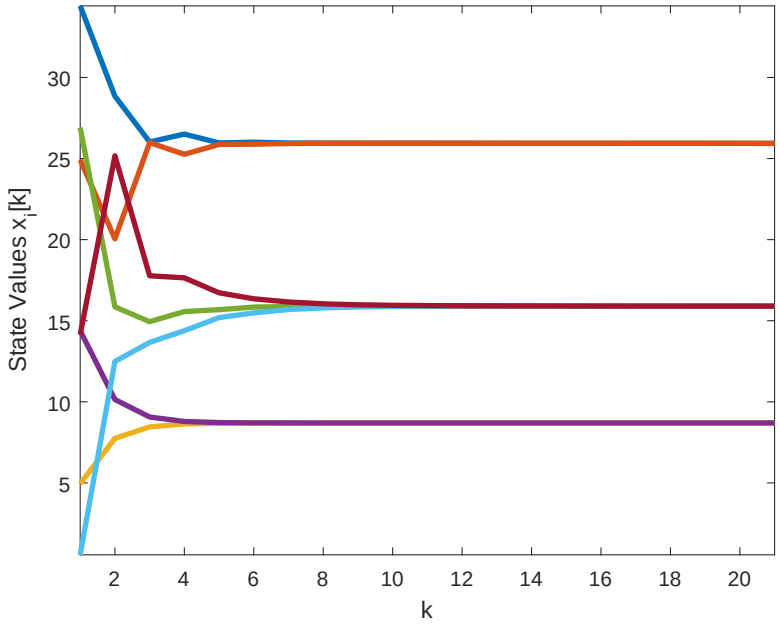


Figure 5.2. The simulation result of the network given in Figure 5.1(a).

Suppose that condition (ii) does not hold. This contradicts Lemma 3.5 since the secondary layer $G_{2,s}$ of the original network G is not preserved in the expanded layer. □

Example 5.1. *Consider the network given in Figure 5.1(a). There are* $l_1 = 2$ *primary layers* $G_{1,1}$, $G_{1,2}$ *and* $l_2 = 1$ *secondary layer* $G_{2,1}$ *in the original network* G. *Let node* 1 *and node* 9 *be StrBYZ agents having* $|N_1^+| = |N_9^+| = 2$. *Since the root node of the secondary layer* $G_{2,1}$ *is non-faulty (node* 6*), it can be concluded from Theorem 5.1 that non-faulty nodes reach* $K = 2 + 1 = 3$ *equilibrium states as it can also be seen from Figure 5.2.*

So far, necessary and sufficient conditions are presented for non-faulty nodes to reach $K = l_1 + l_2$ different equilibrium points when the network has StrBYZ agents. In the following section, we discuss the effect of StrBYZ agents on these equilibrium points and how we can overcome this faulty effect.

5.3. Approximate Byzantine Group Consensus Under StrBYZ Fault Model

In this section, we introduce the approximate Byzantine group consensus problem and present the conditions in order to satisfy not only the agreement condition but also the validity condition. Note that the threat model, the misbehaviour of StrBYZ agents and the expanded network representation are exactly the same as discussed in Section 3.1 and Section 3.3. Hence, the results presented in Subsection 3.5.2 are also valid and extended for the approximate Byzantine group consensus which is defined as follows:

Definition 5.1. (Approximate Byzantine Group Consensus) *A network of non-faulty agents is said to guarantee approximate Byzantine group consensus if each agent starts from an arbitrary initial value and all non-faulty agents achieve group consensus with the following conditions:*

- *Agreement: For any pre-assigned* $\epsilon > 0$*, all non-faulty nodes in the same primary or secondary layer reach on state values that are within* ϵ *of each other.*
- *Number of Equilibrium States: Total number of equilibrium states must be* $K = l_1 + l_2$ *where* l_1 *is the number of primary layers and* l_2 *is the number of secondary layers.*

- *Validity: The state value of each non-faulty node must be in the range of initial values of non-faulty nodes.*

The group consensus problem considered in [22] and [23] is investigated for only the agreement on K different equilibrium points. Actually, it makes sense when there is no faulty agent in the network. However, if faulty behaviour exists in the network, then the validity condition becomes really important in designing fault-tolerant algorithms. It is also important to notice that faulty behaviour can change the network topology. Since (5.1) depends on the network structure, the number of equilibrium states for the network may change due to the faulty behaviour.

For a given directed network G, let $M[k]$ and $m[k]$ denote the maximum and minimum state values of the set of non-faulty nodes at time step k. Furthermore, let $M_{m,s}[k]$ and $m_{m,s}[k]$ respectively denote the maximum and minimum state values of the set of non-faulty nodes in the s-th primary or secondary layer $G_{m,s}$ at time step k. The maximum difference between the state values of non-faulty nodes in $G_{m,s}$ is defined as:

$$e_{m,s}[k] = M_{m,s}[k] - m_{m,s}[k] \tag{5.3}$$

where $e_{m,s}[k] \geq 0$ for all $k \geq 0$ and $e_{m,s}[k] = 0$ if and only if $M_{m,s}[k] = m_{m,s}[k]$. For each $G_{m,s}$, where $s \in \{1, 2, \cdots l_m\}$ and $m \in \{1, 2\}$, approximate Byzantine group consensus is guaranteed for non-faulty nodes if the following conditions hold:

- Agreement: There exists a $k_0 \geq 0$ so that we have $e_{m,s}[k] \leq \epsilon$ for all $s \in \{1, 2, \cdots l_m\}$, $m \in \{1, 2\}$ and $k \geq k_0$.
- Number of Equilibrium States: Total number of equilibrium states must be $K = l_1 + l_2$ where l_1 is the number of primary layers and l_2 is the number of secondary layers.
- Validity: For all $k \geq 0$, $m(0) \leq m_{m,s}[k] \leq M_{m,s}[k] \leq M(0)$.

Given a digraph G with l_1 number of primary and l_2 number of secondary layers, let $G^{sp}_{m,s} = (V^{sp}_{m,s}, E^{sp}_{m,s})$ denote the subgraph of nodes in a normal spanning component of the s-th primary or secondary layer $G_{m,s}$.

Theorem 5.2. *For a given directed network G having l_1 primary layers and l_2 secondary layers, approximate Byzantine group consensus is guaranteed for non-faulty nodes if and only if the following conditions hold:*

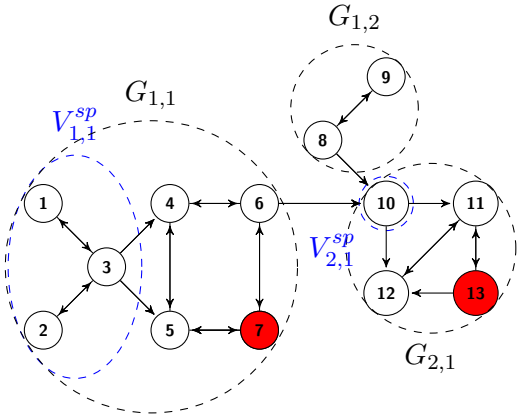
(*i*) *Each StrBYZ agent has at least one in-neighbor from the primary or secondary layer that they belong, i.e., for each $i \in V_{m,s} \cap F$, we have $|N_i^+| \geq 1$ where $s \in \{1, 2, \cdots l_m\}$, $m \in \{1, 2\}$ and $N_i^+ \subseteq V_{m,s}$.*

(*ii*) *Root nodes of all secondary layers in the network G must be non-faulty.*

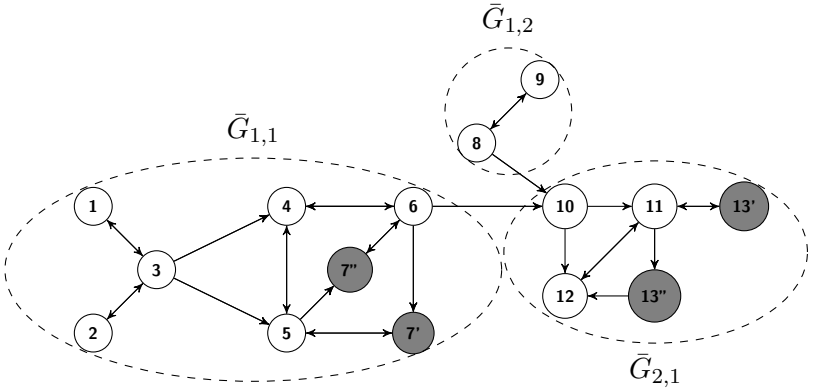
(*iii*) *Each primary and secondary layer in the network G has at least one normal spanning component, i.e., $V^{sp}_{m,s} \subseteq V_{m,s}$ holds for each $m \in \{1, 2\}$ and $s \in \{1, 2, \cdots l_m\}$.*

Proof. The proof directly follows from Theorems 3.3 and 5.1. □

Example 5.2. *Consider the network with* 13 *nodes given in Figure 5.3(a) in which node* 7 *and node* 13 *are StrBYZ. There are two primary layers $G_{1,1}$, $G_{1,2}$ and one secondary layer $G_{2,1}$, where non-faulty node* 3 *is the root node. Furthermore, there are two normal spanning components $V^{sp}_{1,1} = \{1, 2, 3\}$ and $V^{sp}_{2,1} = \{10\}$ in the primary layer $G_{2,1}$ and the secondary layer $G_{2,1}$, respectively. Initial state values for eleven non-faulty nodes are randomly chosen as $x[0] = [45, 50, 48, 32, 37, 35, 25, 30, 27, 18, 15]^T$. As it can be seen from Figure 5.4, equilibrium points of the non-faulty nodes in $G_{1,1}$, $G_{1,2}$ and $G_{2,1}$ are $x^*_{1,1} = 47, 6$, $x^*_{1,2} = 27, 7$, $x^*_{1,1} = 36, 91$, respectively. Therefore, we can say that approximate Byzantine group consensus is guaranteed with $K = 3$ different equilibrium states.*



(a) The original network



(b) The expanded network

Figure 5.3. Original and Expanded Networks where node 7 and node 13 are StrBYZ.

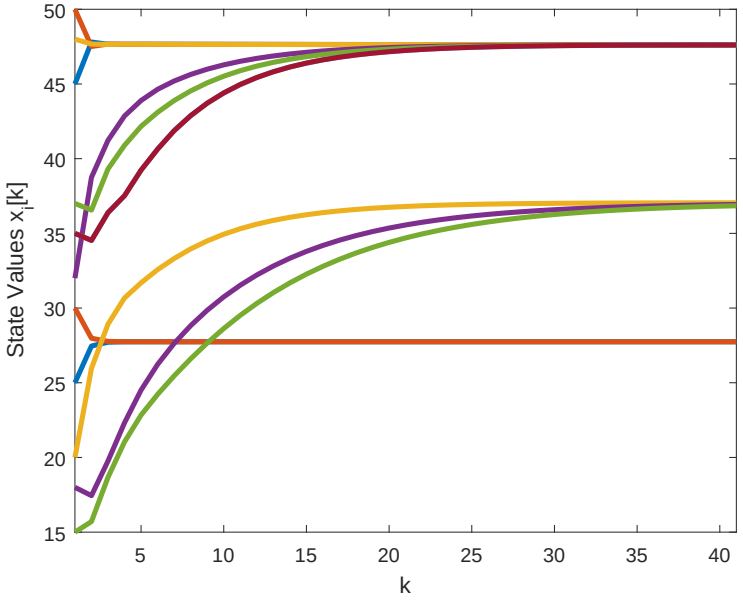


Figure 5.4. The simulation result of the networks given in Figure 5.3.

5.4. Approximate Byzantine Group Consensus under uStrBYZ Fault Model

In this part, we investigate the approximate Byzantine group consensus problem when the threat model is uStrBYZ. Note that the threat model, the misbehaviour of the uStrBYZ agent and the expanded network representation are exactly the same as discussed in Sections 4.1 and 4.2. Due to the layered structure of the network, both R-MSR and L-MSR algorithms can be used to guarantee approximate Byzantine group consensus.

5.4.1. Approximate Byzantine Group Consensus using the L-MSR Algorithm

Recall that the L-MSR algorithm has been used to solve the approximate Byzantine consensus problem in Section 4.4. For approximate Byzantine group consensus, similar results can be obtained as follows:

Theorem 5.3. *Let $f_{m,s}$ denote the number of uStrBYZ agents in the s-th primary or secondary layer $G_{m,s}$ in the network. Approximate Byzantine group consensus is guaranteed for non-faulty nodes of the network G having l_1 primary layers and l_2 secondary layers using the L-MSR algorithm if and only if each of the subgraphs of non-faulty nodes in the primary and secondary layers ($G^{nf}_{m,s}$) is regular and $(f_{m,s}+1)$-robust.*

Proof. (Sufficiency) Suppose that the subgraphs of non-faulty nodes in the primary and secondary layers $G_{m,s}$ are regular and $(f_{m,s}+1)$-robust. According to the L-MSR algorithm, each root node i in the secondary layer $G_{2,s}$ of the network G removes information coming from other layers. Therefore, each of the primary and secondary layers can be considered as individual subgraphs having $f_{m,s}$ uStrBYZ agents. Since each of these layers is regular and $(f_{m,s}+1)$-robust, agreement and validity conditions hold for each of the individual primary and secondary layers from Theorem 4.1. Since there are l_1 disconnected primary layers and l_2 disconnected secondary layers in the network, non-faulty nodes reach $K = l_1 + l_2$ equilibrium states by Lemma 5.1.

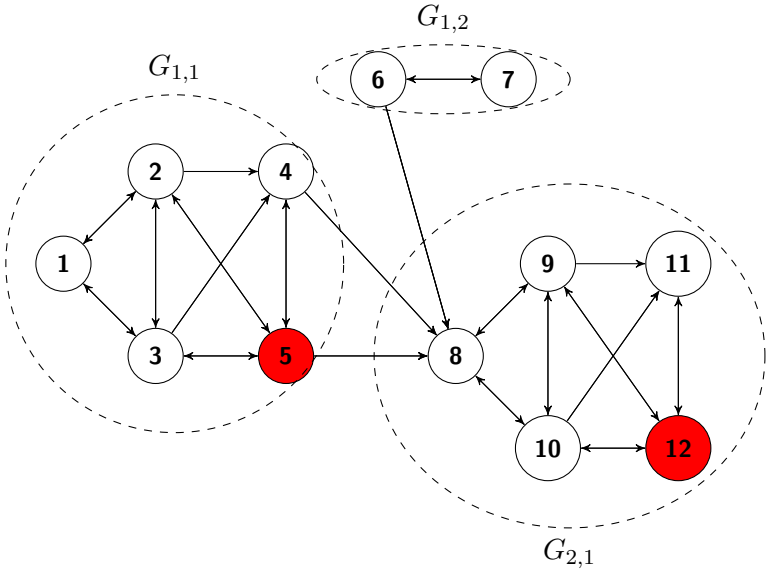


Figure 5.5. A network with 12 nodes where node 5 and node 12 are uStrBYZ.

Hence, approximate Byzantine group consensus is guaranteed for non-faulty nodes of the network G.

(Necessity) Suppose that one of the subgraphs of non-faulty nodes in the primary and secondary layers $G_{m,s}$ is not regular and $(f_{m,s}+1)$-robust. From Theorem 4.1, it is immediately inferred that the agreement condition does not hold. Therefore, the approximate Byzantine group consensus is impossible. □

Example 5.3. *Consider the network given in Figure 5.5 where nodes* 5 *and* 12 *are uStrBYZ. There are two primary layers* $G_{1,1}$ *(*$V_{1,1} = \{1\}$*) and* $G_{1,2}$ *(*$V_{1,2} = \{2\}$*) and one secondary layer* $G_{2,1}$ *(*$V_{2,1} = \{3,4,5,6\}$*), where node* 3 *is the root node, in the network. Since the network of non-faulty agents in the primary layer* $G_{1,1}$ *and the secondary layer* $G_{2,1}$ *are regular and 2-robust, approximate Byzantine group consensus is guaranteed with* $K = 2+1 = 3$ *equilibrium states using L-MSR algorithm as it can be seen from Figure 5.6.*

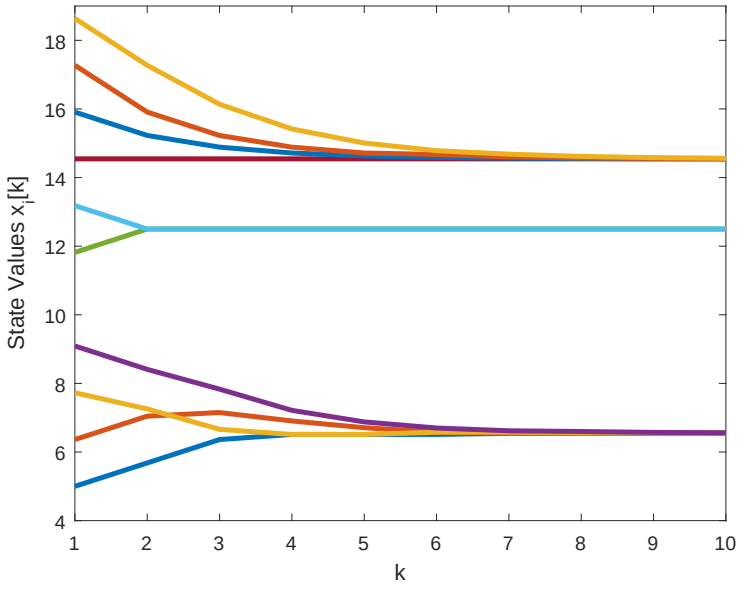


Figure 5.6. The simulation result of the network given in Figure 5.5.

5.4.2. Approximate Byzantine Group Consensus using the R-MSR Algorithm

Similar to the L-MSR algorithm, the R-MSR algorithm can also be used by non-faulty nodes to guarantee approximate Byzantine group consensus.

Theorem 5.4. *Let $f_{m,s}$ denote the number of uStrBYZ agents in the s-th primary or secondary layer $G_{m,s}$ in the network. Approximate Byzantine group consensus is guaranteed for non-faulty nodes of the network G having l_1 primary layers and l_2 secondary layers using the R-MSR algorithm if and only if each of the subgraphs of non-faulty nodes in the primary and secondary layers ($G^{nf}_{m,s}$) is at least $(f_{m,s}+1)$-robust.*

Proof. (Sufficiency) Suppose that the subgraphs of non-faulty nodes in the primary and secondary layers $G_{m,s}$ are $(f_{m,s}+1)$-robust. According to the R-MSR algorithm, each root node i in the secondary layer $G_{2,s}$ of the network G removes information coming from other layers. Therefore, each of the primary and secondary layers can be considered as individual subgraphs having $f_{m,s}$ uStrBYZ agents. Since each of these layers is $(f_{m,s}+1)$-robust, agreement and validity conditions hold for each of the individual primary and secondary layers from Theorem 4.2. Since there are

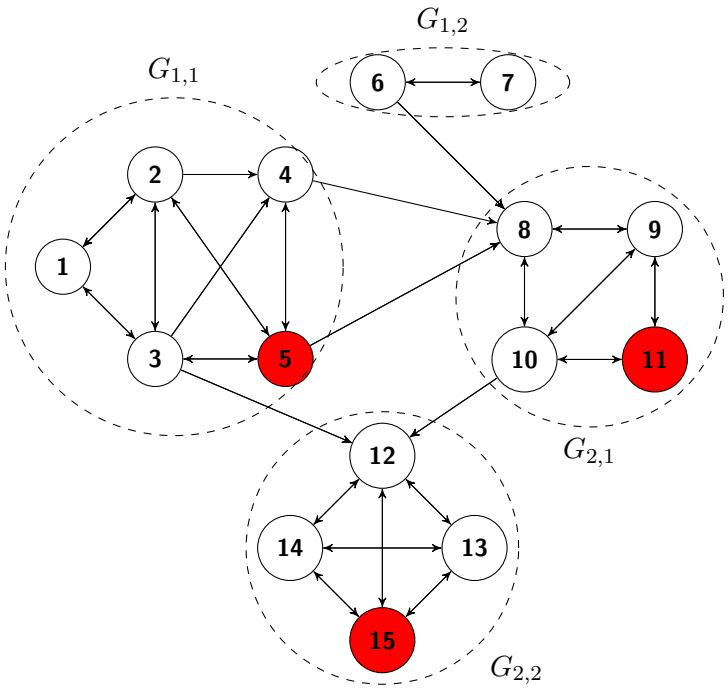


Figure 5.7. A network with 15 nodes where nodes 5, 11 and 15 are uStrBYZ.

l_1 disconnected primary layers and l_2 disconnected secondary layers in the network, non-faulty nodes reach $K = l_1 + l_2$ equilibrium states by Lemma 5.1. Hence, approximate Byzantine group consensus is guaranteed for non-faulty nodes of the network G.

(Necessity) Suppose that one of the subgraphs of non-faulty nodes in the primary and secondary layers $G_{m,s}$ is not $(f_{m,s}+1)$-robust. From Theorem 4.2, it is immediately inferred that the agreement condition does not hold. Therefore, the approximate Byzantine group consensus is impossible. □

Example 5.4. *Consider the network with 15 nodes given in Figure 5.7. Let nodes* 5, 11 *and* 15 *be uStrBYZ. There are two primary layers* $G_{1,1}$, $G_{1,2}$ *and two secondary layers* $G_{2,1}$, $G_{2,2}$ *in which non-faulty nodes* 8 *and* 12 *are the root nodes. Since the network of non-faulty agents in the primary layer* $G_{1,1}$ *is regular and 2-robust, which encloses 2-robustness, and the network of non-faulty agents in the secondary layers* $G_{2,1}$ *and* $G_{2,2}$ *are 2-robust, approximate Byzantine group consensus is guaranteed with* $K = 2 + 2 = 4$ *equilibra using the R-MSR algorithm as it can be seen from Figure 5.8.*

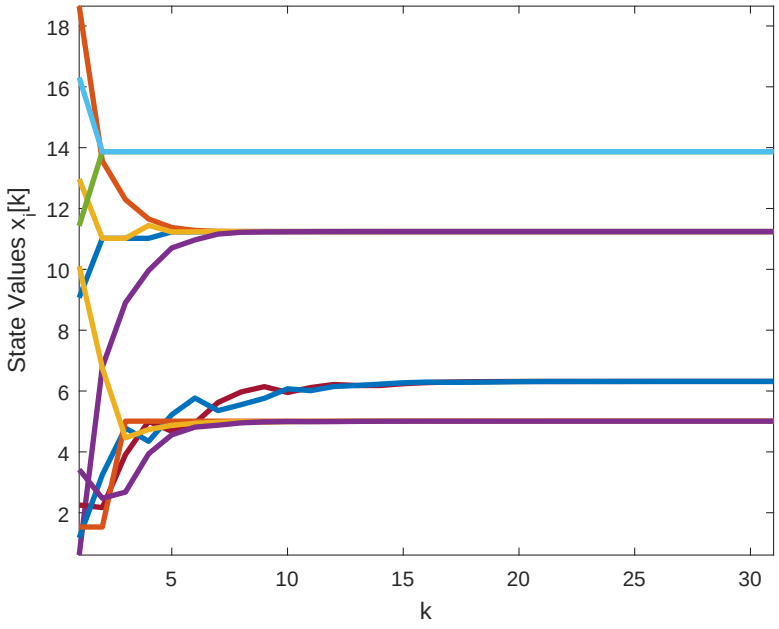


Figure 5.8. The simulation result of the network given in Figure 5.7.

5.5. Summary of the Chapter

In this chapter, group consensus problem has been investigated under the StrBYZ and uStrBYZ fault models. First, necessary and sufficient conditions for non-faulty nodes to achieve group consensus in the presence of StrBYZ agents have been presented. Then, approximate Byzantine group consensus term has been introduced for the StrBYZ fault model and results on approximate Byzantine group consensus problem have been presented.

Approximate Byzantine group consensus has also been investigated under the uStrBYZ fault model. It has been shown that the fault tolerant L-MSR and R-MSR algorithms presented in the previous chapter are also able to solve the approximate Byzantine group consensus problem. Necessary and sufficient conditions for the success of these algorithms are also presented.

6. A PARAMETER-INDEPENDENT FAULT TOLERANT ALGORITHM

In this chapter, a parameter independent MSR algorithm is proposed to address the approximate Byzantine consensus problem using a fully distributed fault detection scheme. In most/all of the studies in the literature, the topology of the network is assumed to be known and some conditions are imposed on the graph topology. In [8], the condition for non-faulty nodes to detect a faulty behaviour is that the whole network must be strongly connected. Even when the given network is strongly connected, only the transmissive symmetric faults can be detected. In that sense, we try to extend the existing approaches such that non-faulty nodes are able to detect both malicious and Byzantine faults. We attach particular importance to the case in which topology is unknown. It is shown that approximate Byzantine consensus is achieved under the same topological conditions presented in [1], [2], [3], [4], [5] without the initial knowledge of the maximum number of Byzantine agents. To this end, a fully distributed Byzantine fault detection filter is implemented. Finally, the theoretical results are also illustrated with numerical studies.

6.1. Fault Detection Scheme

In this section, a fully distributed fault detection scheme is proposed in order to provide the parameter f to the MSR algorithm.

6.1.1. Fault Detection Filter

Consider a network G consisting of n agents with the consensus dynamics given as follows:

$$\begin{aligned} x[k+1] &= Wx[k] \\ y_i[k] &= C_i x[k] \end{aligned} \tag{6.1}$$

where $x[k] \in \mathbb{R}^{\mathrm{n}\times 1}$, $y_i[k] \in \mathbb{R}^{n_i\times 1}$, $n_i = |N_i|$, $C_i = [e_{k_1}...e_{k_{n_i}}]^T \in \mathbb{R}^{n_i\times \mathrm{n}}$, $k_1, ..., k_{n_i} \in N_i$. Here $y_i[k]$ denotes the state values available to node i from its neighbors and $W \in \mathbb{R}^{\mathrm{n}\times\mathrm{n}}$ is the matrix with the entries w_{ij} for which the following conditions hold:

- $w_{ij} = 0$ whenever $j \notin N_i$, $i \in V$, $k \in \mathbb{Z}_{\geq 0}$;
- $0 < \alpha \leq w_{ij}$, $\forall j \in N_i, i \in V$, $k \in \mathbb{Z}_{\geq 0}$;
- $\sum_{j=1}^{n} w_{ij} = 1$, $\forall i \in V$, $k \in \mathbb{Z}_{\geq 0}$.

Suppose that each non-faulty node i in the network builds the following filter:

$$\begin{aligned} z[k+1] &= (\tilde{W} - L)z[k] + Ly_i[k] \\ \tilde{x}[k] &= z[k] \end{aligned} \tag{6.2}$$

where $z[k], \tilde{x}[k] \in \mathbb{R}^{n_i\times 1}$, $I_{n_i} \in \mathbb{R}^{n_i \times n_i}$ is the identity matrix. The matrices $L \in \mathbb{R}^{n_i\times n_i}$ and $\tilde{W} \in \mathbb{R}^{n_i\times n_i}$ are the filter parameters to be designed by non-faulty node i. Note that each non-faulty node estimates the state values of its neighbors by using the received state values.

Lemma 6.1. *A non-faulty node i can estimate the state values of its in-neighbors using the filter given in (6.2) if and only if the following conditions hold:*

(i) *$\tilde{W}$ must be chosen as a row-stochastic matrix,*
(ii) *All eigenvalues of $\tilde{W} - L$ must be inside the unit circle.*

Proof. (Sufficiency) Suppose that conditions (i) and (ii) hold. The error term of the fault detection filter for a non-faulty node i is computed as

$$e_i[k] = y_i[k] - \tilde{x}[k] \tag{6.3}$$

where $e_i[k] = \left[e_i^1[k], \cdots, e_i^{n_i}[k]\right]^T \in \mathbb{R}^{n_i\times 1}$, $1, ..., n_i \in N_i$.

Using (6.1) and (6.2), $e_i[k]$ can be rewritten as follows:

$$\begin{aligned} e_i[k+1] &= y_i[k+1] - \tilde{x}[k+1] \\ &= C_i x[k+1] - \tilde{x}[k+1] \\ &= C_i W x[k] - (\tilde{W} - L) z[k] - L y_i[k] \\ &= C_i W x[k] - (\tilde{W} - L) z[k] - L C_i x[k] \end{aligned} \tag{6.4}$$

Substituting $z[k] = C_i x[k] - e_i[k]$ yields

$$\begin{aligned} e_i[k+1] &= C_i W x[k] - (\tilde{W} - L)(C_i x[k] - e_i[k]) - L C_i x[k] \\ &= (C_i W - L C_i) x[k] - (\tilde{W} - L)(C_i x[k] - e_i[k]) \\ &= (C_i W - L C_i) x[k] - (\tilde{W} - L) C_i x[k] + (\tilde{W} - L) e_i[k] \\ &= (C_i W - L C_i - (\tilde{W} - L) C_i) x[k] + (\tilde{W} - L) e_i[k] \\ &= (\tilde{W} - L) e_i[k] + (C_i W - \tilde{W} C_i) x[k] \\ &= (\tilde{W} - L) e_i[k] + v[k] \end{aligned} \tag{6.5}$$

where $v[k] = (C_i W - \tilde{W} C_i) x[k]$. The solution of (6.5) can be written as

$$e_i[k] = (\tilde{W} - L)^k e_i[0] + \sum_{m=0}^{k-1} (\tilde{W} - L)^{k-m-1} v[m] \tag{6.6}$$

Since all eigenvalues of $(\tilde{W} - L)$ are inside the unit circle, the system (6.5) is exponentially stable when $v[k] = 0$ holds for all k, i.e., there exists $0 < \alpha < 1$ and $m > 0$ such that for all $e_i(0)$ and k, we have

$$\begin{aligned} \|e_i[k]\|_\infty &= \left\| (\tilde{W} - L)^k e_i[0] \right\|_\infty \\ &\leq m\alpha^k \|e_i[0]\|_\infty \end{aligned} \tag{6.7}$$

where $\left\| (\tilde{W} - L)^k \right\|_\infty \leq m\alpha^k$.

Since W and $\tilde{W}$ are row-stochastic matrices, $\lim_{k\to\infty} x[k] = \lim_{k\to\infty} W^k x[0] = \mathbf{1}_{\text{nx1}} c$ holds for $c > 0$ and therefore $\lim_{k\to\infty} v[k]$ exits and can be computed as

$$\begin{aligned}
\lim_{k\to\infty} v[k] &= \lim_{k\to\infty} (C_i W - \tilde{W} C_i) x[k] \\
&= (C_i W - \tilde{W} C_i) \lim_{k\to\infty} x[k] \\
&= (C_i W - \tilde{W} C_i) \mathbf{1}_{\text{nx1}} c \\
&= \mathbf{1}_{n_i \text{x1}} c - \mathbf{1}_{n_i \text{x1}} c \\
&= 0
\end{aligned} \tag{6.8}$$

Using (6.7), (6.6) can be manipulated as follows:

$$\begin{aligned}
\|e_i[k]\|_\infty &\leq \left\|(\tilde{W} - L)^k\right\|_\infty \|e_i[0]\|_\infty + \sum_{m=0}^{k-1} \left\|(\tilde{W} - L)^{k-m-1}\right\|_\infty \|v[m]\|_\infty \\
&\leq m\alpha^k \|e_i[0]\|_\infty + \sum_{m=0}^{k-1} m\alpha^{k-m-1} \|v[m]\|_\infty \\
&\leq m\alpha^k \|e_i[0]\|_\infty + \frac{1-\alpha^k}{1-\alpha} \|v[m]\|_\infty
\end{aligned} \tag{6.9}$$

which can be further coupled with (6.8) to conclude that $\lim_{k\to\infty} e_i[k] = 0$.

(Necessity) Suppose that condition (i) does not hold. Then, $\lim_{k\to\infty} v[k] = 0$ does not hold for any other choice of $\tilde{W}$ because of the zero column of the term $\tilde{W} C_i$. Therefore, it can be concluded that $\lim_{k\to\infty} e_i[k] \neq 0$ holds regardless of the condition (ii).

Suppose that condition (ii) does not hold. Then, we have $\lim_{k\to\infty} e_i[k] \neq 0$ since the eigenvalues of the matrix $(\tilde{W} - L)$ are not inside the unit circle. □

In Lemma 6.1, it is shown that a non-faulty node i estimates the state values of its neighbours in a fully distributed fashion when there is not any faulty agent in the network. Note that each non-faulty node uses only local information available to itself.

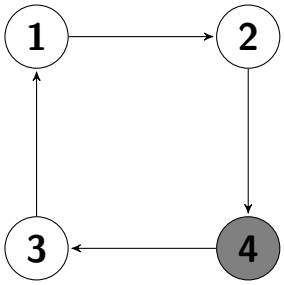


Figure 6.1. The network with 4 nodes where node 4 is Byzantine.

Remark 6.1. *Note that the term $e_i[k]$ converges to zero as k tends to infinity. After a certain amount of time, we expect that the error term becomes very small, i.e., for all $j \in N_i$ and for $k \geq \bar{k}$, $\bar{k} > 0$, $|e_i^j[k]| < \delta_k$ where δ_k is a pre-determined small positive constant. However, if there is a faulty agent in the network, then $|e_i^j[k]| \not< \delta_k$ holds for all $i \in V_{nf}$, $j \in N_i$ and for $k \geq \bar{k}$, $\bar{k} > 0$. Hence, non-faulty node i is able to detect existing faulty behaviour in the network.*

The fully distributed fault detection filter given in (6.2) allows us to observe the error terms of each individual non-faulty node of the given network as illustrated in the following examples.

Example 6.1. *Consider the network of 4 nodes given in Figure 6.1. Nodes 1, 2 and 3 build the fault detection filter given in (6.2) satisfying the conditions stated in Lemma 6.1. For this example, we assume that node 4 is not Byzantine. The pre-determined parameter is chosen as $\delta_k = 0.0001$ by each non-faulty node. As it can be seen from Figure 6.2, the error terms of nodes 1,2 and 3 satisfy $|e_i^j[k]| < \delta_k$ for all $i \in \{1, 2, 3\}$, $j \in N_i$ and for $k \geq \bar{k}$ where $\bar{k} > 20$. Since there is no Byzantine behaviour in the network, all nodes achieve consensus as in Figure 6.3.*

Example 6.2. *Consider the network of 4 nodes given in Figure 6.1. Nodes 1, 2 and 3 build the fault detection filter given in (6.2) satisfying the conditions stated in Lemma 6.1. Assume that node 4 is Byzantine. The pre-determined parameter is chosen as $\delta_k = 0.0001$ by each non-faulty node. As it can be seen from Figure 6.4, for the error terms of non-faulty nodes 1,2 and 3, $|e_i^j[k]| \not< \delta_k$ holds for all $i \in \{1, 2, 3\}$, $j \in N_i$ and for $k \geq 0$. Since there exists a Byzantine behaviour in the network, non-faulty nodes cannot achieve consensus as shown in Figure 6.5.*

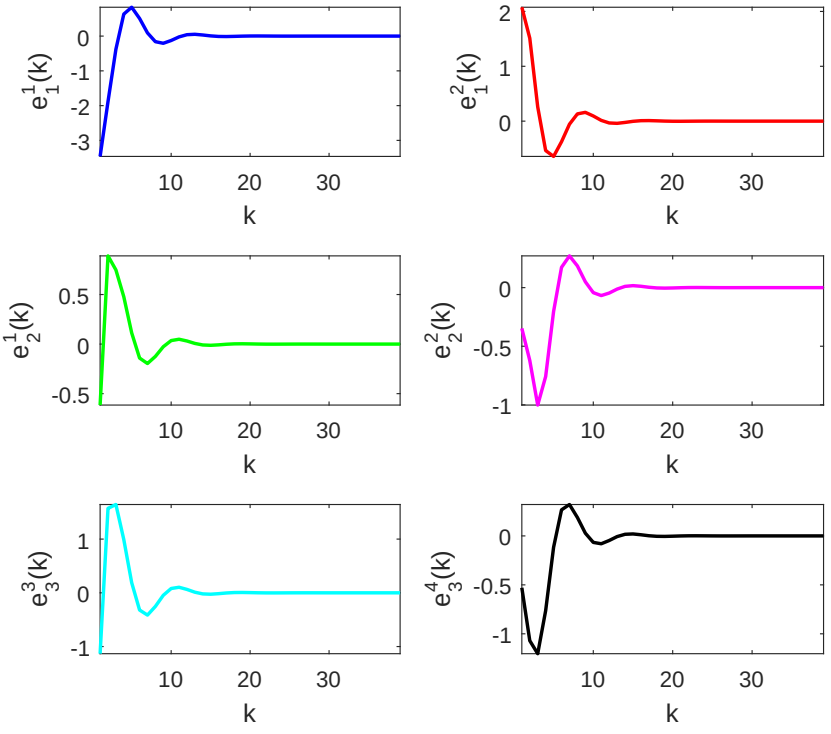


Figure 6.2. Error terms of nodes in Example 6.1.

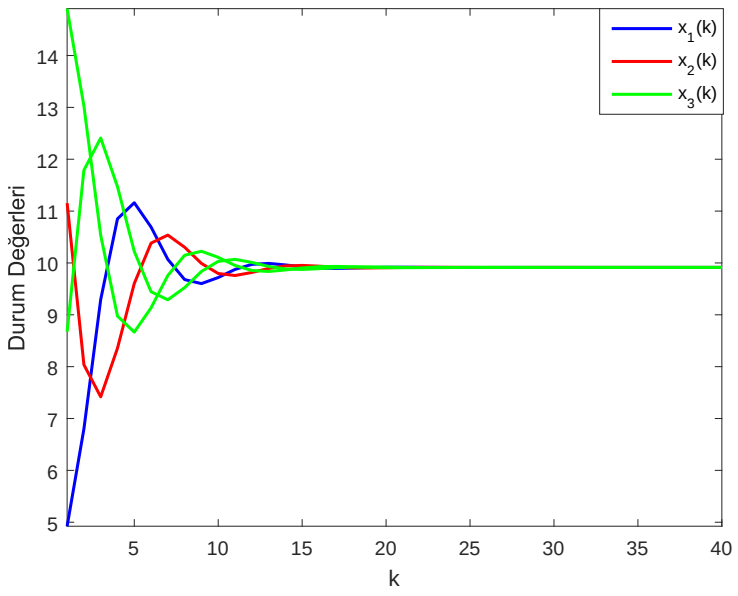


Figure 6.3. Consensus of nodes in Example 6.1.

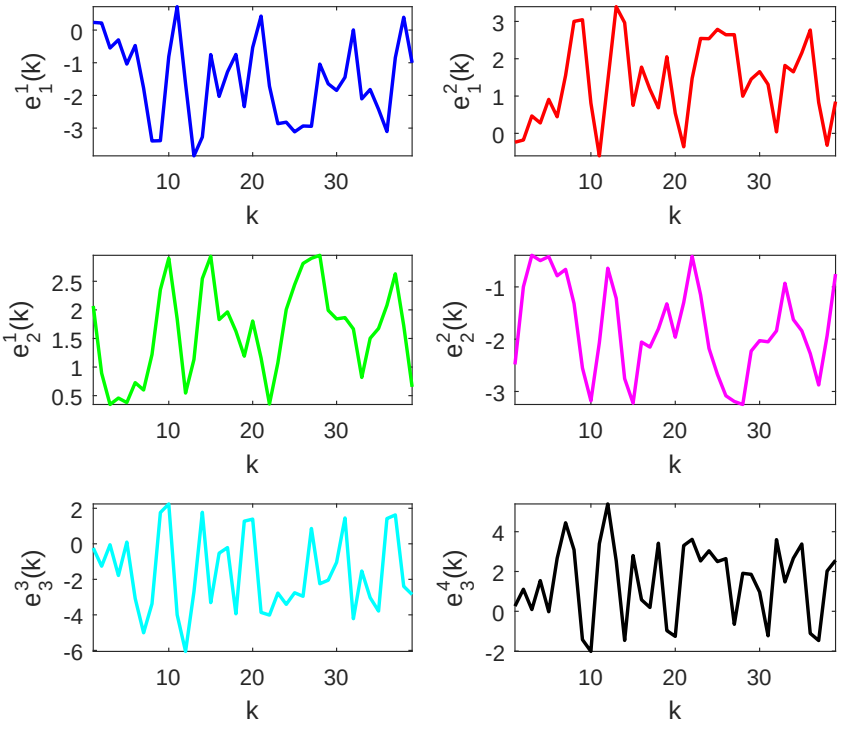


Figure 6.4. Error terms of non-faulty in Example 6.2.

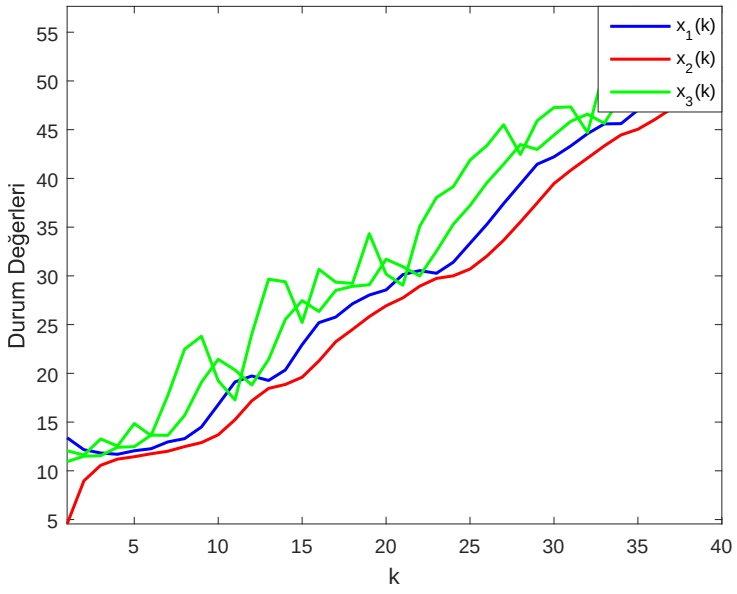


Figure 6.5. Consensus of non-faulty nodes in Example 6.2.

6.2. Parameter-Independent MSR Algorithm (PIMSR)

In this section, a parameter independet MSR algorithm (PIMSR) is proposed. In [1], [29], [30], [31], the maximum number of faulty agents that exist in the network is assumed to be known. Note that each non-faulty node can adjust the parameter f adaptively by using the fault detection filter in (6.2).

The PIMSR algorithm, which relies on this idea, is given as follows:

The PIMSR algorithm:

(i) Each non-faulty node i starts to implement the algorithm with $f_i = 0$ and sends its state value $x_i[k]$ to its out-neighbors at time step k.

(ii) Each non-faulty node i receives the state values of its in-neighbors at time step k and creates a sorted list in ascending order.

(iii) If there are less than f_i values strictly larger than its own state value $x_i[k]$, then each non-faulty node i removes all values that are strictly larger than its own. Otherwise, it removes precisely the largest f_i values in the sorted list. Likewise, if there are less than f_i values strictly smaller than its own value, then node i removes all values that are strictly smaller than its own. Otherwise, it removes precisely the smallest f_i values. Let $R_i[k]$ denote the set of nodes whose values are removed by non-faulty node i.

(iv) Let $d_i = |N_i[k] \setminus R_i[k]| = n_i - f_i$ denote the number of state values in the sorted list. Each non-faulty node i builds the fault detection filter in (6.2) where the parameters of the filter $\tilde{W} \in \mathbb{R}^{d_i \times d_i}$ and $L \in \mathbb{R}^{d_i \times d_i}$ are chosen to satisfy the conditions stated in Lemma 6.1. Each time the value of d_i is changed due to a change in the parameter f_i, the filter parameters $\tilde{W}$ and L are chosen accordingly.

(v) Each non-fautly node i applies the update rule

$$x_i[k+1] = \sum_{j \in N_i \setminus R_i[k]} w_{ij}[k] x_j[k] \tag{6.10}$$

The averaging coefficients $w_{ij}[k]$ are supposed to satisfy the following conditions:

- $w_{ij}[k] = 0$ whenever $j \notin N_i[k] \setminus R_i[k]$;
- $0 < \alpha \leq w_{ij}[k] \leq 1$, $\forall j \in N_i[k] \setminus R_i[k]$;
- $\sum_{j=1}^{n} w_{ij}[k] = 1$, $\forall i \in V$, $\forall j \in N_i[k] \setminus R_i[k]$.

(vi) After a pre-determined number of time steps $k \geq \bar{k}$ ($\bar{k} > 0$), each non-faulty node i does not detect any Byzantine agent in the network if the condition $|e_i^j[k]| < \delta_k$, where δ_k is a pre-selected positive small constant, holds for all $j \in N_i$. Otherwise, a faulty behaviour is detected and each node i updates the algorithm parameter as $f_i = f_i + 1$. Then, go to (S1) and continue to implement the algorithm with $f_i = f_i + 1$.

Remark 6.2. *The PIMSR algorithm requires a certain amount of iterations in order to detect a faulty behaviour. The choice of δ_k and $\bar{k}$ depend on the application and can also be selected by the designer of the communication network. With the proper choice of δ_k and $\bar{k}$, the PIMSR algorithm can solve the approximate Byzantine consensus problem without the initial knowledge of the parameter f.*

Theorem 6.1. *Approximate Byzantine consensus is guaranteed for the non-faulty nodes using PIMSR algorithm if and only if the network of non-faulty nodes is (f+1)-robust.*

Proof. Once the conditions given in Lemma 6.1 are satisfied and with proper choice of δ_k and $\bar{k}$ as explained in Remark 6.2, the proof directly follows from Theorem 1 in [4]. □

Example 6.3. *Consider the network with 14 nodes given in Figure 6.6 where node 14 is faulty. Each non-faulty node implements the PIMSR algorithm with $\bar{k} = 100$ and $\delta_k = 0.0001$. When $k = \tilde{k} = 100$, each non-faulty node i detects a faulty behaviour and increases f_i to 1. Using the parameter $f_i = 1$, none of the non-faulty nodes detect a faulty behaviour and non-faulty nodes reach approximate consensus as it can be seen from Figure 6.7.*

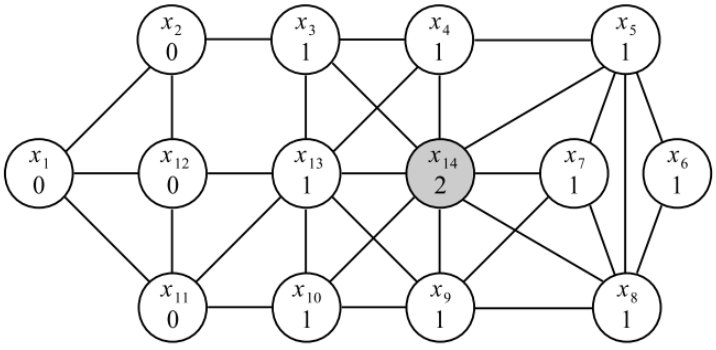


Figure 6.6. The network with 14 nodes given in [1] where node 14 is Byzantine.

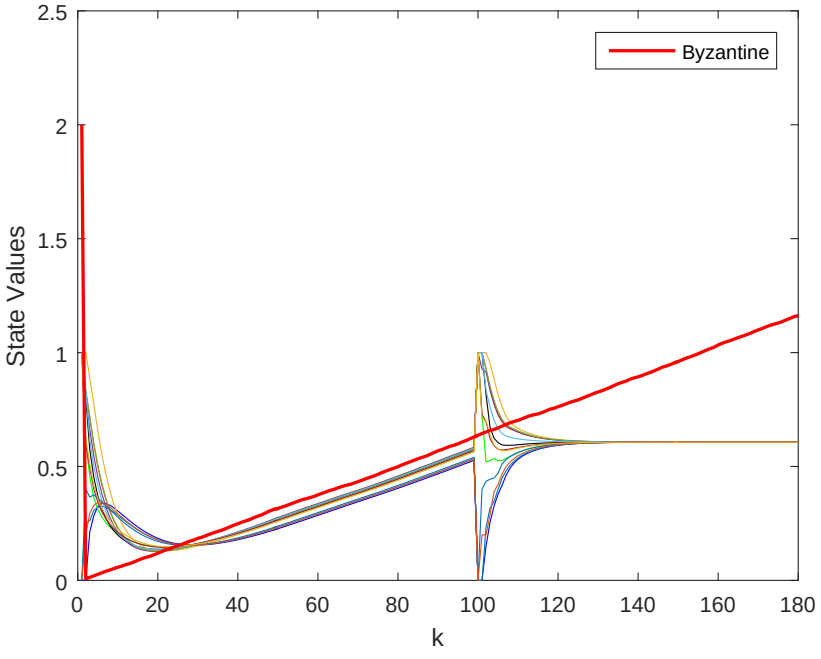


Figure 6.7. The simulation result of the network given in the Figure 6.6 using the PIMSR algorithm.

6.3. Summary of the Chapter

In this chapter, a parameter independent MSR algorithm has been proposed in order to remove the assumption that the maximum number of faulty agents is known by non-faulty nodes of the network. In the proposed PIMSR algorithm, the fault parameter f_i is adjusted adaptively starting with $f_i = 0$, and each time a faulty behaviour is detected, the value of f_i is increased until no fault is detected.

It is shown that up to f_i Byzantine agents, approximate Byzantine consensus is guaranteed for non-faulty nodes using the distributed PIMSR algorithm if and only if the network of non-faulty nodes is $(f_i + 1)$-robust which is the same condition given in [1], [2]. In summary, approximate Byzantine consensus is guaranteed without using the initial knowledge on the parameter f or without putting extra restrictive conditions on the network topology.

7. CONCLUSIONS

In this thesis, consensus under the StrBYZ and uStrBYZ fault models has been discussed. In Chapter 1, a novel fault model classification has been introduced. Consensus computation in the presence of StrBYZ agents has been discussed in Chapter 3. We have shown that non-faulty nodes of a given network can achieve consensus without employing a fault tolerant algorithm. In addition, we have studied the approximate Byzantine consensus problem under the StrBYZ fault model as StrBYZ agents usually disrupt the equilibrium point that non-faulty nodes are supposed to reach. Necessary and sufficient conditions for non-faulty nodes to achieve approximate Byzantine consensus have been presented.

In Chapter 4, consensus computation in the presence of uStrBYZ agents has been discussed. Two fault tolerant algorithms, L-MSR and R-MSR, have been proposed. Convergence analysis of both algorithms has been carried out. Note that the proposed algorithms are more general than the MSR algorithms given in [1], [2], [4].

Fault tolerant group consensus has been investigated for both StrBYZ and uStrBYZ fault models in Chapter 5. First, necessary and sufficient conditions for non-faulty nodes to achieve group consensus objective in the presence of StrBYZ agents have been presented. Then, the approximate Byzantine group consensus has been introduced and results on the approximate Byzantine group consensus problem have been presented. It has been shown that non-faulty nodes can use the L-MSR and R-MSR algorithms to achieve approximate Byzantine group consensus when the thread model is uStrBYZ.

In Chapter 6, a fully distributed fault detection scheme has been introduced so that non-faulty nodes are able to detect a faulty behaviour and adjust the parameter f adaptively. Subsequently, the PIMSR algorithm has been proposed and it has been shown that it can solve the approximate Byzantine consensus problem under the same topological conditions given in [1], [2], [4].

Future work related on the topics presented in this thesis can be stated as follows:

- Design of faster fault tolerant algorithms by appropriately choosing the averaging coefficients.
- Investigating the StrBYZ and uStrBYZ fault models when the consensus dynamics are not linear. To this end, one may want to design fault tolerant algorithms for nonlinear systems.
- Extension of these results to continuous time linear and nonlinear systems.
- In this thesis, first order systems have been investigated. Another direction for research is to extend the results to higher order systems.

REFERENCES

1. LeBlanc, H. J., H. Zhang, X. Koutsoukos and S. Sundaram, "Resilient asymptotic consensus in robust networks", *IEEE Journal on Selected Areas in Communications*, Vol. 31, No. 4, pp. 766–781, 2013.

2. LeBlanc, H. J. and X. Koutsoukos, "Resilient asymptotic consensus in asynchronous robust networks", *50th Annual Allerton Conference on Communication, Control, and Computing*, pp. 1742–1749, 2012.

3. Dolev, D., N. A. Lynch, S. S. Pinter, E. W. Stark and W. E. Weihl, "Reaching approximate agreement in the presence of faults", *Journal of Association for Computing Machinery*, Vol. 33, No. 3, pp. 499–516, May 1986.

4. Haseltalab, A. and M. Akar, "Approximate byzantine consensus in faulty asynchronous networks", *American Control Conference*, pp. 1591–1596, 2015.

5. Haseltalab, A. and M. Akar, "Convergence rate analysis of a fault-tolerant distributed consensus algorithm", *54th IEEE Annual Conference on Decision and Control*, pp. 5111–5116, 2015.

6. Sundaram, S. and C. N. Hadjicostis, "Distributed function calculation via linear iterations in the presence of malicious agents—Part I: Attacking the network", *American Control Conference*, pp. 1350–1355, 2008.

7. Sundaram, S. and C. N. Hadjicostis, "Distributed function calculation via linear iterations in the presence of malicious agents—Part II: Overcoming malicious behavior", *American Control Conference*, pp. 1356–1361, 2008.

8. Pasqualetti, F., A. Bicchi and F. Bullo, "Distributed intrusion detection for secure consensus computations", *46th IEEE Conference on Decision and Control*, pp. 5594–5599, 2007.

9. Pasqualetti, F., A. Bicchi and F. Bullo, "Consensus computation in unreliable networks: A system theoretic approach", *IEEE Transactions on Automatic Control*, Vol. 57, No. 1, pp. 90–104, 2012.

10. Zhang, H. and S. Sundaram, "Robustness of information diffusion algorithms to locally bounded adversaries", *American Control Conference*, pp. 5855–5861, 2012.

11. Krings, A. W. and Z. Ma, "Fault-models in wireless communication: Towards survivable ad hoc networks", *Military Communications Conference*, pp. 1–7, 2006.

12. Azadmanesh, M. H. and R. M. Kieckhafer, "Exploiting omissive faults in synchronous approximate agreement", *IEEE Transactions on Computers*, Vol. 49, No. 10, pp. 1031–1042, 2000.

13. Kieckhafer, R. and M. H. Azadmanesh, "Reaching approximate agreement with mixed-mode faults", *IEEE Transactions on Parallel and Distributed Systems*, Vol. 5, No. 1, pp. 53–63, 1994.

14. Lamport, L., R. Shostak and M. Pease, "The Byzantine Generals Problem", *ACM Trans. Program. Lang. Syst.*, Vol. 4, No. 3, pp. 382–401, Jul. 1982.

15. Vaidya, N. H., L. Tseng and G. Liang, "Iterative approximate byzantine consensus in arbitrary directed graphs", *Proceedings of the ACM symposium on Principles of distributed computing*, pp. 365–374, 2012.

16. Vaidya, N. H., "Iterative Byzantine vector consensus in incomplete graphs", *International Conference on Distributed Computing and Networking*, pp. 14–28, Springer, 2014.

17. Tseng, L. and N. Vaidya, "Iterative approximate byzantine consensus under a generalized fault model", *International Conference on Distributed Computing and Networking*, pp. 72–86, Springer, 2013.

18. Tseng, L. and N. Vaidya, "Iterative approximate consensus in the presence of byzantine link failures", *Networked Systems*, pp. 84–98, Springer, 2014.

19. Öksüz, H. Y. and M. Akar, "Distributed Consensus in Resilient Networks", *56th IEEE Conference on Decision and Control*, 2017 (accepted).

20. Ju, L., H. Li, Y. Liu, W. Xue, K. Li and Z. Chi, "An improved intrusion detection scheme based on weighted trust evaluation for wireless sensor networks", *Proceedings of the 5th International Conference on Ubiquitous Information Technologies and Applications (CUTE)*, pp. 1–6, 2010.

21. Parno, B., A. Perrig and V. Gligor, "Distributed detection of node replication attacks in sensor networks", *IEEE Symposium on Security and Privacy*, pp. 49–63, 2005.

22. Erkan, Ö. F., O. Cihan and M. Akar, "Distributed consensus with multi-equilibria in directed networks", *American Control Conference*, pp. 4681–4685, 2017.

23. Erkan, Ö. F., O. Cihan and M. Akar, "Analysis of Distributed Consensus Algorithms with Multiple Equilibria in Directed Networks", *Journal of the Franklin Institute (*Under review*)*.

24. Ren, W. and R. W. Beard, "Consensus seeking in multiagent systems under dynamically changing interaction topologies", *IEEE Transactions on Automatic Control*, Vol. 50, No. 5, pp. 655–661, 2005.

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

26. Tseng, L. and N. H. Vaidya, "Parameter-independent iterative approximate byzantine consensus", *arXiv preprint arXiv:1208.4867*, 2012.

27. Fang, L., P. J. Antsaklis and A. Tzimas, "Asynchronous consensus protocols: Pre-

liminary results, simulations and open questions", *44th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC'05)*, pp. 2194–2199, 2005.

28. Fang, L. and P. J. Antsaklis, "Information consensus of asynchronous discrete-time multi-agent systems", *American Control Conference*, pp. 1883–1888, 2005.

29. Dibaji, S. M., H. Ishii and R. Tempo, "Resilient randomized quantized consensus", *American Control Conference*, pp. 5118–5123, 2016.

30. Dibaji, S. M., H. Ishii and R. Tempo, "Resilient randomized quantized consensus with delayed information", *55th IEEE Conference on Decision and Control*, pp. 3505–3510, 2016.

31. Saldaña, D., A. Prorok, S. Sundaram, M. F. Campos and V. Kumar, "Resilient consensus for time-varying networks of dynamic agents", *American Control Conference*, pp. 252–258, 2017.