RESILIENT COOPERATIVE CONTROL OF NETWORKED MULTI-AGENT SYSTEMS

By

Heath J. LeBlanc

Dissertation
Submitted to the Faculty of the
Graduate School of Vanderbilt University
in partial fulfillment of the requirements
of the degree of

DOCTOR OF PHILOSOPHY

in

Electrical Engineering

December, 2012

Nashville, Tennessee

Approved:
Professor Xenofon Koutsoukos
Professor Mark Ellingham
Professor Gabor Karsai
Professor Janos Sztipanovits
Professor Yuan Xue

To Jesus, whose teachings have shaped the way I live;

To my loving wife, Alison, whose wisdom and faith in me support my being;

To my mother, who I pray may know of this accomplishment;

To my grandfather, Carol J. LeBlanc, who inspired me to pursue an academic life.

"A man is a mind, not a body, capable of acceptance of responsibility,

and possessing a keen sense of duty based on the laws of God."

- Carol J. LeBlanc (1934–2009)

ACKNOWLEDGMENTS

I am grateful to all of those people with whom I have had the pleasure of working during my time at Vanderbilt. I would like to especially thank my adviser Xenofon Koutsoukos for his advice and guidance with this work. I would also like to thank Shreyas Sundaram and Haotian Zhang at the University of Waterloo for the many insights and ideas shared during our collaborations. I am grateful as well to the members of my committee: Janos Sztipanovits, Gabor Karsai, Yuan Xue, and Mark Ellingham for their guidance. I am also obliged to my colleagues, both past and present, in the ModEs lab and ISIS. Most especially, I express a deep appreciation to Emeka Eyisi, Derek Riley, Ashraf Tantawy, Daniel L. C. Mack, Zhenkai Zhang, Peter Horvath, Mark Yampolskiy, Siyuan Dai, Joe Porter, and Gabor Simko for the various discussions that have provided me feedback that has helped in this research. I would also like to thank Rob Rioux for his guidance and mentorship early in my academic career, and for his friendship and support throughout.

Most importantly, I am indebted to my wife Alison, whose love, support, and wisdom guides me and motivates me to be the best I can be. Finally, this work would not have been possible without the financial support of the National Science Foundation CAREER CNS-0347440 grant awarded to Xenofon Koutsoukos and the National Science Foundation grants CCF-0820088 and CNS-1035655.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

1.1 Motivation

In the past few decades, advances in semiconductor technology and manufacturing techniques have fueled the digital revolution. Throughout this time, the cost of computational resources and memory have consistently decreased, while the computational performance has increased exponentially. This simultaneous trend of decreasing cost with increasing performance has facilitated the rise of embedded systems. The proliferation of embedded systems is profoundly affecting the modern world, from how systems are engineered, to how people socialize. And the transformation is changing the very nature of computing, from interactive to proactive [187].

The rise of embedded systems, along with advances in networking technology, have facilitated a paradigm shift in engineering system design, from centralized to distributed. This shift has been propelled by low-cost, high performance embedded systems along with numerous applications, and has lead to significant interest in the design and analysis of *multi-agent networks*.

A multi-agent network, or *networked multi-agent system*, consists of a set of individuals called *agents*, or *nodes*.[1] The agents are equipped with some means of *sensing* or *communicating*, along with computational resources and possibly *actuation*. Through a medium, which is referred to as the *network*, the agents share information in order to achieve specific *group objectives*. Some examples of group objectives include consensus, synchronization, formation control, and cooperative load transport. In order for the group objectives to be achieved, *distributed algorithms* are used to coordinate the behavior of the agents.

There are several advantages to using multiple agents over a single one. First, the objective may be complex and challenging, or possibly even infeasible for a single agent to achieve. Second, employing many agents can provide some robustness in the case of failures or faults. Third, networked multi-agent systems are flexible and can support reconfigurability. Finally, there are performance

[1]The terms 'agent' and 'node' are used synonymously throughout this manuscript, with preference toward 'agent' whenever the individuals have physical dynamics and 'node' for the more general setting in which the individuals may be processors, robots, etc.

advantages that can be leveraged from multiple agents. For example, in surveillance and monitoring applications, a multi-agent network provides redundancy and increased fidelity of information [31, 99].

1.2 Research Challenges

Along with the advantages come certain challenges. Perhaps the most fundamental challenge in the design of networked multi-agent systems is the restriction that the coordination algorithms use only *local information*, i.e., information obtained by the individual agent through sensor measurements, calculations, or communication with neighbors in the network. In this manner, the algorithms and feedback control laws must be *distributed*.

A second challenge lies in the fact that not only is each agent a dynamical system, but the network itself is dynamic. Therefore, the distributed algorithms must be designed to handle time-varying network topologies. In particular, information may not be able to be relayed across the dynamic network in a reliable manner. Time-varying networks arise from agent mobility and environment factors such as interference. Since the distributed algorithms depend directly on the network, this additional source of dynamics can affect the stability and performance of the networked system.

A third challenge is caused by uncertainties introduced by the network and in the implementation of the control algorithms. As described above, the network is a dynamical system. Depending on the different time constants involved, delays in sensing or communication may lead to instability. Moreover, information may be lost in the network, and the implementation of the control algorithms may be subject to quantization. How these concerns affect stability and performance is a difficult problem.

Finally, multi-agent networks, like all large-scale distributed systems, have many entry points for malicious attacks or intrusions. If one or more of the agents are compromised in a security breach, it is crucial for the networked system to continue operating with minimal degradation in performance. Most importantly, the success of the global objective should be assured. To achieve this, it is necessary for the cooperative control algorithms to be designed in such a way that they can withstand the compromise of a subset of the nodes and *still guarantee some notion of correct behavior at a minimum level of performance*. We refer to such a multi-agent network as being *resilient* to adversaries. Given the growing threat of malicious attacks in large-scale cyber-physical systems, this is an important and challenging problem [30].

1.3 Outline and Overview

This section outlines the contents of this manuscript, and describes in broad terms the problems addressed within each chapter.

Chapter 2 sets the stage for subsequent chapters by providing a detailed review of a subset of the literature on cooperative control of networked multi-agent systems, as well as some background in passivity. To effectively present results from the domain of multi-agent networks, which spans several disciplines (with a myriad of differences in jargon and notation), we present a unified view of multi-agent systems by characterizing the key aspects and attributes of such systems. We focus mainly on consensus problems, synchronization, and passivity-based techniques, which are directly related to the contributions of this work. We provide a background in graph theory that contains the terminology and notation for modeling the network component of the networked multi-agent system used throughout the manuscript. Chapter 2 ends with a brief discussion on the relationship of the related work to the research presented here.

Chapter 3 studies the deployment of passive networked multi-agent systems in uncertain networks. The *deployment problem* requires that a set of mobile agents asymptotically converge to predetermined points in space. For simple agent models (e.g., point mass) in an ideal environment (e.g., deterministic), the deployment problem is trivial to solve and requires no coordination among the agents. Indeed, each agent may simply focus on its task of converging to its preassigned point in space, without the need for feedback from other agents (and therefore, no need for a network).

On the other hand, allowing the agents to exchange information over a network may be beneficial in order to coordinate the time of arrival of the agents (i.e., when the agents arrive at their destinations), among other reasons. But, communication networks introduce uncertainties such as time-varying network delays and data loss. Handling network uncertainties while ensuring stability of networked dynamical systems is a challenging problem. Chapter 3 presents a passivity-based approach to ensure global passivity and stability in the presence of data loss and time-varying delays.

Chapters 4-8 comprise our results on resilient cooperative control of networked multi-agent systems. These chapters study different cooperative control problems in the presence of compromised nodes, or *adversaries*. The nodes are assumed to be compromised in an undetected security breach. The goal is for the uncompromised nodes, or simply *normal nodes*, to still achieve the group objective in the presence of the adversary nodes. Therefore, the networked multi-agent system should be *resilient* to adversaries. It is important to emphasize that the only type of security breach we study is compromised (adversary) nodes, as opposed to attacks on the communication network (e.g., denial of service or deception attacks). That being said, the models considered are general enough so that from a local perspective (i.e., from the point of view of the individuals in the network), the difference between compromising the node or the outgoing communications of the same node is indistinguishable [186].

The adversary models studied here can be characterized by *threat models* and *scope of threat assumptions*. The threat model determines the feasible behaviors of the adversary nodes. Examples of different threat models include *non-colluding* [159], *malicious* [159, 181, 113], *Byzantine* [110, 1, 114, 194], or *crash* [1, 46] nodes. Crash nodes fail by simply stopping their movement and possibly communication. Non-colluding nodes are unaware of the network topology, which other nodes are misbehaving, or the states of non-neighboring nodes. On the other hand, malicious nodes have full knowledge of the networked system and therefore, worst case behavior must be assumed. The only difference between malicious and Byzantine nodes lies in their capacity for deceit. Malicious nodes must convey the same information to each neighbor, whereas Byzantine can convey different information to different neighbors. Of these threat models, the two that are worst-case models are malicious and Byzantine nodes. Algorithms that succeed against these worst-case models also succeed against the less general models, such as non-colluding or crash nodes.

In much of the distributed computing literature, the scope of faults (or in the context of this research, *scope of threats*) is assumed to be bounded by a constant, i.e., at most F out of n nodes fail (or are compromised) [130]. We refer to this as the *F-total model*. Alternatively, the scope may be local; e.g., at most F neighbors of any normal node fail (*F-local model*) [210], or at most a fraction f of neighbors are compromised (*f-fraction local model*) [115]. While the F-total model typically requires certain bounds on the fraction of nodes that may be compromised, the local and fractional models are dependent on the network topology and do not, in general, imply a bound on the fraction of compromised nodes.

In this work, we focus on deterministic scope of threat models as opposed to stochastic models.

While stochastic *failure* models have been studied extensively, we assert that deterministic scope of threat models are more suitable in the context of a security breach by omniscient, worst-case adversaries. This is because worst case analysis must be performed in this case, which can be formulated in terms of deterministic bounds. The assumptions that the adversaries are omniscient and behave in a worst-case manner are reasonable due to the uncertain and adversarial nature of security breaches.

We begin our study of resilient cooperative control in the presence of adversaries in Chapter 4, where we introduce, for the first time, a *resilient consensus problem* formulated in *continuous time*. The problem formulation includes a novel definition of Byzantine agents with continuous-time semantics. In many cases, the dynamics of the normal agents in a multi-agent network are modeled in continuous time using Ordinary Differential Equations (ODEs) [137]. On the other hand, consensus is a fundamental group objective. Often more complex group objectives (such as synchronization, formation control, or coordinated path following) require solving a consensus problem as a constituent objective. Therefore, a major advantage to the continuous-time formulation of resilient consensus and the adversary models is that it provides a semantic framework that can be reused to introduce the notion of resilience in more complex group objectives using ODE models for the dynamics of the normal agents.

Chapter 4 defines the continuous-time resilient asymptotic consensus (CTRAC) problem and introduces a resilient, continuous-time protocol, which is referred to as the Adversarial Robust Consensus Protocol (ARC-P). The protocol, ARC-P, is very light-weight. It uses only local information (i.e., information obtained from neighbors or computed locally), it requires no historical information, and it is low complexity. Through the analysis of ARC-P, we show that traditional graph theoretic metrics are inadequate to characterize the tight topological conditions under which convergence is assured. In more detail, pathological examples are presented in which ARC-P does not succeed in achieving consensus among the normal agents, even in the case where there are no adversaries present. Indeed, any resilient distributed algorithm that is capable of succeeding in the presence of adversaries, without *a priori* knowledge of the adversaries or network topology, must filter the information from neighboring nodes with some measure of skepticism. This amounts to the need for redundancy of information from neighbors in the network, and is the basis of the novel property referred to as *network robustness*, introduced in [210].

Chapter 5 continues the study of CTRAC carried out in Chapter 4 by introducing a definition of network robustness, or just *robustness*, that has finer granularity than the one introduced in [210]. Equipped with this finer granularity definition of robustness, we are able to characterize the necessary and sufficient conditions on the network topology required for a variant of ARC-P, referred to as ARC-P2, to achieve CTRAC in time-invariant networks under the malicious F-total model. We also analyze the convergence of ARC-P2 under the crash and Byzantine threat model, in combination with the F-local and f-fraction local scope of threat models, and in both time-invariant and time-varying robust networks. Several important properties of robust networks are given, and the implications of robustness on other metrics, such as connectivity and minimum degree, are explored.

Chapter 6 analyzes the resilient asymptotic consensus (RAC) problem in discrete-time synchronous networks. We study the class of Weighted Mean-Subsequence-Reduce (W-MSR) algorithms introduced in [210]. As in Chapter 5, we study W-MSR for all combinations of threat models with scope of threat assumptions, and either fixed or dynamic networks. Some recent results obtained in the literature for the F-total Byzantine model are restated under the unifying framework of network robustness [194, 195].

Chapter 7 generalizes the RAC problem studied in Chapter 6 to an asynchronous setting. The model of asynchrony studied in this work is similar to what is presented in [51], except in this work we study communication under a local broadcast model (i.e., each node transmits the same message to any neighbor when transmitting). The necessary and sufficient conditions for the point-to-point model have appeared recently in [195]. The network model is asynchronous in the sense that there is no common clock and the nodes may execute their algorithms at different rates. The model of the network also allows for time-varying delays in transmission and out of order delivery of messages, but assumes there is some bound on the transmission time (unknown to the nodes). This chapter focuses only on the malicious threat model in time-invariant networks under the F-total and F-local scope of threat assumptions. The results may readily be extended to the other models. We also consider a quantization scheme that ensures finite-time approximate agreement, where the maximum error is given by the precision of the quantizer.

Chapter 8 introduces the resilient asymptotic synchronization (RAS) problem in continuous time and demonstrates how CTRAC algorithms may be used in more complex group objectives. The key factor making RAS more complex and challenging than RAC is the additional complexity in the dynamics of the normal agents. In the CTRAC problem, the agents have simple integrator dynamics and the dynamic behavior of the normal agents arises purely from their interactions with their neighbors. On the other hand, in RAS, the "open-loop" dynamics (i.e., the dynamics excluding the coupling with neighbors in the network) may be nontrivial. A multi-agent network achieves RAS if the normal nodes converge to a common "open-loop" solution. In general, this problem is very challenging (e.g., with nonlinear chaotic dynamics in time-varying networks). However, in this chapter, we restrict our attention to linear time-invariant (LTI) systems with no unstable modes.[2] In this case, it is shown that under the additional assumption of stabilizability and detectability, a resilient dynamic control law that uses the same functions as ARC-P2 is used to reduce the synchronization problem to a consensus problem. The results on RAS focus on the F-total malicious model, but modifications may be made to apply the results to the other adversary models in an analogous manner to the consensus results.

Throughout the dissertation, network robustness is shown to be useful, and at times the key property for analyzing the resilient cooperative control problems studied. Because of the importance of network robustness, Chapter 9 examines algorithms for determining the robustness of a network. A centralized algorithm is introduced to determine the robustness of any network, regardless of its connectedness properties, but assumes the topology of the network is given as input to the algorithm. Using this centralized algorithm, a modification is given to enable the individual nodes of a (connected) network to compute the robustness of the network by broadcasting information about their neighborhood. Finally, a modification to this algorithm is proposed in which the individual node only checks the reachability conditions for subsets in which it is *not* included. Afterwards, the nodes broadcast their estimates of the network's robustness and the minimum of the estimates is taken as the true robustness of the network. The complexity of the algorithms and improvement incurred by the distributed algorithm are analyzed. Finally, a growth model for constructing robust networks is demonstrated that entails the preferential-attachment growth model of scale-free networks [2].

The final chapter includes a summary of the work contained in this manuscript, and provides directions for future research.

[2]Observe that since in RAS, the nodes converge to an "open-loop" solution, it seems undesirable to include systems with unstable modes. Moreover, assuming the system is stabilizable, even a system with unstable modes may be locally stabilized and the results of the chapter are then applicable.

1.4 Contributions

This section enumerates the contributions of this dissertation, and is presented in terms of the chapters. The technical contributions are contained in Chapters 3-9.

Chapter 3

- We prove a novel compositional result for the interconnection of discrete-time passive systems in arbitrary bidirectional networks. It is shown that under certain message passing rules and interface components, the passivity of the networked system is maintained even in the presence of time-varying delays and data loss.
- We demonstrate the passivity result by applying it to the deployment of passive systems in uncertain networks. We show that deployment is possible under certain assumptions on the dynamics of the agents and if the graph describing the overlay network is not bipartite.

Chapter 4

- We introduce, for the first time, a resilient consensus problem in continuous time, called the Continuous-Time Resilient Asymptotic Consensus (CTRAC) problem.
- We define several continuous-time threat models, which includes, for example, the Byzantine agent.
- We formulate the Adversarial Robust Consensus Protocol (ARC-P), which is the first resilient consensus algorithm introduced in continuous time.
- We prove existence and uniqueness of solutions whenever normal agents use ARC-P. These results pave the way for "continualizing" discrete algorithms that include sorting and removal of extreme values.
- We prove that ARC-P solves the CTRAC problem in a class of network topologies defined by degree conditions of the nodes in the network, under the F-total malicious and Byzantine model (and therefore, also the crash model).

- We demonstrate that the degree conditions are *sharp* in the sense that if one of the conditions is relaxed, then there exist pathological examples in which ARC-P does not assure CTRAC. The pathological examples also motivate the need for network robustness [210].

Chapter 5

- We introduce a definition of network robustness that has finer granularity than the one introduced in [210]. The novel definition is referred to as (r, s)-robustness.
- We prove several properties of robust networks, including the minimum in-degree and connectivity of robust networks.
- We prove a tight necessary and sufficient condition for ARC-P2 to achieve CTRAC in time-invariant networks under either the F-total malicious or crash models (with an additional assumption of uniform continuity).
- We prove a sufficient condition for time-invariant robust networks under the F-local malicious model.
- We prove a tight necessary and sufficient condition on the normal network (which is the sub-network that removes the adversary nodes and any directed edges incident with an adversary node) under the F-local and F-total Byzantine model.
- We prove separate necessary and sufficient conditions under the f-fractional malicious model.
- We prove necessary and sufficient conditions under the f-fractional Byzantine model.
- We prove sufficient conditions on all of the adversary models in time-varying networks that satisfy certain robustness properties over time.

Chapter 6

- As in the previous chapters, we prove necessary and sufficient conditions under the various adversary models and for both time-invariant and time-varying networks with appropriate robustness properties.[3]

[3]Note that some of these results have been proven by other researchers. In particular, the results for the F-local and f-fraction local models (for both malicious and Byzantine agents and both time-invariant and time-varying networks) are

Chapter 7

- We prove necessary and sufficient conditions for existence of an algorithm that achieves DTRAC in asynchronous networks with the local broadcast model under the F-total malicious model.
- Separate necessary and sufficient conditions are given for the F-local malicious model.

Chapter 8

- We introduce, for the first time, a resilient synchronization problem in continuous time, called the Resilient Asymptotic Synchronization (RAS) problem.
- We show RAS is achieved in sufficiently robust networks (for both time-invariant and time-varying networks) whenever the normal nodes are identical LTI systems with no unstable modes, under the F-total malicious model (with uniformly continuous state, control state, and observer state trajectories). We consider systems with full state feedback with the assumption of stabilizability and output feedback with the additional assumption of detectability.

Chapter 9

- We define centralized algorithms for checking and determining the robustness of any network and analyze their complexity.
- We define a decentralized algorithm that uses the centralized algorithm to determine the robustness of the network, assuming the network is connected.
- We define a distributed algorithm that distributes the calculation of robustness across the nodes. We analyze the improvement realized from this algorithm over the decentralized one.

1.5 Notation

The set of *natural numbers*, *integers*, *real numbers*, and *complex numbers* are denoted by $\mathbb{N} = \{1, 2, \dots\}$, $\mathbb{Z}$, $\mathbb{R}$ and $\mathbb{C}$, respectively. Let $\mathbb{Z}_{>0} = \mathbb{N}$, $\mathbb{Z}_{\geq 0}$, $\mathbb{R}_{>0}$, and $\mathbb{R}_{\geq 0}$ denote the set of positive

contributions by Zhang and Sundaram in our collaborations; however, the proofs presented here are improvements to the original proofs that demonstrate better performance of the algorithms. The first necessary and sufficient conditions under the F-total Byzantine model are given in [194]; alternative conditions are analyzed in [195]. The sufficient condition stated here in terms of network robustness is also a contribution of Zhang and Sundaram in our collaborations.

integers, nonnegative integers, positive reals, and nonnegative reals, respectively. Given $z \in \mathbb{C}$, the real part of $z = a + b\mathsf{i}$ (with $a, b \in \mathbb{R}$) is denoted $\mathrm{Re}(z) = a$ and the imaginary part of z is denoted $\mathrm{Im}(z) = b$, with $\mathsf{i} = \sqrt{-1}$. Given $a \in \mathbb{R}$, the *ceiling* of a, denoted $\lceil a \rceil$, is the smallest integer that is greater than or equal to a. Similarly, the *floor* of a, denoted $\lfloor a \rfloor$, is the largest integer less than or equal to a. The *absolute value* of a is denoted $|a|$. Given sets $\mathcal{S}_1, \mathcal{S}_2$, the *union* and *intersection* of $\mathcal{S}_1$ and $\mathcal{S}_2$ are denoted $\mathcal{S}_1 \cup \mathcal{S}_2$ and $\mathcal{S}_1 \cap \mathcal{S}_2$, respectively. The *reduction* of $\mathcal{S}_1$ by $\mathcal{S}_2$ is denoted $\mathcal{S}_1 \setminus \mathcal{S}_2 = \{x \in \mathcal{S}_1 \colon x \notin \mathcal{S}_2\}$. The *cardinality* of a set $\mathcal{S}$ is given by $|\mathcal{S}|$. The *empty set* is denoted $\emptyset$. The *Cartesian product* of $\mathcal{S}_1$ and $\mathcal{S}_2$ is defined as $\mathcal{S}_1 \times \mathcal{S}_2 = \{(i, j) \colon i \in \mathcal{S}_1, j \in \mathcal{S}_2\}$ and the Cartesian product of $m \in \mathbb{N}$ copies of a set $\mathcal{S}$ is denoted $\mathcal{S}^m$. The set containing all combinations of pairs of distinct elements from a given set $\mathcal{S}$ is denoted $\binom{\mathcal{S}}{2}$.

The m-dimensional Euclidean space is $\mathbb{R}^m$ and the set of all m by n matrices over the real numbers is $\mathbb{R}^{m \times n}$. The *transpose* of a (column) vector $x \in \mathbb{R}^m$ and matrix $M \in \mathbb{R}^{m \times n}$ are denoted x^{T} and M^{T}, respectively. The *Hermitian transpose* of complex matrix $M \in \mathbb{C}^{m \times m}$ is denoted M^{H}. The determinant of a matrix $A \in \mathbb{R}^{m \times m}$ is denoted $\det(M)$. The elements of a vector $x \in \mathbb{R}^m$ are indexed with subscripts in $\{1, 2, \ldots, m\}$; e.g., $x = [x_1, x_2, \ldots, x_m]^{\mathsf{T}}$. The vector of ones in $\mathbb{R}^m$ is denoted $1_m = [1, 1, \ldots, 1]^{\mathsf{T}}$ and the *identity matrix* in $\mathbb{R}^{m \times m}$ is denoted I_m. Given $B \in \mathbb{R}^{m \times n}$ and $C \in \mathbb{R}^{p \times q}$, the *Kronecker product* $B \otimes C \in \mathbb{R}^{mp \times nq}$ is defined as [80]

$$B \otimes C \triangleq \begin{bmatrix} b_{11}C & \ldots & b_{1n}C \\ \vdots & \ddots & \vdots \\ b_{m1}C & \ldots & b_{mn}C \end{bmatrix}.$$

For real number $p \geq 1$ and vector $x \in \mathbb{R}^m$, the *p-norm* of x is defined as

$$||x||_p = \left(\sum_{i=1}^{m} |x_i|^p \right)^{1/p}.$$

The *spectral norm* of a matrix $M \in \mathbb{C}^{m \times m}$ is given by $||M||_2 = (\max\{|\lambda| \colon \lambda(M^{\mathsf{H}}M)\})^{1/2}$, where, in general, $\lambda(A)$ denotes the set of eigenvalues of matrix A. The distance of a vector $v \in \mathbb{R}^m$ from a set $\mathcal{S} \subset \mathbb{R}^m$ is given by $\mathrm{dist}(v, \mathcal{S}) = \inf_{y \in \mathcal{S}} ||v - y||_2$. The span of a set of vectors $v_1, v_2, \ldots, v_N$ in $\mathbb{R}^m$ is denoted by $\mathrm{span}\{v_1, v_2, \ldots, v_N\}$. In particular, we will be interested in $\mathrm{span}\{1_m\}$, which is referred to as the *agreement space*, and is the equilibrium set of interest when studying consensus problems.

A function f with *domain* $\mathcal{U}$ and *codomain* $\mathcal{Y}$ is denoted $f\colon \mathcal{U} \to \mathcal{Y}$. The time derivative of a function $y\colon \mathbb{R} \to \mathbb{R}^m$ at time $t \in \mathbb{R}$ is denoted $\dot{y}(t)$, or just $\dot{y}$ for brevity. With an abuse of notation, the one-sided time derivative is denoted in a similar manner. For example, we may define the Ordinary Differential Equation (ODE) $\dot{y}(t) = f(t, y)$ for $t \geq 0$, and it is understood that the right-sided derivative is used at time $t = 0$.

CHAPTER II

RELATED WORK

Applications of multi-agent networks span several disciplines, including physics, electrical engineering, computer science, mechanical engineering, and the biological sciences. Therefore, the breadth of material in the literature is exorbitant. In order to be germane, this chapter discusses works most closely related to the contributions of this dissertation. Specifically, we focus mainly on consensus problems, synchronization, and passivity-based techniques. For the material on consensus, which is the major focus of this chapter, we provide the most detail on methods that address faults and failures that are adversarial in nature. In addition to presenting material on multi-agent networks, some background is provided on related topics that are important to our contributions, such as graph theory [23], with an emphasis on algebraic graph theory [73, 36], passivity [48, 96], wave variables [62, 147], and distributed computing [130, 19]. We begin with a brief review of graph theory in Section 2.1. Then we discuss passivity and wave variables in Section 2.2. The bulk of the content of the chapter is contained in Section 2.3, which discusses different research directions within the scope of multi-agent networks by organizing along different group objectives. Finally, Section 2.4 discusses how the contributions of this dissertation fit within the scope of the literature.

2.1 A Brief Review of Graph Theory

In this section we review some fundamentals of graph theory germane to this manuscript. It is common when dealing with multi-agent networks to model the networked multi-agent system with a (finite, simple, labeled) *undirected graph*, or just *graph*, $\mathcal{G} = (\mathcal{V}(\mathcal{G}), \mathcal{E}(\mathcal{G}))$ [136]. The *vertex set*, or *node set*, $\mathcal{V}(\mathcal{G}) = \mathcal{V} = \{1, \ldots, n\}$ abstracts the n agents as *vertices*, or *nodes*, and the *edge set* $\mathcal{E}(\mathcal{G}) = \mathcal{E} \subseteq \binom{\mathcal{V}}{2}$ models the information flow or influence between the agents, and is typically realized through communication or sensing (however, in some cases influence arises from physics, such as action-reaction force pairing in cooperative load transport [141]). Each edge $e = \{i, j\} \in \mathcal{E}$ is a distinct unordered pair that links two distinct vertices in the graph; in this case i and j. An edge models *bidirectional* influence or information flow. In this case, i and j are said to be *incident* with

e, and i is *adjacent* to j (and vice versa).

Sometimes, it is important to model *directed*, or *unidirectional* information flow. For example, if the agents have asymmetric sensing or communication capabilities, agent i may have information concerning agent j, but not the other way around. For this case, the mathematical abstraction used is a directed graph, or *digraph* [13]. Similar to a graph, a digraph $\mathcal{D} = (\mathcal{V}(\mathcal{D}), \mathcal{E}(\mathcal{D}))$ consists of a *vertex set* $\mathcal{V}(\mathcal{D}) = \mathcal{V} = \{1, \ldots, n\}$ and a *directed edge set* $\mathcal{E}(\mathcal{D}) = \mathcal{E}$. In this case, a directed edge is an ordered pair $(i, j) \in \mathcal{E}$, such that information flows from agent i to agent j. Vertex i is called the *tail* of the directed edge and vertex j is called the *head*. Vertex i is said to be an *in-neighbor* of agent j and agent j is an *out-neighbor* of agent i.

There are some obvious relationships between graphs and digraphs. Given a graph $\mathcal{G}$, we obtain an *orientation* of the graph by replacing each undirected edge with a (single) directed one. An orientation of $\mathcal{G}$ is a digraph $\mathcal{D}_{\mathcal{G}}$. Conversely, given a digraph $\mathcal{D}$, the *underlying graph* of $\mathcal{D}$, denoted $\mathcal{G}_{\mathcal{D}}$, is obtained by replacing all directed edges with undirected ones (and consolidating multiple edges created from any pairs of directed edges (i, j) and (j, i) in $\mathcal{E}(\mathcal{D})$ in the process). Clearly, there is no unique orientation of a nontrivial graph with nonempty edge set. But, potentially many digraphs have the same underlying graph. This is because graphs are a special subclass of digraphs (equivalent up to isomorphism), such that the symmetry property of the edge set holds. Namely, $(i, j) \in \mathcal{E}$ if and only if $(j, i) \in \mathcal{E}$. Because of this, throughout the rest of the section we introduce properties with respect to digraphs and provide the specific terminology with respect to graphs whenever appropriate. Furthermore, we refer to the digraph (or graph) and the network synonymously.

Next, we review certain structural properties of digraphs. We say that a digraph $\mathcal{D}_1 = (\mathcal{V}_1, \mathcal{E}_1)$ is a *subdigraph* of $\mathcal{D}$, written $\mathcal{D}_1 \subseteq \mathcal{D}$, if $\mathcal{V}_1 \subseteq \mathcal{V}$ and $\mathcal{E}_1 \subseteq \mathcal{E}$. Given two subdigraphs $\mathcal{D}_1$ and $\mathcal{D}_2$, the *union* $\mathcal{D}_1 \cup \mathcal{D}_2$ is the subdigraph with vertex set $\mathcal{V}_1 \cup \mathcal{V}_2$ and directed edge set $\mathcal{E}_1 \cup \mathcal{E}_2$. The subdigraph of $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ *induced* by the nonempty vertex set $\mathcal{V}' \subseteq \mathcal{V}$ is the subdigraph of $\mathcal{D}$ whose vertex set is $\mathcal{V}'$ and whose directed edge set contains only those directed edges of $\mathcal{E}$ whose head *and* tail are both in $\mathcal{V}'$. A digraph $\mathcal{D}' = (\mathcal{V}', \mathcal{E}')$ is *isomorphic* to $\mathcal{D}$ if there exists a bijection $\psi\colon \mathcal{V} \to \mathcal{V}'$ such that $(i, j) \in \mathcal{E}$ if and only if $(\psi(i), \psi(j)) \in \mathcal{E}'$. A sometimes useful fact is that every digraph on n vertices is isomorphic to a subdigraph of the *complete digraph*, $K_N = (\mathcal{V}_{K_N}, \mathcal{E}_{K_N})$, defined by $\mathcal{E}_{K_N} = \{(i, j) \in \mathcal{V}_{K_N} \times \mathcal{V}_{K_N} | i \neq j\}$, also referred to as the *complete network* (all-to-all communication or sensing).

To describe the information flow in the network, both local and nonlocal properties are useful. With respect to local properties, we consider the set of *in-neighbors* of agent j, defined by $\mathcal{N}_j^{\text{in}} = \{i \in \mathcal{V} | (i,j) \in \mathcal{E}\}$. The *in-degree* of j is denoted $d_j^{\text{in}} \triangleq |\mathcal{N}_j^{\text{in}}|$ (or just d_j), and the *minimum in-degree* of $\mathcal{D}$ is denoted $\delta^{\text{in}}(\mathcal{D})$. Whenever the underlying graph $\mathcal{G}_{\mathcal{D}}$ is considered, the *minimum degree* of $\mathcal{G}_{\mathcal{D}}$ is denoted $\delta(\mathcal{G}_{\mathcal{D}})$, with $\delta(\mathcal{G}_{\mathcal{D}}) \geq \delta^{\text{in}}(\mathcal{D})$. Similarly, the *maximum in-degree* of $\mathcal{D}$ and *maximum degree* of $\mathcal{G}_{\mathcal{D}}$ are denoted $\Delta^{\text{in}}(\mathcal{D})$ and $\Delta(\mathcal{G}_{\mathcal{D}})$, respectively, with $\Delta(\mathcal{G}_{\mathcal{D}}) \geq \Delta^{\text{in}}(\mathcal{D})$. Since agent j often has local feedback, the *inclusive in-neighbor* set, $\mathcal{J}_j^{\text{in}} = \mathcal{N}_j^{\text{in}} \cup \{j\}$, is also of interest. There are, of course, analogous definitions for *out-neighbors*, e.g., the *out-degree* of j is $d_j^{\text{out}} \triangleq |\mathcal{N}_j^{\text{out}}|$ and the *minimum out-degree* of $\mathcal{D}$ is $\delta^{\text{out}}(\mathcal{D})$. Other definitions of properties with regard to out-neighbors have similar changes in notation. Whenever the in-degree is equal to the out-degree for each node in the network, the digraph is said to be *balanced*. Clearly, all undirected graphs are balanced, and for each $j \in \mathcal{V}(\mathcal{G})$ of an undirected graph, $\mathcal{N}_j^{\text{in}} = \mathcal{N}_j^{\text{out}} \triangleq \mathcal{N}_j$, known simply as the *neighbor set*. In this case, $d_j \triangleq |\mathcal{N}_j|$ is the *degree* of $j \in \mathcal{V}(\mathcal{G})$.

In order to describe information flow across the network, we consider the following definitions. A *walk* is a finite non-null sequence of vertices $i_0, i_1, \ldots, i_k$ such that $(i_j, i_{j+1}) \in \mathcal{E}$, $j = 0, 1, \ldots, k-1$. The *end vertices* of this walk are i_0 and i_k (possibly identical). A walk with distinct (directed) edges is a *trail*. A trail in which the ends are identical but all other vertices are distinct is a *cycle*. A trail with distinct vertices is a *path*. A *trivial path* is a path with just a single vertex. Clearly, when discussing the underlying graph $\mathcal{G}_{\mathcal{D}}$, there may be walks, trails, cycles, and paths in $\mathcal{G}_{\mathcal{D}}$ that do not exist in $\mathcal{D}$. A vertex j is said to be *reachable* from vertex i if there exists a path starting at i and ending at j. Note that every vertex i is reachable from itself by the trivial path containing i.

Connectedness

We use the notion of path to define different forms of connectedness of graphs and digraphs. Given a vertex i in $\mathcal{V}(\mathcal{G})$, the *component* of i is the subgraph induced by the set of vertices reachable from i. Because of symmetry in undirected graphs, the component of j is equal to the component of i for any j in the component of i. Therefore, an undirected graph can be decomposed into its components. A graph $\mathcal{G}$ is said to be *connected* if it has a single component, and *disconnected* otherwise. On the other hand, connectedness of digraphs is more nuanced. In this case, we define *strong components* by considering the property of mutual reachability between vertices. Then a *strongly connected digraph* is a digraph with a single strong component. That is, for every $i, j \in \mathcal{V}(\mathcal{D})$, there exists a path starting at i and ending at j, and vice versa. If, on the other hand, for every pair $i, j \in \mathcal{V}$, there is possibly only a path from i to j or from j to i, then $\mathcal{D}$ is *unilateral*. If the underlying graph is connected, then $\mathcal{D}$ is *weakly connected*. Alternatively, if the underlying graph is disconnected, then $\mathcal{D}$ is *disconnected*.

A digraph has a *rooted out-branching* if there exists a node r, the root, such that every $i \in \mathcal{V}$ is reachable from r. Whether a digraph has a rooted out-branching is a particularly important property to information dissemination in the network because it implies that a distributed algorithm executed in the network admits information flow from the root to every other node in the network. If the network has only one rooted out-branching, or more importantly only one root for any rooted out-branching, then, depending on the distributed algorithm, the root may be the leader of the network. All unilateral digraphs contain a rooted out-branching; however, the converse is not true.

To measure the robustness and redundancy of information flow, we define a *vertex cut* as a set of vertices $\mathcal{K}$ such that the removal of $\mathcal{K}$ results in either a trivial digraph (a single vertex) or a disconnected digraph. The *vertex connectivity* $\kappa(\mathcal{D})$, or just *connectivity* κ, is the smaller of the following two quantities: (i) the size of a minimal vertex cut, or (ii) $|\mathcal{V}| - 1$. A digraph is said to be *k-vertex connected*, or simply *k-connected*, if $\kappa(\mathcal{D}) \geq k$. A simple consequence of defining connectivity in this manner is $\kappa(\mathcal{D}) = \kappa(\mathcal{G}_{\mathcal{D}})$ [69].[4]

Algebraic Graph Theory

Algebraic graph theory elucidates properties of graphs and digraphs by examining algebraic objects associated with graphs and digraphs, such as polynomials and matrices. Of particular interest are some of the matrices associated to graphs and digraphs, such as the adjacency matrix, degree matrix, incidence matrix, and Laplacian [73, 36]. Using these matrices, the network feedback of many multi-agent network control laws can be mathematically modeled. Assuming linear agent dynamics, the closed-loop system can then be analyzed using techniques of linear system theory. For this, the

[4] In [69], this form of connectivity is defined as $\kappa_1(\mathcal{D})$ and other forms of connectivity in digraphs are studied (most notably strong connectivity). For our purposes, the definition given here suffices.

spectra of these matrices are of special interest. In what follows, we will define these objects with respect to a *weighted digraph*, $\mathcal{D}_w = (\mathcal{V}, \mathcal{E}, w)$, which has, in addition to a vertex set and directed edge set, a *weight function* $w\colon \mathcal{E} \to \mathbb{R}$. In this case, a *weight* $w_{ij} \triangleq w(e) \in \mathbb{R}$ is associated to each directed edge $e = (i, j) \in \mathcal{E}$. For the purposes of defining the matrices, we say that a digraph $\mathcal{D}$ is the specific weighted digraph in which $w\colon \mathcal{E} \to \{1\}$; that is, all directed edges have unit weight.

The *weighted adjacency matrix*, $A_w(\mathcal{D}_w) = [a_{ij}]$, associated with weighted digraph $\mathcal{D}_w$ is the $|\mathcal{V}| \times |\mathcal{V}|$ matrix defined by[5]

$$a_{ij} = \begin{cases} w_{ji} & (j,i) \in \mathcal{E}; \\ 0 & (j,i) \notin \mathcal{E}. \end{cases} \tag{1}$$

In this case, along any given row i of the weighted adjacency matrix, nonzero entries may only occur in the columns corresponding to in-neighbors of node i. For digraphs, we define the *adjacency matrix*, $A(\mathcal{D}) = [a_{ij}]$, or just A, as in (1), but with $w_{ji} \equiv 1$, for all $(j, i) \in \mathcal{E}$. Observe that for undirected graphs, $A(\mathcal{G}) = A^{\mathsf{T}}(\mathcal{G})$; that is, the adjacency matrix is a symmetric matrix. Note that the weighted adjacency matrix of an undirected weighted graph is not necessarily symmetric.

The *weighted in-degree matrix* D_w^{in} of a weighted digraph $\mathcal{D}_w$ is the diagonal $|\mathcal{V}| \times |\mathcal{V}|$ matrix with entries along the main diagonal given by

$$[D_w^{\text{in}}]_{ii} = \sum_{j \in \mathcal{N}_i^{\text{in}}} w_{ji}, \quad i \in \{1, 2, \dots, n\}. \tag{2}$$

Observe that $A_w(\mathcal{D}_w)1_n = \textbf{Diag}(D_w^{\text{in}})$, where 1_n is an $n \times 1$ column vector of ones and $\textbf{Diag}(M)$ is the main diagonal of an $n \times n$ matrix M represented as an $n \times 1$ column vector. Similarly, the *weighted out-degree matrix* D_w^{out} is a diagonal matrix with entries

$$[D_w^{\text{out}}]_{ii} = \sum_{j \in \mathcal{N}_i^{\text{out}}} w_{ij}, \quad i \in \{1, 2, \dots, n\}.$$

In this case, $1_n^{\mathsf{T}} A_w(\mathcal{D}_w) = \textbf{Diag}(D_w^{\text{out}})^{\mathsf{T}}$. For an undirected graph, $D^{\text{in}} = D_w^{\text{out}} \triangleq D$ is the *degree matrix* of $\mathcal{G}$.

[5]We define the weighted adjacency matrix as the transpose of the usual weighted adjacency matrix defined in most of the literature in graph theory. This is typical in the area of multi-agent networks because local node strategies that use linear combinations of their in-neighbors' states or outputs can be conveniently written as a left multiplication of a weighted adjacency matrix defined in this manner.

Let $e_1, e_2, \ldots, e_{|\mathcal{E}(\mathcal{D})|}$ denote the edges of a digraph $\mathcal{D}$. Then the *incidence matrix* $M(\mathcal{D})$ of $\mathcal{D}$ is the $|\mathcal{V}| \times |\mathcal{E}|$ matrix $M(\mathcal{D}) = [m_{ij}]$ defined by

$$m_{ij} = \begin{cases} -1 & \text{if } i \text{ is incident with the tail of } e_j \text{ in } \mathcal{D}; \\ 1 & \text{if } i \text{ is incident with the head of } e_j \text{ in } \mathcal{D}; \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Likewise, the incidence matrix of an undirected graph $\mathcal{G}$ may be defined by first fixing an orientation of the graph $\mathcal{D}_{\mathcal{G}}$. Then, $M(\mathcal{D}_{\mathcal{G}})$ is defined as in (3) by substituting $\mathcal{D} = \mathcal{D}_{\mathcal{G}}$.

Finally, the *(in-degree) weighted Laplacian* is the $|\mathcal{V}| \times |\mathcal{V}|$ matrix defined by[6]

$$L(\mathcal{D}_w) = D_w^{\text{in}} - A_w(\mathcal{D}_w). \tag{4}$$

A direct consequence of this definition is that $1_n \in \mathscr{N}(L(\mathcal{D}_w))$, where $\mathscr{N}(L(\mathcal{D}_w))$ is the *null space* of $L(\mathcal{D}_w)$. Therefore, zero is always an eigenvalue corresponding to eigenvector 1_n of $L(\mathcal{D}_w)$. Likewise, we may define the *(out-degree) weighted Laplacian* $L^{\text{out}}(\mathcal{D}_w)$ as

$$L^{\text{out}}(\mathcal{D}_w) = D_w^{\text{out}} - A_w^{\mathsf{T}}(\mathcal{D}_w),$$

which again always has zero as an eigenvalue corresponding to eigenvector 1_n. However, the structure of the (in-degree) weighted Laplacian is more useful to describe linear consensus protocols in multi-agent networks, and therefore, we will focus our attention on it.

Whenever we consider bidirectional communication, the *graph Laplacian*, or just *Laplacian*, $L(\mathcal{G})$ is of interest (and is defined as in (4) by $L(\mathcal{G}) = D - A$). In this case, $L(\mathcal{G})$ is a real symmetric matrix, so the eigenvalues are real. In fact, $L(\mathcal{G})$ is symmetric and positive semidefinite, so that all eigenvalues are nonnegative [73]. It is advantageous to label the eigenvalues of $L(\mathcal{G})$ in nondecreasing order by

$$\lambda_1(\mathcal{G}) \leq \lambda_2(\mathcal{G}) \leq \cdots \leq \lambda_n(\mathcal{G}). \tag{5}$$

As described above, $\lambda_1(\mathcal{G}) \equiv 0$. It turns out, the multiplicity of the zero eigenvalue is equal to

[6]Note that because we define the weighted adjacency matrix as the transpose of the usual weighted adjacency matrix, the weighted Laplacian considered here is also the transpose of the usual weighted Laplacian.

the number of components of $\mathcal{G}$ [73]. For connected graphs, $\lambda_2(\mathcal{G}) > 0$, and is called the *Fiedler eigenvalue* or the *algebraic connectivity* of the graph. This is because the magnitude of $\lambda_2(\mathcal{G})$ is directly related to the sparsity of the graph. That is, $\lambda_2(\mathcal{G})$ is small for sparse graphs and large for dense graphs.

2.2 Passivity and Wave Variables

The purpose of this section is to provide a brief introduction to passivity and wave variables, and then present a review of some of the important works that are foundational to the passivity deployment protocol discussed in Chapter III. Toward this end, we first describe passivity and the wave variable formalism from a high-level perspective in Section 2.2.1. Then, we describe the related works from *teleoperation*, *virtual environments*, and *haptic interfaces* in Section 2.2.2. Finally, we discuss related works from *networked control systems* (NCS) in Section 2.2.3.

2.2.1 Introduction to Passivity and Wave Variables

Passivity is a system theoretic concept that is useful for constructive nonlinear control design and analysis [48, 96]. The basic idea is that passive systems may store or dissipate "energy", but never create it. That is, when measuring the inputs and outputs of the system, the net energy generated by the system, as seen from an external perspective (i.e., looking only at the inputs and outputs), must never exceed the cumulative sum of stored and input energy. Thus, whether a given system is passive or not often depends on how the system boundaries are defined. Moreover, a lack of passivity in one part of a system may be compensated by additional dissipativity (also referred to as additional passivity) elsewhere in the system. Because these intuitive properties are defined from an input-output perspective, passive systems exhibit profound robustness to unmodeled disturbances and uncertainties, including those arising from implementation effects [62].

Passivity has several attractive features. First, it is a sufficient condition for stability. Second, it applies to both linear and nonlinear systems. Third, it has several nice compositional properties. Namely, parallel and negative feedback configurations of passive systems are also passive (c.f., Figures 1 and 2, respectively). Furthermore, multiplying the inputs of a passive system by a matrix and then multiplying the outputs by the transpose of the same matrix also maintains passivity (c.f., Figure 3). Together, these features facilitate conclusions concerning global passivity and stability under appropriate compositions of subsystems based only on the passivity of the subsystems.

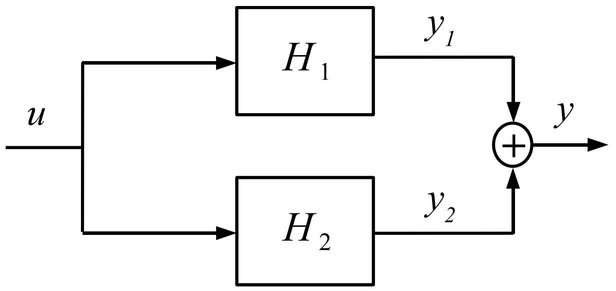


Figure 1: Parallel interconnection of passive systems is passive.

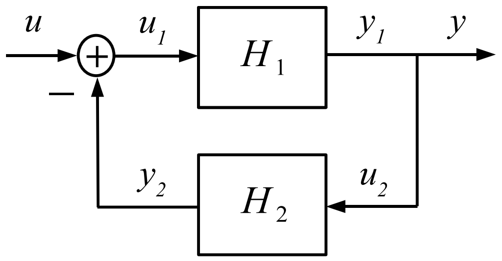


Figure 2: Negative feedback interconnection of passive systems is passive.

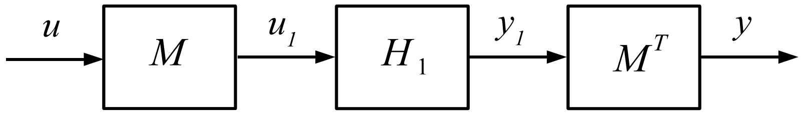


Figure 3: Multiplication of inputs and outputs by a matrix and its transpose maintain passivity.

Because of the beneficial qualities and broad-reaching underlying intuition behind passivity, oftentimes various approaches to system-theoretic problems are unified under passivity-based techniques. For example, it has been shown that various stability results dealing with congestion control in the physical layer of computer networks may be unified under passivity theory [203]. It turns out that many cooperative control techniques may also be unified under passivity theory. For a comprehensive treatment of passivity in cooperative control design – especially focusing on how passivity enables adaptive designs – see [11].

Another useful tool – complementary to passivity – is the *wave variable formalism*. The main idea behind the wave variable formalism is to transform two variables of the same dimension, often the input and output variables of a system, so that the resulting variables are linear combinations of the original variables. The canonical example is the *scattering transformation*, which transforms the variables into transmitted and reflected waves, and thus are called *wave variables*. Whenever the input and output variables of a system are effort and flow variables,[7] the scattering transformation has a powerful intuition. In this case, the physical unit of the square of each wave variable is Watts, and the direction of transmission of each wave variable represents *power flow*. Then, the difference of the transmitted and reflected waves is the *net power flow*. Even whenever the physical units do not admit this interpretation of power flow, the notion of *mathematical power flow* allows the intuition to be generalized. Because the energy of a wave is conserved when it is delayed by a constant value, wave variables are well-suited to passivity-based techniques while providing additional robustness to delays. As with passivity, wave variables have been used to incorporate resilience to delays in multi-agent networks [201, 35].

2.2.2 Teleoperation, Virtual Environments, and Haptics

Teleoperation is the control of a remote robot using a local robotic device, possibly a simple remote control. Oftentimes, in practice, the only feedback available to the operator of the local robotic device is visual. The goal within the teleoperation literature is to design interfaces with the remote robot that provide richer forms of feedback than just visual feedback. The premise is that multisensory feedback provides the operator with a greater sense of control of the remote robot, and therefore improves the precision and dexterity of the operator when performing tasks remotely. Because remote surgery, or *telesurgery*, is one of the focal applications of teleoperation, improving the surgeon's ability to reliably and safely operate the surgical robot is of utmost importance.

In order to train operators in teleoperation applications, such as telesurgery or UAV remote piloting, *virtual environments* are often used. In this case, the remote robot is replaced by virtual interactions between objects in a simulated environment. As in teleoperation, it is beneficial to have multisensory feedback.

One way to provide additional sensory information is through *haptic technology*, or *haptics*,

[7] *Effort* and *flow* variables are a generalization of force and velocity, and provide common terminology across multiple physical domains, i.e., translational mechanics (force and velocity), rotational mechanics (torque and angular velocity), electrical (voltage and current), magnetic (current and voltage), and hydraulic (pressure and volume flow). In all cases, the product of effort and flow is power.

which provides tactile feedback, and may be used when interfacing with a remote robot as in teleoperation, or with a virtual environment. In order for the haptic device to provide useful tactile information, it is important for the operator to "feel" in some sense the forces exerted on the remote robot (or within the virtual environment). One way to do this is by closing the loop on the forces acting on the remote robot (or actuate virtual forces from the virtual environment). Teleoperation with force feedback is known as *bilateral teleoperation*. Whenever a haptic device is used to interface with a virtual environment, it is referred to as a *haptic interface*.

For both bilateral teleoperation systems and haptic interfaces, there is a single interface between the operator and the environment (whether it be remote or virtual). In either case, information and energy are exchanged bidirectionally. This property is conjugate to ideas from classical electrical network theory, in which subsystems interact through "ports" that share effort and flow variables when connected. This port-based perspective is well-suited to both bilateral teleoperation systems and haptic interfaces because, as illustrated in Figure 4, in both cases the operator manipulates the local device (effectively applying input velocities, f_o) and the environment reciprocates via reactive forces and torques, e_e, which are transmitted back to the local device, and in turn the operator, e_o. Imposing an order of actions and reactions in this manner is referred to as imposing *causality* on the network. The causality exhibited from the view of the operator in this example is *impedance causality*, in which the input to the local device is a flow variable and the output is an effort variable. Alternatively, effort input and flow output is known as *admittance causality*.

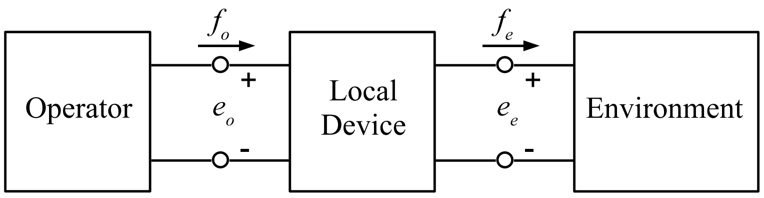


Figure 4: Port-based view of bilateral teleoperation and haptic interfaces.

2.2.2.1 Haptic Interfaces

Haptic interfaces connect an operator with a virtual environment and provide force feedback whenever the operator encounters virtual objects. This is done through actuators in the haptic interface. A major issue in the design of haptic interfaces is to provide the operator with realistic force feedback for a variety of virtual objects while maintaining stable behavior. Unstable behavior such as violent vibrations or divergent motions of the haptic device typically occur whenever the operator encounters virtual walls, where the response must be abrupt and hard. In such a setting, passivity can be used to maintain stable behavior. However, passivity can sometimes be overly conservative and provide degraded performance (unrealistic feedback) of the haptic interface.

One way to get around the conservativeness of an overarching passivity approach is to monitor the input and output variables for their energy content and only dissipate energy whenever the cumulative energy indicates active (nonpassive) behavior. This is the technique used in the haptic interface design of [76]. In more detail, a *passivity observer* is designed for the network to monitor the passivity of the haptic interface. The passivity observer is derived using *Tellegen's Theorem*, which states that the net power created in a network of elements satisfying Kirchoff's laws may be observed by examining only the unterminated ports in the network (i.e., those external to the network). The notation in the network is defined in such a way that energy generated by the system is negative.

In order to compensate for active behavior, a *passivity controller* dissipates energy whenever the cumulative energy is negative. The implementation of the passivity controller depends on whether the virtual environment has impedance causality or admittance causality. If it has impedance causality, then the passivity controller is placed in series; if it has admittance causality, the passivity controller is placed in parallel. With respect to Figure 4, the passivity controller is placed between the local device and the environment. Whenever the local device and operator exhibit dissipative behavior, it is helpful for performance to dissipate less energy, and therefore the threshold for the activation of the passivity controller is changed from zero to a negative value. This value depends on the damping of the load (haptic device and operator) and parameter estimation may be used to determine it. Together, the passivity observer and passivity controller can be implemented in software and run in real time. Some problems that can degrade the performance of the passivity controller include actuator saturation (which prevents dissipation of all the excess energy in one time step) and noise magnification whenever the velocity is small in magnitude.

2.2.2.2 Bilateral Teleoperation

Bilateral teleoperation is teleoperation with force feedback. A bilateral teleoperation system, like a haptic interface, should be designed in such a way so as to provide the operator with realistic force feedback in a stable manner. However, bilateral teleoperation systems have an additional source of instability caused by the network through which the operator controls the remote robot. The network is a source of delays and data loss, which can destabilize even passive systems. Therefore, the wave variable formalism has been combined with passive techniques to address delays [146].

More recently, performing bilateral teleoperation over the internet has been considered [18]. In this work, it is shown that passivity in the communication channel can be maintained with both fixed and time-varying delays provided the controller properly manages dropped data and reordering of data. This idea is extended to both continuous-time and discrete-time models of the teleoperation system by using the scattering transformation with communication management modules (CMMs) in [34]. The CMMs are introduced to properly handle the wave variables output from the network so as to maintain passivity. The CMM consists of *interpolation components* and *queue management*. The interpolation components allow for compensation from packet loss and time-varying delays. The queue management helps to mitigate bursty network behavior by compressing data during periods of decreasing delay and expanding the data during periods of increasing delays and packet loss. In this manner, the queue management prevents both extremes; namely, empty queues and queue overflow.

Port-Hamiltonian Systems

Port-Hamiltonian systems are generalizations of port concepts from electrical network theory. The port-Hamiltonian formalism is very expressive and is well-suited for modeling teleoperation systems and haptic interfaces. This is because port-Hamiltonian systems are inherently passive and are easily augmented with wave variables [48]. Recently, it has been shown how to model a sampled data system using the port-Hamiltonian formalism [175]. In [175] it is shown that the interconnection between discrete and continuous port-Hamiltonian systems can be made to preserve passivity. An important factor for preserving passivity in a sampled data system is to ensure discrete energy leaps are handled properly. The developed port-Hamiltonian sampled data theory is then applied to telemanipulation and haptic interfaces to show that passivity is maintained even with dropped data and time-varying delays. This is done by construction rather than requiring the passivity observer and passivity controller as in [76].

2.2.3 Networked Control Systems

Networked control systems (NCS) are a class of control systems where the sensors, actuators, and controllers are physically distributed and connected over a network. Unlike supervisory control and data acquisition (SCADA) systems, which have hierarchical control structures for coordinating local controllers over a network, NCS are flat, and real-time control is done over the network. Therefore, NCS are susceptible to network delays and data loss. For this reason, passivity and wave variables have been used in the design of NCS.

The first use of the scattering transformation in NCS is given in [134]. It is shown that for a linear time-invariant (LTI) single-input, single-output (SISO) plant and controller feedback configuration, the scattering transformation results in a delay independent stability approach that can be used to ensure stability (asymptotic stability) of the closed loop system. The stability (asymptotic stability) result holds even for nonpassive plant and controller as long as a positive real (strictly positive real) condition on the transfer function relationship, denoted $K(s)$, induced by the scattering transformation and loop delay is satisfied. Design considerations are discussed and it is shown that it is important for $K(s)$ to be near unity over a large range of frequencies so as to minimize the sensitivity with respect to time delay. Furthermore, if $K(0) = 1$, the steady-state performance is the same as without delays and without the scattering transform. In fact, if $K(s) = 1$ then the performance is identical to the zero delay case with a constant time shift given by the forward time delay (i.e., the delay from controller to plant). The scattering approach is compared to delay dependent approaches: the Smith predictor and PI control, and is shown to be much less sensitive to delay. Moreover, it is shown through simulation example that the sufficient stability condition is not overly conservative. Finally, the scattering approach is used with a single degree of freedom pendulum and shown to produce good performance results in the presence of network delays.

In [33], passivity of the traditional feedback configuration of two continuous-time systems is explored whenever constant and time-varying delays exist in the feedback loop. The author uses a definition of passivity with the internal storage function and shows that with constant delays, the overall configuration is always passive. For the case of increasing delays, small gains bounded by the time derivative of the time-varying delays are used to keep the configuration passive. The time delays are assumed to change no faster than real time (a causality condition). Even using wave variables in continuous time, this approach requires the gains in the feedback loop to maintain passivity with increasing delay, but cannot handle decreasing delays.

A generalization of the scattering transformation is introduced in [79] for input-feedforward output-feedback-passive (IF-OFP) nonlinear systems (which are essentially conic systems [207]). Here it is desired to maintain L_2-stability[8] in the presence of arbitrarily large and unknown constant time delays. The approach allows for separation of concerns as follows. The controller may be designed beforehand to meet very aggressive performance criteria under the assumption of no delays. Then the generalized transform is introduced to ensure stability of the system in the presence of constant delays and can be designed with low sensitivity to delay. By limiting the class of transforms, a sufficient condition for stability is given. For SISO LTI systems, a necessary and sufficient condition for stability is provided. The approach is compared to the traditional small-gain approach and to the design without the generalized transform in simulations. The approach is shown to meet much better performance criteria while maintaining stability in the presence of constant delays.

In [105], a similar approach to [79] is given for maintaining passivity. In this case, a passive sampled data approach is used for controlling a continuous-time plant (robotic arm) over a discrete-time network with a discrete-time controller. In order to do this, the scattering transformation is used to produce continuous-time wave variables and passive sample-and-hold device is introduced to transform the wave variables into discrete-time in a manner that preserves passivity. This passive sample-and-hold borrows ideas from the sampled data approach of [175] (i.e., the port-Hamiltonian approach). As in [175], the approach maintains passivity even with certain types of time-varying delays and data loss. The approach is later generalized to conic systems in multirate networks in [106].

Another approach to passive control of NCS using wave variables is given by Kottenstette and Antsaklis in [101]. This approach seeks to maintain passivity in NCS with time-varying delays and data loss using component based design. The goal is to be able to add and remove elements (plants

[8]See Chapter 3 for a definition of l_2-stability, which is the discrete time version of L_2-stability. L_2-stability is a form of bounded input, bounded output stability applicable to nonlinear systems

and controllers) from the NCS while maintaining passivity and L_2-stability. To do this, a *power junction* is introduced as a hub for the network, and interfaces between the plants and controllers within the star network. The power junction is in general a dissipative element (but can be lossless) and is essentially a wave digital filter [62], designed to guarantee passivity of the network. More specifically, the power junction takes as input wave variables and generates the same number of output wave variables in such a way that the output power is never more than the input power (recall, the power of a wave variable is given by its square). In [101], an *averaging power junction* is introduced, which averages the input power and generates all output waves equal to the average input power. This lossless power junction is shown to maintain passivity in the network. It is also shown how to distribute the functionality of the power junction over all nodes in a ring network configuration. An extension of this work is given in [103], which provides a refined averaging power junction that exhibits better performance in the simulations provided.

A couple of different implementations of power junctions are explored in [102]. Two lossless power junctions, the averaging power junction and a daisy-chain power junction (called a consensus power junction in the paper), are compared using simulations. Also, the behavior of the power junctions is explored using steady-state analysis in order to relate the inputs and outputs. The steady-state results are provided for both the average and daisy-chain power junctions.

2.3 Multi-Agent Networks

Research in networked multi-agent systems, from an engineering perspective, may be organized based on the *group objective* of the multi-agent network. Examples of different group objectives include consensus, pattern formation, formation control, synchronization, coordinated path tracking, flocking, foraging, and cooperative load transport. As we will see, consensus is related in one way or another to nearly all of these objectives. The specific taxonomy of these group objectives on which this section is organized is given in Figure 5. In the figure, numbered and unnumbered section or subsection titles are written in boldface font, with the specific section numbers given if applicable. Within the tree, child nodes are contained within sections given by parents.

Along with the group objectives given in Figure 5, there are several important aspects of multi-agent networks that help to further delineate the related work. These attributes are: diversity of the agents, agent dynamics, the level of uncertainty with respect to the agent behavior, assumptions

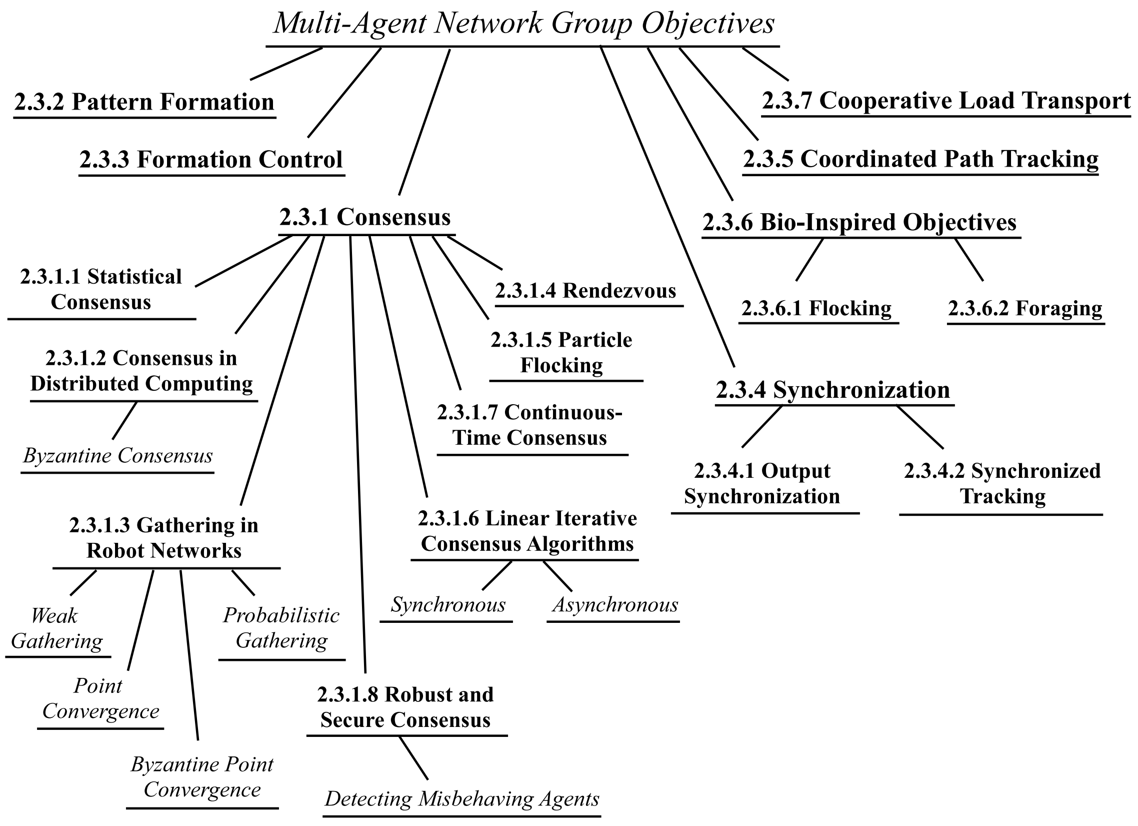

Multi-Agent Network Group Objectives
2.3.2 Pattern Formation
2.3.3 Formation Control
2.3.1 Consensus
2.3.7 Cooperative Load Transport
2.3.5 Coordinated Path Tracking
2.3.6 Bio-Inspired Objectives
2.3.1.1 Statistical Consensus
2.3.1.4 Rendezvous
2.3.6.1 Flocking
2.3.6.2 Foraging
2.3.1.2 Consensus in Distributed Computing
2.3.1.5 Particle Flocking
2.3.4 Synchronization
Byzantine Consensus
2.3.1.7 Continuous-Time Consensus
2.3.4.1 Output Synchronization
2.3.4.2 Synchronized Tracking
2.3.1.3 Gathering in Robot Networks
2.3.1.6 Linear Iterative Consensus Algorithms
Weak Gathering
Probabilistic Gathering
Synchronous
Asynchronous
Point Convergence
2.3.1.8 Robust and Secure Consensus
Byzantine Point Convergence
Detecting Misbehaving Agents

Figure 5: Taxonomy of multi-agent network related work.

made concerning the network, and the design or analysis methodology.

- **Agent Diversity:** Are the agents identical or not? Is there a leader, or is the multi-agent network leaderless?

- **Agent Dynamics:** How are the dynamics of the individual agents modeled? Do the agents evolve continuously or in discrete steps? If they are discrete, are they modeled by a discrete-time dynamical system or a finite automaton? Are they linear or nonlinear? If they are linear, do they have first order or higher order dynamics? Are they deterministic or nondeterministic?

- **Agent Duplicity:** Does the behavior of the agents adhere strictly to their dynamical models? Or, is model uncertainty a possibility? What about faults, failures, or security breaches? Is it possible that some of the agents are adversaries?

- **Network Assumptions:** Is the network continuous or discrete? Is is synchronous or asynchronous? Is it bidirectional or directed? Is it fixed or is it dynamic, with time-based or state-based switching? Is it stochastic, lossy, delay-ridden, or ideal?

- **Design or analysis methodology:** How is the distributed control protocol designed or analyzed? Does it use algebraic graph theory, passivity, Lyapunov techniques, contraction theory, or some other techniques?

Each contribution in the literature is categorized by each of these aspects of the multi-agent network, but in some cases combines multiple components of a given aspect. For example, [37] looks at synchronized trajectory tracking as the group objective, which is a special case of coordinated path tracking (in this case all the paths are the same). All of the agents are cooperative and modeled as Lagrangian systems coupled over an ideal continuous (synchronous) network. The authors prove synchronization with and without tracking, so the work could be categorized under both coordinated path tracking and synchronization. To further complicate things, the authors consider both identical and non-identical agent dynamics and use both contraction theory and algebraic graph theory (by introducing the modified Laplacian) to prove exponential convergence results.

Because of the unavoidable multiplicity often seen in categorizing the related work, we organize this section by looking at different objectives, and discuss different contributions that address these objectives. Keeping in mind the different attributes helps not only in distinguishing the individual contributions, but also provides a means to characterize research initiatives from different communities.

Multi-agent Group Objectives

There are many possible group objectives of a multi-agent network. Here we consider several possible group objectives, discussing each in turn. Many objectives not directly described here can be reformulated to fit one or more of the objectives discussed. Moreover, some objectives are used as intermediate steps or components of a more complex objective. In particular, consensus is often used in this capacity. In what follows, we discuss consensus, synchronization, flocking, foraging, coordinated path tracking, cooperative load transport, and distributed estimation.

2.3.1 Consensus

Reaching consensus is fundamental to group coordination, and involves reaching agreement on a quantity of interest. This concept is deeply intuitive, yet imprecise. Hence, there are several variations on how consensus problems are defined. Different variations on consensus problems are illustrated in Figure 6. The first variation concerns whether the agreement quantity is constrained or not. At one extreme, consensus may be *unconstrained*, and there is no restriction on the agreement quantity. In other cases, consensus may be *partially constrained* by some rule or prescribed to lie in a set of possible agreement values that are in some way reasonable to the problem at hand. At the other extreme, consensus may be *function constrained*, or χ-constrained, in which case the consensus value must satisfy a particular function of the initial values of the agents [41, 178]. Examples of χ-constrained consensus include average consensus [156, 166, 205], power-mean consensus [15], and geometric-mean consensus [153].

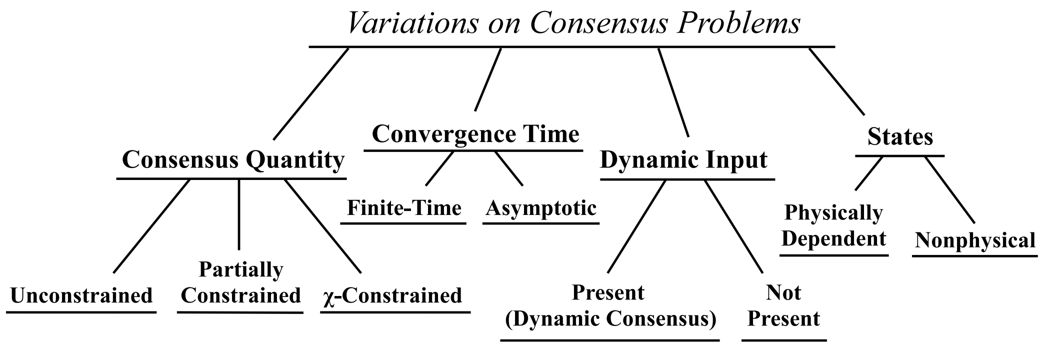


Figure 6: Variations on consensus problems.

Another issue is whether the agents must reach consensus in finite-time or asymptotically. In the distributed computing literature, finite termination is required [130], whereas in the cooperative control literature, asymptotic consensus is more common [168, 152, 167]. This discrepancy is due to the assumptions placed on the individual agents. In control applications, it is typical that the agents have continuous dynamics modeled by ordinary differential equations with certain continuity and smoothness properties, which naturally lead to asymptotic results [137]. Without imposing any discrete modes [121], discontinuities, or nonsmoothness [39] on the continuous dynamics, the best that can be hoped for is exponential convergence. On the other hand, a few recent promising results use nonsmooth techniques to achieve finite-time convergence [40, 41, 84].

A third issue is whether the agreement quantity is fixed or dynamically changing. In most cases, consensus is reached on a specific quantity of interest that is fixed with time, or is a limit point to which the consensus process converges. However, in distributed estimation, and more generally sensor network applications, the measured or estimated quantity may change dynamically over time. We refer to this as *dynamic consensus*; otherwise we say the consensus is *static*. Each agent (or sensor) has a measurement input (possibly multidimensional). The goal of the consensus process is to reach agreement dynamically either on the measurements directly or on quantities being estimated dynamically from the measurements. The feasibility of achieving dynamic consensus in this case is dependent both on the frequency spectrum of the particular measurement signal and on the time constants, or modes, of both the underlying process being estimated and the multi-agent network (sensor network). The Fourier transform of the multi-agent network, when the agents are linear systems and communication is bidirectional, is a function of the spectrum of the graph Laplacian.

Finally, an important distinction is whether the consensus process directly affects a physical process, or not. Obviously, for the former, the agents must have state variables corresponding to some physical process that are manipulable and dependent on the consensus process. In this case, the consensus protocol is a control law, directly affecting a subset of the physical states of the agents. We refer to this type of consensus as *physically dependent*, and is a concept that is important to most cooperative control problems, such as *rendezvous* [42, 123, 124] or *gathering* (as it is known in the mobile robotics community) [182, 163, 1], *flocking* [151], and *synchronization* [91, 35, 190]. Examples where the quantities are not manipulable physical state variables include consensus of sensor measurements [206, 173], probability distributions [47], or decision values [130].

2.3.1.1 Statistical Consensus

Some of the earliest formal study of consensus lies in management science [47], in which the individual "agents" each have a probability distribution for the unknown value of a parameter. Consensus in this case refers to an alignment of the distributions, resulting in *statistical consensus*. By consolidating the distributions to single values, statistical consensus can also handle consensus on a set of values as a special case. Statistical consensus is closely related to *belief consensus*, which is used for distributed hypothesis testing in [153].

2.3.1.2 Consensus in Distributed Computing

Consensus problems also have a rich history in distributed computing [130]. In this case, the "agents" are stationary processors modeled by finite automata, which communicate over a network also modeled by a finite automaton [130]. Since neither the agents nor the network are subject to physical constraints (i.e., no physical dynamics), the problem of reaching consensus on a set of values is trivial in the ideal setting (a simple flooding algorithm does the trick). Therefore, in the distributed computing literature, consensus problems are defined with respect to faults and failures of the agents and with respect to unreliable communication networks. Other ways additional complexity is added to the problem is by considering different timing models (i.e., levels of synchronicity of the network) and by either requiring determinism or allowing for randomness in the consensus algorithms.

The consensus problems tend to be defined with conditions on agreement, validity, and termination. The validity condition effectively restricts the agreement value, thereby resulting in at least partially constrained, and in some cases function constrained consensus problems. Within these well-defined problem settings, computer scientists commonly provide impossibility results with regard to reaching consensus under the specific conditions of a particular problem. For example, the *coordinated attack problem* is a binary consensus problem in which all of the agents must decide to commit or abort, and they coordinate in an unreliable network in which link failures are possible [130]. There are deterministic [74] and nondeterministic [198] forms of the problem, and in the deterministic case, it is proven that no algorithm can always solve the coordinated attack problem [74].

There are various failure models used for agent (processor) failures. Two of the most popular models are *stopping failures* and *Byzantine failures* [130]. Stopping failures occur whenever the agent simply stops, and models a processor crash. Byzantine failures, on the other hand, may exhibit arbitrary behavior within the limitations of the automaton models (both the processor automaton and the network automaton; at least the components of the network automaton to which the faulty processor has access). In particular, Byzantine faulty processors may send different messages to different processors in the network. Therefore, worst case executions must be considered. Because of this, processors undergoing Byzantine failures are adversarial in nature.

Byzantine Consensus Problems

The earliest work on reaching consensus in the presence of Byzantine faulty processors can be found in [161] and [110], where the Byzantine agreement problem is introduced. The term "Byzantine" is coined in [110] to describe the problem with respect to generals in ancient Byzantium. The generals communicate by sending messengers and want to decide on a plan of action on whether to attack or retreat. But, a subset of the generals may be traitors, and the loyal generals want to devise a plan in order to reliably reach an agreement on the plan of action that is not subverted by the traitors. Hence, [110] addresses a binary Byzantine consensus problem that is partially constrained. All generals are able to send messengers to all other generals (i.e., the communication graph is complete). It is shown that the maximum number of traitors F, which is an upper bound on the total number of traitors assumed *a priori*, must be less than one-third the total number of generals n, or $n > 3F$. Thus, any algorithm which successfully solves this consensus problem must satisfy this criterion. In [161], the authors again demonstrate this necessary bound on the number of Byzantine processors for the case whenever the values are not necessarily binary. Both [161] and [110] use exponential data structures in the algorithms proposed for the problem. These algorithms can handle the case $n = 3F + 1$, so the condition $n > 3F$ is also sufficient.

The necessary and sufficient condition of $n > 3F$ derived for complete networks in [161, 110] is shown to also hold, more generally, in synchronous networks with less connectivity in [50]. Here it is shown that it is also necessary that the connectivity κ of the network be more than twice the number of Byzantine processors, or $\kappa > 2F$. The connectivity condition is also shown to be sufficient by using a majority voting algorithm that succeeds with $\kappa = 2F + 1$.

The Byzantine agreement problem is relaxed in [51] to allow the decision values of the nonfaulty processors to differ within any fixed margin of error. In this case, the values are real numbers. Given any arbitrarily small $\epsilon > 0$, any pair of nonfaulty decision values must lie within ϵ of each other. In addition to this agreement condition, the validity condition requires that the nonfaulty decision values lie within the range of the initial values of the nonfaulty processors (the *range* of a multiset of values is defined to be the smallest interval containing all the values). These conditions, along with the termination condition (all nonfaulty processors eventually terminate), comprise the *Byzantine approximate agreement problem*.

In [51], both asynchronous and synchronous forms of the Byzantine approximate agreement problem are considered. In fact, one of the reasons [51] considers approximate agreement is because of an impossibility result showing that guaranteed termination with exact agreement is not possible in asynchronous networks even with a single stopping failure [64]. However, if termination with probability one is considered, then exact agreement is possible [16, 29]. As an additional motivation, approximate agreement is also useful for clock synchronization [129]. For a comprehensive survey of Byzantine consensus problems in distributed computing, along with applications, see [14].

The algorithms introduced in [51] for the Byzantine approximate agreement problem are designed for complete networks. The algorithms use so-called *approximation functions*, which iteratively shrink the range of values of nonfaulty processors in each round of the algorithms. (This idea of iteratively moving toward consensus is analogous to the linear iterative strategies discussed in Section 2.3.1.6.) The approximation functions of [51] use a composition of *reduce functions*, each of which removes the largest and smallest elements of a multiset. Another function used is the *select function* with respect to positive integer $k > 0$, which chooses the smallest element and every kth element thereafter (in an ordered list of sorted values in ascending order).

The basic idea of the synchronous algorithm, which we refer to as the *ConvergeApproxAgreement* algorithm (as in [130]), is for every node to share its own value with every other node in the network (complete network) at the beginning of each round. The node's own value is combined with the received values in a multiset and sorted in ascending order. Since F is an upper bound on the number of traitors, the largest and smallest F values are removed from the multiset (with ties broken arbitrarily). Then, the select function with respect to F is applied. Finally, the selected values are averaged to obtain the new value to be used in the subsequent round. This choice of operations for the approximation function is shown to result in an optimal convergence rate for algorithms of this type (that is, no other approximation function achieves faster convergence). The termination criterion is derived by using the range of initial values from the first round, the tolerance ϵ, and the known rate of convergence. The synchronous algorithm can handle the case $n = 3F + 1$.

The asynchronous algorithm requires $n \geq 5F + 1$. The asynchronous algorithm is similar to the synchronous version, but in this case each processor cannot wait for all $n - 1$ values to arrive at each round. This is because there is no guarantee the faulty processors will ever send their values. So, in each round, each processor waits until it receives $n - F$ values. Then, the largest and smallest F of these values (including its own value) are removed, the select function with respect to $2F$ is applied, and the selected values are averaged to obtain the new value. Again, this choice of operations for the approximation function is shown to result in an optimal convergence rate.

Although the Byzantine approximate agreement problem was posed more than twenty-five years ago, the necessary and sufficient topological condition on the network for a successful iterative algorithm to exist has eluded researchers until very recently [194, 193, 195]. Synchronous networks are studied in [194, 193], and both synchronous and asynchronous networks are studied in [195]. In [194], Vaidya et al. provide the tight condition required in synchronous directed networks to ensure convergence (instead of finite termination) of any iterative consensus algorithm in the presence of up to F Byzantine faulty nodes. In order to state the condition, we require the following definition.

Definition 2.1 ([194]). *For nonempty, disjoint sets of nodes $A, B \subset \mathcal{V}$, $A \overset{r}{\Rightarrow} B$ if and only if there exists a node $v \in B$ that has at least r in-neighbors in A; i.e., $|\mathcal{N}_v^{in} \cap A| \geq r$. $A \overset{r}{\nRightarrow} B$ if and only if $A \overset{r}{\Rightarrow} B$ is not true.*

Given the relation of Definition 2.1, the tight condition may be stated as follows. For all quadruples of sets of nodes $\mathcal{F}, L, C, R$ that form a partition[9] of $\mathcal{V}$ such that $0 \leq |\mathcal{F}| \leq F$, $|L| > 0$, and $|R| > 0$, at least one of the two following conditions must hold true: (i) $R \cup C \overset{F+1}{\Rightarrow} L$ or (ii) $L \cup C \overset{F+1}{\Rightarrow} R$. Observe that this condition requires sufficient redundancy of directed edges between subsets of normal nodes in the network.

The necessary and sufficient condition stated above is not very intuitive. For this reason, Vaidya et al. present an equivalent condition that has more intuitive appeal in [195], which uses the following concepts. The *decomposition digraph* $\mathcal{D}^d = (\mathcal{V}^d, \mathcal{E}^d)$ of $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ is constructed from $\mathcal{D}$ by associating a node $v_k \in \mathcal{V}^d$ to each strongly connected component $\mathcal{C}_k$ of $\mathcal{D}$. A directed edge $(i, j) \in \mathcal{E}^d$ exists if and only if there is a node in component $\mathcal{C}_j$ reachable from every node in component $\mathcal{C}_i$. Note that the decomposition digraph is always a directed acyclic graph [45]. A *source component* of $\mathcal{D}$ is a strongly connected component $\mathcal{C}_k$ of $\mathcal{D}$ such that v_k is not reachable from any other node in $\mathcal{D}^d$. Finally, a *reduced digraph* $\mathcal{D}_\mathcal{F} = (\mathcal{V}_\mathcal{F}, \mathcal{E}_\mathcal{F})$ of $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ is any subdigraph of $\mathcal{D}$ such that $\mathcal{F} \subset \mathcal{V}$, $\mathcal{V}_\mathcal{F} = \mathcal{V} \setminus \mathcal{F}$, and $\mathcal{E}_\mathcal{F}$ is obtained by first removing all directed edges in $\mathcal{E}$ that are incident with nodes in $\mathcal{F}$ and then removing up to F other incoming edges at each node in

[9] Here, sets $\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_p \subseteq \mathcal{S}$ are said to form a **partition** of set $\mathcal{S}$ if $\cup_{i=1}^p \mathcal{S}_i = \mathcal{S}$ and $\mathcal{S}_i \cap \mathcal{S}_j = \emptyset$ for $i \neq j$. Note that in this context, some of the sets in the partition may be empty.

$\mathcal{V}_{\mathcal{F}}$. The alternative condition states that every reduced digraph $\mathcal{D}_{\mathcal{F}}$ with $|\mathcal{F}| < |\mathcal{V}|$ and $|\mathcal{F}| \leq F$ must contain *exactly* one source component. It is shown that the unique source component in any such reduced digraph must contain at least $F+1$ nodes. By associating $\mathcal{F}$ with the set of Byzantine nodes, these results say there must be a set of nodes (the source nodes in the reduced digraph) that are capable of disseminating their information resiliently throughout the rest of the network. Moreover, the number of source nodes in any reduced digraph $\mathcal{D}_{\mathcal{F}}$ must outnumber the Byzantine faulty nodes.

The synchronous algorithm studied in [194] demonstrates sufficiency of the network conditions stated above and is similar to the *ConvergeApproxAgreement* algorithm. The only difference is that the select function is *not* used. This algorithm has also been analyzed using transition matrices [193]. Likewise, the asynchronous algorithm studied in [195] is similar to the asynchronous algorithm of [51]. Again, the only difference between the two algorithms is that the select function is *not* used. The necessary and sufficient condition for asynchronous networks can also be stated using the relation of Definition 2.1. For all quadruples of sets of nodes $\mathcal{F}, L, C, R$ that form a partition of $\mathcal{V}$ such that $0 \leq |\mathcal{F}| \leq F$, $|L| > 0$, and $|R| > 0$, at least one of the two following conditions must hold true: (i) $R \cup C \overset{2F+1}{\Rightarrow} L$ or (ii) $L \cup C \overset{2F+1}{\Rightarrow} R$.

2.3.1.3 Gathering in Robot Networks

Similar to the distributed computing community, the goal of the research pursued by the mobile robotics community is to determine the minimal required attributes of the multi-agent network (including the robots, the network, etc.) that are required to achieve distributed tasks in finite time [163]. (It is assumed the robots are *dimensionless*, so they occupy just a single point in space, and multiple robots may occupy the same point; the number of robots occupying a single point is referred to as the *multiplicity* of the point). One such task is *gathering*, which is a consensus problem that requires a set of autonomous robots to gather at a single point in space in finite time [182]. Because the algorithms used to solve this problem modify the physical positions of the robots over time, the gathering problem is an example of a physically-dependent consensus problem. Moreover, the point at which the robots gather is unimportant, so the consensus is of the unconstrained type.

The focus of the research is on the computational models, schedulers, and high-level planning algorithms required to eventually ensure the objective – in this case gathering – is achieved. The question of whether the robots are capable of reaching the waypoints determined by the algorithms at the desired times is ignored, or at best simple bounds on the movement are assumed. That is, the physical dynamics of the robots are abstracted away from the problem. In this manner, the researchers are able to answer more fundamental questions on the feasibility of coordinated tasks under very weak assumptions.

In order to consider the weakest assumptions on the capabilities of the robots, it is common to assume that the robots are *anonymous* (i.e., indistinguishable, no identifiers), have different local coordinate frames (orientation and scale), *dimensionless* (points in space that neither obstruct the vision nor hinder the movement of the other robots), no communication (only sensing), and *oblivious* (or memoryless, meaning the robots do not remember past observations or computations performed in previous steps). The robots are assumed to move in either the plane or along a line. Additionally, it is assumed that the robots have multiplicity detectors; that is, they can determine whether more than one robot occupies a point. This is a necessary condition for gathering using deterministic algorithms [164, 46]. The timing assumptions on the network also vary between synchronous [1], asynchronous [163], and partially synchronous [182]. Sometimes the robots are assumed to have unlimited visibility, and therefore know the positions of all other robots (complete network) [38]. While at other times they are assumed to have limited visibility [65]. In the case of limited visibility, the consensus is sometimes local, allowing for gathering in each component of the mutual visibility graph [5].

Assuming the robots are oblivious has certain advantages with regard to resilience and robustness properties of the algorithms designed for such robots. Namely, if the algorithm can handle any initial configuration, then it is necessarily *self-stabilizing* [182]. A self-stabilizing algorithm has the property that the correctness of the algorithm is not jeopardized by transient uncertainties, disturbances, or faults in the system, as long as eventually the environment is free from such anomalies for a long enough period of time for the goal to be achieved. This property follows from the memoryless property of the robots because the anomaly-ridden history is essentially wiped clean after the anomalies cease to be present. Another benefit of designing algorithms for oblivious robots is that the algorithms are more amenable to a changing environment, where the robots may leave or enter the network dynamically.

Model of Computation and Scheduler. As in the distributed computing approaches, the agents are modeled discretely as finite automata, but here the network is a parallel composition of the agent automata (instead of being an automaton itself that directly coordinates the processor automata). The evolution of individual robots occurs in computational cycles, most typically the *Look-Compute-Move* cycle. In the Look stage, the robot observes the *configuration* (positions) of the other robots within its visibility range. With this information, the robot computes its next desired position using some algorithm. Finally, the robot moves to the desired position in the Move stage. Whenever a robot is in the middle of a cycle, it is said to be *activated*. When a robot is not activated it waits in a stationary position, which is sometimes referred to as the *Wait stage*.

The *scheduler* is a daemon that controls the activation times of the robots in the network [46]. The scheduler's control over the activation times is limited by the level of synchronicity of the network. For example, whenever the network is *fully synchronous*, the scheduler activates all the robots at each step in the evolution of the multi-agent network, and the activated robots execute each stage in lockstep throughout the cycle. In this case, the synchronicity of the network determines the semantics of the computational model. When the network is *semi-synchronous*, the first condition above is relaxed to allow only a subset of the robots to be activated, but still implies the activated robots execute their cycle in lockstep. The *asynchronous* case is a bit more complicated and requires the definition of the computational models (to be explicated shortly). Together, the scheduler and the level of synchronicity define the parallel composition of agent automata comprising the network. Along with the computational model, the timing semantics of the multi-agent network are fully determined.

The two most common computational models are the ATOM model [182] and the CORDA model [163]. In the ATOM model, the entire Look-Compute-Move sequence is executed atomically, and any activated robots execute the sequence in lockstep. This assumption effectively forces the execution of the system to occur in rounds, and ensures that the robots observe other robots only whenever they are all stationary [5]. In the CORDA model, each stage of a cycle requires a (nonzero) finite amount of time to complete, and any non-null move action results in a (nonzero) finite distance moved. Moreover, the CORDA model is asynchronous, so the scheduler can allow another robot to preempt an activated robot in the middle of its cycle. For example, say robot 1 begins a Look-Compute-Move cycle. In the CORDA model, the scheduler may interrupt robot 1 at any time, say in the middle of its compute stage. It can then activate robot 2 and have it complete an entire cycle before allowing robot 1 to continue is computation. Once resumed, robot 1 may have an outdated status on the position of robot 2.

To avoid a robot being ignored by the scheduler, it is typically assumed that the scheduler is *fair*. That is, in any infinite number of cycles, each robot is activated infinitely often. Another stronger notion of fairness is given by a *k-bounded* scheduler, in which case between any two activations of a given robot, no other robot may be activated more than k times. More precisely, for each robot i, with activation times $t_{i,1}, t_{i,2}, \ldots$, no other robot has more than k activation times within the interval $[t_{i,j}, t_{i,j+1})$, for $j = 1, 2, \ldots$ (notice the interval is closed on the left, meaning if another robot r has a simultaneous activation time at $t_{i,j}$ then $t_{i,j}$ is counted as one of up to k activation times of robot r allowed before $t_{i,j+1}$ with a k-bounded scheduler). Another strong assumption is that the scheduler is *centralized*, which means that at each configuration only a single robot is allowed to perform its action [46]. Finally, a scheduler is *bounded regular* if between two consecutive activations of any given robot, all other robots perform their actions exactly once. Thus, a bounded regular scheduler imposes a round robin schedule.

Weak Gathering: Gathering with Faults

The gathering problem has also been studied in the presence of failures, including stopping failures (referred to as *crash faults*) and Byzantine failures. In the presence of failures, the gathering problem is defined so that only the nonfaulty robots are required to gather at a point, which is called the *weak gathering problem* [46]. For Byzantine failures, the scheduler is treated as an adversary that controls the activation times of the robots (as always), the behavior of faulty robots, as allowed by the model of computation, and any unspecified characteristics of the nonfaulty robots. A common assumption is that the scheduler can stop a nonfaulty robot before reaching its desired position, but only if the desired position is further than some minimum distance [1, 27].

In the ATOM model, as described above, the scheduler cannot interrupt any activated robot. Therefore, since the activated robots execute in lockstep, the faulty robots cannot hide their true positions during the compute stage of a nonfaulty robot's computation cycle. In this case, we say the faulty robot is *malicious*. However, in the CORDA model, the scheduler *can* interrupt any robot at any point in their computational cycles. Thus, by carefully scheduling alternately nonfaulty robots with a particular faulty one, the scheduler can effectively make each nonfaulty robot in the visibility range of a faulty robot have different beliefs on the current position of the faulty robot when each nonfaulty robot computes its next position. (Notice this is not possible if the scheduler is k-bounded for sufficiently small k). In this case, the faulty behavior is truly Byzantine, as defined in the distributed computing failure model.

The first work to consider both crash and Byzantine failures for the weak gathering problem is [1]. In this work, the typical assumptions are made concerning the robots: dimensionless, anonymous, oblivious, no common coordinate system, and unlimited visibility (complete network). An algorithm is given that can handle a single crash for any network of $n \geq 3$ robots using the ATOM computational model. Then, it is shown that in a semi-synchronous network with the ATOM model, no deterministic algorithm can solve the weak gathering problem with even a single Byzantine failure (without additional assumptions on the scheduler). Finally, under a fully-synchronous timing model, an algorithm is given that can solve the gathering problem with up to F Byzantine failures as long as $n \geq 3F + 1$ (the algorithm also solves the problem for the special case of $n = 3, F = 1$).

Probabilistic Gathering

Variants of the (weak) gathering problem have also been considered. For example, [46] looks at *probabilistic gathering* in both fault-free and fault-prone environments, where the (nonfaulty) robots gather at a point with probability 1. The assumptions on the robots are the same as [1]. Interestingly, it is shown in [46] that multiplicity detection[10] is not necessary for probabilistic gathering with an algorithm that can introduce randomness as long as the scheduler is fair and k-bounded (this possibility result includes the CORDA model). On the other hand, in crash prone networks, multiplicity detection is again necessary for the deterministic case, and is also necessary in the probabilistic case with a fair scheduler that is not k-bounded. However, whenever the scheduler is k-bounded, an algorithm is provided which can handle a single crash without multiplicity detection. If the robots have knowledge of the multiplicity of all points, then it is shown that a probabilistic algorithm can achieve weak gathering with 2 or more crashes under any scheduler. With respect to Byzantine failures, it is shown that no deterministic algorithm can achieve gathering with even a single Byzantine failure with a centralized $(n-1)$-bounded scheduler. Then, the necessary condition is extended to the

[10]The *multiplicity* of a point occupied by one or more robots is the number of robots at that point, and *multiplicity detection* is the capability of determining whether a point occupied by one or more robots has multiplicity one or greater than one.

case of $F \geq 2$ failures, and lower bounds are provided on k for k-bounded, centralized schedulers so that for any k larger than the bounds no deterministic algorithms can achieve weak gathering, even with multiplicity knowledge. In the probabilistic case, an algorithm is given that can handle Byzantine failures in the ATOM model with a bounded scheduler and multiplicity knowledge, as long as $n > 3F$.

Point Convergence

Another variant on gathering is the *point convergence problem*, or just *convergence*, which relaxes the assumption of gathering in finite time to allow for asymptotic convergence toward gathering [5]. More precisely, given any $\epsilon > 0$ there exists a point in space c and a time $t_\epsilon > 0$ such that for all $t \geq t_\epsilon$, all nonfaulty robots lie within the ϵ-neighborhood of c, and therefore, the maximum distance between any two nonfaulty robots is 2ϵ [26]. In [5], the usual assumptions are made concerning the robots, except that there is a maximum distance σ that each robot can move in a single activation cycle and the robots have limited visibility of radius r. A circumcenter algorithm is proposed for achieving point convergence which ensures that the number of components of the mutual visibility graph never increase. The ATOM model is assumed with a semi-synchronous fair scheduler, along with the additional assumption that the Look-Compute-Move cycle occurs instantaneously. The basic idea of the circumcenter algorithm is that each robot i, when activated, moves toward the center, c_i, of the smallest circle enclosing the set of points occupied by robots within its visibility range (see Figure 7 for an example). The precise point to which it moves is constrained to be at most a distance σ from its current position p_i. For each robot $j \neq i$ in its visibility range, define D_j to be the circle defined with radius $r/2$ and center given by the midpoint m_j of the line segment between the robots i and j. Let l_j denote the maximum distance robot i can move without exiting D_j. Then the robot moves in the direction of c_i with a distance given by the minimum of the following distances: dist(c_i, p_i), σ, and $\min_j\{l_j\}$. Notice that if no other robots, other than i itself, are in its visibility range, it does not move. In this manner, the algorithm ensures local point convergence; namely, within the components of the visibility graph.

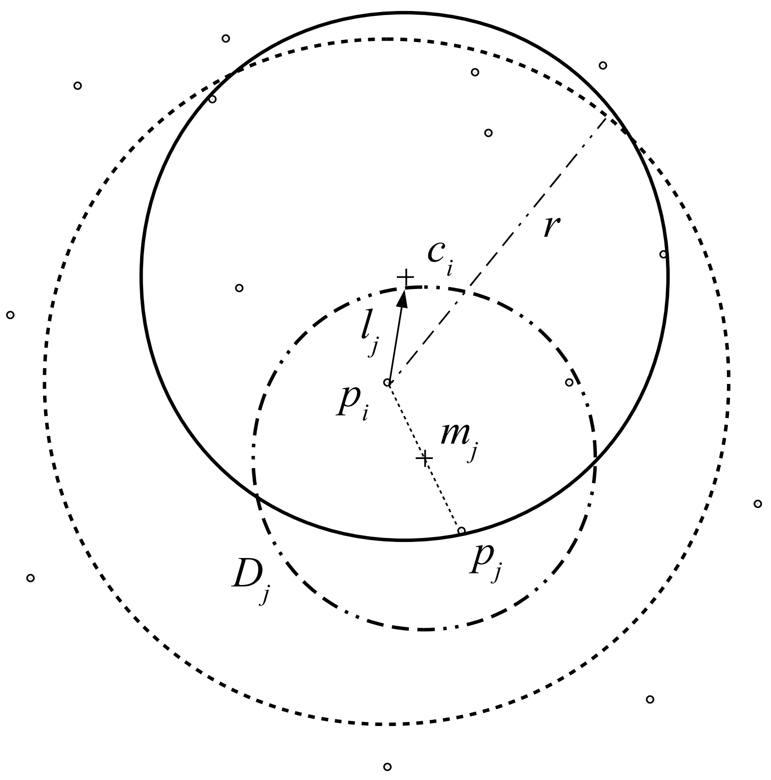


Figure 7: Scenario for the circumcenter algorithm.

Byzantine Point Convergence

The *Byzantine point convergence problem* is first addressed in [25] for the unidimensional case with the usual assumptions (including *strong multiplicity sensor*, i.e., a sensor which can detect the exact multiplicity of each occupied point). An algorithm is introduced that uses an approximation function – similar to the one in [51] – that can ensure point convergence in the presence of up to F Byzantine failures in complete networks of size n with k-bounded schedulers whenever $n \geq 2F + 1$ for fully synchronous (ATOM), $n \geq 3F + 1$ for semi-synchronous (ATOM), and $n \geq 4F + 1$ for asynchronous timing (CORDA). The approximation function used in the algorithm uses a composition of F reduce functions and then averages the unique positions in the resulting multiset (called the *center* function) to determine the next position (see Section 2.3.1.2 for a definition of approximation functions and the reduce function).

Algorithms that use approximation functions are said to be *cautious*, which means nonfaulty robots always move inside the range of positions of other nonfaulty robots and eventually a nonfaulty robot will move if point convergence has not been achieved [26]. If the algorithm is also *shrinking*, that is the range of the union of positions and destinations held by nonfaulty robots decreases by a constant factor over time (not necessarily uniformly over time), then the algorithm achieves point convergence [26]. In [26], it is shown that $n > 3F$ is necessary in semi-synchronous ATOM networks with a 2-bounded scheduler. A deterministic algorithm is also given that is both cautious and shrinking in k-bounded complete CORDA networks whenever $n > 3F$. The algorithm uses another approximation function, similar to the one described above. Instead of always eliminating the F positions that have largest and smallest coordinates, it only removes up to F positions larger or smaller than the given robot's current position (starting at the largest and smallest coordinate, respectively). This function is called the F-trimming function. Then the center function is applied to determine the next position.

Some impossibility results for single dimensional Byzantine point convergence are given in [27] that are similar to the results of [46] for gathering. The results focus on whether the robots are equipped with multiplicity sensors and if so, what kind they have. A *weak multiplicity sensor* is capable of detecting multiplicity of a point, but not the precise number of robots occupying the point, whereas *strong multiplicity sensors* are able to detect precisely how many robots occupy a point. In [27], it is shown that strong multiplicity sensors is a necessary condition for Byzantine point convergence. It is also shown that $n > 2F$ and $n > 3F$ are necessary conditions for Byzantine point convergence in ATOM networks with fully synchronous and semi-synchronous timing, respectively.

Finally, the analysis of Byzantine point convergence in one dimension is extended to fully asynchronous complete networks with a fair scheduler (but no boundedness assumptions) in [24]. Otherwise, all other assumptions are the same as before. Here it is shown that $n \geq 5F + 1$ is both necessary and sufficient for solving the Byzantine point convergence problem. For sufficiency, a refinement of the algorithm described in the previous section is introduced. Whenever a nonfaulty robot is activated, it observes the positions of all other robots and computes the multiset resulting from application of the $2F$-trimming function. It then uses the center function to determine the center of the remaining points. The novel aspect of the algorithm is an election method, which determines if the point of the given robot is an extremal point, defined as less than or equal to the $(F + 1)$st smallest coordinate or greater than or equal to the $(F + 1)$st largest coordinate. If this is the case, the robot is elected and moves to the center point calculated above.

2.3.1.4 Rendezvous

Rendezvous is an unconstrained, physically-dependent consensus problem, just as gathering and point convergence. In fact, it is equivalent to point convergence. That is, the goal of rendezvous is for the agents to converge to a single unspecified point in space (not necessarily in finite time). The difference lies in the assumptions surrounding the problem and the techniques used to design and analyze the algorithms. Typically, researchers studying rendezvous tend to focus more on limitations of information dissemination (either arising from sensing or communication limitations), and therefore employ graph theoretic methods. Also, because of this, the control laws considered are required to maintain connectivity properties of the network as the system evolves. The researchers also allow for communication between robots and have studied robustness to communication failures [42] and implementation effects (such as quantization [58]). But, there have not been studies on crash or Byzantine failures, as in the mobile robotics literature [1, 46, 24]. Finally, dynamical models have been imposed on the robots, often with nonholonomic constraints [49]. These differences are due to the fact that the rendezvous problem emerged from the control community instead of the mobile robotics community.

The first work from the control community to describe the rendezvous problem – and, because of this, it retains many of the assumptions and attributes of the problem from the gathering and point convergence perspective – is given by Lin, Morse, and Anderson in [122], and addresses the problem in both synchronous and asynchronous settings. The assumptions in [122] are similar to the assumptions on the agents in [5] (e.g., the robots reside in the plane), except that here the assumption of instantaneous Look-Compute-Move cycle is dropped. That is, the times required to sense other agents in the visibility range and to move to the calculated waypoint are nonzero. These times are synchronized to a common clock in the synchronous case and in the asynchronous case, only worst case times are considered, with no common clock. A family of control algorithms is introduced, whose form is a generalization of the circumcenter algorithm used in [5]. As such, it is considerably more complicated, but has the same underlying intuition as the algorithm of [5] (described above in Section 2.3.1.3). The main idea in the family of control laws considered is to ensure that the number of components in the mutual visibility graph does not increase over time. To do this, it is sufficient for the agents to be constrained in their movements so that no agent loses any of its neighbors during its maneuver phase. The asynchronous case is especially difficult because this constraint must be enforced in the state-based dynamically switching network without a common clock. To ensure the desired connectivity properties hold in the network in such a case, the authors use a dwell time assumption, along with some other technical assumptions similar to the synchronous case, to maintain connectivity and prove rendezvous. This work is presented in full detail in [123] for the synchronous case and in [124] for the asynchronous case.

Another work that extends the results of [5] is [42]. In [42], the authors analyze the circumcenter algorithm of [5], under more general conditions than considered in [5, 123]. In this paper, the network is synchronous and the robots are again dimensionless (points in space), anonymous (indistinguishable), oblivious (memoryless, with static feedback), sense or communicate with other robots within radius r, and may have different coordinate systems (if information exchange arises from sensing, but *not* communication). Here the robots may move in higher dimensions than the plane and the assumptions on the visibility graph are more general. In more detail, rendezvous is proven to occur using the notion of *proximity graphs* in either a finite number of rounds (i.e., in *finite time*) under more general topological conditions or asymptotically with robustness to link failures. Specifically, as long as there exists a positive integer l such that over any l consecutive rounds the union of the l consecutive digraphs is strongly connected, then rendezvous is achieved. The sequence of digraphs are subdigraphs of the proximity graph and represent directed link failures. This connectivity property is referred to as *periodic strong connectivity*. Also, it is shown that the robots may use different proximity graphs (under appropriate assumptions) in the circumcenter algorithm and still achieve convergence. Together these results help to explain the robustness properties observed in the simulations of [5].

In [133] and [132], a formal framework is presented for modeling synchronous robotic networks, group objectives, and complexity of the group objectives. This model for robotic networks formally defines control and communication laws, coordination tasks (i.e., group objectives), and time and communication complexity. Worst-case time complexity is defined as the minimum time (number of rounds) required to achieve the coordination task given the worst initial conditions (i.e., the initial conditions such that the minimum time to achieve the coordination time is maximized). Within this framework, formal definitions of the *exact rendezvous task* and *ϵ-rendezvous task* are given [132].

The exact rendezvous task is basically local gathering, whereas the ϵ-rendezvous task is local point convergence. The time complexity of two distributed control laws for achieving rendezvous are considered: the move-to-average law and the circumcenter law. The *move-to-average algorithm* progresses over a proximity graph induced by the r-radius visibility assumption described above, and at each round, each agent transmits its position to its neighbors. Then each agent moves to the average of the points in its inclusive neighborhood (that is, including its own position). Unlike the circumcenter algorithm, the move-to-average algorithm does not preserve components of the visibility graph. The move-to-average algorithm is shown to have time complexity of $\mathcal{O}(n^5)$ and $\Omega(n)$ when executed in a single dimension [132].[11] For the circumcenter law, two variants are considered. In addition to the circumcenter algorithm described in the preceding paragraph, another circumcenter algorithm, called the *parallel circumcenter law*, is introduced. In the parallel circumcenter law, instead of moving directly to the circumcenter of the inclusive neighborhood, the positions of the neighbors are projected onto the canonical axes of the agent's coordinate frame. The single dimensional version of the circumcenter law is applied to each canonical axis to determine the coordinates of the next desired position. Provided the local coordinate frames of each of the agents are *parallel frames* (meaning for any pair of coordinate frames of the agents, one may be obtained from the other by a rotation of 0°, 90°, 180°, or 270°), the parallel circumcenter law is shown to maintain connectivity of the visibility graph, and achieve exact rendezvous (gathering) with time complexity $\Theta(n)$ for arbitrary number of dimensions. In a single dimension, the traditional circumcenter algorithm is also shown to achieve exact rendezvous with time complexity $\Theta(n)$.

Another variant of the circumcenter algorithm for achieving rendezvous is considered in [131]. Here, the $1/2$ *circumcenter algorithm* is introduced, which does not require calculation of the constraint sets in order to guarantee connectivity is maintained. The main idea of the $1/2$ circumcenter algorithm is for each agent i to compute its next desired position as the circumcenter given by the midpoints of the line segments between i and neighbors in the visibility graph. Additionally, two variants of the traditional circumcenter and $1/2$ circumcenter algorithms are given that can handle noisy measurements. The main assumption is that the measurements of any neighbor's position lies within the disk centered at the true position with radius $\sigma_e < r$ (where r is the radius of visibility

[11]Recall, given $f, g\colon \mathbb{R} \to \mathbb{R}$, $f \in \mathcal{O}(g(x))$ if there exists $c \in \mathbb{R}_{>0}$ and $x_0 \in \mathbb{R}$ such that $|f(x)| \leq c|g(x)|$ for all $x \geq x_0$. Likewise, $f \in \Omega(g(x))$ if there exists $c \in \mathbb{R}_{>0}$ and $x_0 \in \mathbb{R}$ such that $|f(x)| \geq c|g(x)|$ for all $x \geq x_0$. Finally, if $f \in \mathcal{O}(g(x))$ and $f \in \Omega(g(x))$, then $f \in \Theta(g(x))$.

that induces the visibility graph). The first variant restricts even further the constraint sets to limit the motion of each agent. In the second variant, each agent filters the measurements to ensure they are within radius r. Both of these variants are shown to preserve connectivity with noisy measurements under the assumption $\sigma_e < r$. Then, the variants on the circumcenter algorithms are analyzed in a single dimension and show lower bounds on r/σ_e required to achieve rendezvous. Also, the practical stability ball to which all agents converge has diameter bounded above by $2\sigma_e$.

2.3.1.5 Particle Flocking

Particle flocking consists of a network of point masses moving with constant speed, where the objective is to asymptotically align the heading of all of the "particles". Therefore, particle flocking is a physically dependent, partially constrained consensus problem (partially constrained because headings must lie in the interval $[0, 2\pi)$). A simple model for demonstrating particle flocking is the Vicsek model [199]. In [199], it is shown that particle flocking can arise given a simple discrete-time update law, whereby each agent updates its heading at each round to be the average of the headings in its inclusive neighborhood given by the r radius visibility graph (as in the move-to-average algorithm described in the previous section). The updated heading is then perturbed by a small additive random disturbance. Given this simple control law, it is shown through simulations that particle flocking emerges. This model has motivated much of the recent study of consensus from the cooperative control perspective.

For example, in [90], Jadbabaie, Lin, and Morse formally analyze the Vicsek model (without the additive disturbance) using convergence properties of infinite products of nonnegative matrices [204]. The agent update model is given by the first order difference equation

$$\theta_i(t+1) = \frac{1}{d_i(t)+1}\left(\theta_i(t) + \sum_{j\in\mathcal{N}_i(t)} \theta_j(t)\right), \tag{6}$$

where $\theta_i \in [0, 2\pi)$ is the heading angle of agent i and $d_i(t) = |\mathcal{N}_i(t)|$. Note that this update rule may at times cause the agents to abruptly change directions. For example, consider the case where two agents with heading 0.01 and $2\pi - 0.01$ are the only two agents in each of their inclusive neighborhoods. Then, the updated heading for each agent in this case is π.

This work is the first consensus work to study dynamic (or switching) network topology, and it is shown that as long as there exists an infinite sequence of contiguous, nonempty, bounded time intervals $[t_k, t_{k+1})$, $k \geq 0$, starting at $t_0 = 0$, with the property that within any such interval the union of the communication graphs (induced by a switching signal) is connected, then the headings asymptotically align. Another interesting property of the Vicsek model demonstrated in [90] is that a single particle not following the coordination law given in (6) can fix its heading as it pleases, and so long as the conditions on the communication graph just described hold, the particle behaves as a leader of the network. That is, all other particles converge to the leader's heading. Obviously, this is not a desirable property to have if it is desired that the distributed control law be resilient to adversaries.

2.3.1.6 Linear Iterative Consensus Algorithms

The move-to-average strategy of the Vicsek model for particle flocking defined in (6) is just one example of a broader class of *linear iterative consensus strategies*. Within this class of consensus algorithms, the basic idea is to combine the values within each agent's inclusive neighborhood as a convex combination in order to determine the next value [191]. One benefit of linear iterative strategies is that agents only have to transmit a single value at each iteration. This technique has been applied to unconstrained, partially constrained, and χ-constrained consensus problems. Sometimes, the consensus process is physically-dependent, as in particle flocking, whereas other times it is not. Most often, these techniques are asymptotic; however, it is possible to reach consensus with linear iterations in finite time [178]. Both synchronous and asynchronous network paradigms have been studied using linear iterative techniques. In what follows, we first describe some synchronous techniques and then look at asynchronous techniques.

Synchronous Linear Iterative Consensus

In general, linear iterative consensus algorithms in discrete-time synchronous multi-agent networks can be modeled using a time-varying weighted digraph $\mathcal{D}_w(t)$ augmented with local feedback weights. The weighted digraph is time-varying in the case of switching network topology. Without loss of generality, we assume each round occurs at times given by the nonnegative integers, $\mathbb{Z}_{\geq 0}$. For all $t \in \mathbb{Z}_{\geq 0}$, the weighted digraph has the same vertex set $\mathcal{V} = \{1, 2, \ldots, n\}$. Thus, switching occurs between the possible directed edge sets. Then, for each $(j, i) \in \mathcal{E}(\mathcal{D}_w(t))$ and each $i \in \mathcal{V}$, let

$w_{ij}(t) \in \mathbb{R}_{\geq 0}$ and $w_{ii}(t) \in \mathbb{R}_{\geq 0}$ denote the time-varying directed edge weights and local feedback weights, respectively. That is, $w_{ij}(t)$ is the weight i gives to information received from $j \in \mathcal{N}_i^{\text{in}}$ along (j, i). However, because the network topology is changing with time, it is useful to define $w_{ij}(t), i \neq j$, even in the case $(j, i) \notin \mathcal{E}(\mathcal{D}_w(t))$ (in which case we may define $w_{ij}(t) = 0$). For simplicity, let $x_i(t) \in \mathbb{R}$ denote the state of agent $i \in \mathcal{V}$, and let $x(t) = [x_1(t), x_2(t), \ldots, x_n(t)]^{\mathsf{T}}$. By combining the weights into an $n \times n$ matrix $W(t) = [w_{ij}(t)]$, a linear iterative consensus protocol (LICP) is given by the time-varying first order linear difference equation

$$x(t+1) = W(t)x(t). \tag{7}$$

In (7), $W(t)$ is a nonnegative matrix (meaning $w_{ij}(t) \geq 0$, for all i, j) and row stochastic (row sums are equal to 1). Additionally, the following is assumed [22]. There exists a small positive constant $\alpha > 0$ such that

- $w_{ii}(t) \geq \alpha$ for all $i \in \mathcal{V}, t \in \mathbb{Z}_{\geq 0}$;
- $w_{ij}(t) = 0$ if $j \notin \mathcal{N}_i^{\text{in}}(t)$ for all $i \neq j, t \in \mathbb{Z}_{\geq 0}$;
- $w_{ij}(t) \geq \alpha$ for all $j \in \mathcal{N}_i^{\text{in}}(t), t \in \mathbb{Z}_{\geq 0}$;
- $\sum_{j=1}^{N} w_{ij}(t) = 1$, for all $i \in \mathcal{V}, t \in \mathbb{Z}_{\geq 0}$.

The lower bound on the weights, α, is given because there are counterexamples in which no consensus is reached without a lower bound [127].

Xiao and Boyd study the average consensus problem with the goal of finding the optimal weights to maximize the speed of convergence in fixed network topologies [205]. Two metrics, *asymptotic convergence factor* and *per-step convergence factor*, are defined to characterize the optimization problem. A theorem is proven that relates these metrics to the spectral radius and spectral norm of the weighting matrix, thereby recasting the optimization problem into traditional matrix related terms. By assuming a symmetric weighting matrix, both forms aforementioned collapse to a single optimization problem that can be solved by expressing the spectral norm constraint as a linear matrix inequality. The authors then describe heuristics for selecting sufficient weights for guaranteeing convergence and compare the speed of convergence of these with the optimal one on a couple of examples. Finally, one extension of the method is discussed for sparse graph design.

In [166], Ren and Beard address the problem of information consensus in a dynamic network of interacting agents. Both discrete-time and continuous-time linear consensus protocols are considered. Dynamic switching topology is considered, where the only constraint is that the union of digraphs over uniformly bounded time intervals has a rooted out-branching. These uniformly bounded time intervals must occur infinitely often, and so this is referred to as the *existence of a rooted out-branching periodically over time*. In addition, the time-varying weights in the consensus protocol must be nonnegative. Under these conditions, the multi-agent network is shown to reach unconstrained consensus asymptotically. A complementary result is given by Moreau in [144], where it is shown for any iteration scheme that satisfies a strict convexity assumption (not necessarily linear), the existence of a rooted out-branching periodically over time is both necessary and sufficient to ensure asymptotic consensus is reached. In addition, it is shown by Moreau that the periodicity of the existence of a rooted out-branching can be relaxed in the case of bidirectional communication. In this case, a necessary and sufficient condition is for each point in time $t_0 \in \mathbb{Z}_{\geq 0}$, the union of graphs over $[t_0, \infty)$ must contain a rooted out-branching.

The more general problem of distributed calculation of any function of the initial states of agents in the network using linear iterations is studied by Sundaram and Hadjicostis in [178]. By using observability theory, each agent i observes the evolution of values in its inclusive in-neighborhood over a finite time interval in order to determine precisely the initial values of all nodes from which i is reachable. The method requires fixed network topology, but works for almost any choice of weights. The amount of time required is upper bounded by the difference between the number of nodes to which i is reachable and the in-degree of i. Therefore, if the network is *strongly connected*, all nodes in the network can calculate *any* function of the initial states in at most $n - 1$ time steps.[12] This is a very general result and includes χ-consensus, for any function χ on the initial states.

Asynchronous Linear Iterative Consensus

In an *asynchronous* setting, there are several issues not present in the linear iteration of (7). First, and most importantly, it is necessary to make some assumption that each agent eventually performs an update. Otherwise, the agents that never update will obviously not reach consensus with the

[12]It is $n - 1$ instead of n because in a strongly connected digraph all nodes must have in-degree greater than or equal to one.

others. Second, the values used in a given update may be outdated. Third, because there is no common clock, it is typically assumed that agents perform pairwise updates, i.e., only using a single neighbor's value per update.

In [59], results from asynchronous systems theory are applied to stability properties of consensus in multi-agent networks with asynchronous communication and fixed interaction topology. The following assumptions are made:

- *regularity*: finite number of updating instants in any finite time interval;
- *eventual update*: the number of updates of an agent as time approaches infinity is unbounded for every agent in the network;
- *local state access with positive initial value*: there exists at least one agent whose initial value is greater than zero and which can access its own state without delay;
- *rooted out-branching*: the interaction digraph has a rooted out-branching.

By ordering the time instances in which updates are made, a one-to-one correspondence can be made with the nonnegative integers. At each update instance, each agent i either updates its state according to the ith row of (7) or remains the same (if it does not update at the time instance). Given the above assumptions, the asynchronous protocol is shown to converge. Also, it is described how synchronous protocols with time-varying interaction topologies may be cast into the asynchronous framework proposed if the union of interaction digraphs across an infinite sequence of contiguous, nonempty, bounded time intervals has a rooted out-branching. The consensus in this case is unconstrained.

Asynchronous linear iterations are studied in [135] for the *average consensus problem* in packet-switched networks. A message passing mechanism is used for communication in fixed connected undirected network topologies. Two algorithms are studied, one with blocking and another with no blocking. For each algorithm, the basic primitive is a pairwise update. In the first algorithm, a blocking mechanism is used to prevent a node from concurrently participating in multiple pairwise updates. A problem with the blocking mechanism is that it can lead to deadlock. To address this, the authors describe a round-robin implementation of the blocking mechanism, which is deadlock-free. The work assumes that for each link, and at any time t, there exists a finite time $\tau \geq t$ in which an update will occur on the link. The second algorithm does not have blocking, but instead requires that each node keep in memory the sum of all pairwise updates on each channel. The first algorithm is shown to have an exponential convergence rate. No analytical rate of convergence is provided for the second algorithm, but in simulations it demonstrates exponential convergence.

2.3.1.7 Continuous-Time Consensus

A natural extension of the linear iterative strategies described in the previous section is to "continualize" the iterations by considering ordinary differential equations instead of difference equations. Doing so enables the use of a vast array of tools from smooth and nonsmooth analysis to design and analyze *continuous-time consensus protocols*. Moreover, if the differential equations are interpreted as physical state dynamics of the agents, then there is a natural physical interpretation. In this case, the agents have integrator dynamics and can be viewed as point masses or particles as in the rendezvous or particle flocking problems. The tradeoff, however, is that only synchronous networks can be analyzed in the continuous-time setting.

The canonical continuous-time linear consensus protocol (LCP) is given as follows. Let $x_i(t)$ denote the scalar state of agent $i \in \mathcal{V}$, which has integrator dynamics $\dot{x}_i = u_i$. Then

$$\dot{x}_i(t) = \sum_{j \in \mathcal{N}_i} a_{ij}(t) \left(x_j(t) - x_i(t) \right), \tag{8}$$

where $A(\mathcal{D}(t)) = [a_{ij}(t)]$ is the (possibly) time-varying adjacency matrix. By expressing

$$x(t) = [x_1(t), x_2(t), \ldots, x_n(t)]^\mathsf{T},$$

we may write

$$\dot{x} = -L(\mathcal{D}(t))x(t), \tag{9}$$

which describes the global evolution of the multi-agent network. The asymptotic behavior of (9) is studied in fixed and switching network topology in [156]. For fixed topology, the solution of (9) is given by

$$x(t) = \exp(-L(\mathcal{D})t)x(0). \tag{10}$$

In [156], it is shown that if $\mathcal{D}$ is strongly connected with the Laplacian satisfying $L(\mathcal{D})z_r = 0$, $z_l^\mathsf{T} L(\mathcal{D}) = 0$, and $z_l^\mathsf{T} z_r = 1$ (i.e., z_l and z_r are the left and right eigenvectors associated with the zero eigenvalue that satisfy the constraint $z_l^\mathsf{T} z_r = 1$), then

$$\lim_{t\to\infty} x(t) = z_r z_l^\mathsf{T} x(0).$$

Thus, if $L(\mathcal{D})$ has a left and right eigenvector of 1 associated with the zero eigenvalue, then the constraint $z_l^\mathsf{T} z_r = 1$ implies that (9) solves the average consensus problem asymptotically. It turns out that this is true if and only if the weighted digraph is *balanced*. Whenever the digraph is strongly connected, the convergence to the average of the initial values is exponential, with rate determined by the algebraic connectivity of the symmetric part of the weighted Laplacian, i.e., $L_s = 1/2(L(\mathcal{D}) + L(\mathcal{D})^\mathsf{T})$.[13]

The convergence properties of (9) are also studied in [156] for the case of switching network topology. It is shown that as long as each digraph in the set of feasible digraphs is strongly connected and balanced, then average consensus will be reached asymptotically with a rate of convergence bounded by the smallest Fiedler eigenvalue in the set of feasible digraphs. Also, networks with identical constant delays are considered in [156]. Whenever the network is fixed, connected, and undirected, the average consensus problem is achieved if the time delay is no larger then $\pi/(2\lambda_n(L))$.

In order to increase the rate of convergence in average consensus, Epstein *et al* decompose the (connected, undirected) network into an M-layer hierarchical network [54]. An algorithm for implementing the canonical linear consensus protocol that utilizes this hierarchical decomposition is shown to reach consensus faster than when implemented on the same network without the imposed hierarchy. This is achieved by effectively increasing the algebraic connectivity in the hierarchical subgraphs, which in turn drastically increases the rate of consensus. The tradeoff is steady-state error. That is with the proposed algorithm the average may be reached within certain error bounds. The consensus error is analyzed and closed-form bounds are derived.

Another approach to increasing the rate of convergence of the linear consensus protocol is to perform a rewiring process on a regular graph in order to create a *small-world network* [150]. Small-

[13]The second smallest eigenvalue of L_s is justified to be called the algebraic connectivity of L_s because L_s is a valid Laplacian matrix of the so-called mirror graph of $\mathcal{D}$ [156].

world networks are highly clustered, like regular graphs, but also have a small characteristic path length, like random graphs[14] [202]. It is shown in [150] that the mean of the bulk Laplacian spectrum is invariant throughout the rewiring process. Surprisingly, the algebraic connectivity of the resulting small-world network can be up to three orders of magnitude larger than the regular graph. Note that this is equivalent to increasing the rate of convergence of the consensus protocol by three orders of magnitude.

In [143], Moreau analyzes the canonical linear consensus protocol in a network of integrators. Here the agents communicate weighted copies of their state to their nearest neighbors. The main contribution of this paper is to relax the constraints on the topology and still guarantee stability (i.e., convergence of the protocol to an aligned equilibrium). Furthermore, the case of fixed and identical time delay in the communication between agents is shown to reach an aligned state as well, provided the self-feedback of the individual integrators is delay-free. In this work, there is a slight difference in the approach to analyzing the convergence properties. Olfati-Saber and others have used graph theoretic tools like the graph Laplacian to apply convergence analysis to the network of interacting agents, whereas Moreau takes the complementary approach of considering first the "system" (i.e., the multi-agent network) described by a time-varying system matrix that is Metzler (off diagonal elements are all nonnegative and at least one is strictly positive to avoid the trivial case of the zero matrix) with zero-row sums, and then associating to the system dynamics a directed graph describing the communication. A nonnegative tolerance parameter is also associated to the graph so that only weights in the system matrix which are strictly greater than the tolerance will have associated links in the graph. To achieve the convergence results (uniform exponential stability), the Lyapunov function of Tsitsiklis is used [191], defined as follows: Suppose there are n integrator agents with states given by $x_1, x_2, \ldots, x_n$. Then define

$$V(x) = \max\{x_1, x_2, \ldots, x_n\} - \min\{x_1, x_2, \ldots, x_n\}.$$

With this, the milder connectivity conditions may be stated as follows. Provided there is a rooted

[14]The *characteristic path length* is the average distance between nodes in the graph, where the distance between nodes is the length of a shortest path between the nodes. The *clustering coefficient* of a graph is defined as follows. Given that a node i has degree d_i, the maximum number of edges *within* the subgraph of inclusive neighbors of i is $d_i(d_i-1)/2$. Then i's clustering coefficient is the ratio of edges in the subgraph of the inclusive neighborhood to the number of possible edges. The clustering coefficient of the graph is then the average of this ratio over all nodes.

out-branching and the system matrix is Metzler with zero-row sums for every time t, then the aligned equilibrium point is uniformly exponentially stable. Note that this paper does not address the average consensus problem, but instead the consensus is unconstrained because there is no requirement that the network be balanced.

Wang and Slotine provide a contraction analysis of time-delayed versions of the canonical linear consensus protocol in [201]. This work brings together much of the prior work in the literature on consensus under a common framework. It shows how in a leaderless group the time delays can cause the steady-state value of consensus to differ from a version of the same system without delays. Here the time delays are not required to be identical. For leader-follower groups, the leader is able to dictate the consensus value. In order to ensure stability (convergence), wave variables are used in communication. The analysis is done for both continuous-time and discrete-time models.

Lee and Spong extend the unconstrained consensus results from integrator agents to more general dynamical agents [118]. The agents are modeled as a class of linear systems and are not necessarily identical. The network has directed information flow with possibly nonidentical constant delays. The main stipulation is that each agent's frequency response must have unity magnitude at DC (zero frequency) and strictly less than unity magnitude for all nonzero frequencies. This may be achieved using local feedback control to shape the closed-loop frequency response as desired. For the agents to reach consensus, the directed information graph must have at least one rooted out-branching. Furthermore, the information digraph is weighted to model relative reliability of link communication amongst neighboring nodes under the following constraint: the sum of the weights for each node's set of neighbors is unity. This constraint means that the weighted adjacency matrix has an eigenvalue 1 associated with 1_n. Further, imposing the constraint of at least one rooted out-branching, causes this unity eigenvalue to be simple (multiplicity one). Combining these constraints allows for the application of Geršgorin's disk theorem coupled with the spectral radius stability theorem to show that all nonzero frequencies decay over time, leaving only a DC component. Taking this with the assumption on the weighted adjacency matrix, it is shown that consensus is reached if and only if the digraph has a rooted out-branching. Finally, a simulation of a network of agents with fully-actuated point-mass dynamics (double-integrator dynamics) and a simulation of a heterogeneous network of single and double-integrator agents is given to verify the theory applied to the rendezvous problem.

Almeida et al. study the problem of reaching consensus in continuous time with communication performed at discrete instances [3]. The main consensus variable at each node behaves as an integrator. In order to deal with discrete communication instances with possibly outdated information from neighboring nodes, an auxiliary variable is introduced that has continuous dynamics between communication instances, but is allowed to have discrete jump discontinuities at the instances in which communication takes place. The auxiliary variable takes into account the information from neighboring nodes and is introduced in the main variable's dynamics in order to drive the variables to consensus. The results require assumptions of bounded delays, nontrivial convex interactions, existence of a periodic rooted out-branching, and a bound in the time difference between communication instances [3].

Finally, Cortés studies continuous-time consensus under a very general framework in [41]. Here, distributed χ-constrained consensus is considered in weighted directed networks for any continuous function χ. In this case, the agents have integrator dynamics, but the input is not required to be continuous. In the case of discontinuous input, the solution is understood in the sense of Filippov [63].[15] The feasible functions χ on which consensus may be reached are characterized using several technical assumptions for both fixed and switching topology. An algorithm – designed for the special case of power-mean consensus – is shown to converge exponentially in weakly connected weighted digraphs given an appropriate selection of the weights. The rate of convergence is a function of the Fiedler eigenvalue of the symmetric part of the weighted Laplacian. A particularly noteworthy result of [41] is that in strongly connected weighted digraphs, a nonsmooth algorithm that achieves finite-time consensus for $\min$ and $\max$ consensus is analyzed. The time to converge is bounded by the maximum difference of the initial values.

2.3.1.8 Robust and Secure Consensus

Robustness to uncertainties and *security* are two important issues to consider when designing distributed protocols for multi-agent networks. Because consensus is arguably one of the most fundamental group objectives, it is especially important to develop secure and robust consensus algorithms. In this section we discuss consensus that is robust to uncertainties and security in multi-agent

[15]Given a differential equation $\dot{x} = f(x)$ over an interval $[t_0, t_1]$ in which the vector field f is discontinuous, the *Filippov solution* is an absolutely continuous function ξ such that $\dot{\xi} = f(\xi)$ for almost all $t \in [t_0, t_1]$.

networks.

We have already discussed various forms of robustness to uncertainties in the consensus algorithms described above. For example, in continuous-time consensus, we have discussed robustness to identical constant delays [156, 143] and nonidentical constant delays [201, 118]. Because switching topology can occur because of temporary link failures, several works already described can handle certain types of data loss (assuming the technical assumptions for convergence are still met) [156, 41]. Various other forms of uncertainty have been considered in consensus protocols for multi-agent networks. Reaching average consensus in a wireless network with interference is studied in [197]. Additive channel noise is addressed in [83]. Packet loss in ring networks is studied in [81]. Robustness in terms of sensitivity to model uncertainty has been addressed in [85].

The problem of quantized average consensus whenever the consensus variables are integers is studied in [94]. With quantization, it is not possible in general to consider exact average consensus. Therefore, the authors define the notion of *quantized consensus distribution*, which is a distribution for the values of the nodes such that no two values differ by more than one and the sum of the values in the distribution is equal to the sum of the initial values. Then, given a class of algorithms that satisfy mild assumptions, convergence to a quantized consensus distribution is shown to hold. In particular, quantized gossip algorithms are shown to belong to this class. In addition, the authors study expected convergence time of quantized gossip algorithms and shown that in complete and linear (path) networks, the expected convergence time is bounded by polynomials in the number of nodes, n.

A work that considers robust statistical consensus is [142]. We will describe this algorithm in detail because it deals with the notion of outlier removal, which is similar to the idea behind the approximation functions used in the consensus protocols of distributed computing and the Byzantine-resilient gathering algorithms. However, instead of considering outliers in terms of normal behavior of the protocol, this paper considers statistical outliers. Therefore, the "outlier nodes" are considered reliable with respect to communication.

In more detail, [142] studies robust consensus in synchronous sensor networks in which measurements are taken of a phenomenon or process that is assumed to follow a consistent statistical model. A subset of the measurements taken are assumed to be outliers with respect to this model, and the goal is to instantiate the model that fits the inliers and reach average consensus on just the inlier measurements in a distributed way, so that each node in the network – including the outliers – have the best estimate of the the true average. The algorithm to achieve this goal is a distributed version of the RANdom SAmple Consensus (RANSAC) algorithm, called De-RANSAC (RANSAC is widely used in the field of computer vision). The authors demonstrate that the distributed version achieves the same results as the centralized version of RANSAC. Because the RANSAC algorithm requires the total number of nodes, n, the authors present a finite-time consensus protocol (applicable even in switching topologies), and then combine this with a leader election mechanism to determine n.

Before describing how De-RANSAC works, it is useful to describe the centralized version (RANSAC) first. The RANSAC algorithm generates K hypothetical models to fit the data using random subsets of measurements, and then selects one of the models using a voting system. RANSAC utilizes a number of assumptions in its analysis, including

(i) each measurement has independent probability, p_{in}, of being an inlier;

(ii) there exists a procedure to estimate a hypothetical model using at least c samples;

(iii) if a model is generated using c inlier samples, then the model fits all inliers.

The number K is chosen by selecting the number of measurements, c, and the desired probability to generate one hypothesis using only inlier measurements, p_{suc}. Then using the equation

$$K = \frac{\log(1 - p_{\text{suc}})}{1 - (p_{\text{in}})^c},$$

one determines K. The voting mechanism requires beforehand a threshold parameter τ and error function. The vote on whether a measurement is an inlier with respect to a hypothetical model is a binary result, determined by taking the measurement and hypothesis as input to the error function, which returns a real value. If the value is less than τ, the measurement is then an inlier. The hypothesis that receives the most votes is the winner. Finally, the hypothesis is refined using the only the known inliers from the winning hypothesis of the voting process.

In the distributed version with switching topology, first a partial flooding algorithm is run so that each node has a larger enough sample size of measurements to create at least K hypotheses. A multi-hop sample sharing algorithm is proposed to do this. However, a problem arises because some of the hypotheses generated may be duplicates across different nodes in the network. Therefore, a refinement on the number of hypotheses each node must produce is necessary. The analysis of how this should be done assumes that each node has access to all measurements and that no local duplicates are made. The former does not hold unless the flooding algorithm is run to completion, while the latter is easy to ensure locally. The results, which use combinatorial arguments, provide a conservative bound on the number of hypotheses each node must produce to ensure other nodes are able to produce enough unique hypotheses. This bound is then reduced by using Newton's method to find a less conservative bound. Next, the distributed, finite-time consensus on the number of votes each hypothesis in the union of hypotheses receives is presented. The hypothesis with the most votes is then selected. In order to perform the refinement on this hypothesis in a distributed manner, a consensus protocol using homogeneous and normalized coordinates is given. Finally, simulation examples for robust average consensus with outliers and robust formation of teams of robots are given.

Next, we discuss a framework for determining the robustness of distributed algorithms given in [75]. The framework applies to discrete-time, synchronous algorithms on undirected graphs. In the framework, the agent, networked multi-agent system, cooperative task, and cooperative algorithm are all formally defined, along with the notion of faulty agent and a new concept for robustness. The consensus algorithm of [156], along with the sensor deployment algorithm of [43], are analyzed for their robustness properties. Some discussion is given on how to render algorithms more robust by introducing mechanisms for detection and isolation of faulty agents.

In more detail, the agent is defined as a 4-tuple (X, U, X_0, f), where the state $x(t)$ lies in space X, the control input $u(t)$ is in U, the initial state $x_0(t)$ is in X_0, and $f\colon X \times U \to X$ is a map that defines the dynamics of the agent. The networked multi-agent system is a triple $(\mathcal{I}, \mathcal{A}, \mathcal{G}_{\text{comm}})$, where

- $\mathcal{I} = \{1, 2, \ldots, n\}$ is the set of unique identifiers for each of the n agents;
- $\mathcal{A} = \{A_i\}_{i\in\mathcal{I}}$ is the set of agents;
- $\mathcal{G}_{\text{comm}}$ is the set of allowed communication graphs. At each time step t, the communication graph defined by $\mathcal{E}_{\text{comm}}$ over n vertices is an element of $\mathcal{G}_{\text{comm}}$. The graph may be due to sensing or communication.

On top of this, there may be a set of additional variables involved in the problem specification. These are considered *environmental variables*, denoted by V. The *cooperative task* is defined as a cost function C defined over all state and input trajectories, initial states, and environmental variables. The minimizer of the task cost function should occur on a subset of the domain where the networked multi-agent system achieves the task. The *cooperative algorithm* is an assignment of communication and control laws to each agent. The failure modes considered in the paper are stopping failures, constant output failures, and Byzantine failures.

The definition of robustness of an algorithm is defined as follows. The *performance cost* $\mathcal{PC}$ of an algorithm is the difference between the task cost achieved by the algorithm and the minimum task cost. The *worst-case performance cost* $\mathcal{PC}_{wc}$ is the supremum of the $\mathcal{PC}$ as the initial conditions and values of the environmental variables are varied across a given set. Consider an algorithm being executed on a system of n agents out of which F agents fail according to a particular failure model. Denote the worst-case performance cost achieved through the remaining $n - F$ agents as $\mathcal{PC}_{wc}(n, F)$, where the supremum is taken over the initial conditions, values of the environmental variables, and all groups of F agents that can fail. An algorithm is said to be *worst-case robust* to a particular failure mode up to F agents if

$$\mathcal{PC}_{wc}(n, F) = \mathcal{O}(\mathcal{PC}_{wc}(n - F)),$$

as $n \to \infty$. If $\mathcal{PC}_{wc}(n, F) = \Omega(\mathcal{PC}_{wc}(n-F))$ but $\mathcal{PC}_{wc}(n, F) \neq \Theta(\mathcal{PC}_{wc}(n-F))$, the algorithm is said to be *worst-case non-robust*.

The most important results of [75] are that the consensus protocol of [156] is worst-case non-robust if the F failures make the graph disconnected, and is worst-case robust if the F failures do not make the graph disconnected with respect to stopping failures. It is worst-case non-robust to constant and Byzantine failures. Because of this, there is a need for consensus protocols in cooperative control that are robust to these types of failures.

Detecting Misbehaving Agents

Here we consider techniques for detecting and identifying misbehaving agents in multi-agent networks. Typically, two types of misbehaving agents have been considered: non-colluding and malicious agents. Non-colluding agents are unaware of the network topology, the states of the other agents, or the other misbehaving agents. Malicious agents, on the other hand, are aware of the network topology, the states of the other agents, and which agents are adversaries. In this way, malicious agents are similar to Byzantine agents because worst-case behavior must be considered; however, malicious agents must transmit the same information to each neighbor, and are therefore less devious than Byzantine agents. Each of the identification techniques requires at least some non-local information. And, the algorithms tend to be of high complexity, i.e., exponential complexity in the case of exact identification. But, more computationally efficient *approximate* algorithms have been developed [159].

In [157], the issue of detecting and identifying a single misbehaving agent using a linear iterative strategy in discrete-time synchronous networks is introduced. Then, Sundaram and Hadjicostis show in [179] that $\kappa(\mathcal{G}) \geq 2F + 1$ is a necessary condition for detecting and identifying up to F malicious agents using linear iterations in synchronous networks. In the companion paper [180], $\kappa(\mathcal{G}) \geq 2F+1$ is shown to also be sufficient for the problem. In this case, observability properties of the linear multi-agent network are exploited so that every node is able to calculate the initial values exactly, and thus any function of the initial states, in at most n steps. The results of [179, 180] are generalized in [181] to characterize under which conditions any subset of nodes can obtain all of the initial values.

The authors of [157] later extend the analysis done in [179, 180] by characterizing the type of behavior of the malicious agents that is most troublesome to the linear network and by characterizing the network connectivity required to tolerate both malicious agents and non-colluding agents in [158]. A computationally expensive but exact algorithm is presented in [158] to detect and identify up to F malicious agents in networks with connectivity at least $2F + 1$. This exact algorithm requires each node to know the topology of the entire network. In [160], two approaches are considered to reduce the computational complexity and require only partial network information. The first assumes the network is comprised of weakly interconnected subcomponents and restricts the behavior of the misbehaving nodes. The second imposes a hierarchical structure to detect and isolate the malicious agents. These results are combined and extended in [159].

The idea of using a sequence of maneuvers in order to detect faulty or malicious agents in discrete-time linear consensus networks is considered in [67]. The so-called motion probes, defined as inputs to the network, are shown to leave the stationary point of the consensus process unchanged. However, the authors do not consider how the motion probes achieve the detection objective, and no algorithms are given for the motion probes. Also, a fault recovery maneuver is considered, whereby the uncompromised agents undo the effects of the malicious agents. This is achieved by maintaining in memory for each agent, all contributions of its neighbors on its own trajectory. Under appropriate assumptions, subtracting these contributions will recover the initial stationary point of the consensus process including only uncompromised agents. This idea was originally used in [174] for reaching the correct consensus value with dynamic network topology in continuous-time and later adapted to an asynchronous setting in [135].

Another closely related research problem is the issue of security of consensus in multi-robot systems [57]. A distributed intrusion detection system (IDS) has been developed using a hybrid model of robotic agents that monitors neighboring agents to detect non-cooperative agents using only local information [57]. The distributed IDS has been extended to deal with the case where some of the monitors provide false information [56]. This paper describes a distributed service for detecting malicious robots in multi-agent networks. The framework is a high-level architecture that uses a set of decentralized cooperation rules to try to detect and isolate malicious robots based on known physical behavior and logical constraints requiring a majority vote. The approach is exemplified using a simple automated highway example with simulations. The IDS of [56] is improved by using past information in [55].

2.3.2 Pattern Formation

Pattern Formation, or *formation stability*, is a physically dependent generalization of gathering, point convergence, and rendezvous. The goal is for the agents to autonomously and cooperatively form a pattern in space, which could be predetermined by a desired curve or unknown prior to deployment of the agents. Pattern formation has been studied in the mobile robotics community [182] and the cooperative control community [154]. The differences between the two approaches are essentially the same as the differences between gathering and rendezvous, so they will not be repeated here.

One simple scenario is whenever the agents are point masses and they wish to achieve desired inter-agent distances between neighbors. A simple modification of the canonical continuous-time consensus protocol can achieve the objective [152]. To be concrete, let r_{ij} denote the inter-agent distance between agents i and j desired by agent i. Then, the following protocol achieves convergence to a formation with the desired inter-agent distances, assuming the directed network is balanced:

$$\dot{x}_i = \sum_{j \in \mathcal{N}_i^{\text{in}}} (x_j - x_i) + p_i,$$

where $p_i = \sum_{j \in \mathcal{N}} r_{ij}$.

The basic idea described above is used in [125] to make formations about the centroid of the group of integrators in a cyclic pursuit scenario. In this situation, the communication network forms a directed cycle. Here the authors also consider counterclockwise and clockwise star formations and show that collisions are avoided. Similarly, in [154], structural potential functions are used to achieve planar formations of double-integrator agents while avoiding collisions. Another completely different approach to pattern formation uses a virtual layer for self-stabilization of a network of robots to a desired curve [71]. In this case, the robots converge uniformly along the desired curve, and because the algorithm is self-stabilizing, it is guaranteed to converge whenever there are intermittent disturbances in the network that eventually subside.

2.3.3 Formation Control

Formation control is concerned with establishing and preserving a formation of the agents while the agents perform maneuvers. Therefore, formation control requires pattern formation, or formation stability, as a subtask. Formation control is important for coordination of spacecraft for interferometry, control of unmanned aerial vehicles (UAVs), and mobile robotics. There are four major approaches to formation control: leader follower, virtual leaders, behavior-based, and graph theoretic. In the leader follower approaches, the leader provides the reference trajectory to the other agents [185]. Virtual leaders, on the other hand, are synthesized by the multi-agent network, so that the network is more robust to leader failure [53]. Behavior-based approaches typically have a set of primitive actions, and the collective behavior is formed by the selection of actions made by the individuals, with rules governing how they are selected [12]. Graph theoretic methods employ results from algebraic graph theory in order to achieve formation control using local strategies [60]. The graph theoretic approach is the most relevant to our concerns.

In [60], the problem of formation stability and control is considered using graph theory. Specifically, the eigenvalues of the Laplacian are used to state a Nyquist criterion on formation stability for the proposed distributed controller. The effects of the network topology on formation control is studied in detail, and a separation principle, which allows for the separate design of an individual stable controlled vehicle and a stable formation network is considered. The focus of the paper is on formation stability and control of LTI systems.

In [7], Arcak analyzes a passive interconnection structure suitable for formation control and group agreement problems that unifies other prior work [148, 184, 155], and addresses data loss for the agreement problem by showing that the communication topology may be time-varying, as long as it remains connected in an integral sense. In [7], Arcak presents a passivity-based design framework for controlling nonlinear systems which have been rendered passive through an internal feedback configuration. The problem addressed is to satisfy two design constraints. The first deals with zero steady-state error in velocity control. The second is a relative positioning of the agents (or formation control) defined by a compact set. The underlying assumption is that each agent in the network is capable of receiving a global velocity trajectory command.

2.3.4 Synchronization

Synchronization is a dynamic form of agreement on *all* of the state variables of the agents. From one perspective, it is a strict generalization of consensus, where the agents have more complicated dynamics than simply an integrator or accumulator. On the other hand, there are fewer variants of synchronization problems. This is because synchronization problems tend to deal exclusively with agreement on physically dependent quantities, instead of more general quantities. Therefore, more precisely, synchronization is a strict generalization of physically dependent consensus, and entails agreement whenever the states are oscillatory or converge to limit cycles. Because the synchronization literature is vast, we consider only synchronization problems in multi-agent networks.

In [171], synchronization of identical linear systems is studied. Sufficient conditions for synchronizing to open loop dynamics $\dot{x} = Ax$ require (A, B) stabilizable, (A, C) detectable, A has all eigenvalues in closed left-half plane, $L(t)$ (Laplacian) piecewise continuous and bounded, and $\mathcal{D}(t)$ uniformly connected (meaning there exists a time horizon T such that for all t, the union of digraphs over $[t, t+T]$ has a rooted in-branching). The dynamic output feedback controller uses a Luenberger observer [128] and stabilizing gain matrix K. Extensions to discrete-time and static feedback law are given without proof. Examples including double integrators and passive/nonpassive harmonic oscillators are given to explicate the technique.

Synchronization of the agents whenever the agents are nonidentical is extremely challenging. A recent work that looks at the asymptotic synchronization with nonidentical agents is [211]. The agents are modeled as nonlinear input-affine systems with input given by a weighted sum of neighbors' states. Using the error dynamics, conditions for synchronization are given by matrix inequalities. Two cases are considered and treated separately, (i) the agents have a common equilibrium solution, and (ii) they do not. In the first case, the origin of the error coordinates is an equilibrium point for the error dynamics, and an asymptotic stability approach is used. For the latter case, which is more difficult, the error dynamics are defined with respect to the average dynamics of the network of agents. In this case, synchronization is shown by proving global attractivity to the manifold defined by the points where the error is zero. Controllers for achieving synchronization are then developed to meet the needed criteria. The controllers may be obtained by solving an optimization problem.

2.3.4.1 Output Synchronization

One important variant on synchronization is *output synchronization*. Output synchronization is less restrictive than synchronization because it does not require all of the states to asymptotically synchronize. Rather, only a subset of the outputs are required to asymptotically synchronize. One of the first works to consider output synchronization is [35]. Here, each agent has a positive definite storage function, which is used in the stability analysis with Lyapunov based arguments [96]. Static linear and nonlinear control laws are considered that only act on the relative outputs of agents, so no inertial coordinate system is required. For the linear coupling, the results prove output synchronization on weakly connected balanced digraphs with static topology. For switching topologies with piecewise constant switching signal (i.e., a finite number of switches in any finite interval of time), the graph must be balanced pointwise in time and jointly connected (meaning the union of graphs over the time interval is connected). By additionally assuming only a single path exists between any two nodes in the communication graph, output synchronization is shown to hold with constant time delays.

In the case of nonlinear coupling, all results are on undirected connected graphs. For this case, it is assumed that the agents employ a nonlinear static control law that is odd symmetric, continuous, locally Lipschitz, and passive. With these assumptions, stability is proven, and output synchronization is possible given the right choice of the nonlinear function governing the control law. If, in addition, the function is conic and zero only at zero, then output synchronization is guaranteed. By using the scattering transformation in each of the channels of the network, the above results are generalized to the case of constant time delay, assuming only a single path exists between any two nodes in the communication graph. The above general results are then applied to the synchronization of Kuramoto Oscillators and mechanical systems (Lagrangian systems). Finally, simulations of networked pendulums and position control of robots are given.

In [87], the problem of output synchronization of rigid body holonomic robots modeled in the special Euclidean group SE(3) is considered. A velocity control law is proposed that takes the position and orientation measurements from the neighbors of a node and produces a velocity input to the local controller. Each node has its own coordinate system and the rotation matrix mapping it to an inertial coordinate system is assumed to be known. The nodes communicate in a strongly connected weighted digraph (the weights provide information of channel quality). A passivity-like condition is shown between the velocity and the vector form of the rigid body motion at each node. This is used to construct a Lyapunov function. Using LaSalle's invariant set theorem [96], the position and orientation of each robot is shown to synchronize asymptotically, thus solving the output synchronization problem. The approach is extended to the case where nonuniform constant time delays exist in the channels of the network. Finally, numerical simulations and full-scale experiments with nonholonomic robots are provided to validate the theoretical results.

Building on the results of [87], the specific case of *attitude synchronization* is considered in [88]. Here, attitude synchronization means the translational velocities and orientation of all the agents asymptotically converge to a common value. Given that all of the orientation matrices are positive definite, a velocity input law is shown to achieve attitude synchronization in strongly connected digraphs. The convergence is shown to be exponential with rate proportional to the algebraic connectivity of the symmetric part of the weighted Laplacian matrix. Also, it is shown that leader-following is possible if the leader has constant velocity and orientation. The control law is shown to have certain robustness properties. For example, attitude synchronization is shown to be achieved with nonuniform, time-invariant, and finite time delays. Finally, given a large enough dwell time, switching network topology – even with some of the communication networks *not* strongly connected – can be sustained.

Another work that addresses attitude synchronization is [10]. In [10], the design approach described in [7] is applied to a group of rigid bodies to asymptotically synchronize the group in a rotational maneuver in two situations. The first situation assumes each agent has access to the desired angular velocity. The second assumes a leader-follower architecture, where the other agents reconstruct the desired angular velocity adaptively.

2.3.4.2 Synchronized Tracking

A variant on output synchronization is *synchronized tracking*, or *concurrent synchronization*, where the outputs of the agents must synchronize while following a common reference trajectory. Although an uncoupled tracking law would "synchronize" the agents' outputs, without explicit synchronization, the cohesion of the agents' tracking performance would not be robust to disturbances. In [37], concurrent synchronization of Lagrangian systems is studied in networks modeled as regular balanced digraphs. The nonlinear stability tool used is contraction theory, and therefore the convergence results are exponential [126]. A control structure to achieve concurrent synchronization is given that uses a modified Laplacian matrix. Synchronization of the agents is shown with and without reference tracking. Moreover, the agents may be modeled as different Lagrangian systems. The concurrent synchronization scheme with the modified Laplacian is particularly nice because it allows for separate tuning of the tracking and synchronization gains.

2.3.5 Coordinated Path Following

Coordinated path following, also known as *synchronized path tracking*, is a generalization of synchronized tracking in which the entire trajectory, or path, of the reference signal is known to each agent and may be different. This generalization is nontrivial because the paths are parameterized by a dynamic variable, often time, speed, or acceleration, to which each agent must also conform *in addition* to staying "on path." This is often done in the interest of maintaining a particular formation while following the desired paths, which requires synchronizing the path parameters in an indirect manner in such a way that the desired formation is maintained. Because of this, coordinated path following may also be viewed as a generalization of formation control.

In [70], coordinated path following of nonholonomic wheeled robots in connected bidirectional networks is studied. Two goals are desired: (i) achieve asymptotic tracking of a path for each vehicle, and (ii) coordinate the vehicles while tracking their individual paths by asymptotically synchronizing the coordination states, which encapsulate formation information that may possibly be time varying. The interaction between these two goals lies in the vehicles' velocities and a disturbance-like function of the heading and distance from path errors. First, a controller for ensuring path following is proven to globally asymptotically stabilize the error dynamics. By using a decomposition of the incidence matrix based on a spanning tree contained within the connected network, uniform global asymptotic stability results are given for a synchronizing controller. This controller synchronizes the coordination states and uses saturation to ensure the velocities satisfy the constraints (boundedness, uniform continuity, and nonzero limit). These velocity constraints are required so that the path following and coordination are properly decoupled and can be combined. Time-varying pattern tracking is then discussed along with detailed simulations illustrating the results.

In [89], passivity is used to design distributed controllers for synchronized path following in a bidirectional network. Two control designs are considered. The first control law uses both path error and synchronization error (with respect to the path parameters) formed with neighbors in the network. There are separate gains for each error, which can be tuned to obtain the desired tradeoff between path error and synchronization error. The drawback is this design is only applicable to static network topology. The second control law only uses the synchronization error, but is applicable to dynamic network topology, and therefore, data loss. It utilizes the passive design technique of [7]. Finally, a sampled data approach is applied to the first control law, in which the path parameters are synchronized over the network in discrete time, while the rest of the control framework is modeled in continuous time.

2.3.6 Biologically Inspired Objectives

2.3.6.1 Flocking

Flocking is similar to coordinated path following, but is inspired by behavior observed in nature. Moreover, flocking has an explicit requirement that the agents avoid collisions with each other and with obstacles. Although biologically inspired, flocking has also been studied for the purposes of computer animation [169]. In [169], Reynolds identifies three important criteria for flocking:

- *polarization* (or cohesion): avoid becoming separated from the group;
- *noncolliding* (or separation): avoid colliding with obstacles or other members of the group;
- *aggregate motion* (or alignment): align velocity with the velocities of other members.[16]

In [151], Olfati-Saber seeks to characterize flocking from an engineering perspective. He looks to address the three criteria of Reynolds by introducing algorithms that achieve cohesion, separation, and alignment quantitatively. To do this, first r-disk proximity graphs are used to define the connectivity properties of the multi-agent network. Then cohesion is equivalent to maintaining connectivity of the proximity graph (which is assumed to hold). To address separation, a spatial lattice is used to specify inter-agent distances. Alignment is achieved by performing consensus on velocity. A more recent work introduces a technique to strictly enforce cohesion by explicitly maintaining connectivity of the proximity graph [208].

2.3.6.2 Foraging

Foraging addresses swarming behavior given utility functions that form surfaces over an environment. The basic idea is that a swarm of agents is deployed in an environment that is either rich in resources or toxic. In the case of resource-rich environments, the agents seek to move to local maxima of the resource utility function over the environment while interacting with the swarm. On the other hand, in toxic environments, the agents move to minima to minimize the level of toxicity within the environment. In either case, gradient methods are often used and are therefore sensitive to the initial distributions of the swarms and the shape of the utility surface over the environment.

[16]Whereas flocking is concerned with all three behaviors described here, the special case of particle flocking (discussed in Section 2.3.1.5) is only concerned with aggregate motion.

In [68], an attractant/repellent profile over an arbitrary dimensional environment is studied. In this work, the motion of individual agents is determined by long distance attraction, short distance repulsion, and attraction to resource-rich areas (or repulsion from toxic regions). The stability behavior of the swarm is determined by these factors. The authors analyze conditions for convergence given specific nutrient profiles, such as a plane profile, quadratic profile, Gaussian profile, and multimodal Gaussian profile. For more information on foraging, and more generally swarming behavior, see [68].

2.3.7 Cooperative Load Transport

Cooperative load transport is an objective in which the agents attempt to cooperatively move an object in space, and interact only through incident and reactive forces on the load. Therefore, the information exchange is inherently bidirectional. Although it has essentially no relation to the work of this dissertation, it is an interesting example of a group objective in which the communication in the network arises *implicitly* by mutual interactions with the payload. The interested reader is referred to [11] for more details.

2.4 Comparison to this Dissertation

The goal of the research presented in this dissertation is to develop distributed multi-agent network consensus and synchronization algorithms that are *resilient* to malicious attacks and uncertainties caused by implementation effects. While some of the literature has focused on uncertainties caused by implementation effects, there are few works dealing with resilient consensus and synchronization. The work here differs from these related works because, as described in Section 2.3.1.8, *no cooperative control methods have been developed to handle adversaries resiliently using only local information*. Moreover, as of yet, *no researchers have studied synchronization in the presence of adversaries*.[17]

[17]Resilient clock synchronization has been studied [129, 109, 120]. However, these techniques achieve agreement resiliently on logical clock values, instead of agreement on the oscillator dynamics. In the jargon of this manuscript, this type of clock synchronization is physically dependent consensus, but *not* synchronization.

CHAPTER III

DEPLOYMENT OF PASSIVE SYSTEMS IN UNCERTAIN NETWORKS

Modern surveillance and convoy tracking applications often require deploying groups of unmanned robotic vehicles, such as unmanned aerial vehicles (UAVs). The benefit of using multiple vehicles is redundancy, which reduces the likelihood of missing interesting events on the ground, in the presence of obstructions caused by nonuniform terrain, vegetation, or man-made structures. Further, the additional vehicles provide greater breadth of coverage and increased fidelity of information [31, 99]. A central task for such multi-agent systems is to establish a formation around an area of interest. For example, an n-gon with a target as its center, at the appropriate radius, may simultaneously provide significant redundancy and breadth of coverage. While there are many benefits to networked multi-agent systems in such scenarios, there are also many challenges. One major challenge concerns implementation. For any distributed control approach to be feasible over packet-switched networks, the protocol should be robust with respect to network uncertainties, such as time-varying delays and data loss.

Many works recently have sought to at least partially address network uncertainties, while focusing on specific classes of coordination tasks. In [7], Arcak analyzes a passive interconnection structure suitable for formation control and group agreement problems that unifies other prior work [148, 184, 155], and addresses data loss for the agreement problem by showing that the communication topology may be time-varying, as long as it remains connected in an integral sense. The passive design technique of [7] is used in one of the designs in [89] for synchronized path following, and likewise, can tolerate data loss. A similar work, predating [7], is [35], which studies output synchronization of passive agents with linear and nonlinear coupling among agents and can handle switching topology and constant time delays under appropriate assumptions. In [88], attitude synchronization is studied in the presence of constant time delays and data loss. Using partial contraction analysis and wave variables, the agreement problem with time delays is studied in [201]. In [200], an iterative algorithm is proposed to dynamically track the average of the time-varying signals measured by individual sensors whenever the directed links are subjected to multiplicative and additive noise.

The implementation challenge in multi-agent networks is a special case of the more general integration problem in Cyber-Physical Systems (CPS) [183]. As in CPS, the agents in a multi-agent network have physical dynamics, and the control computation and coordination is done using embedded microprocessors and communication devices. This inherent heterogeneity leads to complex cross-layer interactions between the cyber layers of the system (e.g., the software, network, and platform) and the physical layers of the system (e.g., control and actuation). Because of the high cost of integration, it is important to develop methods that decouple the design concerns across layers without neglecting the interactions that arise in integration. The work presented in this chapter makes a step in this direction by focusing on decoupling the stability of the networked multi-agent system from uncertainties in the network.

The coordination task studied in this chapter is for the networked multi-agent system to achieve *deployment*. The deployment problem is similar to *pattern formation* [182] or *formation stabilization* [154] (see Section 2.3.2), which are generalizations of the *rendezvous problem* [42, 123, 124] (see Section 2.3.1.4). Specifically, the *deployment problem* requires that a set of mobile agents asymptotically converge to predetermined points in space.[18] Since the destinations are known beforehand, it is not necessary that the agents coordinate to achieve this objective. However, networking the agents may be beneficial in order to coordinate the time of arrival of the agents (i.e., when the agents arrive at their destinations), among other reasons. But, the introduction of the network is accompanied by associated uncertainties such as time-varying network delays and data loss.

In contrast to what has been studied in the literature, we consider persistent network disturbances in the form of both time-varying delays and data loss [111, 112]. Here, each agent is assumed to know its destination beforehand, and communicates with other agents in the network in order to approximately synchronize the agents' times of arrival at their destination. We propose a passivity-based deployment coordination algorithm, or protocol, that leverages the inherent stability and compositionality properties of passive systems. In order to make the communication robust to network uncertainties, we use the *wave variable formalism* that has been used successfully in bilateral teleoperation [34], port-Hamiltonian systems [175], and network control [103, 79, 108, 104].

The main contribution of this work is that we demonstrate through our deployment protocol the

[18]Deployment differs from pattern formation because in the deployment problem, the points to which the individual agents converge are preassigned. In pattern formation, the formation pattern may be given beforehand, but the specific destinations of the agents are not predetermined.

feasibility of maintaining passivity and stability in general bidirectional networks in the presence of both time-varying delays and data loss. Under appropriate assumptions, we show that the protocol achieves deployment in connected, non-bipartite overlay networks in the presence of uncertainties. We further demonstrate through a set of simulation examples that the protocol induces an emergent flocking-like behavior in which the agents arrive at their destinations at approximately the same time. For these examples, we discuss how to parameterize the protocol to achieve good performance in this regard, and describe how to implement the protocol without algebraic loops.

The rest of the chapter is organized as follows. Section 3.1 provides background material on passivity and l_2-stability needed for the technical results. The system model and problem statement are given in Section 3.2. The deployment protocol is described in Section 3.3. The analytical results on passivity and stability are given in Section 3.4, and the deployment results are presented in Section 3.5. Section 3.6 discusses the design concerns when parameterizing the protocol to achieve good performance. The design techniques described in Section 3.6 are then used in a set of simulations of a network of velocity-limited vehicles in Section 3.7. The network of planar vehicles is simulated over a range of network conditions. Finally, a summary is provided in Section 3.8.

3.1 Background

Recall that a signal $f\colon \mathbb{Z}_{\geq 0} \to \mathbb{R}^m$ is in the l_2-space of functions if it has finite energy; i.e.,

$$\sum_{t=0}^{\infty} f^{\mathsf{T}}(t)f(t) < \infty.$$

Because l_2 includes only a subset of those functions that asymptotically approach zero, it is useful to consider the *extended l_2-space* of functions, denoted l_{2e} (or l_{2e}^m to emphasize the dimensionality of the signals). A signal is in l_{2e}^m whenever any finite truncation of the signal has finite energy. Therefore, l_{2e}^m includes all signals that do *not* have finite escape time. In formal terms, define the N-truncation operator $(\cdot)_N$, $N \in \mathbb{N}$, which nullifies function values for indices larger than $N-1$. That is,

$$(f)_N = \begin{cases} f(t) & 0 \leq t \leq N-1; \\ 0 & \text{otherwise.} \end{cases}$$

For all signals f and g in l_{2e}^m define

$$\langle f, g \rangle_N \triangleq \sum_{t=0}^{N-1} f^{\mathsf{T}}(t) g(t)$$

and

$$\|(f)_N\|_2^2 \triangleq \langle f, f \rangle_N.$$

Then, $f \in l_{2e}^m$ whenever $\|(f)_N\|_2^2 < \infty$ for all $N \in \mathbb{Z}_{\geq 0}$. The agents communicate and process signals in l_{2e}^m.

Because of the nondeterministic nature of network uncertainties, it is useful to formulate the relationships between inputs and outputs of the network in terms of relations instead of maps. A *relation* R is a subset of $l_{2e}^m \times l_{2e}^m$ which identifies the set of possible outputs in l_{2e}^m for each input $u \in l_{2e}^m$. We use definitions for l_2^m-stability and passivity for discrete-time systems, which are analogous to the continuous-time counterparts in [48]:

Definition 3.1. *The discrete-time system with relation $R \subset l_{2e}^m \times l_{2e}^m$ is l_2^m-**stable** if*

$$u \in l_2^m, (u, y) \in R \implies y \in l_2^m.$$

Definition 3.2. *Let $R \subset l_{2e}^m \times l_{2e}^m$. Then,*

1. *R is **passive** if there exists some constant $\beta \in \mathbb{R}_{\geq 0}$ (the bias) such that*

$$\langle y, u \rangle_N \geq -\beta, \quad \forall (u, y) \in R \text{ and } N \in \mathbb{N};$$

2. *R is **strictly output passive** if there exists some constants $\beta \in \mathbb{R}_{\geq 0}$ and $\epsilon > 0$ such that*

$$\langle y, u \rangle_N \geq \epsilon \|(y)_N\|_2^2 - \beta, \quad \forall (u, y) \in R \text{ and } N \in \mathbb{N}.$$

Finally, all signals are assumed to be zero for all $t < 0$.

3.2 System Model and Problem Statement

3.2.1 Multi-Agent Network Model

The agents share an inertial coordinate system, have ideal clocks, and communicate over a synchronous network, where execution progresses in discrete time steps of size $T \in \mathbb{R}_{>0}$ and indexed by $t \in \mathbb{Z}_{\geq 0}$. The network is modeled by a connected undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where the node set $\mathcal{V} = \{1, 2, \ldots, n\}$ describes the n agents and $\mathcal{E} \subseteq \binom{\mathcal{V}}{2}$ models the bidirectional communication. The network $\mathcal{G}$ is a logical *overlay graph* that describes how packets are routed in the network. Therefore, $\{i, j\} \in \mathcal{E}$ indicates that information can be exchanged between agents i and j, possibly over multiple hops in the network. It is assumed that the physical layer of the network maintains the appropriate connectivity to ensure the logical overlay network remains connected. Additionally, time-varying, nonnegative integer weights on the directed edges of the associated digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E}_{\mathcal{D}})$ of $\mathcal{G}$[19] are used to model asymmetric, time-varying delays required to route the packets throughout the network. The delays are denoted by $d_{ij}(t) \in \mathbb{Z}_{\geq 0}$ for each directed edge $(i, j) \in \mathcal{E}_{\mathcal{D}}$. In order to denote packet loss and delays, we define the outgoing value sent from node i to node j on directed edge $(i, j) \in \mathcal{E}_{\mathcal{D}}$ at time $t \in \mathbb{Z}_{\geq 0}$ by $u_{ij}(t) \in \mathbb{R}^m$. The incoming value received by node j from node i on directed edge $(i, j) \in \mathcal{E}_{\mathcal{D}}$ at time $t \in \mathbb{Z}_{\geq 0}$ is denoted $v_{ij}(t) \in \mathbb{R}^m$. Since it is entirely possible that no value is received by node j from node i at time t, we set $v_{ij}(t) = 0$ for those time indices $t \in \mathbb{Z}_{\geq 0}$ whenever no value is received (either because the value is delayed in the channel, or it has been lost). To illustrate the notation, consider the three node example with nodes i, j, and q in Figure 8.

3.2.2 Problem Statement

The problem of deployment of a networked multi-agent system is defined as follows.

Definition 3.3. *A networked multi-agent system with overlay network modeled by graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, is said to achieve **deployment** if*

$$\lim_{t \to \infty} ||p_i(t) - \nu_i||_2 = 0, \ \forall i \in \mathcal{V}, \tag{11}$$

[19]$\mathcal{E}_{\mathcal{D}}$ contains directed edges (i, j) and (j, i) for each undirected edge $\{i, j\} \in \mathcal{E}$.

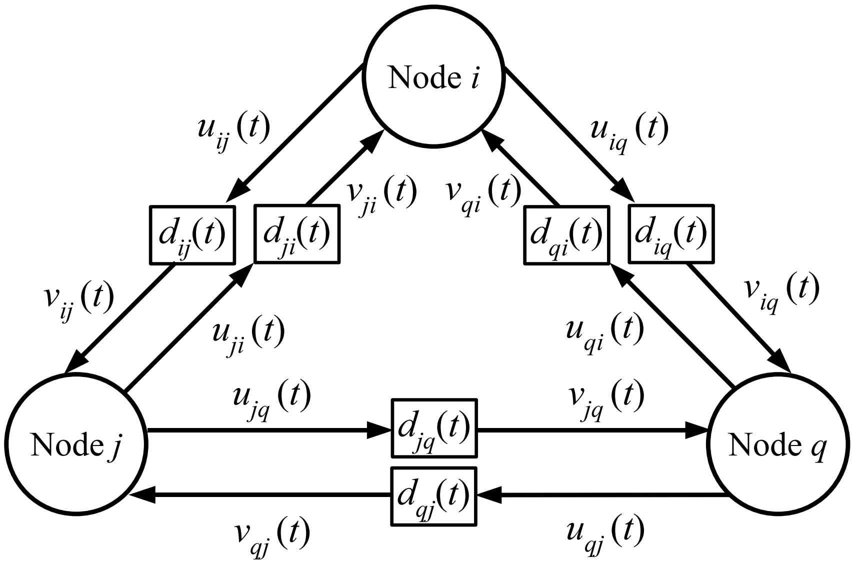


Figure 8: Three agent network example with nodes i, j, and q.

where the position of agent $i \in \mathcal{V}$ is denoted $p_i(t)$ and its destination is given by ν_i.

For a multi-agent network to achieve deployment in the presence of time-varying delays and data loss, (11) must be satisfied for a given model of the time-varying delays and data loss. For the time-varying delays this amounts to constraining the set of allowable delays to some set $\Delta \subseteq \mathbb{Z}_{\geq 0}$, for all $d_{ij}(t)$ such that $(i, j) \in \mathcal{E}_{\mathcal{D}}$, $t \in \mathbb{Z}_{\geq 0}$. Associated with Δ is a sequence of probability mass functions defined on $\mathbb{Z}_{\geq 0}$ from which the values of $d_{ij}(t) \in \Delta$ are drawn to determine the delay of $u_{ij}(t)$ at time t on the directed edge $(i, j) \in \mathcal{E}_{\mathcal{D}}$. In the absence of data loss, we have $v_{ij}(t) = u_{ij}(t - d_{ij}(t))$. To introduce data loss, associate to each pair $(t, d_{ij}(t)) \in \mathbb{Z}_{\geq 0} \times \Delta$ a binary mass function that determines $v_{ij}(t) \in \{u_{ij}(t - d_{ij}(t)), 0\}$. For the sake of generality, we allow for $\Delta = \mathbb{Z}_{\geq 0}$, and place no restrictions on the probability mass functions that determine the values of either $d_{ij}(t) \in \Delta$ or $v_{ij}(t) \in \{u_{ij}(t - d_{ij}(t)), 0\}$. Note that if two or more messages arrive at the same time, we assume that all messages except the latest message are dropped.

3.3 Deployment Protocol

Figure 9 illustrates the agent model, in which r_i is the *reference input*, x_i is the *network feedback*, y_i is the *output*, and $H_i \colon l_{2e}^m \to l_{2e}^m$ is the agent *input-output map* that describes the agent dynamics. The variables x_i and y_i are transformed into the wave domain through the *scattering transforma-*

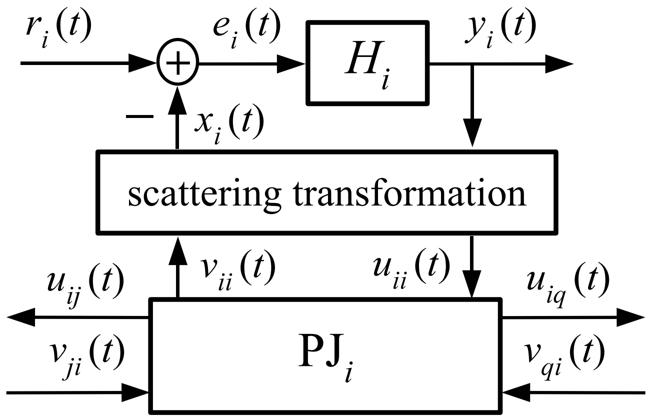


Figure 9: Agent model with protocol components for agent i.

tion [147]. The main idea behind the scattering transformation is to combine two variables (x_i and y_i), as linear combinations, in such a way that models transmitted and reflected waves, u_{ii} and v_{ii}, respectively. Because the energy of a wave is conserved upon delay (assuming no attenuation), wave variables are well-suited to passivity-based techniques while providing additional robustness to delays. The agent's wave variables u_{ii} and v_{ii} are coupled to neighboring agents through the power junction, PJ_i [103]. The deployment protocol combines these transformations and wave variable manipulations with assumptions on the message passing mechanism and agent dynamics. First, we describe the scattering transformation and power junction. Then, we state the assumptions on the packet handling and agent dynamics before formally defining the protocol.

Definition 3.4. *For each $i \in \mathcal{V}$, the **scattering transformation** relates y_i and v_{ii} to x_i and u_{ii} by*

$$x_i(t) = b_i(t)\, y_i(t) - \sqrt{2b_i(t)}\, v_{ii}(t), \tag{12a}$$

$$u_{ii}(t) = \sqrt{2b_i(t)}\, y_i(t) - v_{ii}(t). \tag{12b}$$

This definition is similar to the one in [147], but expressed with the outputs as functions of the inputs. Additionally, the characteristic impedance, $b_i(t) \geq 0$, is allowed to be time-varying. The characteristic impedance has a physical interpretation whenever the variables x_i and y_i are effort and flow[20] variables, respectively. In this case, the product of $b_i(t)$ with a flow variable is an effort

[20]*Effort* and *flow* variables are a generalization of force and velocity, and provide common terminology across multiple physical domains, e.g., translational mechanics (force and velocity), rotational mechanics (torque and angular velocity), electrical (voltage and current), magnetic (current and voltage), and hydraulic (pressure and volume flow). In all cases,

variable, and the scattering transformation has a powerful intuition. The physical unit of the square of each wave variable is *Watts*, and the direction of transmission of each wave variable represents *power flow*. Then, the difference of the transmitted and reflected waves, $u_{ii} - v_{ii}$ is the *net power flow* (where the sign determines the direction in which the net energy is flowing). Even whenever the physical units of x_i and y_i do not admit this interpretation of power flow, the notion of *mathematical power flow* allows the intuition to be generalized. In the latter case, the scattering transformation and characteristic impedance $b_i(t)$ lose their physical interpretation, but the wave variables retain the robustness properties with respect to time delays.

Next, we define the *power junction*, which allows two or more systems to be connected in the wave domain in a passivity-preserving manner [103]. The intuition behind the power junction is to ensure that the net (mathematical) power flow into the junction is always greater than or equal to the net (mathematical) power flow out of the junction. That is, the power junction is defined so that passivity is conserved.

Definition 3.5. *Let $m, p \in \mathbb{N}, p \geq 2$. Then, a* **power junction** *is a map $F\colon l_{2e}^{mp} \to l_{2e}^{mp}$, which satisfies for all $\xi \in l_{2e}^{mp}$ and all $t \in \mathbb{Z}_{\geq 0}$ the inequality $\xi^{\mathsf{T}}(t)\xi(t) \geq F(\xi(t))^{\mathsf{T}} F(\xi(t))$.*

In the definition of the power junction, $\xi(t)$ is formed by concatenating the p inputs of size $m \times 1$. For analyzing the system, it is useful to pair the p inputs to their corresponding outputs in $F(\xi(t))$. For each agent $i \in \mathcal{V}$, the power junction inputs u_{ii} and v_{ji} with $j \in \mathcal{N}_i$ correspond, respectively, to the power junction outputs v_{ii} and u_{ij} with $j \in \mathcal{N}_i$. We implement each agent's power junction as a linear set of equations in order to simplify the steady-state analysis. Specifically, we use the following equations. For each $i \in V$, $j \in \mathcal{N}_i$, and $t \in \mathbb{Z}_{\geq 0}$, the output waves are computed as

$$u_{ij}(t) = \tfrac{1}{\sqrt{d_i}} u_{ii}(t), \tag{13a}$$

$$v_{ii}(t) = \tfrac{1}{\sqrt{d_i}} \sum_{j \in \mathcal{N}_i} v_{ji}(t). \tag{13b}$$

Assuming that $\mathcal{G}$ is connected ensures $d_i > 0$. Recall from Section 3.2.1 that $v_{ji}(t) = 0$ in those time instances t whenever no value is received at node i from node j. We now prove that (13) implements a power junction.

the product of effort and flow is power.

Lemma 3.6. *The implementation defined by (13) satisfies the power junction constraint.*

Proof. From the remarks following the power junction definition, we must show for all $t \in \mathbb{Z}_{\geq 0}$,

$$u_{ii}^{\mathsf{T}}(t)u_{ii}(t)+\sum_{j\in\mathcal{N}_i} v_{ji}^{\mathsf{T}}(t)v_{ji}(t) \geq v_{ii}^{\mathsf{T}}(t)v_{ii}(t)+\sum_{j\in\mathcal{N}_i} u_{ij}^{\mathsf{T}}(t)u_{ij}(t). \tag{14}$$

Clearly, a sufficient condition for satisfying (14) is to enforce the following constraints for each component $l = 1, 2, \ldots, m$:

$$\sum_{j\in\mathcal{N}_i} u_{ij_l}^2(t) \leq u_{ii_l}^2(t) \text{ and } v_{ii_l}^2(t) \leq \sum_{j\in\mathcal{N}_i} v_{ji_l}^2(t).$$

The first inequality holds with equality by direct substitution of u_{ij} from (13a). To show the second inequality, combine (13b) with the Cauchy-Schwarz inequality to get

$$v_{ii_l}^2(t) = \tfrac{1}{d_i}\left(\sum_{j\in\mathcal{N}_i} v_{ji_l}(t)\right)^2 \leq \sum_{j\in\mathcal{N}_i} v_{ji_l}^2(t).$$

□

We now discuss the assumptions on the message passing mechanism. Due to the presence of time-varying delays and data loss, some (or all) of the $v_{ji}(t)$ may not be received at time t, in which case $v_{ji}(t) = 0$. The assumptions on packet handling are summarized as follows:

A1: No retransmission of packets.

A2: Process null packets whenever input buffers are empty.

These assumptions ensure that each channel $(i, j) \in \mathcal{E}$ satisfies the following passive inequality despite time-varying delays and data loss, as described in [34]:

$$\|(v_{ij})_N\|_2^2 \leq \|(u_{ij})_N\|_2^2,\ \forall N \in \mathbb{Z}_{>0}, \tag{15}$$

where

$$v_{ij}(t) \in \{u_{ij}(t - d_{ij}(t)), 0\} \quad \forall t \in \mathbb{Z}_{>0}.$$

Note that reordering of packets is allowed.

Finally, we outline the assumptions on the agent dynamics, which require the following definition.

Definition 3.7. *A dynamic system defined by map $H\colon l_{2e}^m \to l_{2e}^m$ with input u and output y* ***settles at steady state*** *with* ***steady-state gain matrix*** $G \in \mathbb{R}^{m\times m}$ *if*

$$\lim_{t\to\infty} y(t) = G\left(\lim_{t\to\infty} u(t)\right),$$

whenever $\lim_{t\to\infty} u(t)$ exists.

For each agent $i \in \mathcal{V}$, H_i is assumed to be a causal strictly output passive mapping, which settles at steady state with diagonal positive-definite steady-state gain matrix G_i. Furthermore, we assume that the position of the agent p_i is equal to the output y_i at steady state. We are now able to formally define the deployment protocol as follows:

Definition 3.8. *The* ***Deployment Protocol*** *is given by the 6-tuple, $(\mathcal{V}, \mathcal{E}, \mathcal{M}, \mathcal{H}, \mathcal{S}, \mathcal{PJ})$, which consists of*

- $\mathcal{V} = \{1, 2, \ldots, n\}$*, a set of n agents;*
- $\mathcal{E}$*, the edge set that models the bidirectional overlay routing scheme and must result in a connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$;*
- $\mathcal{M}$*, a message passing mechanism ensuring assumptions* ***A1*** *and* ***A2*** *hold;*
- $\mathcal{H} = \{H_1, H_2, \ldots, H_n\}$*, a set of causal, strictly output passive maps describing the agents' dynamics, with diagonal, positive-definite steady-state gain matrices G_1,G_2,…,G_n, respectively, such that $y_i = p_i$ at steady state $\forall i \in \mathcal{V}$;*
- $\mathcal{S}$*, a set of n scattering transformations defined by (12), in which $b_i(t) \geq 0$ may be time-varying for each agent $i \in \mathcal{V}$;*
- $\mathcal{PJ}$*, a set of n power junctions implemented as in (13) for each agent $i \in \mathcal{V}$.*

3.4 Passivity and Stability Analysis

In this section, we present the main contribution of the chapter; namely, that the deployment protocol ensures strict output passivity and l_2^m-stability of the networked multi-agent system in the presence of network uncertainties. For compactness we use the following notation. Define the global input r and output y by $r = [r_1^\mathsf{T}, \ldots, r_n^\mathsf{T}]^\mathsf{T}$ and $y = [y_1^\mathsf{T}, \ldots, y_n^\mathsf{T}]^\mathsf{T}$, respectively. Similarly, define $x = [x_1^\mathsf{T}, \ldots, x_n^\mathsf{T}]^\mathsf{T}$ and $e = [e_1^\mathsf{T}, \ldots, e_n^\mathsf{T}]^\mathsf{T}$. Denote the mapping $H\colon l_{2e}^{nm} \to l_{2e}^{nm}$, where $y = H(e)$ is defined by each agent mapping $y_i = H_i(e_i)$ for each $i \in \mathcal{V}$. Finally, denote the relation relating r to y by $R_{ry} \subset l_{2e}^{nm} \times l_{2e}^{nm}$. The following lemma proves that the network, described by the relation R_{yx} shown in Figure 10, is passive despite time-varying delays and data loss.[21]

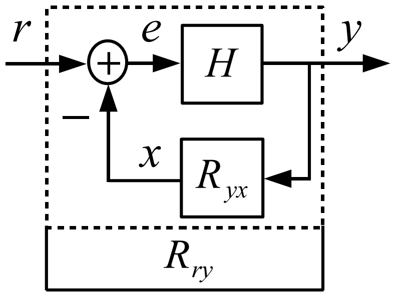


Figure 10: Feedback interconnection of the global agent map with the network relation.

Lemma 3.9. *Consider a multi-agent system modeled by connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ that satisfies the deployment protocol of Definition 3.8. Then,*

$$\sum_{i=1}^{n} \left(\|(u_{ii})_N\|_2^2 - \|(v_{ii})_N\|_2^2 \right) \geq 0, \ \forall N \in \mathbb{N}, \tag{16}$$

even in the presence of time-varying delays and data loss. Moreover, the relation R_{yx} relating y to x is passive.

Proof. Sum the power constraints (14) of each agent i from time $t = 0$ to $t = N - 1$ and then sum the resulting inequalities over all $i \in \mathcal{V}$. Then, invoke (15) to obtain

$$\begin{aligned}\sum_{i=1}^{n} &(\|(u_{ii})_N\|_2^2 - \|(v_{ii})_N\|_2^2) \\ &\geq \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i} \left(\|(u_{ij})_N\|_2^2 - \|(v_{ij})_N\|_2^2 \right) \geq 0, \ \forall N \in \mathbb{N}.\end{aligned}$$

To show that R_{yx} is passive, consider the following power constraint, which may easily be derived

[21]Observe that R_{yx} is a relation and not a map because of the nondeterminism caused by network uncertainties.

from (12a) and (12b), regardless of the value of $b_i(t)$:

$$\tfrac{1}{2}(u_{ii}^{\mathsf{T}}(t)u_{ii}(t) - v_{ii}^{\mathsf{T}}(t)v_{ii}(t)) = x_i^{\mathsf{T}}(t)y_i(t),\ \forall t \in \mathbb{N}. \tag{17}$$

Substitute (17) into (16) to obtain

$$\sum_{i=1}^{n} \langle x_i, y_i \rangle_N \geq 0,\ \forall N \in \mathbb{N}. \tag{18}$$

Then, it follows that $\langle x, y \rangle_N \geq 0$, which is true with network uncertainties, and therefore holds for all $(y, x) \in R_{yx}$, which satisfies the definition of passivity in Definition 3.2, with $\beta = 0$. □

Next, we show that the networked multi-agent system, illustrated in Figure 10, is strictly output passive and l_2^m-stable.

Theorem 3.10 (Passivity and Stability)**.** *The relation R_{ry} is strictly output passive and l_2^m-stable despite time-varying delays and data loss.*

Proof. First we show that R_{ry} is strictly output passive and then show that this implies l_2^m-stability. Since each H_i is strictly output passive, there exists $\epsilon_i > 0$ and β_i, for all $i \in V$, such that

$$\langle y_i, e_i \rangle_N \geq \epsilon_i \|(y_i)_N\|_2^2 - \beta_i, \quad \forall N \in \mathbb{N}.$$

Summing this inequality over all $i \in \mathcal{V}$ leads to

$$\langle y, e \rangle_N \geq \epsilon \|(y)_N\|_2^2 - \beta, \quad \forall N \in \mathbb{N},$$

in which $\epsilon = \min_i\{\epsilon_i\}$ and $\beta = \sum_{i=1}^{n} \beta_i$. This proves that H in Figure 10 is strictly output passive. Now, rewriting this inequality by substituting $e = r - x$, then adding $\langle x, y \rangle_N$ to both sides of the resulting inequality, and finally invoking (18) results in

$$\langle y, r \rangle_N \geq \epsilon \|(y)_N\|_2^2 - \beta, \quad \forall N \in \mathbb{N},$$

which is true with network uncertainties, and therefore holds for all $(r, y) \in R_{ry}$.

Now, it is well known in continuous-time [48] and has been shown for discrete-time [100] that a sufficient condition for a system to have finite l_2^m-gain is for the system to be strictly output passive. Moreover, finite l_2^m-gain is a sufficient condition for l_2^m-stability [48, 100]. $\square$

Although the agents are strictly output passive and therefore l_2^m-stable, the results of this section ensure that the interactions through the network do not destabilize the system of agents. Next, we address the deployment problem.

3.5 Deployment using Steady-State Analysis

To analyze the behavior of the networked multi-agent system, we consider the system at steady state by assuming the network is ideal. Without network uncertainties, the relations, R_{yx} and R_{ry}, become mappings H_{yx} and H_{ry}, respectively. By assuming H_{yx} and H_{ry} settle at steady state, we have the following result. For notational brevity, define $\lim_{t\to\infty} f_i(t) = f_{i,\infty}$ for any function f_i for which the limit exists.

Theorem 3.11. *Suppose the multi-agent network is modeled by connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and satisfies the deployment protocol of Definition 3.8. Further, suppose there are no network uncertainties, and that both H_{yx} and H_{ry} settle at steady state. Assume that limits $b_{i,\infty}$ and $r_{i,\infty}$ exist for all $i \in \mathcal{V}$, and either $b_{i,\infty} = 0$ for all $i \in \mathcal{V}$ or $b_{i,\infty} > 0$ for all $i \in \mathcal{V}$. Then the steady-state output of agent i is given by*

$$y_{i,\infty} = \begin{cases} M_i \left(r_{i,\infty} + \frac{\sqrt{2b_{i,\infty}}}{\sqrt{d_i}} \sum_{j\in\mathcal{N}_i} \frac{1}{\sqrt{2b_{j,\infty}d_j}} \left(r_{j,\infty} + K_j y_{j,\infty}\right)\right) & \text{if } b_{i,\infty} > 0, \forall i \in \mathcal{V}; \\ G_i r_{i,\infty} & \text{if } b_{i,\infty} = 0, \forall i \in \mathcal{V}; \end{cases} \tag{19}$$

where $M_i = (b_{i,\infty}G_i + I_m)^{-1}G_i$, $K_j = (b_{j,\infty}G_j - I_m)G_j^{-1}$, and I_m is the identity matrix in $\mathbb{R}^{m\times m}$.

Proof. Since H_{ry} and H_{yx} settle at steady state and $r_{i,\infty}$ exists for all $i \in \mathcal{V}$, it follows that y_∞ exists and, in turn, x_∞ also exists. Therefore, e_∞ exists. For the case $b_{i,\infty} = 0$ for all $i \in \mathcal{V}$, it follows from (12a) that $x_{i,\infty} = 0$ for all $i \in \mathcal{V}$. Using $e_{i,\infty} = r_{i,\infty} - x_{i,\infty}$ and $y_{i,\infty} = G_i e_{i,\infty}$, we conclude $y_{i,\infty} = G_i r_{i,\infty}$.

If, on the other hand, $b_{i,\infty} > 0$ for all $i \in \mathcal{V}$, we deduce from (12) and (13) that all steady-state values of wave variables exist. Using $e_{i,\infty} = r_{i,\infty} - x_{i,\infty}$ and $y_{i,\infty} = G_i e_{i,\infty}$ we get $y_{i,\infty} =$

$G_i(r_{i,\infty} - x_{i,\infty})$. Substitute (12a) into $y_{i,\infty} = G_i(r_{i,\infty} - x_{i,\infty})$, and solve for $y_{i,\infty}$ to get

$$y_{i,\infty} = (b_{i,\infty} G_i + I_m)^{-1} G_i \left(r_{i,\infty} + \sqrt{2b_{i,\infty}} v_{ii,\infty} \right). \tag{20}$$

Combining $v_{ji,\infty} = u_{ji,\infty}$ with (13a) at node j (roles of j and i are reversed) produces $v_{ji,\infty} = u_{jj,\infty}/\sqrt{d_j}$. Substituting this into (13b) for node i yields

$$v_{ii,\infty} = \tfrac{1}{\sqrt{d_i}} \sum_{j \in \mathcal{N}_i} \tfrac{1}{\sqrt{d_j}} u_{jj,\infty}. \tag{21}$$

Now, solve (12a) at node j for $v_{jj,\infty}$ and substitute the result into (12b) at node j to get

$$u_{jj,\infty} = \frac{1}{\sqrt{2b_{j,\infty}}} \left(x_{j,\infty} + b_{j,\infty} y_{j,\infty} \right).$$

Then, solve $y_{j,\infty} = G_j(r_{j,\infty} - x_{j,\infty})$ for $x_{j,\infty}$ and substitute into the previous result to get

$$u_{jj,\infty} = \frac{1}{\sqrt{2b_{j,\infty}}} \left(r_{j,\infty} + (b_{j,\infty} G_j - I_m) G_j^{-1} y_{j,\infty} \right).$$

Substitute this into (21) to get

$$v_{ii,\infty} = \tfrac{1}{\sqrt{d_i}} \sum_{j \in \mathcal{N}_i} \tfrac{1}{\sqrt{2b_{j,\infty} d_j}} \left(r_{j,\infty} + (b_{j,\infty} G_j - I_m) G_j^{-1} y_{j,\infty} \right).$$

Finally, substitute this equation into (20) to obtain the first case of (19). □

Theorem 3.11 provides a system of equations for a network of non-identical agents at steady state. The following theorem characterizes the system of equations for a network of identical agents. Before stating the theorem, we require the definition of the Kronecker product of two matrices. Given $B \in \mathbb{R}^{m \times n}$ and $C \in \mathbb{R}^{p \times q}$, the *Kronecker product* $B \otimes C \in \mathbb{R}^{mp \times nq}$ is defined as [80]

$$B \otimes C \triangleq \begin{bmatrix} b_{11}C & \dots & b_{1n}C \\ \vdots & \ddots & \vdots \\ b_{m1}C & \dots & b_{mn}C \end{bmatrix}.$$

Theorem 3.12 (Identical Agents)**.** *Given the assumptions of Theorem 3.11, but with identical agents* *(i.e., $G_i \equiv G$, $b_{i,\infty} \equiv b_\infty \; \forall i \in \mathcal{V}$), let $A_w = \Delta^{-1/2} A \Delta^{-1/2}$, where A is the adjacency matrix and Δ is the degree matrix of the connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Define $M = (b_\infty G + I_m)^{-1} G$ and $K = (b_\infty G - I_m) G^{-1}$. Then, whenever $b_\infty > 0$,*

$$y_\infty = ((A_w + I_n) \otimes M) r_\infty + (A_w \otimes MK) y_\infty. \tag{22}$$

Assuming the inverse of $(A_w + I_n)$ exists, then the limit r_∞ that ensures that the agents will converge to the desired locations $p_\infty = [p_{1,\infty}^\mathsf{T}, \ldots, p_{n,\infty}^\mathsf{T}]^\mathsf{T}$ is given by

$$r_\infty = \left((A_w + I_n)^{-1} \otimes M^{-1}\right) (I_{mn} - (A_w \otimes MK)) p_\infty. \tag{23}$$

If $b_\infty = 0$, then $r_{i,\infty} = G_i^{-1} p_{i,\infty}$.

Proof. If $b_\infty = 0$, it follows directly from Theorem 3.11 that $r_{i,\infty} = G_i^{-1} y_{i,\infty}$. Since we assume $y_{i,\infty} = p_{i,\infty}$ for all $i \in \mathcal{V}$ (see Definition 3.8), the result follows. Therefore, assume $b_\infty > 00$, and observe the following equivalencies:

$$\frac{1}{\sqrt{d_i}} \sum_{j \in \mathcal{N}_i} \frac{1}{\sqrt{d_j}} r_{j,\infty}, \; \forall i \in \mathcal{V} \iff (\Delta^{-1/2} A \Delta^{-1/2} \otimes I_m) r_\infty,$$

$$M r_{i,\infty}, \; \forall i \in \mathcal{V} \iff (I_n \otimes M) r_\infty.$$

An analogous equivalency holds for left multiplication by K. Therefore, the system of steady-state equations is given by

$$y_\infty = (I_n \otimes M) \left(r_\infty + (A_w \otimes I_m)(r_\infty + (I_n \otimes K) y_\infty)\right).$$

This equation can be reduced to (22) by using the distributive property along with the following property of Kronecker products: $(B \otimes C)(D \otimes F) = BD \otimes CF$, which holds whenever the dimensions of the matrix products agree [80]. Finally, (23) follows from (22) by the fact that $p_\infty = y_\infty$ and the following property: $((A_w + I_n) \otimes M)^{-1} = (A_w + I_n)^{-1} \otimes M^{-1}$, which is valid by hypothesis since G and $(A_w + I_n)$ are invertible. □

Theorem 3.12 lays the groundwork necessary to show the class of overlay networks in which deployment is achieved.

Theorem 3.13 (Deployment of Identical Agents with No Uncertainties)**.** *Consider a multi-agent network of identical agents (i.e., $G_i \equiv G$, $b_i \equiv b \; \forall i \in \mathcal{V}$) modeled by connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Suppose the networked multi-agent system satisfies the deployment protocol of Definition 3.8 and there are no network uncertainties present. Suppose further that H_{yx} and H_{ry} settle at steady state, and that b_∞ exists. Then deployment is achieved whenever $\mathcal{G}$ is non-bipartite, with r given by*

$$r(t) = \left((A_w + I_n)^{-1} \otimes M^{-1}\right)\left(I_{mn} - (A_w \otimes MK)\right)\nu, \tag{24}$$

for all $t \geq 0$, with $\nu = [\nu_1^\mathsf{T}, \nu_2^\mathsf{T}, \ldots, \nu_n^\mathsf{T}]^\mathsf{T}$, $M = (b_\infty G + I_m)^{-1} G$, and $K = (b_\infty G - I_m)\, G^{-1}$.

Proof. To prove the result, we first show that $A_w + I_n$ is invertible if and only if $\mathcal{G}$ is not bipartite. To this end, we show that A_w has spectral radius $\rho(A_w) = 1$, so that $A_w + I_n$ is singular if and only if $-\rho(A_w)$ is an eigenvalue of A_w. Indeed, the Laplacian is defined as $I_n - A_w$ in [36] and is shown to have its spectrum within the interval $[0, 2]$, with 2 as an eigenvalue if and only if there is a connected component of $\mathcal{G}$ that is bipartite and nontrivial [36, Lemma 1.7]. Therefore, (24) is well defined. Since $r(t) \equiv r_\infty$, the conditions of Theorem 3.12 are met. Substituting the value for r in (24) into (22) yields

$$y_\infty - \nu = (A_w \otimes MK)(y - \nu),$$

which has solution $y - \nu \neq 0$ if and only if 1 is an eigenvalue of $(A_w \otimes MK)$. By the reasoning above, the spectrum of A_w lies within the interval $(-1, 1]$ for non-bipartite communication graphs. A simple calculation shows that the eigenvalues of MK have the form $(b_\infty g_i - 1)/(b_\infty g_i + 1)$, in which g_i are the positive diagonal entries of G, for $i = 1, 2, \ldots, m$. Since $b_\infty g_i \geq 0$, the spectrum of MK lies within the interval $[-1, 1)$. Thus, no product of eigenvalues of A_w and MK can be unity. Hence, $p_{i,\infty} = y_{i,\infty} = \nu_i$ for all $i \in \mathcal{V}$, and (11) is satisfied. $\square$

Next, we consider the problem of eliminating steady-state error in the presence of network uncertainties, which is achieved by using a time-varying characteristic impedance and eventually setting it to zero.

Theorem 3.14 (Deployment of Identical Agents with Uncertainties)**.** *Consider a multi-agent network of identical agents modeled by connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ in which the deployment protocol* *of Definition 3.8 is satisfied. Then (11) is achieved in the presence of network uncertainties*[22] *by setting* $b_i(t) = 0$ *and* $r_i(t) = G_i^{-1}\nu_i$ *for all* $t \geq t_0$*, in which* $t_0 \in \mathbb{Z}_{\geq 0}$.

Proof. Since $x_i(t) \equiv 0$ whenever $b_i(t) = 0$ (c.f., (12a) and Figure9), it follows that

$$\lim_{t\to\infty} p_i(t) = \lim_{t\to\infty} y_i(t) = \lim_{t\to\infty} G_i r_i(t) = \nu_i,\ \forall i \in \mathcal{V}.$$

□

3.6 Design of Deployment Protocol Parameters

In Section 3.5, we showed how the protocol can achieve (11) in the presence of network uncertainties. In this section, we examine qualitatively how the network connectivity, characteristic impedance $b(t)$, and the times to switch between these values affect how well the deployment protocol achieves synchronized deployment in networks of identical agents. Synchronized deployment is defined as follows.

Definition 3.15. *A networked multi-agent system achieves* (ϵ, δ)***-synchronized deployment*** *if (11) is satisfied and* $t_d = t_{max} - t_{min} < \delta$*, in which*

$$t_{min} = \min_t \left\{ t \in \mathbb{Z}_{\geq 0} \mid |p_i(t) - \nu_i|_2 < \epsilon,\ i \in \mathcal{V} \right\},$$
$$t_{max} = \max_t \left\{ t \in \mathbb{Z}_{\geq 0} \mid |p_i(t) - \nu_i|_2 \geq \epsilon,\ i \in \mathcal{V} \right\}.$$

We refer to t_d as the *time difference of arrival*, where t_{min} is the minimum ϵ-rise time and t_{max} is the maximum ϵ-settling time. Note that ϵ and δ are independent of each other in the definition of synchronized deployment. These parameters are used to compare the synchronized deployment performance between different protocols and scenarios. Clearly, smaller values of δ for any small fixed ϵ indicate better performance.

The network connectivity determines how y affects x in the feedback connection of Figure 10. This influence is complicated by the scattering transformation and manipulations performed by the power junction. However, to clarify, the power junction produces a weighted average of the wave

[22] Recall from Section 3.2.2 that the class of network uncertainties considered allow for $\Delta = \{d_{ij}(t) \in \mathbb{Z}_{\geq 0} \colon (i,j) \in \mathcal{E}_{\mathcal{D}}, t \in \mathbb{Z}_{\geq 0}\}$, and place no restrictions on the probability mass functions that determine the values of either $d_{ij}(t) \in \Delta$ or $v_{ij}(t + d_{ij}(t)) \in \{u_{ij}(t), 0\}$.

variables of neighboring agents, causing neighboring agents to cluster as they progress toward their destinations. For this reason, it is beneficial to have at least one highly connected agent to better synchronize the time of arrival.

The characteristic impedance, $b(t)$, modulates the amount of influence the network feedback component, x_i, has on the error term, e_i. For realistic networks, with packet loss and time-varying delays, it is important to maintain $b(t) \leq 1$ to limit the degradation of performance caused by a lossy and delay-ridden network. However, a larger b strengthens the coupling in the network, thereby improving synchronization. Hence, there is a natural tradeoff between maintaining a small b for robustness to uncertainties and ensuring b is large for improving synchronization. By switching the values of b over time, it is possible to mitigate these factors and achieve better performance.

For adjusting the times to switch between values of $b(t)$, we use an approach based on the initial and final positions of the agents, and the maximum velocity of the agents in each direction, all of which are assumed to be known by the command center. The method considers two time horizons: t_{rel}, the minimum time for the farthest agent to reach the initial coordinate of the closest agent (or a preset value if the closest agent is too close) in each dimension and t_m, the minimum time for the farthest agent to reach its destination. Both of these times are easily calculated using the maximum velocity, v_{max} of the agent as follows. Let p_{max} denote the largest difference between the initial and final positions in all dimensions. Similarly, let p_{min} denote the smallest difference between the initial and final positions for the dimension corresponding to p_{max} (or a preset value if the closest agent is too close). Then,

$$t_{\text{rel}} = \frac{p_{\text{max}} - p_{\text{min}}}{v_{\text{max}}} \quad \text{and} \quad t_m = \frac{p_{\text{max}}}{v_{\text{max}}}. \tag{26}$$

Using these time horizons as a baseline, the switching times can be set to occur at fractions of t_{rel} or t_m, or at fractions between the two times, in the interval $[t_{\text{rel}}, t_m]$. By experimenting with different switching times, one can determine a set of switching times providing reasonable performance.

3.7 Simulations

The simulations evaluate how well the deployment protocol achieves synchronized deployment in a network of identical agents. We consider two designs of the protocol parameters: the *nominal*

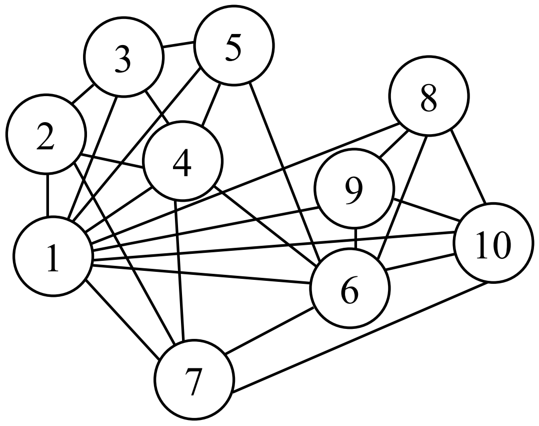


Figure 11: Overlay topology $\mathcal{G}$ of the ten agent network.

design, which is designed for the ideal network, and the *2% packet loss (PL) design*, which is designed for a moderate level of network uncertainties. Finally, we compare the deployment protocol with the case where the agents deploy without a network to show the importance of the network in (ϵ, δ)-synchronized deployment.

Scenario. The experimental setup involves a network of ten identical velocity-limited vehicles with communication overlay network given by the undirected graph shown in Figure 11. The agents move in the plane, and the goal is to form a decagon with each agent 30m from a target centered at the point $(500\text{m}, 500\text{m})$ in an inertial coordinate frame. The initial points of the agents in the inertial coordinate frame are (-270m,-300m), (-100m,100m), (-50m,350m), (50m,150m), (350m,450m), (390m,160m), (340m,-150m), (485m,470m), (500m,125m), and (490m,-180m), respectively. We implement the agent dynamics in Simulink, while TrueTime [149] is used to simulate the network dynamics and communication between agents. The network protocol used is IEEE 802.11b, with a speed of 11 Mbps. The network sampling interval is set to $T = 0.01\text{s}$.

Agent Design. For simplicity, each agent $i \in \mathcal{V}$ is modeled as the point mass, $H_{M_i} : f_i \to p_i$, in which $f_i \in \mathbb{R}^2$ is the control force, M is the mass of the vehicle, and $p_i \in \mathbb{R}^2$ is the position, as depicted in Figure 12. The equations of motion are

$$\dot{p}_i(t) = v_i(t),$$
$$M\dot{v}_i(t) = f_i(t).$$

With this model, we design the position control system H_{cp_i}, shown in Figure 12. The inner

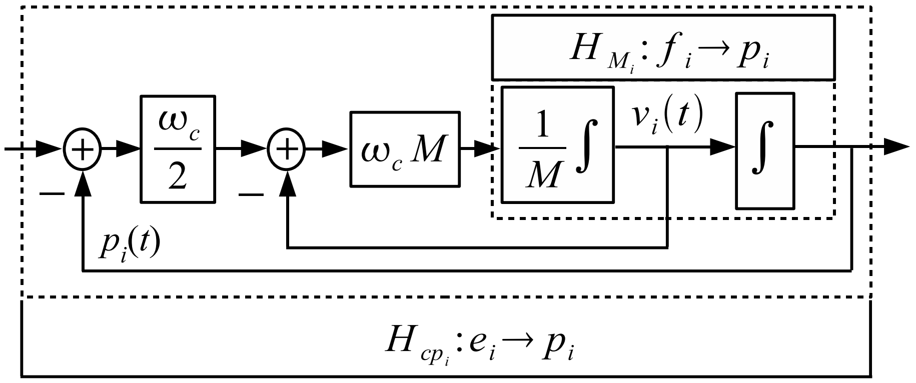
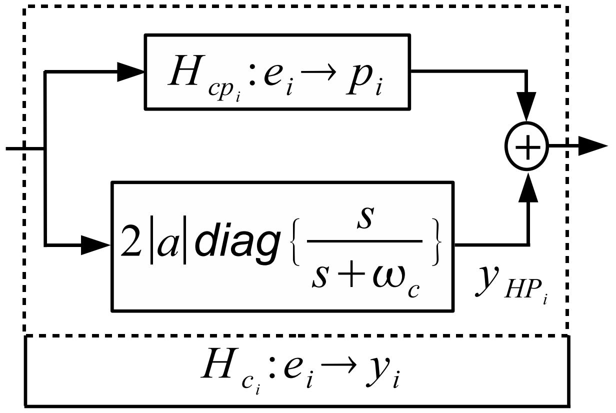


Figure 12: Continuous-time point mass model.

loop gain of the compensator is $\omega_c M$ ($\omega_c > 0$) and the outer loop gain $\omega_c/2$. The dynamics,

$$\ddot{p}_i = -\omega_c \dot{p}_i - \omega_c^2/2(p_i - e_i) = -2\zeta\omega_n \dot{p}_i - \omega_n^2(p_i - e_i),$$

indicate a stable second order system with natural frequency $\omega_n = \omega_c/\sqrt{2}$ and damping coefficient $\zeta = 1/\sqrt{2}$, where $p_{i,\infty} = e_{i,\infty}$. It can be shown that the position control system is inside the sector $[a, 1]$, where $a = -1/(2 + 2\sqrt{2})$ [207], [107]. Therefore, H_{cp_i} is not strictly output passive, but by adding a high-pass filter in parallel, the system may be rendered strictly output passive, as depicted in Figure 13. Since $e_{i,\infty} = p_{i,\infty} = y_{i,\infty}$, it follows that $G = I_2$.



$H_{cp_i}: e_i \rightarrow p_i$
$2|a| diag\{\frac{s}{s+\omega_c}\}$
y_{HP_i}
$H_{c_i}: e_i \rightarrow y_i$

Figure 13: Continuous-time passive model.

Next, we discretize the continuous-time design above using a zero-order hold, which in this case maintains strict output passivity. To prevent a feedforward term in the transfer function of the double integrator, we treat each integrator in the point mass model separately, with zero-order holds.

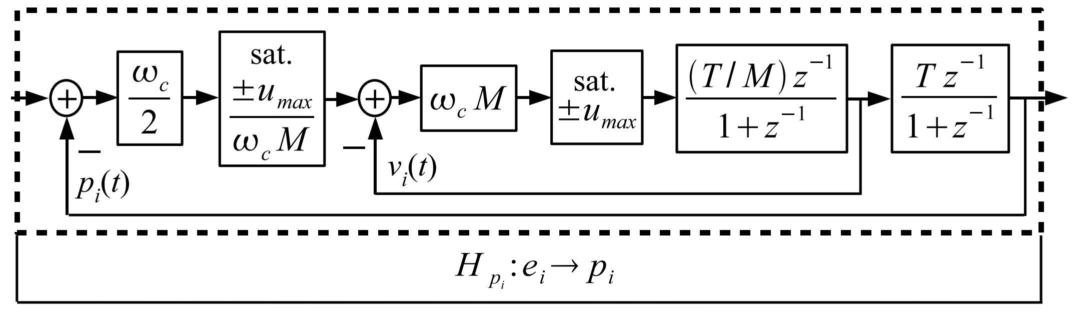

$\frac{\omega_c}{2}$
sat. $\frac{\pm u_{max}}{\omega_c M}$
$\omega_c M$
sat. $\pm u_{max}$
$\frac{(T/M)z^{-1}}{1+z^{-1}}$
$\frac{Tz^{-1}}{1+z^{-1}}$
$-$
$p_i(t)$
$-$
$v_i(t)$
$H_{p_i}: e_i \rightarrow p_i$

Figure 14: Saturated discrete-time model.

This model is necessary to prevent algebraic loops once the saturation blocks and scattering transformation are incorporated into the model. The saturated point mass model is shown in Figure 14. The saturation in the inner loop bounds the control input so that the magnitude for each degree of freedom is at most $u_{\max}$. The saturation in the outer loop bounds the maximum velocity, $v_{\max}$, of the vehicle in each direction.

Agent Implementation. Because the scattering transformation and power junction introduce algebraic loops when implemented as described in Section 3.3, it is necessary to combine the two components in the software implementation. Additionally, due to the feedforward term in the discretized high-pass filter, it is necessary to integrate the feedback term of the scattering transformation, $b(t)y_i(t)$. Since it is not straightforward to integrate this feedback term with the saturated point mass, and $y_i = p_i + y_{HP_i}$, we integrate the feedback of the high-pass filter output locally, which is shown in Figure 15. The modified transfer function of the high-pass filter is then

$$H_{mHP_i}(z) = \frac{2|a|\left(1+z^{-1}\right)}{2b|a| + 1 - (2b|a| + e^{-\omega_c T})z^{-1}}.$$

Finally, Figure 16 shows how the feedback term is implemented to form H_{m_i}. The modified high-pass filter output is saturated using the velocity bound of the vehicle.

In order to combine the scattering transformation with the power junction, first define

$$x_{mi}(t) = x_i(t) - b(t)y_i(t)$$

as the modified network feedback. By substituting (13b) into (12a), and then substituting the result

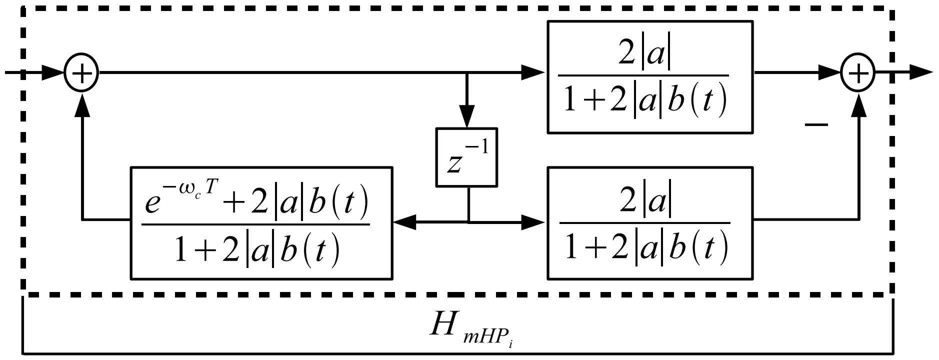

$\frac{2|a|}{1+2|a|b(t)}$
$-$
z^{-1}
$\frac{e^{-\omega_c T}+2|a|b(t)}{1+2|a|b(t)}$
$\frac{2|a|}{1+2|a|b(t)}$
H_{mHP_i}

Figure 15: Modified high-pass filter.

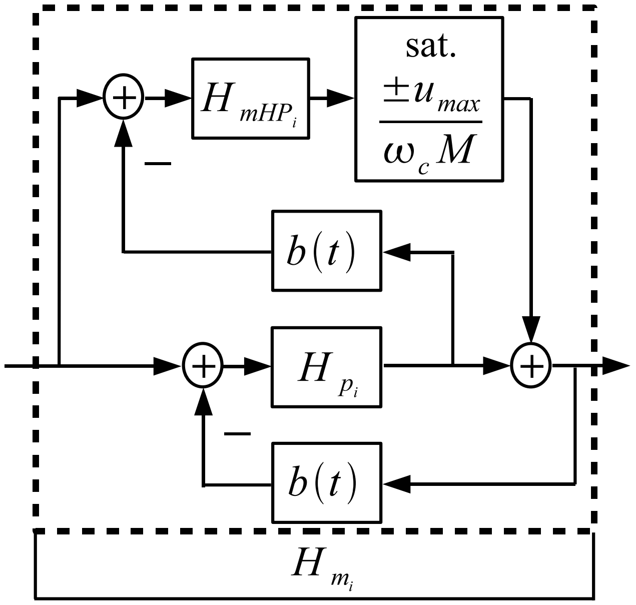


Figure 16: Modified H_i.

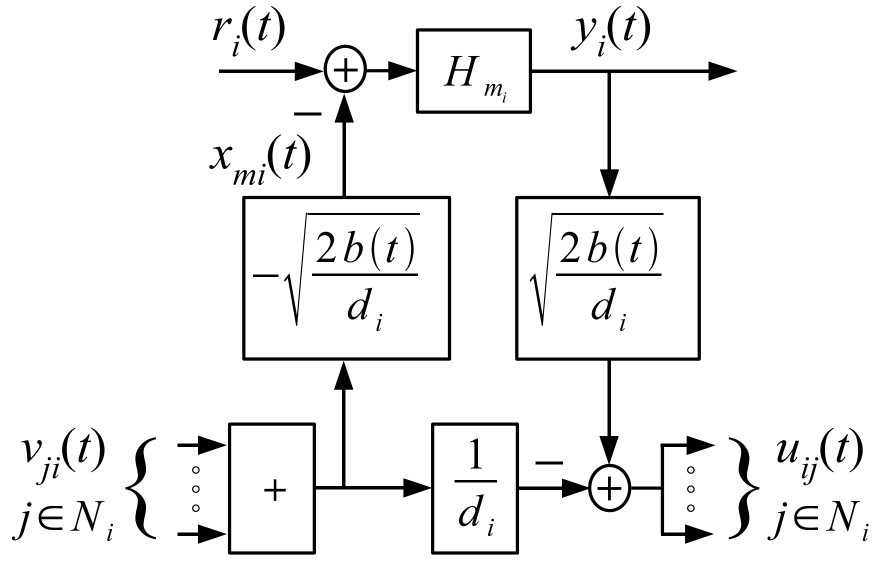


Figure 17: Agent implementation.

into the expression for x_{mi}, we get

$$x_{mi}(t) = -\sqrt{\frac{2b(t)}{d_i}} \sum_{j \in \mathcal{N}_i} v_{ji}(t).$$

Similarly, by substituting (13b) into(12b) and then substituting the result into (13a), we get

$$u_{ij}(t) = \sqrt{\frac{2b(t)}{d_i}} y_i(t) - \frac{1}{d_i} \sum_{j \in \mathcal{N}_i} v_{ji}(t), \quad \forall j \in \mathcal{N}_i.$$

The final implementation is illustrated in Figure 17, where the parameters used for simulation are $\omega_c = 6.283$Hz, $M = 1.826$kg, and $u_{\max} = 40$N (which gives $v_{\max} = 3.49$m/s). These parameters closely match those of a Hummingbird quadrotor that we have in the lab.

Network Implementation. The network dynamics are implemented using both Simulink and TrueTime. We implement constant delays using Simulink, whereas the rest of the network dynamics are implemented using TrueTime (e.g., message passing, packet loss, time-varying delays). Since we do not focus on the physical layer of the network, we set all of the positions of the agents in TrueTime at the origin. We then implement packet loss using Bernoulli random variables in each of the channels of the overlay network. For time-varying delays, we introduce a disturbance node in the network. The disturbance node floods the network with disturbance packets of size P_d with

Table 1: Network conditions: parameterizations used in simulations

Label	PL	Constant Delay	T_d (s)	P_d (bits)
Nominal	0%	0	–	–
1% PL	1%	10ms–500ms	0.01	1000
2% PL	2%	500ms–1s	0.01	2000
10% PL	10%	1s–2s	0.01	10000
25% PL	25%	1s–2s	0.01	10000

Table 2: Characteristic impedance values

$b_{\text{nom}}(t)$	$\mathcal{I}_{\text{nom}}$ ($t \in \mathcal{I}_{\text{nom}}$)	$b_{2\%}(t)$	$\mathcal{I}_{2\%}$ ($t \in \mathcal{I}_{2\%}$)
1	$[0, 159.47)$	1	$[0, 148.08)$
0.8	$[159.47, 189.08)$	0.5	$[148.08, 170.86)$
0.6	$[189.08, 209.59)$	0.2	$[170.86, 182.25)$
0.3	$[209.59, 227.81)$	0.1	$[182.25, 193.64)$
0.2	$[227.81, 228.63)$	0.05	$[193.64, 205.03)$
0.05	$[228.63, 229.04)$	0.01	$[205.03, 225.53)$
0	$[229.04, \infty)$	0	$[225.53, \infty)$

probability 0.5 at each sampling instance given by sampling interval T_d, which is an integer multiple of the network sampling interval T.

Evaluation. For evaluating the deployment protocol, we analyze the synchronized deployment performance by considering statistics on the time difference of arrival, t_d, over a set of simulation runs. To do this, we consider five different operating conditions for the network, shown in Table 1. For each parameterization of the network, we compare the statistics of t_d, for two different designs: the first designed for an ideal network (nominal design) and the second for moderate network uncertainties (2% PL case).

The parameterization for the two designs is summarized in Table 2 with the values of the characteristic impedance and the switching times. In order to compare the performance of the deployment protocol for the two designs, we performed 20 simulations for each design at each operating condition (except for the nominal network, for which we only ran one simulation). We summarize the statistics on the time difference of arrival, t_d, for the nominal design in Table 3 and for the 2% PL design in Table 4, in which $\epsilon = 0.5$m.

As one may expect, in each case, the performance is best at the operating condition for which it was designed. The nominal design has a very small time difference of arrival for the nominal network, at just 2.88s. But, the mean of t_d increases an order of magnitude with even small amounts

Table 3: Statistics of t_d (s) for $\epsilon = 0.5$m; designed for nominal case

Net. Cond.	mean	std dev	median	min	max
Nominal	2.88	0.0	2.88	2.88	2.88
1% PL	27.51	12.82	22.70	11.37	54.81
2% PL	26.20	16.67	19.54	7.74	61.73
10% PL	25.83	11.15	21.78	15.63	56.98
25% PL	26.46	7.50	27.08	10.77	45.17

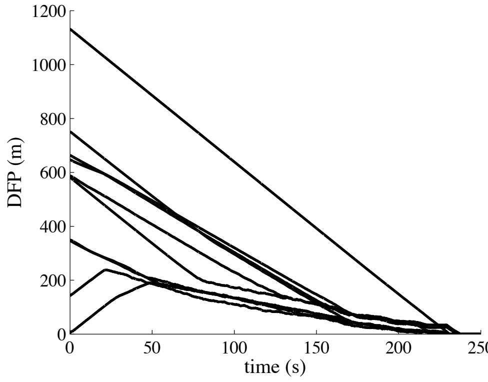
Table 4: Statistics of t_d (s) for $\epsilon = 0.5$m; designed for 2% packet loss (PL)

Net. Cond.	mean	std dev	median	min	max
Nominal	31.98	0.0	31.98	31.98	31.98
1% PL	15.96	19.53	6.29	3.25	73.93
2% PL	12.69	11.43	7.39	3.75	40.48
10% PL	16.85	19.34	7.16	3.36	66.35
25% PL	17.60	20.76	8.07	3.71	83.23

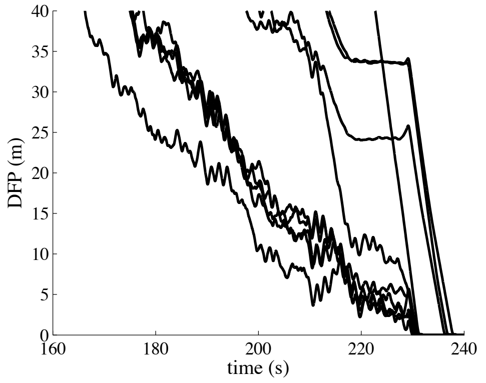
of network uncertainty. However, after the initial drop in performance, the design is robust to increasing network uncertainty. Similarly, the protocol designed for 2% PL performs the best at 2% PL with a sample mean of 12.69s, which is less than half the sample mean of the nominal design under the same network conditions. This design is also robust to increasing network uncertainty, although not as much so as the nominal design. Perhaps the most surprising statistic is how poorly the 2% PL design performs in the nominal case. This is caused by setting $b(t)$ too small.

Consider the distance from final position (DFP) and trajectory plots shown in Figures 18(a)-(d), where we examine only the network with 2% PL conditions. Figures 18(a)-(b) show the DFP plot for the nominal design and Figure 18(c) shows the DPF plot for the 2% PL design. In these figures, we can see the tradeoffs in selecting $b(t)$. Keeping $b(t)$ too large for too long in the presence of network uncertainties can cause the agents to maintain a larger distance from their destinations, degrading the performance as seen in Figure 18(b). On the other hand, selecting $b(t)$ too small can sufficiently decouple the agents from each other so that some of the agents enter the ϵ-neighborhood of their destination. This nearly happens around 177s in Figure 18(c), and is generally why the 2% PL design has greater standard deviation.

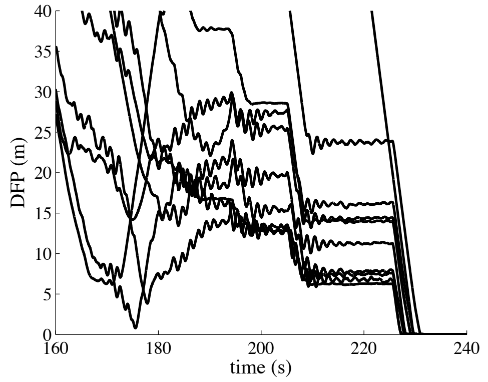
Finally, we compare the synchronized deployment performance of the deployment protocol with a simple approach not requiring a network. Since each agent has steady-state gain matrix $G = I_2$, deployment may be achieved by setting the input of each agent as its destination. But, without the network feedback, the agents do not coordinate to reduce t_d, as seen in Figures 18(e)-(f). In this case



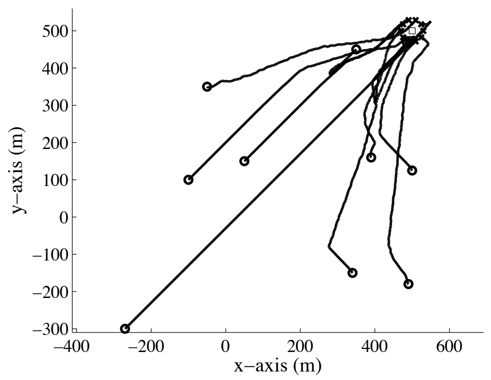
(a) Nominal design DFP with 2%PL.



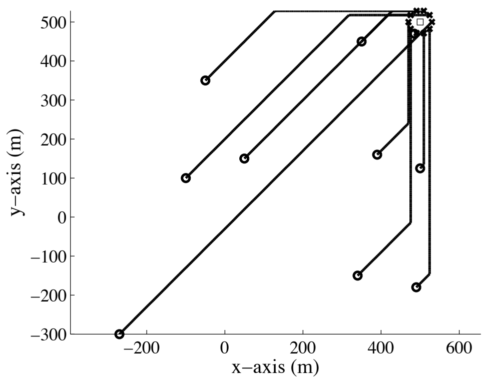
(b) Nominal design DFP with 2%PL.



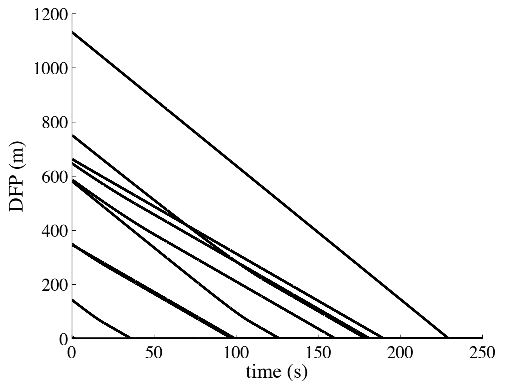
(c) 2%PL design DFP with 2%PL.



(d) Nominal design trajectory with 2%PL.



(e) Trajectory with no network.



(f) DFP with no network.

Figure 18: Distance from final position (DFP) and trajectory plots.

t_d is 227.17s, an order of magnitude greater than the mean of either design using the deployment protocol and two orders of magnitude greater than the nominal design with an ideal network.

3.8 Summary

In this chapter, we present a discrete-time, passivity-based protocol for networked multi-agent systems to address deployment in the presence of network uncertainties. We prove that the deployment protocol ensures passivity and stability of the networked system. We show that deployment is achieved despite network uncertainties for all connected non-bipartite overlay network topologies. Finally, we illustrate the performance qualitatively using simulations of a network of velocity-limited vehicles. The simulations are automatically generated using a modeling language developed in the Generic Modeling Environment (GME) [117] with an interpreter that creates simulation code in Simulink [188] and TrueTime [149].

CHAPTER IV

CONTINUOUS-TIME RESILIENT ASYMPTOTIC CONSENSUS

Due to recent improvements in computation and communication, control system design has made a shift in many applications from centralized to decentralized and distributed approaches. This trend has been fueled by the need for increased flexibility, reliability, and performance in applications such as coordination of vehicle formations [60] and flocking [93]. For these applications and many others, reaching some form of consensus is fundamental to coordination [130, 152]. However, large-scale distributed systems have many entry points for malicious attacks and intrusions. If a security breach occurs, traditional consensus algorithms fail to produce desirable results, and therefore lack robustness [75]. Hence, there is a need for resilient consensus algorithms that guarantee correct behavior even after sustaining security breaches.

Of course, there is a long history in distributed computing of studying consensus problems in the presence of faults and adversarial processors [161, 130]. The most potentially harmful form of adversary is the Byzantine processor, which may behave arbitrarily within the limitations set by the model of computation [110]. Therefore, worst case executions must be considered. Typically, the number of processors that may be Byzantine are bounded and fundamental tight bounds have been established on the ratio of Byzantine to normal processors [110, 50], as well as on the connectivity of the graph representing the communication network [50]. Specifically, the number of Byzantine processors must be less than one third the total number of processors and less than one half the graph connectivity.

From a control theoretic viewpoint, consensus in the presence of adversaries has only been considered recently, and has focused on detection and identification of misbehaving agents in linear consensus networks [157, 158, 160, 159, 181, 186]. While detection is clearly an important problem, these techniques require each agent to have information of the network topology beyond its local neighborhood. This requirement of *nonlocal information* renders these techniques inapplicable to general time-varying networks. Further, the detection algorithms are computationally expensive and do not consider safety constraints on the states of the agents. Using these approaches, it is possible that the adversaries may drive the states of the agents outside of a predetermined safe set during the detection phase, which may not be suitable for certain safety critical applications.

In this chapter, we study a consensus protocol that is low complexity and uses *only local information* to achieve resilience against adversaries in the network. In order to codify a notion of correct behavior of the normal agents in the presence of adversaries, we define a continuous-time resilient consensus problem that has conditions on agreement and safety. For this problem, we study both time-invariant and time-varying (or switching) network topologies with directed information flow. The agents have continuous dynamics and convey state information to each other over a network that switches between a finite number of discrete topologies.

The adversaries studied in this work model compromised nodes that are hijacked in a security breach. The model for the adversaries consists of a threat model and a scope of threat assumption. The threat model defines the types of behaviors allowed by the adversaries. We study Byzantine, malicious, and crash threat models. A Byzantine adversary is similar to a Byzantine fault; it may behave in an arbitrary fashion (within the scope of the model of computation) and may convey different information to different neighbors in the network. The malicious adversary is similar to the Byzantine model, but malicious nodes must convey the same information to all neighbors (i.e., the malicious adversary is a local broadcast version of the Byzantine adversary). Finally, the crash adversary may select a time to crash the compromised node, which forces the states of the compromised node to remain constant. All adversaries are considered omniscient. This means they have access to global information concerning the multi-agent network, including the full network topology, the algorithms used by the other nodes, the states of the other nodes, which nodes are compromised, and the plans of the other adversary nodes. One may take the viewpoint that the individual adversary nodes are controlled by a central commanding adversary. Therefore, worst-case executions must be accounted for.

The scope of threat assumptions bound the number or fraction of compromised nodes either in the entire network or in the neighborhood of any normal node. Whenever, the total number of compromised nodes in the network is assumed to be bounded above by some constant $F \in \mathbb{Z}_{\geq 0}$, we say the adversaries satisfy the F-total model. In cases where it is preferable to make *no global assumptions*, we are interested in other threat assumptions that are strictly local. For example, whenever each node assumes that at most F nodes in its *neighborhood* are compromised (but there is no other bound on the total number of compromised nodes), the scope of threat is

F*-local*. Alternatively, if it is assumed that there is an upper bound on the *fraction*, $0 \leq f \leq 1$, of compromised nodes in any normal node's neighborhood, we say the adversaries satisfy the f-fraction local model.

This chapter does not include our most general results for the continuous-time resilient asymptotic consensus problem. The purpose of this chapter is to introduce the novel continuous-time consensus protocol, the *Adversarial Robust Consensus Protocol* (ARC-P), and to motivate the need for the novel graph theoretic property of *network robustness* (introduced in [210] and extended in [116, 115]). While this chapter is in large part a bridge to the more general results of the following chapter, it provides considerable insight into the nature of the problem and the limitations of classical graph theoretic properties for studying this problem. Moreover, the results of this chapter show exponential convergence rather than just asymptotic convergence. From this perspective, the performance results of this chapter are superior to those presented in the following chapter.

The consensus protocol, ARC-P, is inspired by the *ConvergeApproxAgreement* algorithm [51, 130] and the linear consensus protocol (LCP) [156] (refer to equation (8) on page 52). Specifically, it combines the elimination of extreme values used in the *ConvergeApproxAgreement* algorithm in distributed computing [130, 51], with the standard consensus technique in cooperative control of summing the neighboring relative states as input to an integrator agent [152]. By combining these ideas from distributed computing and control, we obtain a new consensus protocol that is resilient to adversaries, and can be analyzed using system theoretic techniques. In particular, we analyze the Lipschitz continuity of the protocol to ensure the uniqueness of solutions. We prove that ARC-P yields a unique solution that solves the Continuous-Time Resilient Asymptotic Consensus (CTRAC) problem with an exponential rate of convergence (under restricted network topologies).

The rest of the chapter is organized as follows. Section 4.1 outlines the multi-agent network model, the update mode, the CTRAC problem statement, and the adversary models. Section 4.2 describes and defines a simple resilient consensus protocol, ARC-P. Section 4.3 studies the existence and uniqueness of solutions and Lipschitz continuity whenever ARC-P is used in the dynamics of the normal nodes. Section 4.4 defines the Lyapunov candidate function and presents a couple of useful lemmas dealing with the Lyapunov candidate. Section 4.5 then focuses on the special case of complete networks. Section 4.6 studies the limitations of traditional graph theoretic metrics for characterizing the topologies under which ARC-P achieves agreement. To do this, we first consider classes of network topologies defined by in-degree and out-degree conditions and examine the exponential convergence of ARC-P in these classes of networks in Sections 4.6.1–4.6.2. Then, in Section 4.6.3, we examine the degree conditions and present pathological examples of digraphs with high (weak) connectivity that do not enable ARC-P to achieve consensus. Finally, Section 4.7 illustrates the results through simulation, and Section 4.8 summarizes the work.

4.1 System Model and Problem Statement

4.1.1 Multi-Agent Network Model

Consider a time-varying network modeled by the *digraph* $\mathcal{D}(t) = (\mathcal{V}, \mathcal{E}(t))$, where $\mathcal{V} = \{1, ..., n\}$ is the *node set* and $\mathcal{E}(t) \subset \mathcal{V} \times \mathcal{V}$ is the *directed edge set* at time t. The nodes represent the agents in the multi-agent network. Without loss of generality, the node set is partitioned into a set of N *normal agents* $\mathcal{N} = \{1, 2, \ldots, N\}$ and a set of M *adversary agents* $\mathcal{A} = \{N+1, N+2, \ldots, n\}$, with $M = n - N$. The directed edges model the *influence* relationships between the nodes. Each directed edge $(j, i) \in \mathcal{E}(t)$ indicates that node i can be influenced by node j at time t. In this case, we say that agent j *conveys* information to agent i. Let $\Gamma_n = \{\mathcal{D}_1, \ldots, \mathcal{D}_d\}$ denote the set of all digraphs on n nodes, which is of course a finite set. Note that $\mathcal{D}(t) \in \Gamma_n$ for all $t \in \mathbb{R}_{\geq 0}$.

The time-varying topology of the network is governed by a piecewise constant switching signal $\sigma \colon \mathbb{R}_{\geq 0} \to \{1, \ldots, d\}$. At each point in time t, $\sigma(t)$ dictates the topology of the network at time t, and σ is continuous from the right everywhere. In order to emphasize the role of the switching signal, we denote $\mathcal{D}_{\sigma(t)} = \mathcal{D}(t)$. Note that time-invariant networks are represented by defining $\mathcal{D}_{\sigma(t)} \equiv \mathcal{D}$, or by simply dropping the dependence on time t.

The agents share state information with one another according to the topology of the network. Each normal agent's state (or value[23]) at time t is denoted $x_i(t) \in \mathbb{R}$. However, in order to handle the deceptive Byzantine adversaries, we let $x_{(j,i)}(t)$ denote the state of agent j *intended* for agent i at time t. For consistency of notation, we define $x_{(j,i)}(t)$ for all[24] $j, i \in \mathcal{V}$, even if $(j, i) \notin \mathcal{E}(t)$. In the case that $j \in \mathcal{N}$ is normal, we define $x_{(j,i)}(t) \equiv x_j(t)$ (and, in particular, $x_{(j,j)}(t) \equiv x_j(t)$). On the other hand, if $j \in \mathcal{A}$ is an adversary, then $x_{(j,i)}(t)$ is the state trajectory that adversary j

[23]Throughout this manuscript we refer to a agent's value and state interchangeably.

[24]Although we are not concerned with the values received by adversaries, we still define $x_{(j,i)}(t)$ for $i \in \mathcal{A}$ for consistency.

would like to convey to agent i, but the topological constraints on the network prevent it from doing so. With this terminology, we denote the collective states of all agents in $\mathcal{N}$, $\mathcal{A}$, and $\mathcal{V}$ intended for agent i by

$$x_{(\mathcal{N},i)}(t) = [x_{(1,i)}(t), \ldots, x_{(N,i)}(t)]^\mathsf{T} = [x_1(t), \ldots, x_N(t)]^\mathsf{T} \in \mathbb{R}^N,$$

$$x_{(\mathcal{A},i)}(t) = [x_{(N+1,i)}(t), \ldots, x_{(n,i)}(t)]^\mathsf{T} \in \mathbb{R}^M,$$

and

$$x_{(\mathcal{V},i)}(t) = [x_{(1,i)}(t), \ldots, x_{(n,i)}(t)]^\mathsf{T} \in \mathbb{R}^n,$$

respectively. Since $x_{(\mathcal{N},i)}(t) \equiv x_{(\mathcal{N},j)}(t)$ for all $i, j \in \mathcal{V}$, we unambiguously define $x_\mathcal{N}(t) = x_{(\mathcal{N},i)}(t)$ for any $i \in \mathcal{V}$. Finally, we denote the vector containing all adversary states intended for the normal agents by $x_{(\mathcal{A},\mathcal{N})}(t) = [x^\mathsf{T}_{(\mathcal{A},1)}(t), \ldots, x^\mathsf{T}_{(\mathcal{A},N)}(t)]^\mathsf{T} \in \mathbb{R}^{MN}$.

4.1.2 Update Model

Each normal agent $i \in \mathcal{N}$ has scalar state $x_i(t) \in \mathbb{R}$ and integrator dynamics given by $\dot{x}_i = u_i$, where $u_i = f_{i,\sigma(t)}(t, x_\mathcal{N}, x_{(\mathcal{A},i)})$ is a control input. The states of the neighboring adversaries, within $x_{(\mathcal{A},i)}$, are analyzed as uncertain inputs; however, because there is no prior knowledge about which agents are adversaries, the control input must treat the state information from neighboring agents in the same manner. The dynamics of the system of normal agents are then defined for $t \in \mathbb{R}_{\geq 0}$ by

$$\dot{x}_\mathcal{N}(t) = f_{\sigma(t)}(t, x_\mathcal{N}, x_{(\mathcal{A},\mathcal{N})}), \;\; x_\mathcal{N}(0) \in \mathbb{R}^N, \mathcal{D}_{\sigma(t)} \in \Gamma_n, \tag{27}$$

where $f_{\sigma(t)}(\cdot) = [f_{1,\sigma(t)}(\cdot), \ldots, f_{N,\sigma(t)}(\cdot)]^\mathsf{T}$. Each of the functions $f_{i,\sigma(t)}(\cdot)$ must satisfy appropriate assumptions to ensure existence of solutions, and may be different for each agent, depending on its role in the network. These functions should be designed *a priori* so that the normal agents reach consensus, *without prior knowledge about the identities of the adversaries*.[25] Observe that the dynamics of (27) define a switched system without impulse effects, so the trajectory of any solution is absolutely continuous [121].

[25]Note that for existence of solutions on $\mathbb{R}_{\geq 0}$, the $f_{i,\sigma(t)}(\cdot)$'s must be bounded and piecewise continuous with respect to the adversaries' trajectories.

4.1.3 Problem Statement

The Continuous-Time Resilient Asymptotic Consensus (CTRAC) problem is a continuous-time analogue to the Byzantine approximate agreement problem [130, 51], and is defined as follows. Let $M_{\mathcal{N}}(t)$ and $m_{\mathcal{N}}(t)$ be the *maximum* and *minimum* values of the normal agents at time t, respectively. That is,

$$M_{\mathcal{N}}(t) = \max_{i \in \mathcal{N}} \{x_i(t)\},$$

and

$$m_{\mathcal{N}}(t) = \min_{i \in \mathcal{N}} \{x_i(t)\}.$$

Definition 4.1 (CTRAC)**.** *The normal agents are said to achieve* ***continuous-time resilient asymptotic consensus*** *(CTRAC) in the presence of adversary agents (given a particular adversary model) if*

(i) $\exists L \in \mathbb{R}$ *such that* $\lim_{t \to \infty} x_i(t) = L$ *for all* $i \in \mathcal{N}$*, and*

(ii) $x_i(t) \in \mathcal{I}_0 = [m_{\mathcal{N}}(0), M_{\mathcal{N}}(0)]$ *for all* $t \in \mathbb{R}_{\geq 0}$*,* $i \in \mathcal{N}$ *(i.e.,* $\mathcal{I}_0$ *is a positively invariant set for each normal agent),*

for any choice of initial values $x_{\mathcal{N}}(0) \in \mathbb{R}^N$*.*

The CTRAC problem is defined by two conditions, agreement and safety, along with the type of adversary considered. Condition (i) in Definition 4.1 is an *agreement condition* that requires the states of the normal agents to converge to a common limit, despite the influence of the adversaries. Condition (ii) is a *safety condition* that requires the state trajectory of each normal agent to be contained in the interval formed by the initial states of the normal agents, despite the influence of the adversaries. Equivalently, the safety condition can be stated in terms of $x_{\mathcal{N}}$. Let $\mathcal{H}_0 = \mathcal{I}_0^N \subset \mathbb{R}^N$ denote the hypercube formed by the Cartesian product of N copies of $\mathcal{I}_0$. Then the safety condition requires $x_{\mathcal{N}}(t) \in \mathcal{H}_0$ for all $t \geq 0$, despite the influence of the adversaries. In the agreement condition it is important to explicitly require that the limit exists because a single asymptotic consensus value is desired.

The safety condition in (ii) is similar to the validity condition defined in [113], which in turn is motivated by the validity condition of the *Byzantine approximate agreement problem* [130, 51]. The definition ensures that the value chosen by each normal agent lies within the range of good values. This is important in applications where the values are measurements and only measurements within the range obtained by the normal agents are considered valid. The safety condition entails this notion along with an invariant condition, which is important for safety critical applications. Specifically, it is applicable whenever $\mathcal{I}_0$ is a known safe set.

4.1.4 Adversary Models

The adversary models studied throughout this work have two aspects: a *threat model* and *scope of threat model*. The threat model defines the behavioral semantics of the individual compromised nodes. The scope of threat model defines the topological semantics of the adversary model. That is, the scope of threat model may stipulate the number of adversaries (nodes), or the number of interactions among other nodes (directed edges) that are allowed under the model. The scope, in terms of either nodes or directed edges, may be limited by global or local bounds, and the scope may also be fractional in nature. Put simply, the scope of threat model limits the *scope* of the adversaries.

4.1.4.1 Threat Models

The threat model defines the types of behaviors allowed by the individual adversary node. The behavior of the adversaries may be quantified in terms of limitations on their dynamics (or state trajectories) as well as how the adversaries are allowed to convey information, or more generally, how they interact with the network. One important aspect of their dynamics is the timing semantics (e.g., continuous or discrete time). In this chapter, we are interested only in continuous-time behavior. Therefore, we define the threat models here in terms of continuous-time semantics. In later chapters, we revisit these threat models under different timing semantics and with different degrees of synchrony of the multi-agent network.

There are three threat models that we consider in this work. The least general threat is the *crash adversary*. The inspiration for the crash adversary is the *crash fault*, which is studied in mobile robotics; e.g., in consensus problems such as gathering [1, 46]. As a fault model, crash-faulty robots fail by simply stopping their movement. In an analogous manner, we define a crash adversary as a node that behaves normally until it is compromised. Once compromised, the crash node stops changing its state. We assume that the crash adversary has control over when the node is compromised, but otherwise cannot modify the state of the compromised agent or the values conveyed to other nodes. Therefore, the same values are conveyed to each out-going neighbor and the values remain fixed for all time in the interval $[t_0, \infty)$, where the crash time $t_0 \geq 0$ is chosen by the adversary. Finally, crash adversaries are assumed to be omniscient (i.e., they know all other states and the full network topology; they are aware of the update rules $f_{i,\sigma(t)}(\cdot), \forall i \in \mathcal{N}$; they are aware of which other agents are adversaries; and they know the plans of the other adversaries[26]). For this reason, the worst case crash times for the adversaries must be considered when analyzing multi-agent networks with crash adversaries. This behavior is summarized in the following definition.

Definition 4.2 (Continuous-Time Crash Adversary)**.** *An agent $k \in \mathcal{A}$ is a* ***crash adversary*** *(or simply* ***crash node****) if it is omniscient, and there exists $t_k \in \mathbb{R}_{\geq 0}$, selected by the adversary, such that*

- *agent k behaves normally before $t = t_k$, according to its prescribed update rule; i.e.,*

$$\dot{x}_k = f_{k,\sigma(t)}(t, x_{\mathcal{N}}, x_{(\mathcal{A},k)}) \textit{ for all } t < t_k;$$

- *agent k stops changing its state for all $t \geq t_k$; i.e., $x_k(t) = x_k(t_k)$ for all $t \geq t_k$;*
- *agent k conveys the same state to each out-neighbor (i.e., $x_{(k,i)} \equiv x_{(k,j)}$ for all $i, j \in \mathcal{N}_k^{out}$);*

Conversely, the most general threat studied here is the *Byzantine adversary*. The Byzantine adversary is motivated by Byzantine faulty nodes studied in distributed computing [110, 130, 165, 16, 61, 17], communication networks [82, 92], and mobile robotics [1, 46, 27]. As a fault model, Byzantine faulty nodes are allowed to behave arbitrarily within the limits set by the model of computation. For networked automata, this means that the Byzantine node's state-transition and message-generation functions may be modified in an arbitrary – or even worst-case – manner [130]. Byzantine nodes are capable of deceit. Specifically, the values conveyed to out-neighbors need not be the same. These behaviors make Byzantine faults adversarial in nature. Therefore, no modification to the Byzantine model is needed for its interpretation as an adversary.

As far as the author is aware, Byzantine nodes have not been studied in continuous-time systems prior to this work. For the switched system model of (27), a couple of additional assumptions are

[26]One may take the viewpoint that a centralized omniscient adversary informs and directs the behavior of the individual adversary agents.

needed. In discrete-time systems (such as networked automata), it is possible for Byzantine nodes to choose whether to send messages or not at any given time step or round. For the switched system model of (27) the most obvious way to mathematically model this behavior is through the switching signal (i.e., allow the topology of the network to change by the temporary removal of outgoing edges from a Byzantine node at time instances in which it chooses not to convey its state). Because the switching signal is piecewise constant, this limits the switching behavior an adversary is capable of inducing. At times, we require stronger assumptions on the switching signal (such as imposing a dwell time assumption). A limitation to this model of a Byzantine agent's capability of selectively withholding information is that in time-invariant networks, whenever $\mathcal{D}_{\sigma(t)} \equiv \mathcal{D}$, this capability is effectively eliminated.

A second assumption deals with the continuity of the state trajectories of Byzantine agents. Technically, only piecewise continuity of the trajectories of $x_{(\mathcal{A},\mathcal{N})}(t)$ (combined with certain regularity conditions on $f_{\sigma(t)}(\cdot)$) is needed for existence of solutions to (27). However, stronger continuity assumptions are sometimes needed for analysis. Moreover, the trajectories of normal nodes are continuous; therefore, it is feasible that normal nodes could use discontinuities in the state trajectories to detect adversaries. For these reasons, we restrict the trajectories of Byzantine adversaries to be continuous for all t. The behavior of Byzantine agents in continuous time is summarized as follows.

Definition 4.3 (Continuous-Time Byzantine Agent)**.** *An agent $k \in \mathcal{A}$ is a **Byzantine adversary** (or just **Byzantine node**) if it is omniscient, and*

- *agent k always conveys state information to out-neighbors in the network;*
- *agent k may selectively remove directed edges incident with itself in time-varying networks, but must maintain piecewise continuity of the switching signal $\sigma(t)$;*
- *agent k's state trajectory intended for i may be different than the one intended for j (i.e., $x_{(k,i)}(t) \neq x_{(k,j)}(t)$ is allowed);*
- *agent k's state trajectories intended for other nodes, $\{x_{(k,i)}(t)\colon i \in \mathcal{V}\}$, are continuous functions of time on $[0, \infty)$;*

Note that we allow the possibility that a Byzantine agent behaves exactly as prescribed (so that it is indistinguishable from a normal agent).

The third and final threat model studied here is the *malicious adversary*, or *malicious node*. The malicious node is essentially a Byzantine node restricted to a local broadcast model of conveying information. A malicious adversary may behave arbitrarily and is omniscient, in the same manner as a Byzantine adversary. However, malicious nodes are not capable of deceit; i.e., every out-neighbor receives the same information. Furthermore, the additional assumptions on continuous-time Byzantine nodes are also imposed on the malicious model. The malicious adversary is defined as follows.

Definition 4.4 (Continuous-Time Malicious Agent)**.** *An agent $k \in \mathcal{A}$ is a* ***Byzantine adversary*** *(or just* ***Byzantine node****) if it is omniscient, and*

- *agent k always conveys state information to out-neighbors in the network;*
- *agent k may selectively remove directed edges incident with itself in time-varying networks, but must maintain piecewise continuity of the switching signal $\sigma(t)$;*
- *agent k's state trajectory intended for i must be the same as the one intended for j; i.e., $x_{(k,i)}(t) \equiv x_{(k,j)}(t)$;*
- *agent k's state trajectories intended for other nodes, $\{x_{(k,i)}(t) \colon i \in \mathcal{V}\}$, are continuous functions of time on $[0, \infty)$;*

Like the Byzantine adversary, it is possible that a malicious node behaves exactly as prescribed (so that it is indistinguishable from a normal agent).

The malicious adversary is the main threat model that has been considered in prior works from the point of view of security. In particular, malicious nodes have been studied in the context of detection and identification of misbehaving nodes in linear consensus networks [157, 179, 180, 158, 160, 181, 159, 186]. In these works, a malicious node is modeled with the same update rule as the normal nodes, but with a disturbance on the output. The disturbance allows the malicious node to convey essentially any information it chooses. The general approach used in these works is for the normal nodes to use unknown input observers, along with nonlocal topological information concerning the (time-invariant) network, in order 'invert' the dynamics and detect the malicious nodes. Clearly, for this approach to be successful it is necessary for a malicious node to behave abnormally at some point in time. Otherwise, the malicious nodes cannot be distinguished from normal ones. For this reason, it is commonly assumed in these works that malicious nodes *cannot always behave normally*. We do not require this restriction here. Note that malicious adversaries have also been studied in resilient asymptotic consensus problems [113, 114, 116, 115, 210].

From the point of view of analysis, Byzantine adversaries are the worst-case threat model considered in this work. Note that malicious and crash adversaries are particular restrictions to the Byzantine model. Therefore, any sufficient condition proven to hold in the presence of Byzantine adversaries must also hold for malicious and crash adversaries. Conversely, any necessary condition shown to hold in the presence of crash nodes also holds for malicious and Byzantine nodes.

4.1.4.2 Scope of Threat Models

Having defined the limitations on the behavior of the threat models, we now look at the scope of threat assumptions. The scope of threat model defines the topological assumptions placed on the adversaries. While stochastic models may be used, we consider only deterministic scope of threat models. This is in large part because deterministic bounds are well-suited to worst-case analysis. Given the omniscient types of threat models studied, this restriction is reasonable.

A simple scope of threat model is to assume there are at most $F < n$ adversaries in the entire network. We refer to this scope of threat model as the *F-total model*. The F-total model has been studied extensively with respect to node failures [130]. Alternatively, the scope may be local; e.g., at most F neighbors of any normal node fail. This is referred to as the *F-local model* [210]. To account for varying degrees of different nodes, we also introduce a fault model that considers an upper bound on the *fraction* of compromised nodes in any node's neighborhood. This is called the *f-fraction local model* [115]. While the F-total model typically requires certain bounds on the fraction of nodes that may be compromised (e.g., $n > 3F$ for Byzantine nodes and $n > 2F$ for malicious nodes), the local and fractional models are dependent on the network topology and do not, in general, imply a bound on the fraction of nodes compromised in the entire network.

In what follows, we first define subsets of nodes as either F-total, F-local, or f-fraction local. From these definitions, we then define the scope of threat models.

Definition 4.5 (F-Total Set)**.** *A set $\mathcal{S} \subset \mathcal{V}$ is F**-total** if and only if it contains at most F nodes in the network, i.e., $|\mathcal{S}| \leq F$, $F \in \mathbb{Z}_{\geq 0}$.*

Definition 4.6 (F-Local Set)**.** *A set $\mathcal{S} \subset \mathcal{V}$ is F**-local** if and only if it contains at most F nodes in the neighborhood of the other nodes for all t, i.e., $|\mathcal{N}_i^{in}(t) \bigcap \mathcal{S}| \leq F$, $\forall i \in \mathcal{V} \setminus \mathcal{S}$, $F \in \mathbb{Z}_{\geq 0}$.*

Definition 4.7 (f-Fraction Local Set)**.** *A set $\mathcal{S} \subset \mathcal{V}$ is f**-fraction local** if and only if it contains at most a fraction f of agents in the neighborhood of the other agents for all t, i.e., $|\mathcal{N}_i^{in}(t) \bigcap \mathcal{S}| \leq \lfloor f|\mathcal{N}_i^{in}(t)| \rfloor$, $\forall i \in \mathcal{V} \setminus \mathcal{S}$, $f \in [0, 1]$.*

It should be emphasized that in time-varying network topologies, the local properties defining an F-local set (or an f-fraction local set) must hold for all points in time. These definitions facilitate the following scope of threat models.

Definition 4.8 (Scope of Threat Models)**.** *A set of adversary nodes $\mathcal{A}$ is F**-totally bounded**, F**-locally bounded** or f**-fraction locally bounded** if and only if it is an F-total set, F-local set or f-fraction local set, respectively. We refer to these scope of threat models as the F**-total**, F**-local**, and f**-fraction local models**, respectively.*

F-totally bounded faults have been studied in distributed computing [110, 130, 194] and mobile robotics [1, 46, 27] for crash, Byzantine, and stopping failures. The F-locally bounded fault model has been studied in the context of fault-tolerant (or resilient) broadcasting [162, 86, 210] and resilient consensus [210]. The f-fraction local model is proposed by Zhang and Sundaram [115]. It is inspired from ideas pertaining to *contagion* in social and economic networks [52], where a node accepts some new information (behavior or technology) if more than a certain fraction of its neighbors has adopted it.

A scope of threat model similar to the f-fraction local model is proposed in [120] for resilient clock synchronization in the presence of Byzantine nodes. The network in [120] is hierarchical and there are two types of nodes: tamper-proof nodes and normal nodes. Either type of node may be compromised, but a tamper-proof node can be trusted because it will destroy itself upon being compromised. The tamper-proof nodes act as leaders of clusters of nodes, where it is assumed that (a) each normal node belongs to a cluster whose head is tamper-proof, (b) any two tamper-proof nodes are within two hops from one another, and (c) at most one-third of the shared neighbors of any pair of tamper-proof nodes can be compromised. Assumption (c) differs from the $\frac{1}{3}$-fraction local model because in the latter, at most one-third of *all* neighbors of *any* normal node may be compromised. The f-fraction local model does not assume a hierarchical network. Instead, the normal nodes are effectively identical. Notice that in the hierarchical network of [120], it is possible that the compromise of a single tamper-proof node results in some uncompromised normal nodes that cannot participate in the consensus process. This precludes the possibility of achieving resilient asymptotic consensus as defined in this work (since *all* normal nodes must reach consensus).

4.2 Adversarial Robust Consensus Protocol (ARC-P)

Here, we describe the Adversarial Robust Consensus Protocol (ARC-P) with respect to parameter $F \in \mathbb{Z}_{\geq 0}$ and with respect to parameter $f \in [0, 1]$. The main idea of the protocol is for each normal agent $i \in \mathcal{N}$ to sort the relative states of its inclusive in-neighbors and then remove some of the extreme values (i.e., the largest and smallest values). The remaining relative state values are summed to determine the first order dynamics of the agent.

The parameter F (or f) of the algorithm determines the *number* (or *fraction*) of largest and smallest values that are removed. For consistency in notation, let $F_i(t) \equiv F$ if ARC-P with parameter F is used by normal node i, and let $F_i(t) = \lfloor f d_i(t) \rfloor$ whenever ARC-P with parameter f is is used by normal node i (note that all normal nodes either use one parameter or the other, depending on the scope of threat model assumed). Given this notation, each normal node i using ARC-P (regardless of the parameter) removes the $F_i(t)$ largest and $F_i(t)$ smallest values in its neighborhood (breaking ties arbitrarily). If $d_i(t) \geq 2F_i(t)$, this results in $d_i(t) - 2F_i(t) + 1$ relative states that are summed to determine the control input u_i. To make the protocol well-defined for all network topologies, i.e., whenever $d_i(t) < 2F_i(t)$, in this case the agent removes all neighboring values from consideration and the input is zero. This approach adheres to the philosophy that whenever there is insufficient information to act in a way that is resilient to adversarial influence, it is best to do nothing. In order to formulate ARC-P with parameter F (or f), we require the following definitions.

Definition 4.9. *Let $k \in \mathbb{N}$, $F_i \in \mathbb{Z}_{\geq 0}$, and $m = \max\{k - 2F_i, 1\}$. Denote the elements of vectors $z, \xi_i \in \mathbb{R}^k$ by z_l and ξ_i^l, respectively, for $l = 1, 2, \ldots, k$. Then:*

(i) The (ascending) ***sorting function*** *on k elements, $\rho_k \colon \mathbb{R}^k \to \mathbb{R}^k$, is defined by $\xi_i = \rho_k(z)$ such*

that ξ_i is a permutation of z which satisfies

$$\xi_i^1 \leq \xi_i^2 \leq \cdots \leq \xi_i^k; \tag{28}$$

(ii) The ***reduce function*** *with respect to F_i and k is defined by $r_{F_i,k} \colon \mathbb{R}^k \to \mathbb{R}^m$, which satisfies $r_{F_i,k}(\xi_i) = 0$ if $k \leq 2F_i$, and if $k > 2F_i$, then*

$$r_{F_i,k}(\xi_i) = [\xi_i^{F_i+1},\, \xi_i^{F_i+2},\, \dots\, ,\, \xi_i^{k-F_i}]^T;$$

(iii) The ***sum function****, $\Sigma_m \colon \mathbb{R}^m \to \mathbb{R}$, is defined for $y \in \mathbb{R}^m$ by*

$$\Sigma_m(y) = \sum_{l=1}^{m} y_l;$$

(iv) The composition of the sorting, reduce, and sum functions with respect to F_i and k is defined by $\phi_{F_i,k} \colon \mathbb{R}^k \to \mathbb{R}$, which satisfies for all $z \in \mathbb{R}^k$,

$$\phi_{F_i}^k(z) = (\Sigma_m \circ r_{F_i,k} \circ \rho_k)(z) = \begin{cases} \sum_{l=F_i+1}^{k-F_i} \xi_i^l & k > 2F_i; \\ 0 & k \leq 2F_i. \end{cases}$$

The switched system that defines the dynamics of the normal agents is given by

$$\dot{x}_{\mathcal{N}} = f_{\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})}), \;\; x_{\mathcal{N}}(0) = x_0 \in \mathbb{R}^N, \mathcal{D}_{\sigma(t)} \in \Gamma_n$$

where $f_{\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})}) = [f_{1,\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},1)}), \dots, f_{N,\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},N)})]^{\mathsf{T}}$ and the $x_{(\mathcal{A},i)}(t)$, $i \in \mathcal{N}$, are adversary trajectories intended for i. Then, ARC-P with parameter $F \in \mathbb{Z}_{\geq 0}$ (or $f \in [0,1]$) determines $u_i{=}f_{i,\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},i)})$ for each normal agent $i \in \mathcal{N}$ at each point in time t by

$$\begin{aligned} f_{i,\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},i)}) &= \phi_{F_i(t)}^{d_i(t)+1}\left(J_i(t)(x_{(\mathcal{V},i)}(t) - x_i(t)1_n)\right) \\ &= \begin{cases} \sum_{l=F_i(t)+1}^{d_i(t)+1-F_i(t)} \left(\xi_i^l(t) - x_i(t)\right) & d_i(t) \geq 2F_i(t); \\ 0 & d_i(t) < 2F_i(t). \end{cases} \end{aligned} \tag{29}$$

where $\xi_i(t) = \rho_{d_i(t)+1}(J_i(t)x_{(\mathcal{V},i)}(t))$ with elements $\xi_i^1, \xi_i^2, \ldots, \xi_i^{d_i(t)+1}$, $F_i(t) \equiv F$ or $F_i(t) = \lfloor f d_i(t) \rfloor$ if the parameter is F or f, respectively, $1_n \in \mathbb{R}^n$ is the vector of ones, and $J_i(t) \in \mathbb{R}^{(d_i(t)+1)\times n}$ is a time-varying sparse matrix with each row corresponding to a distinct $j \in \mathcal{J}_i^{\text{in}}(t)$ such that each row has a single 1 in the j-th column. There is a one-to-one correspondence between $j \in \mathcal{J}_i^{\text{in}}(t)$ and rows in $J_i(t)$.

Figure 19 illustrates the computation that occurs at time t for normal agent i whenever $d_i(t) \geq 2F_i(t)$. In the figure, the state, $x_i(t)$, of the agent, whose dynamics are $\dot{x}_i(t) = u_i(t)$, is subtracted from each of the other states in its inclusive neighborhood, with each of the in-neighbors denoted $x_i^j, j = 1, 2, \ldots, d_i(t)$. The resulting relative state values are sorted and then reduced by eliminating the largest and smallest $F_i(t)$ elements. Finally, the remaining elements are summed to produce the control input $u_i(t)$ to the integrator agent. The only difference if $d_i(t) < 2F_i(t)$ is that the output of the Reduce block is 0.

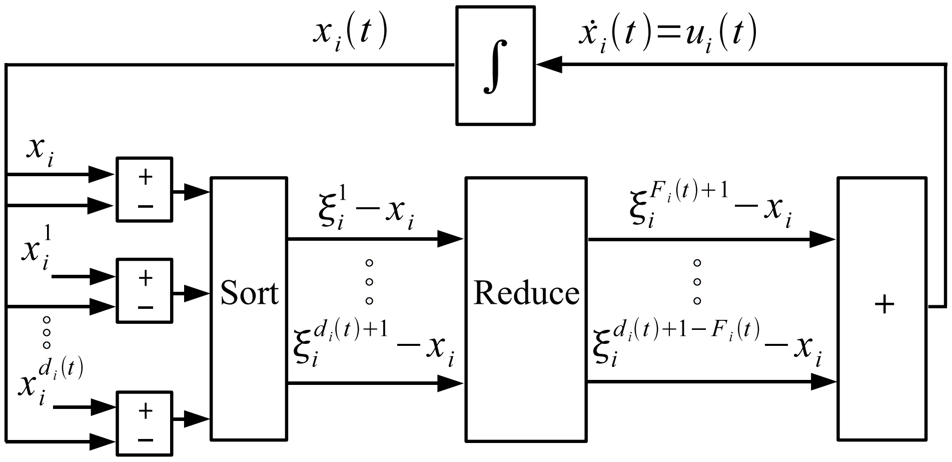


Figure 19: Synchronous data flow model of ARC-P with parameter F (or f) for agent i.

From a complexity standpoint, ARC-P consists of low complexity operations in both time and space, including sort, reduce, and sum methods (see Figure 19). The worst performing subroutine of ARC-P is the sort method. But, if quicksort is used, it is worst-case quadratic in time and linear in space, with respect to the size of the inclusive in-neighborhood. Therefore, ARC-P is also worst-case quadratic in time and linear in space, and hence low complexity.

4.3 Existence and Uniqueness of Solutions with ARC-P

This section examines the issue of existence and uniqueness of solutions to the switched system (27) where the component functions are given by ARC-P with parameter F (or f) as defined in (29). Since the switched system (27) is absent of impulse effects, the trajectory of any solution is continuous everywhere [121]. However, because $f_{\sigma(t)}(\cdot)$ is not a continuous function of time, we do not consider the usual notion of "solution" (i.e., in the sense of Peano). Instead, we mean an absolutely continuous function $\psi\colon [0,T] \to \mathbb{R}^N$ such that

$$\psi(t) = \psi(0) + \int_0^t f_{\sigma(\tau)}(\psi(\tau), x_{(\mathcal{A},\mathcal{N})}(\tau))d\tau, \ \ \forall t \in [0,T]. \tag{30}$$

Such a function ψ is known as a Carathéodory solution to (27) on the interval $[0,T]$. A Carathéodory solution is differentiable almost everywhere (a.e.)[27] and its derivative is Lebesgue integrable on the interval $[0,T]$ (i.e., the integration in (30) is in the sense of Lebesgue). The following classical result provides sufficient conditions for existence and uniqueness of a Carathéodory solution [78].

Theorem 4.10 (Carathéodory Solutions)**.** *Consider the ordinary differential equation*

$$\dot{x} = g(t,x), \ \ x(0) = \bar{x} \in \mathbb{R}^N, t \in [0,T],$$

where $g\colon [0,T] \times \mathbb{R}^N \to \mathbb{R}^N$ is a bounded function. Then:

(i) If the map $t \mapsto g(t,x)$ is measurable for each x and the map $x \mapsto g(t,x)$ is continuous for each t, then the ODE has at least one Carathéodory solution.

(ii) If the map $t \mapsto g(t,x)$ is measurable for each x and the map $x \mapsto g(t,x)$ is Lipschitz continuous for each t, with a uniform Lipschitz constant, then the ODE has a unique Carathéodory solution.

In what follows, we show that $f_{\sigma(t)}(\cdot)$ is piecewise continuous in time, and hence measurable for each $x_{\mathcal{N}}$ (observe that the time-dependent nature of $f_{\sigma(t)}(\cdot)$ is induced by the piecewise constant switching signal $\sigma(t)$ and the continuous adversary trajectories intended for the normal nodes,

[27]A property is said to hold *almost everywhere (a.e.)* if the set of elements for which the property does *not* hold has measure zero.

$x_{(\mathcal{A},\mathcal{N})}$). We also show in Theorem 4.15 that $f_{\sigma(t)}(\cdot)$ is globally Lipschitz continuous in $x_{\mathcal{N}}$ for each t. Then we show in Lemma 4.16 that $f_{\sigma(t)}(\cdot)$ is bounded for each t by a function of $x_{\mathcal{N}}$. After this, we consider the minimal hypercube in $\mathbb{R}^N$ that contains the initial values of the normal agents, denoted $\mathcal{H}_0$. Lemma 4.16 is then used in the proof of Lemma 4.18 to show that $f_{\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})})$ is directed toward the interior or along the boundary of $\mathcal{H}_0$ for all points $x_{\mathcal{N}}$ on the boundary of $\mathcal{H}_0$. From Lemmas 4.16 and 4.18 we are able to deduce that $f_{\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})})$ is bounded on the time interval $[0,\infty)$ by a function of $x_{\mathcal{N}}(0)$. We conclude from Theorem 4.10 that there is a unique Carathéodory solution on $[0,T]$ for all $T \in \mathbb{R}_{\geq 0}$ and hence the solution is unique on the entire interval $[0,\infty)$ (of course, unique up to the switching signal and the trajectories chosen by the adversaries). We begin by recalling the definition of Lipschitz continuity.

Definition 4.11 (Lipschitz Continuity)**.** *Let $||\cdot||$ denote any norm defined on a Euclidean space, and let $g(t,x,u)$, $g\colon \mathbb{R}\times\mathbb{R}^n\times\mathbb{R}^p \to \mathbb{R}^q$, be a piecewise continuous function in t and u. Then g satisfies a* ***global Lipschitz condition*** *with Lipschitz constant $L \in \mathbb{R}_{\geq 0}$ if the following condition holds for all $z, y \in \mathbb{R}^n$, $t \in \mathbb{R}_{\geq 0}$, $u \in \mathbb{R}^q$:*

$$||g(t,z,u) - g(t,y,u)|| \leq L||z-y||.$$

In order to show that ARC-P satisfies a Lipschitz condition, we must show that the sorting function is Lipschitz continuous. First, we consider an interesting property of the sorting function; namely, given any two vectors, then the angle between the vectors will never increase by sorting the vectors. This result is then used to show Lipschitz continuity of the sorting function.

Lemma 4.12. *Given $x, x_0 \in \mathbb{R}^n$, $n \in \mathbb{Z}_{>0}$, with $\xi = \rho_n(x)$ and $\xi_0 = \rho_n(x_0)$, then*[28]

$$\xi^{\mathsf{T}}\xi_0 = \sum_{i=1}^{n} \xi_i \xi_{0_i} \geq \sum_{i=1}^{n} x_i x_{0_i} = x^{\mathsf{T}} x_0. \tag{31}$$

Proof. We prove the result by induction on n. The base step ($n=1$) is obvious (since $\xi = x$, $\xi_0 = x_0$). Now, suppose (31) is true for $1 \leq n \leq m$, and let $n = m+1$, with $x, x_0, \xi, \xi_0 \in \mathbb{R}^{m+1}$. Let j (and k) denote the index of the element with minimum value in x (x_0). If there are multiple

[28]The inequality in (31) is known as the *rearrangement inequality* [77]. A slightly different proof is given here for completeness.

elements with minimum values in either vector, arbitrarily fix the index to correspond to one of the minimum values. There are two cases: $j \neq k$ and $j = k$.

Case 1, $j \neq k$: Swap the elements x_j and x_k in x so that the minimum values of x and x_0 occur in the same index (k in this case). Remove the k^{th} element from each vector and denote the resulting vectors by $x', x'_0 \in \mathbb{R}^m$, and their corresponding sorted versions by ξ' and ξ'_0 respectively. Then, by the inductive hypothesis

$$\sum_{i=1}^{m} \xi'_i \xi'_{0_i} \geq \sum_{i=1}^{m} x'_i x'_{0_i}. \tag{32}$$

But the terms in (32) are related to the terms in $x^\mathsf{T} x_0$ and $\xi^\mathsf{T} \xi_0$ as follows. For the right-hand side, the only elements altered in x are x_j and x_k, which have been swapped (with x_j removed), and only x_{0_k} has been removed from x_0, with no other changes to x_0. Thus, we have

$$\sum_{i=1}^{m} x'_i x'_{0_i} = \sum_{\substack{i=1 \\ i \neq j,k}}^{m+1} x_i x_{0_i} + x_k x_{0_j}. \tag{33}$$

Similarly, for the left-hand side of (32), only one minimum value of each vector has been removed; therefore, the inner product of the resulting sorted vectors ($\xi'^\mathsf{T} \xi'_0$) contain the same terms as $\xi^\mathsf{T} \xi_0$, except for the term $x_j x_{0_k} = \xi_1 \xi_{0_1}$. Hence,

$$\sum_{i=1}^{m} \xi'_i \xi'_{0_i} = \sum_{i=2}^{m+1} \xi_i \xi_{0_i}. \tag{34}$$

Substituting (33) and (34) into (32) and adding $x_j x_{0_k} = \xi_1 \xi_{0_1}$ to both sides of the inequality yields

$$\sum_{i=1}^{m+1} \xi_i \xi_{0_i} \geq \sum_{\substack{i=1 \\ i \neq j,k}}^{m+1} x_i x_{0_i} + x_k x_{0_j} + x_j x_{0_k}. \tag{35}$$

Now, since $x_k \geq x_j$ and $x_{0_j} \geq x_{0_k}$, we have

$$\begin{aligned} &(x_k - x_j)(x_{0_j} - x_{0_k}) \geq 0 \\ \implies\ & x_k x_{0_j} + x_j x_{0_k} \geq x_k x_{0_k} + x_j x_{0_j}. \end{aligned}$$

Finally, combining this with (35) produces the desired result

$$\sum_{i=1}^{m+1} \xi_i \xi_{0_i} \geq \sum_{i=1}^{m+1} x_i x_{0_i}, \tag{37}$$

which completes the inductive step.

Case 2, $j = k$: Fix x' and x'_0 by removing the k^{th} element (the minimum value) of x and x_0, respectively. Then, (32) is true by the inductive hypothesis. Analogous to Case 1, (34) also holds. In this case, the right-hand side of (32) is given by

$$\sum_{i=1}^{m} x'_i x'_{0_i} = \sum_{\substack{i=1 \\ i \neq k}}^{m+1} x_i x_{0_i}. \tag{38}$$

Substituting (34) and (38) into (32) and adding $x_k x_{0_k} = \xi_1 \xi_{0_1}$ to both sides of the inequality yields (37), which completes the inductive step and the proof. □

Lemma 4.13. *The sorting function, $\xi = \rho_n(x) \in \mathbb{R}^n$, defined by (28), satisfies a Lipschitz condition in $x \in \mathbb{R}^n$.*

Proof. Fix $x, x_0 \in \mathbb{R}^n$ and let $\rho_n(x) = \xi$, $\rho_n(x_0) = \xi_0$. Then, using the norm preservation property of permutations and Lemma 4.12, we have

$$\begin{aligned} ||\xi - \xi_0||_2 &= \left(\xi^\mathsf{T}\xi + \xi_0^\mathsf{T}\xi_0 - 2\xi^\mathsf{T}\xi_0\right)^{\frac{1}{2}} \\ &\leq \left(x^\mathsf{T}x + x_0^\mathsf{T}x_0 - 2x^\mathsf{T}x_0\right)^{\frac{1}{2}} = ||x - x_0||_2. \end{aligned}$$

□

Lemma 4.14. *Each function $f_{i,\sigma(t)}(x_\mathcal{N}, x_{(\mathcal{A},i)})$ for $i \in \mathcal{N}$ defined in (29) with parameter $F \in \mathbb{Z}_{\geq 0}$ (or $f \in [0, 1]$), satisfies a global Lipschitz condition in $x = [x_\mathcal{N}^\mathsf{T}, x_{(\mathcal{A},i)}^\mathsf{T}]^\mathsf{T}$, $x_\mathcal{N}$, and $x_{(\mathcal{A},i)}$.*

Proof. Fix $i \in \mathcal{N}$. Since the switching signal $\sigma(t)$ is piecewise constant, it follows that $f_{i,\sigma(t)}(\cdot)$ is piecewise continuous in time t. Fix $t \in \mathbb{R}_{\geq 0}$, $F_i(t) = F \in \mathbb{Z}_{\geq 0}$, and $d_i^{\text{in}}(t) + 1 = k$. The argument to $\phi_F^k(\cdot)$, the reduce function with to F and k, and the sum function are all linear transformations, and are therefore Lipschitz. The sorting function is Lipschitz by Lemma 4.13. Since the composition of Lipschitz functions is Lipschitz, it follows that $f_{i,\sigma(t)}(\cdot)$ satisfies a Lipschitz condition in $x = [x_{\mathcal{N}}^{\mathsf{T}}, x_{(\mathcal{A},i)}^{\mathsf{T}}]^{\mathsf{T}}$. Therefore, there exists $L \in \mathbb{R}_{\geq 0}$ such that

$$\begin{aligned}&||f_{i,\sigma(t)}(y, x_{(\mathcal{A},i)}) - f_{i,\sigma(t)}(z, x_{(\mathcal{A},i)})||_1 \\ &\qquad \leq L \left|\left| \begin{bmatrix} y \\ x_{(\mathcal{A},i)} \end{bmatrix} - \begin{bmatrix} z \\ x_{(\mathcal{A},i)} \end{bmatrix} \right|\right|_1 = L||y - z||_1,\end{aligned}$$

for any $t \in \mathbb{R}_{\geq 0}$, $y, z \in \mathbb{R}^N$, $x_{(\mathcal{A},i)} \in \mathbb{R}^M$. Likewise, for any $t \in \mathbb{R}_{\geq 0}$, $x_{\mathcal{N}} \in \mathbb{R}^N$,$y, z \in \mathbb{R}^M$ we have

$$\begin{aligned}&||f_{i,\sigma(t)}(x_{\mathcal{N}}, y) - f_{i,\sigma(t)}(x_{\mathcal{N}}, z)||_1 \\ &\qquad \leq L \left|\left| \begin{bmatrix} x_{\mathcal{N}} \\ y \end{bmatrix} - \begin{bmatrix} x_{\mathcal{N}} \\ z \end{bmatrix} \right|\right|_1 = L||y - z||_1.\end{aligned}$$

□

Theorem 4.15 (Lipschitz Continuity of ARC-P)**.** *The function $f_{\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})})$ that defines the dynamics of the normal agents, with $f_{i,\sigma(t)}(\cdot)$ defined in (29) with parameter $F \in \mathbb{Z}_{\geq 0}$ (or $f \in [0, 1]$) and any threat model such that $x_{(\mathcal{A},\mathcal{N})}$ is piecewise continuous, satisfies a global Lipschitz condition in $x_{\mathcal{N}}$.*

Proof. Since the switching signal $\sigma(t)$ is piecewise constant, $f_{i,\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},i)})$ is Lipschitz in $x_{(\mathcal{A},i)}$ for all $i \in \mathcal{N}$, and $x_{(\mathcal{A},i)}$ is piecewise continuous in t for all $i \in \mathcal{N}$, it follows that $f_{i,\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},i)}(t))$ is piecewise continuous in t. Therefore, $f_{\sigma(t)}(\cdot)$ is also piecewise continuous in t. By Lemma 4.14, we know that for $t \in \mathbb{R}_{\geq 0}$, $y_{\mathcal{N}}, z_{\mathcal{N}} \in \mathbb{R}^N$ and $x_{(\mathcal{A},i)} \in \mathbb{R}^M$ for $i \in \mathcal{N}$, there exists $L \in \mathbb{R}_{\geq 0}$ such that

$$\begin{aligned}
||f_{\sigma(t)}(y_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})}) - f_{\sigma(t)}(z_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})})||_1 &= \sum_{i=1}^{N} |f_{i,\sigma(t)}(y_{\mathcal{N}}, x_{(\mathcal{A},i)}) - f_{i,\sigma(t)}(z_{\mathcal{N}}, x_{(\mathcal{A},i)})| \\
&\leq \sum_{i=1}^{N} L \left|\left| \begin{bmatrix} y_{\mathcal{N}} \\ x_{(\mathcal{A},i)} \end{bmatrix} - \begin{bmatrix} z_{\mathcal{N}} \\ x_{(\mathcal{A},i)} \end{bmatrix} \right|\right|_1 \\
&\leq \sum_{i=1}^{N} L \left(\sum_{j=1}^{N} |y_j - z_j| \right) = nL||y_{\mathcal{N}} - z_{\mathcal{N}}||_1.
\end{aligned}$$

From this we conclude that $f_{\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})})$ satisfies a Lipschitz condition in $x_{\mathcal{N}}$. □

Theorem 4.15 shows that $f_{\sigma(t)}(\cdot)$ is piecewise continuous in t and Lipschitz in $x_{\mathcal{N}}$. We show next in Lemma 4.16 that $f_{\sigma(t)}(\cdot)$ is bounded by the current normal values $x_{\mathcal{N}}(t)$ for $t \in \mathbb{R}_{\geq 0}$. From these facts, we conclude the local existence and uniqueness of solutions of (27) for all $i \in \mathcal{N}$. Then, we show in Lemma 4.18 that any solution is confined to a compact set, from which we conclude global existence and uniqueness of solutions. For these results, and indeed for the analysis throughout, we define $m_{\mathcal{N}}(t) = \min_{j \in \mathcal{N}}\{x_j(t)\}$, and $M_{\mathcal{N}}(t) = \max_{k \in \mathcal{N}}\{x_k(t)\}$.

Lemma 4.16. *Consider the normal agent $i \in \mathcal{N}$ using ARC-P with parameter $F \in \mathbb{Z}_{\geq 0}$ under the F'-local or F'-total models where $F' \leq F$, or parameter $f \in [0, 1]$ under the f'-fraction local model where $f' \leq f$. Define $F_i(t) \equiv F$ or $F_i(t) = \lfloor f d_i(t) \rfloor$ whenever the parameter is F or f, respectively. Then, for each $t \in \mathbb{R}_{\geq 0}$*

$$B(m_{\mathcal{N}}(t) - x_i(t)) \leq f_{i,\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},i)}) \leq B(M_{\mathcal{N}}(t) - x_i(t)),$$

where $B = n - 2\min_{i \in \mathcal{N}}\{F_i(t)\}$.

Proof. If $d_i(t) < 2F_i(t)$, then $f_{i,\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},i)}) = 0$, and the result follows. Therefore, assume $d_i(t) \geq 2F_i(t)$ and at least one value not equal to $x_i(t)$ is used in the update at time t, say $x_j(t) = \xi_i^l(t)$ for some $l \in \{F_i(t) + 1, F_i(t) + 2, \ldots, d_i(t) + 1 - F_i(t)\}$. Suppose $\xi_i^l(t) > M_{\mathcal{N}}(t)$. Then, by definition j must be an adversary and $\xi_i^l(t) > x_i(t)$. Since i uses $\xi_i^l(t)$ at time t, there must be at least $F_i(t)$ more agents in the neighborhood of i with values at least as large as $\xi_i^l(t)$. Hence, these agents must also be adversaries, which contradicts the assumption of at most $F_i(t)$ adversary agents in the neighborhood of i at time t. Thus, $\xi_i^l(t) \leq M_{\mathcal{N}}(t)$. Similarly, we can show that $\xi_i^l(t) \geq m_{\mathcal{N}}(t)$. Since there are at most n agents in i's inclusive in-neighborhood, of which $2F_i(t)$ values are removed, it follows that

$$(n - 2F_i(t))(m_{\mathcal{N}}(t) - x_i(t)) \leq \sum_{l=F_i(t)+1}^{d_i(t)+1-F_i(t)} \left(\xi_i^l(t) - x_i(t)\right) \leq (n - 2F_i(t))(M_{\mathcal{N}}(t) - x_i(t)).$$

□

Lemma 4.16 bounds $f_{\sigma(t)}(\cdot)$ as a function of the total number of nodes n, the smallest upper bound on the number of adversaries in the neighborhood of any normal agent $\min_{i\in\mathcal{N}}\{F_i(t)\}$, and the current state of the normal agent values $x_{\mathcal{N}}(t)$. The next result shows that for any point $x_{\mathcal{N}}$ along boundary of the hypercube $\mathcal{H}_0$, which is given by $[m_{\mathcal{N}}(0), M_{\mathcal{N}}(0)]^N$, the function $f_{\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})})$ is always directed either toward the interior of $\mathcal{H}_0$ or along the boundary. From this, we conclude that $\mathcal{H}_0$ is a *robustly positively invariant set*, which is defined as follows.

Definition 4.17 (Robustly Positively Invariant Set)**.** *The set $\mathcal{S} \subset \mathbb{R}^N$ is **robustly positively invariant** for the system given by (27) if for all $x_{\mathcal{N}}(0) \in \mathcal{S}$, $x_{(\mathcal{A},\mathcal{N})}(t)$, any solution satisfies $x_{\mathcal{N}}(t) \in \mathcal{S}$ for all $t \geq 0$.*

Lemma 4.18 (Invariant Set)**.** *Suppose the normal agents in $\mathcal{N}$ use ARC-P with parameter $F \in \mathbb{Z}_{\geq 0}$ under the F'-local or F'-total models where $F' \leq F$, or parameter $f \in [0,1]$ under the f'-fraction local model where $f' \leq f$. Then, the hypercube $\mathcal{H}_0 = [m_{\mathcal{N}}(0), M_{\mathcal{N}}(0)]^N$ defined by*

$$\mathcal{H}_0 = \{y \in \mathbb{R}^N \colon m_{\mathcal{N}}(0) \leq y_i \leq M_{\mathcal{N}}(0),\ i = 1, 2, \dots, N\},$$

is robustly positively invariant for the system of normal agents.

Proof. Since $\mathcal{H}_0$ is compact and any solution of (27) using (29) is continuous with $x_{\mathcal{N}}(0) \in \mathcal{H}_0$, we must show that $f_{\sigma(t)}(\cdot)$ is not directed outside of $\mathcal{H}_0$, whenever $x_{\mathcal{N}}(t) \in \partial\mathcal{H}_0$, for all $\mathcal{D}_{\sigma(t)} \in \Gamma_n$ and all $x_{(\mathcal{A},\mathcal{N})} \in \mathbb{R}^{MN}$. The boundary of $\mathcal{H}_0$ is given by

$$\partial\mathcal{H}_0 = \{y \in \mathcal{H}_0 \colon \exists i \in \{1, 2, \dots, N\} \text{ s.t. } y_i \in \{m_{\mathcal{N}}(0), M_{\mathcal{N}}(0)\}\}.$$

Now, fix $x_{\mathcal{N}} \in \partial\mathcal{H}_0$ for some $t \in \mathbb{R}_{\geq 0}$. Let e_j denote the j-th canonical basis vector and denote $\mathcal{I}_{\mathcal{N},\min}, \mathcal{I}_{\mathcal{N},\max} \subseteq \{1, 2, \ldots, N\}$ as the sets defined by

$$j \in \mathcal{I}_{\mathcal{N},\min} \Leftrightarrow x_j = m_{\mathcal{N}}(0) \text{ and } k \in \mathcal{I}_{\mathcal{N},\max} \Leftrightarrow x_k = M_{\mathcal{N}}(0).$$

Then, from the geometry of the hypercube, we require

$$\begin{aligned} e_j^{\mathsf{T}} f_{\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})}) \geq 0 \quad \forall j \in \mathcal{I}_{\mathcal{N},\min}, \\ e_k^{\mathsf{T}} f_{\sigma(t)}(t, x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})}) \leq 0 \quad \forall k \in \mathcal{I}_{\mathcal{N},\max}. \end{aligned}$$

These conditions are true for all $\mathcal{D}_{\sigma(t)} \in \Gamma_n$ and $x_{(\mathcal{A},\mathcal{N})}$ under any of the scope of threat models (assuming the corresponding parameter for ARC-P is used) by Lemma 4.16, in which the lower bound is used for $j \in \mathcal{I}_{\mathcal{N},\min}$ (since $x_j = m_{\mathcal{N}}(0)$), and the upper bound is used for $k \in \mathcal{I}_{\mathcal{N},\max}$ (since $x_k = M_{\mathcal{N}}(0)$). $\square$

Remark 4.1. *Lemma 4.18 guarantees that ARC-P (with sufficiently large parameter) always guarantees the safety condition (i.e., condition (ii) of Definition 4.1). Therefore, the remaining question is under what conditions does ARC-P guarantee the agreement condition.*

The argument made in Lemma 4.16 implies that any time an adversary under any of the scope of threat models is outside of $\mathcal{I}_t = [m_{\mathcal{N}}(t), M_{\mathcal{N}}(t)]$, its influence is guaranteed to be removed by its normal neighbors (provided the corresponding parameter of ARC-P is large enough). Therefore each adversary has the same effect as if it were on the boundary of $\mathcal{I}_t$. Using Lemma 4.18 we conclude $\mathcal{I}_t \subseteq \mathcal{I}_0$, $\forall t \geq 0$. Hence, each adversary is effectively restricted to the compact set $\mathcal{I}_0$. This fact enables us to allow adversary states $x_{(\mathcal{A},i)} \in \mathbb{R}^M$ for $i \in \mathcal{N}$ rather than explicitly restricting them to a compact set (which is typically the assumption made for systems with uncertain inputs [21]), while still ensuring existence and uniqueness of solutions. The preceding results along with the argument made after stating Theorem 4.10 implies the following result.

Theorem 4.19 (Existence and Uniqueness)**.** *Suppose the adversaries have piecewise continuous trajectories on $[0, \infty)$. Suppose each normal agent uses ARC-P with parameter $F \in \mathbb{Z}_{\geq 0}$ under the F'-local or F'-total models where $F' \leq F$, or parameter $f \in [0, 1]$ under the f'-fraction local* *model where $f' \leq f$. Then, the switched system (27) with component functions given in (29) has a unique Carathéodory solution on* $[0, \infty)$ *(given the choice of* $\sigma(t)$ *and* $x_{(\mathcal{A},\mathcal{N})}(t)$ *for* $t \in \mathbb{R}_{\geq 0}$*).*

4.4 Lyapunov Candidate

Much of the analysis of ARC-P (and its variants) hinges upon the behavior of a Lyapunov candidate function that characterizes the maximum disagreement among the normal values. Define $\Psi \colon \mathbb{R}^N \to \mathbb{R}$ by

$$\Psi(x_{\mathcal{N}}) = M_{\mathcal{N}}(t) - m_{\mathcal{N}}(t), \tag{39}$$

where $m_{\mathcal{N}}(t) = \min_{j \in \mathcal{N}}\{x_j(t)\}$ and $M_{\mathcal{N}}(t) = \max_{k \in \mathcal{N}}\{x_k(t)\}$. It can be readily shown that the function Ψ has the following attractive properties (for example, property (ii) below can be shown easily using Lemma 4.13):

(i) Ψ is nonnegative with $\Psi(x_{\mathcal{N}}) = 0$ for all $x_{\mathcal{N}} \in \text{span}\{1_N\}$ and $\Psi(x_{\mathcal{N}}) > 0$ for all $x_{\mathcal{N}} \notin \text{span}\{1_N\}$;

(ii) Ψ is Lipschitz;

(iii) Ψ is increasing away from span$\{1_N\}$ in the sense that $\Psi(y_1) > \Psi(y_2), \forall y_1, y_2 \in \mathbb{R}^N$ satisfying $\text{dist}(y_1, \text{span}\{1_N\}) > \text{dist}(y_2, \text{span}\{1_N\})$;

(iv) Ψ is radially unbounded away from span$\{1_N\}$ in the sense that

$$\Psi(y) \to \infty \quad \text{as } \text{dist}(y, \text{span}\{1_N\}) \to \infty.$$

These properties make Ψ an excellent Lyapunov candidate for proving global convergence to the agreement space, span$\{1_N\}$. Indeed, Ψ has been used to prove convergence of asynchronous consensus algorithms whenever all agents are normal [191].

One issue with Ψ is that it is not everywhere differentiable. Therefore, to study the monotonicity of $\psi(t) = \Psi(x_{\mathcal{N}}(t))$, we consider the *upper-right Dini derivative* $D^+\psi(t)$ of ψ at t, defined by

$$D^+\psi(t) = \limsup_{h \to 0^+} \frac{\psi(t+h) - \psi(t)}{h},$$

and the *upper-directional derivative* of Ψ with respect to (27):

$$D^+\Psi(x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})}) = \limsup_{h\to 0^+} \frac{\Psi(x_{\mathcal{N}} + hf_{\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})})) - \Psi(x_{\mathcal{N}})}{h}.$$

The motivation for considering the upper directional derivative of Ψ is that

$$D^+\psi(t) = D^+\Psi(x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})})$$

for almost all t along solutions of (27) since Ψ is locally Lipschitz [170]. In this case,

$$D^+\Psi(x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})}) = \limsup_{h\to 0^+} \frac{N_1(h)}{h} + \limsup_{h\to 0^+} \frac{N_2(h)}{h}, \tag{40}$$

with

$$\begin{aligned} N_1(h) &= \max_{i\in\mathcal{N}}\{x_i + hf_{i,\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},i)})\} - M_{\mathcal{N}}(t), \\ N_2(h) &= m_{\mathcal{N}} - \min_{i\in\mathcal{N}}\{x_i + hf_{i,\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},i)})\}. \end{aligned}$$

4.4.1 Useful Lemmas Involving the Lyapunov Candidate

In this section, we prove a couple of useful lemmas related to the Lyapunov candidate of (39). The first result characterizes the upper Dini derivative of the Lyapunov candidate whenever ARC-P is used. The characterization is valid for all network topologies and all adversary models. The second lemma bounds the Lyapunov candidate between two functions of the distance from agreement. This result is useful for proving exponential convergence to the agreement space.

Lemma 4.20. *Fix $t \in \mathbb{R}_{\geq 0}$ and $x_{\mathcal{N}}(t) \in \mathbb{R}^N$. Suppose each normal agent in $\mathcal{N}$ uses ARC-P with parameter $F \in \mathbb{Z}_{\geq 0}$ under the F'-local or F'-total models where $F' \leq F$, or parameter $f \in [0,1]$ under the f'-fraction local model where $f' \leq f$. Let $\mathcal{D}_{\sigma(t)} \in \Gamma_n$ and define $\mathcal{S}_{min}(t), \mathcal{S}_{max}(t) \colon \mathbb{R} \to \{1,\ldots,N\}$ by*

$$\begin{aligned} j \in \mathcal{S}_{min}(t) &\Leftrightarrow x_j(t) = m_{\mathcal{N}}(t), \\ k \in \mathcal{S}_{max}(t) &\Leftrightarrow x_k(t) = M_{\mathcal{N}}(t). \end{aligned}$$

Fix $j_t \in \mathcal{S}_{min}(t)$ *such that*

$$f_{j_t,\sigma(t)}(x_\mathcal{N}, x_{(\mathcal{A},j_t)}) \leq f_{j,\sigma(t)}(x_\mathcal{N}, x_{(\mathcal{A},j)}), \quad \forall j \in \mathcal{S}_{min}(t).$$

Likewise, fix $k_t \in \mathcal{S}_{max}(t)$ *such that*

$$f_{k_t,\sigma(t)}(x_\mathcal{N}, x_{(\mathcal{A},k_t)} \geq f_{k,\sigma(t)}(x_\mathcal{N}, x_{(\mathcal{A},k)}), \quad \forall k \in \mathcal{S}_{max}(t).$$

Then, at time t*, we have*

$$D^+\Psi(x_\mathcal{N}(t), x_{(\mathcal{A},\mathcal{N})}(t)) = f_{k_t,\sigma(t)}(x_\mathcal{N}, x_{(\mathcal{A},k_t)}) - f_{j_t,\sigma(t)}(x_\mathcal{N}, x_{(\mathcal{A},j_t)}), \tag{41}$$

and $D^+\Psi(x_\mathcal{N}, x_{(\mathcal{A},\mathcal{N})}) \leq 0$ *for all* $t \in \mathbb{R}_{\geq 0}$.

Proof. Lemma 4.16 implies that

$$-n\Psi(x_\mathcal{N}) \leq f_{i,\sigma(t)}(x_\mathcal{N}, x_{(\mathcal{A},i)}) \leq n\Psi(x_\mathcal{N})$$

holds $\forall i \in \mathcal{N}$ and under any of the scope of threat models (with sufficiently large parameter). If $x_\mathcal{N}(t) \notin \text{span}\{1_N\}$, then there exists $\epsilon_{\text{min}} > 0$ such that $x_i - x_j \geq \epsilon_{\text{min}} > 0$ for all $j \in \mathcal{S}_{\text{min}}(t)$ and $i \in \mathcal{N} \setminus \mathcal{S}_{\text{min}}(t)$. Similarly, there exists $\epsilon_{\text{max}} > 0$ such that $x_k - x_i \geq \epsilon_{\text{max}}$ for all $k \in \mathcal{S}_{\text{max}}(t)$ and $i \in \mathcal{N} \setminus \mathcal{S}_{\text{max}}(t)$. Then, by letting $\epsilon = \min\{\epsilon_{\text{min}}, \epsilon_{\text{max}}\}$ and taking $h \leq \epsilon/(2n\Psi(x_\mathcal{N}(t)))$, we may write

$$\begin{aligned}
x_i + hf_{i,\sigma(t)}(x_\mathcal{N}, x_{(\mathcal{A},i)}) &\geq x_i - hn\Psi(x_\mathcal{N}(t)) \\
&\geq x_i - \epsilon/2 \\
&\geq x_j + \epsilon/2 \\
&\geq x_j + hn\Psi(x_\mathcal{N}(t)) \\
&\geq x_j + hf_{j,\sigma(t)}(x_\mathcal{N}, x_{(\mathcal{A},j)}) \\
&\geq x_{j_t} + hf_{j_t,\sigma(t)}(x_\mathcal{N}, x_{(\mathcal{A},j_t)})
\end{aligned}$$

for all $i \in \mathcal{N} \setminus \mathcal{S}_{\min}(t), j \in \mathcal{S}_{\min}(t)$. Therefore, at time t

$$\min_{i\in\mathcal{N}}\{x_i + hf_{i,\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},i)})\} = x_{j_t} + hf_{j_t,\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},j_t)}).$$

Following a similar argument, we deduce

$$\max_{i\in\mathcal{N}}\{x_i + hf_{i,\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},i)})\} = x_{k_t} + hf_{k_t,\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},k_t)}).$$

Combining this with (40), gives (41). On the other hand, if $x_{\mathcal{N}}(t) \in \text{span}\{1_N\}$ then both $\Psi(x_{\mathcal{N}})$ and $D^+\Psi(x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})})$ are zero. Finally, applying Lemma 4.16 for node j_t and k_t with $x_{j_t} = m_{\mathcal{N}}(t)$ and $x_{k_t} = M_{\mathcal{N}}(t)$, respectively, shows that $D^+\Psi(x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})}) \leq 0$. $\square$

Notice in (41) that the agents acting as k_t and j_t may change with time. It is important in any convergence argument to show that bounds on $D^+\Psi(x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})})$ hold for all $t \in \mathbb{R}_{\geq 0}$, regardless of which normal agents fill the roles of k_t and j_t. Next, we show that $\Psi(x_{\mathcal{N}})$ is bounded by scaled versions of dist$(x_{\mathcal{N}}, \text{span}\{1_N\}) = \inf_{y\in\text{span}\{1_N\}} ||x_{\mathcal{N}} - y||_2$. For this argument, we use the following properties of the min and max functions. If $\alpha \in \mathbb{R}$, then

$$\begin{aligned} \min_{i\in\mathcal{N}}\{x_i + \alpha\} &= \min_{i\in\mathcal{N}}\{x_i\} + \alpha \\ \max_{i\in\mathcal{N}}\{x_i + \alpha\} &= \max_{i\in\mathcal{N}}\{x_i\} + \alpha. \end{aligned} \tag{42}$$

Lemma 4.21. *Given $x_{\mathcal{N}} \in \mathbb{R}^N$, $\Psi(x_{\mathcal{N}})$ is bounded by*

$$\frac{1}{\sqrt{N}}\textit{dist}(x_{\mathcal{N}}, \textit{span}\{1_N\}) \leq \Psi(x_{\mathcal{N}}) \leq 2\textit{dist}(x_{\mathcal{N}}, \textit{span}\{1_N\}), \tag{43}$$

Proof. Consider the decomposition of $x_{\mathcal{N}}$: $x_{\mathcal{N}} = v_{1_N} + v_{1_N}^{\perp}$, in which $v_{1_N} \in \text{span}\{1_N\}$ and $v_{1_N}^{\perp} \in \text{span}\{1_N\}^{\perp}$. Given this decomposition, we conclude $\|v_{1_N}^{\perp}\|_2 = \text{dist}(x_{\mathcal{N}}, \text{span}\{1_N\})$ and $\exists\gamma \in \mathbb{R}$ such that $v_{1_N} = \gamma 1_N$. Because of this, we can use (42) to write

$$\Psi(x_{\mathcal{N}}) = \max_{i\in\mathcal{N}}\{(v_{1_N}^{\perp})_i\} - \min_{i\in\mathcal{N}}\{(v_{1_N}^{\perp})_i\}, \tag{44}$$

in which $(v_{1_N}^{\perp})_i$ is the i-th element of $v_{1_N}^{\perp}$. From this, we obtain the upper bound

$$\Psi(x_{\mathcal{N}}) \leq \max_{i\in\mathcal{N}}\{(v_{1_N}^{\perp})_i\} + |\min_{i\in\mathcal{N}}\{(v_{1_N}^{\perp})_i\}| \leq 2\|v_{1_N}^{\perp}\|_2.$$

On the other hand, since $v_{1_N}^{\perp} \in \text{span}\{1_N\}^{\perp}$, $\sum_{i=1}^{N}(v_{1_N}^{\perp})_i = 0$, so that

$$\max_{i\in\mathcal{N}}\{(v_{1_N}^{\perp})_i\} \geq 0 \quad \text{and} \quad \min_{i\in\mathcal{N}}\{(v_{1_N}^{\perp})_i\} \leq 0.$$

From this and (44) we conclude $\Psi(x_{\mathcal{N}}) \geq |(v_{1_N}^{\perp})_j|$ for all $j \in \mathcal{N}$. Thus, we obtain the lower bound

$$\frac{1}{\sqrt{N}}\|v_{1_N}^{\perp}\|_2 \leq \frac{1}{\sqrt{N}}\sqrt{N\Psi^2(x_{\mathcal{N}})} = \Psi(x_{\mathcal{N}}).$$

□

4.5 ARC-P in Complete Networks

Before analyzing ARC-P in more general networks, we first look at complete networks. Restricting our attention to this special case enables us to build some intuition about the protocol, which is helpful in understanding the more general results. Additionally, for the case of complete networks we may state a result that enables us to terminate the protocol in finite time with a guaranteed ϵ-approximate solution. This result is particular to complete networks because of the availability of global information.

4.5.1 Analysis in Complete Networks

In this section, we prove that ARC-P achieves CTRAC under all of the adversary models. We show that the error between the normal values exponentially vanishes. For the malicious (and crash) model we show that the rate of convergence depends on the number of nodes and the parameter of ARC-P. Then, we prove that the nodes converge to a single limit, which when combined with Lemma 4.18 shows that CTRAC is achieved.

Observe that in complete networks, the different scope of threat models are equivalent. To show this, first observe that for any normal node i,

$$\mathcal{N}_i^{\text{in}} \cap \mathcal{A} = \mathcal{A}.$$

From this, it follows that $\mathcal{A}$ is F-local if and only if it is F-total. Suppose $\mathcal{A}$ is F-local . Then, for all $i \in \mathcal{N}$,

$$|\mathcal{N}_i^{\text{in}} \cap \mathcal{A}| \leq F \leq \lfloor F \rfloor \leq \left\lfloor \left(\frac{F}{n-1} \right) (n-1) \right\rfloor \leq \left\lfloor \left(\frac{F}{n-1} \right) |\mathcal{N}_i^{\text{in}}| \right\rfloor .$$

Therefore, $\mathcal{A}$ is $(\frac{F}{n-1})$-fraction local. Since $\mathcal{N}$ is nonempty, it follows that $\mathcal{A}$ may be at most $(n-1)$-local, which corresponds to it also being 1-fraction local. Conversely, suppose $\mathcal{A}$ is f-fraction local, with $f \in [0,1]$. Let $F = \lfloor f(n-1) \rfloor$. Then, F is between 0 and $n-1$, and $\mathcal{A}$ is F-local. From this, it follows that ARC-P with parameter $F \in \{0,1,2,\dots,n-1\}$ is equivalent to ARC-P with parameter $f \in [\frac{F}{n-1}, \frac{F+1}{n-1}) \cap [0,1]$ (because in the latter case $F_i = \lfloor \frac{F}{n-1} d_i \rfloor = F$).

Theorem 4.22 (CTRAC in Complete Networks)**.** *Consider a complete network where each normal node updates its value according to ARC-P with parameter $F \in \mathbb{Z}_{\geq 0}$ under the F'-local or F'-total models where $F' \leq F < n/2$, or parameter $f \in [0,1]$ under the f'-fraction local model where $f' \leq f$ (and we assume $f \in [\frac{F}{n-1}, \frac{F+1}{n-1}) \cap [0,1]$ for some $F < n/2$ so that the parameters are equivalent). If $n > 3F$ for Byzantine adversaries or $n > 2F$ for malicious adversaries, then the states of the normal agents converge exponentially to the agreement space and CTRAC is achieved. Whenever the adversaries are malicious (or crash) nodes, the rate of convergence is $n - 2F$.*

Proof. First, assume the adversaries are malicious and $n > 2F \geq 2F'$. Then all of the normal nodes have the same multiset of values that are used to form the relative states used in the update. Fix $i, j \in \mathcal{N}$ and define the error term $e_{ij}(t) = x_i(t) - x_j(t)$. It then follows that the error dynamics are $\dot{e}_{ij}(t) = -(n-2F)e_{ij}(t)$. Therefore, the normal nodes converge to the agreement space exponentially with rate $n - 2F \geq 1$.

Now, assume the adversaries are Byzantine and $n > 3F \geq 3F'$. Since there are at most $F' \leq F < \frac{n}{3}$ adversaries, it follows that there are at least $2F+1$ normal nodes. Because the network is complete, all of the normal nodes are influenced by these values. Since ARC-P with parameter F removes at most $2F$ values in the neighborhood, it follows that at least one common normal value among the normal nodes is used at each point in time. In particular, for nodes j_t and k_t of Lemma 4.20, there is a common value, say $\xi_{k_t}^{m_t} = \xi_{j_t}^{l_t}$. Combining this with the expression for $D^+\Psi(\cdot)$ at time t from Lemma 4.20 yields

$$\begin{aligned} D^+\Psi(x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})}) = &\sum_{\substack{m=F+1 \\ m\neq m_t}}^{d_{k_t}(t)+1-F} (\xi_{k_t}^{m} - x_{k_t}) - \sum_{\substack{l=F+1 \\ l\neq l_t}}^{d_{j_t}(t)+1-F} (\xi_{j_t}^{l} - x_{j_t}) \\ &+ \left(\xi_{k_t}^{m_t} - \xi_{j_t}^{l_t}\right) - \Psi(x_{\mathcal{N}}) \leq -\Psi(x_{\mathcal{N}}). \end{aligned}$$

Since for each t there exists some $\xi_{k_t}^{m_t} = \xi_{j_t}^{l_t}$ for the j_t and k_t valid at t, it follow that this inequality holds for all $t \geq 0$. Therefore, it can be shown that (see, for example, [21, Theorem 2.18] and make the appropriate changes for convergence to the set span$\{1_N\}$)

$$\Psi(x_{\mathcal{N}}(t)) \leq \Psi(x_{\mathcal{N}}(0))e^{-t}.$$

From this, we conclude that $x_{\mathcal{N}}$ exponentially converges to span$\{1_N\}$.

Finally, in either case (Byzantine or malicious) the normal values remain in $\mathcal{I}_0$ by Lemma 4.18. It follows from Lemma 4.16 that $M_{\mathcal{N}}(t)$ is a nonincreasing function of time and $m_{\mathcal{N}}(t)$ is a nondecreasing function of time. Therefore, there is a common limit to which the normal values converge. $\square$

4.5.2 Finite Termination in Complete Networks

In this section, we derive an upper bound on the performance of ARC-P in order to terminate in finite time while ensuring an ϵ-approximate solution to CTRAC. By Theorem 4.22, the rate of convergence is exponential with rate no smaller than one. With this, the upper bound on convergence can be made precise.

Corollary 4.23. *Consider a complete network where each normal node updates its value according to ARC-P with parameter $F \in \mathbb{Z}_{\geq 0}$ under the F'-local or F'-total models where $F' \leq F < n/2$, or parameter $f \in [0,1]$ under the f'-fraction local model where $f' \leq f$ (and we assume $f \in [\frac{F}{n-1}, \frac{F+1}{n-1}) \cap [0,1]$ for some $F < n/2$ so that the parameters are equivalent). Suppose $n > 3F$ with Byzantine adversaries or $n > 2F$ with malicious adversaries. Define $\gamma = \max_{k\in\mathcal{V}}\{x_k(0)\} -$*

$\min_{j\in\mathcal{V}}\{x_j(0)\}$. *Then,*

$$\max_{i,j\in\mathcal{N}} e_{ij}(t) \le \gamma e^{-t}, \ \ \forall t \ge 0.$$

with Byzantine adversaries, or

$$\max_{i,j\in\mathcal{N}} e_{ij}(t) \le \gamma e^{-(n-2F)t}, \ \ \forall t \ge 0.$$

with malicious adversaries.

Proof. For all $i, j \in \mathcal{N}$, with at most $F' \le F$ Byzantine adversaries, we have

$$e_{ij}(t) \le e_{ij}(0)e^{-t} \le \gamma e^{-t}, \ \ \forall t \ge 0.$$

With at most $F' \le F$ malicious adversaries, the upper bound is $\gamma e^{-(n-2F)t}$. □

Using Corollary 4.23, an ϵ-approximate solution to CTRAC can be obtained in finite time. Specifically, to ensure that the maximum pairwise error between the values of normal agents is less than $\epsilon > 0$, Corollary 4.23 implies that any normal agent may terminate at any time greater than $|\log(\frac{\epsilon}{\gamma})|$ with Byzantine adversaries or $\frac{1}{n-2F}|\log(\frac{\epsilon}{\gamma})|$ with malicious adversaries (provided $\gamma \neq 0$, in which case $x_{\mathcal{N}}(0) \in \text{span}\{1_N\}$). It is important to emphasize that this method of terminating in finite time is only applicable in complete networks, where all initial values are known to each normal agent, so that γ may be determined.

4.5.3 Simulations with Malicious Agents in Complete Networks

To illustrate the resilience of ARC-P, we consider three examples in which a subset of the agents have been overtaken by malicious agents and redesigned with dubious intent, and a fourth example that illustrates the performance tradeoff for resilience incurred by ARC-P with parameter F. Three out of the four examples consider an eight agent complete network. In this case, at most three agents can be compromised by malicious agents while guaranteeing CTRAC whenever ARC-P with parameter $F = 3$ is used (since $n > 2F$ is required). The first scenario considers the case where two out of the eight agents are malicious adversaries and their goal is to drive the consensus state of the normal agents to an unsafe set $\mathcal{U}$. In the second scenario, three of the agents have been redesigned as oscillators in order to force the normal agents to oscillate at the desired frequency. Finally, in the third scenario a single adversary in a large network tries to force the other agents to follow a sinusoidal trajectory in the unsafe set.

To motivate the need for a consensus protocol that is resilient to adversaries, we compare ARC-P with the linear consensus protocol (LCP) given by

$$\dot{x}_i(t) = \sum_{j \in \mathcal{N}_i^{\text{in}}} \left(x_{(j,i)}(t) - x_i(t) \right), \;\; \forall i \in \mathcal{N}.$$

It is shown that LCP achieves the agreement condition in spite of the behavior of the adversaries, but not the safety condition, under the same conditions applied to ARC-P. For LCP, the states of the normal agents effectively converge to the average of the adversaries' trajectories. Thus, in all three scenarios, the adversaries are able to achieve their goal.

Example 4.1. *Consider a multi-agent network with eight agents, each with unique identifier in* $\{1, 2, \dots, 8\}$*, and with initial states equal to their identifier (e.g., for agent* 1*,* $x_1(0) = 1$*). Suppose that agents* 7 *and* 8 *have been compromised and are malicious (i.e.,* $\mathcal{A}= \{7, 8\}$*). The malicious agents are redesigned with*

$$\dot{x}_i = -10x_i + 10u_i, \; \forall i \in \mathcal{A},$$

where the reference inputs u_i *for the malicious agents are* $u_7 = 25$ *and* $u_8 = 26$*. Therefore, the malicious agents converge exponentially to* 25 *and* 26*, respectively, with rate* 10*.*

The goal of the malicious agents is to drive the states of the normal agents into the unsafe set $\mathcal{U}=\{y \in \mathbb{R} | y \geq 20\}$*. The results for LCP and ARC-P are shown in Figures 20 and 21, respectively. The malicious agents are able to achieve their goal only with LCP. The normal agents use ARC-P with parameter* $F = 2$ *to achieve both the agreement and safety conditions. Because both of the adversaries always have larger state values, the consensus process for the normal agents is unaffected and the final consensus state is the average of the* $n - 2F = 4$ *initial states of the agents kept by ARC-P; in this case,* 4.5*. Also, the rate of convergence is* $n - 2F = 4$*.*

Example 4.2. *Consider the same multi-agent network as Example 4.1, with the same initial conditions, but with agents* 6*,* 7*, and* 8 *as malicious agents (i.e.,* $\mathcal{A}= \{6, 7, 8\}$*). This time the malicious*

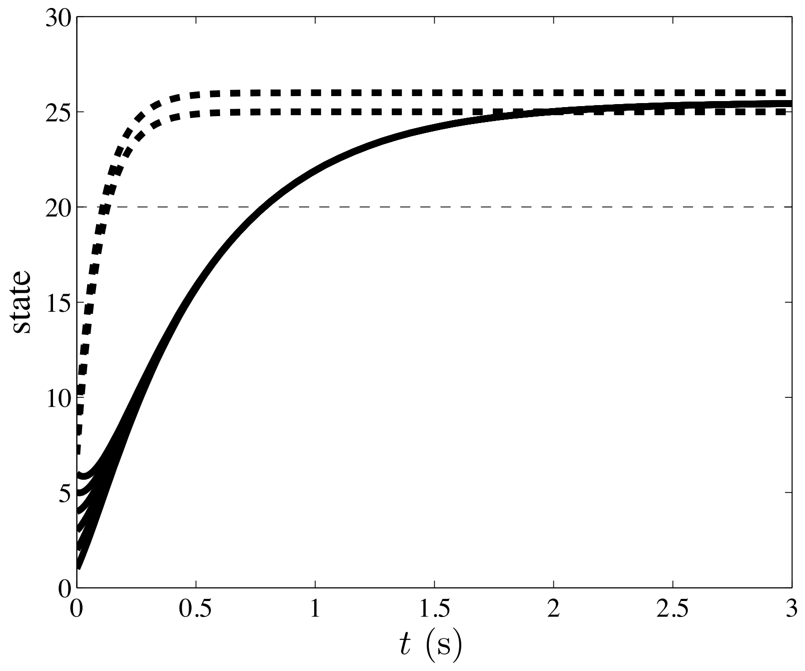


Figure 20: LCP plot where malicious agents try to drive normal agents to the unsafe set $\mathcal{U}$.

agents' dynamics are designed as

$$\ddot{x}_i = -100\pi^2 x_i, \; \forall i \in \mathcal{A}.$$

Thus, they are oscillators with natural frequency 5 *Hz and amplitude given by their initial state. The goal of the malicious agents is to force the states of the normal agents to oscillate at* 5 *Hz.*

The results for LCP and ARC-P are shown in Figures 22 and 23, respectively. As can be seen in Figure 22, the normal agents using LCP synchronize and begin oscillating at 5 *Hz, with a phase lag of* 90° *with respect to the malicious agents. However, for the case of ARC-P with parameter* $F = 3$*, the normal agents achieve the agreement and safety conditions. As the malicious agents move their states inside the range of the values of the normal agents, the limit point for the normal agents is shifted, which can be seen in Figure 23 as a change in the shape of the exponential decay each time the malicious agents move through this range. This shifts the limit point from* 3 *to* 2.6*, without affecting the rate of consensus.*

Example 4.3. *Consider a multi-agent network with* 51 *agents, where only agent* 51 *is an adversary. The initial states of the normal agents are* $x_1(0) = -1$, $x_2(0) = -2$, $\ldots$, $x_{50}(0) = -50$. *The*

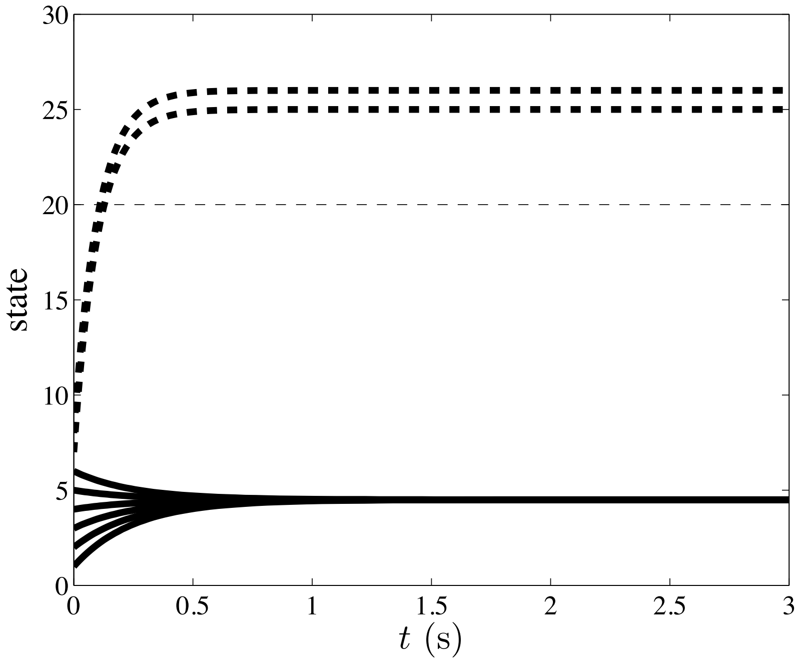


Figure 21: ARC-P plot where malicious agents try to drive normal agents to the unsafe set $\mathcal{U}$.

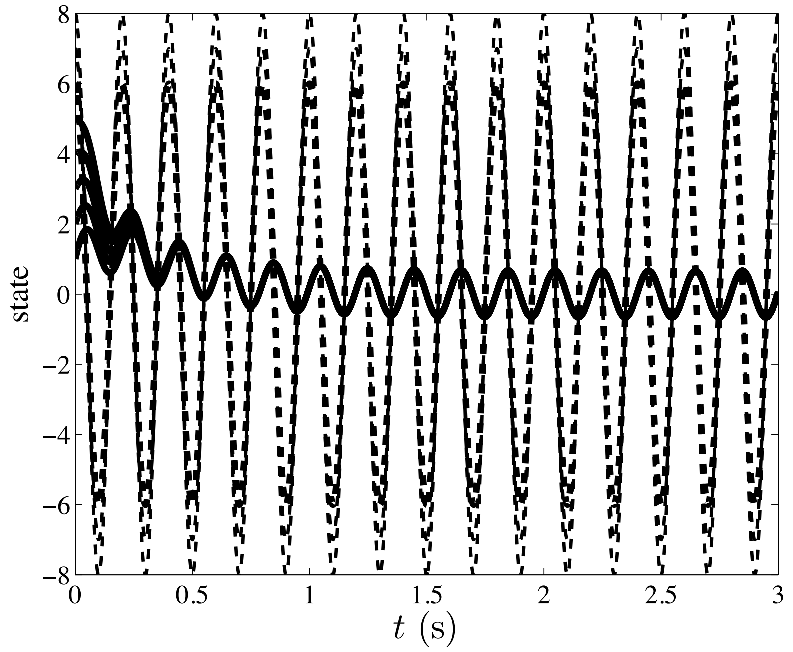


Figure 22: LCP plot where malicious agents try to force normal agents to oscillate at 5 Hz.

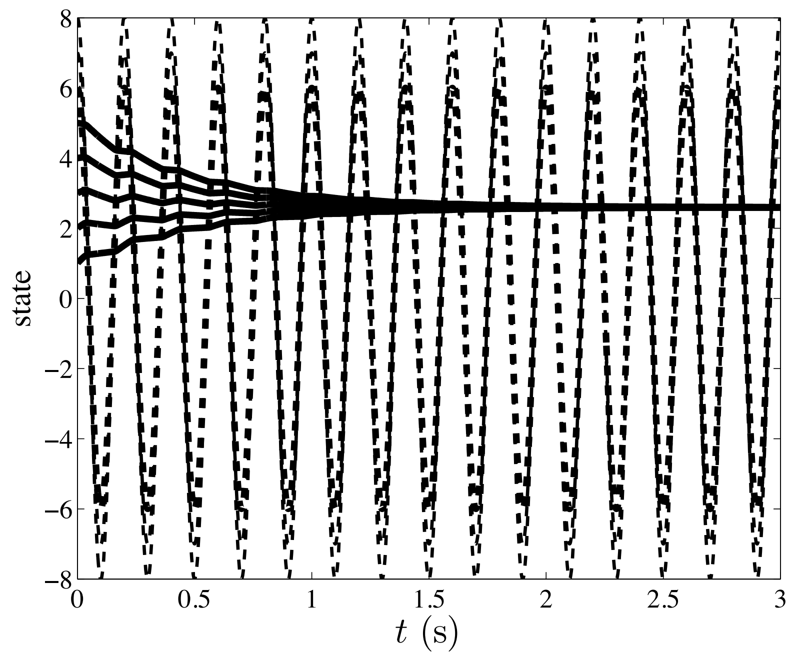


Figure 23: ARC-P plot where malicious agents try to force normal agents to oscillate at 5 Hz.

malicious agents is designed with time-varying dynamics given by the following expressions:

$$\ddot{x}_{51} = -0.25\pi^2 x_{51} \qquad \textit{if } 0 \le t < 1;$$
$$\dot{x}_{51} = -0.4\pi \sin(0.2\pi(t-1)) \quad \textit{if } t \ge 1;$$

and initial conditions $x_{51}(0) = 0$, $\dot{x}_{51}(0) = 15\pi$. The resulting trajectory is

$$x_{51}(t) = \begin{cases} 30\sin(0.5\pi t) & \textit{if } t < 1; \\ 2\cos(0.2\pi(t-1)) + 28 & \textit{if } t \ge 1. \end{cases}$$

The objective of the malicious agent is to bring the states of the normal agents into the unsafe set $\mathcal{U}$ (as in Example 4.1), and force them to oscillate at a frequency of 0.1 Hz. The results for LCP and ARC-P with parameter $F = 1$ are shown in Figures 24 and 25, respectively. In this case, the convergence rates for LCP and ARC-P are 51 and 49, respectively, so the pairwise difference between the states of normal agents becomes negligible by 0.1 s (approximately five time constants) into the simulation. The result is that the trajectory of the normal agents appears to be a single curve

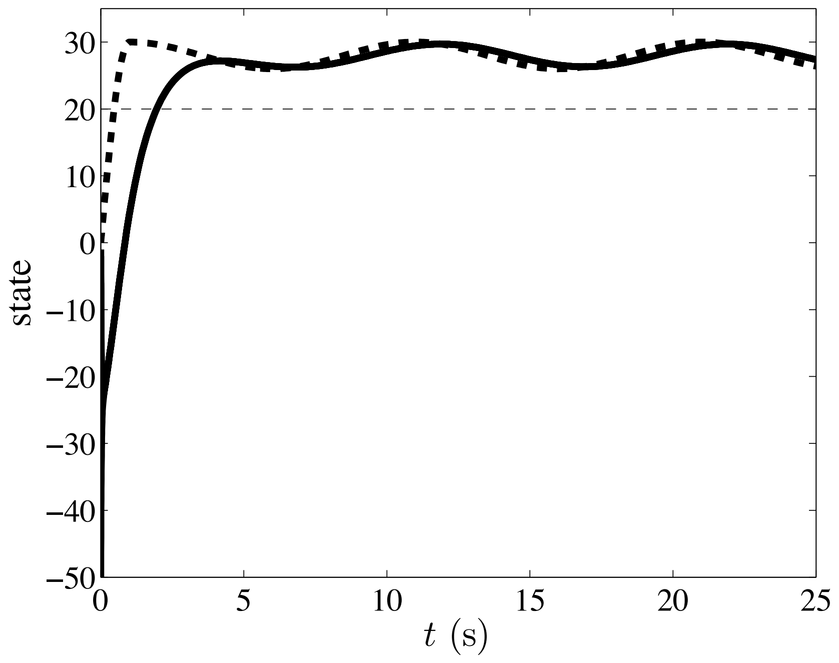


Figure 24: LCP plot where a malicious agent tries to drive 50 normal agents to oscillate in $\mathcal{U}$.

in the figure. Clearly, the adversary is able to achieve its objective only with LCP. The consensus limit point for ARC-P is -25; i.e., the average of $x_1(0),\ldots,x_{49}(0)$.

Example 4.4. *We consider the performance tradeoff required for resilience to adversaries incurred by ARC-P with parameter F. For this purpose, consider the same multi-agent network of Example 4.1, but with no adversaries. As shown in Theorem 4.22, the rate of exponential convergence is $n-2F$. The change in the rate of convergence with F is illustrated in Figure 26 for the eight agent network. Note that ARC-P reduces to LCP in the case $F=0$ (in the upper-left plot of Figure 26). It is also important to note that although the limit observed in the figures is the same in all four cases, this would not be the case with asymmetries in the initial conditions.*

By scaling ARC-P with parameter F in (29) by the factor $\frac{n}{n-2F}$, we may eliminate the tradeoff in performance for robustness to adversaries, and make ARC-P perform as well as LCP. However, LCP may also be scaled to improve its rate of convergence. Moreover, scaling ARC-P incurs a reduction in robustness to time delays as it does with LCP. Indeed, scaling LCP scales the largest eigenvalue of the Laplacian, which reduces the robustness to time delays [150].

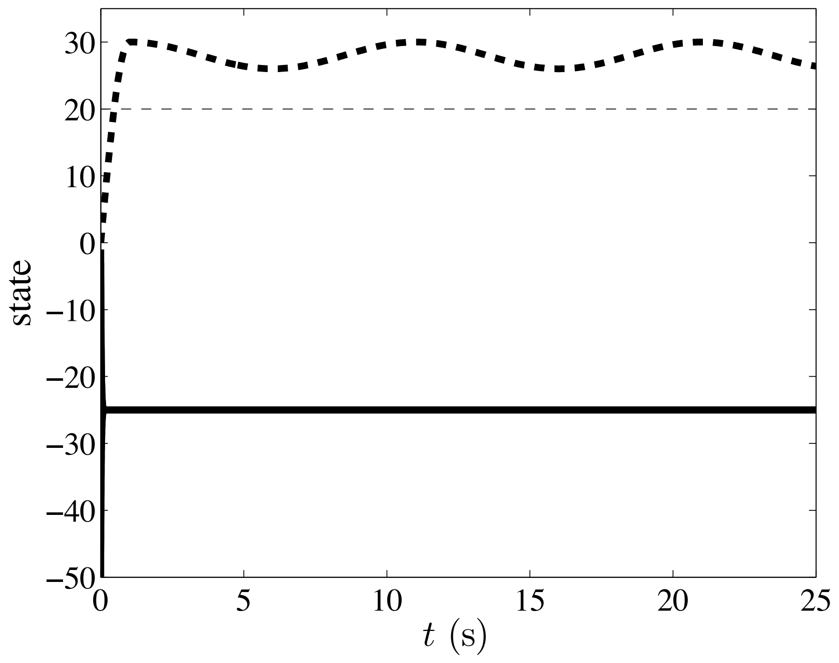


Figure 25: ARC-P plot where a malicious agent tries to drive 50 normal agents to oscillate in $\mathcal{U}$.

4.6 Limitations of Traditional Graph Properties for Analyzing ARC-P

So far we have analyzed ARC-P in complete networks. This analysis shows that ARC-P can achieve CTRAC under any of the adversary models studied in this work (provided the parameter of ARC-P is sufficiently large and the number of adversaries actually present is sufficiently small). However, tighter conditions on the topology require new graph theoretic metrics (which are introduced in Chapter V). The remainder of this chapter is devoted to motivating the need for new metrics. To do this, we introduce conditions on the network topology involving degree properties (i.e., in-degree and out-degree) of nodes in the network and show that these conditions are sufficient to achieve CTRAC using ARC-P with parameter $F \geq F'$ under the F'-total model. Then we show that these degree conditions are *sharp* in the sense that relaxing any of the conditions, even minimally, results in pathological examples where no consensus is reached. For this analysis, we focus on malicious and Byzantine adversaries under the F'-total scope of threat assumption.

The CTRAC analysis proceeds as follows. First, we consider fixed network topology, and prove exponential convergence of $x_{\mathcal{N}}(t)$ to span$\{1_N\}$ whenever the degree conditions are satisfied. This analysis uses properties of $\Psi(x_{\mathcal{N}})$ and $D^+\Psi(x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})})$. We prove safety using an invariant set

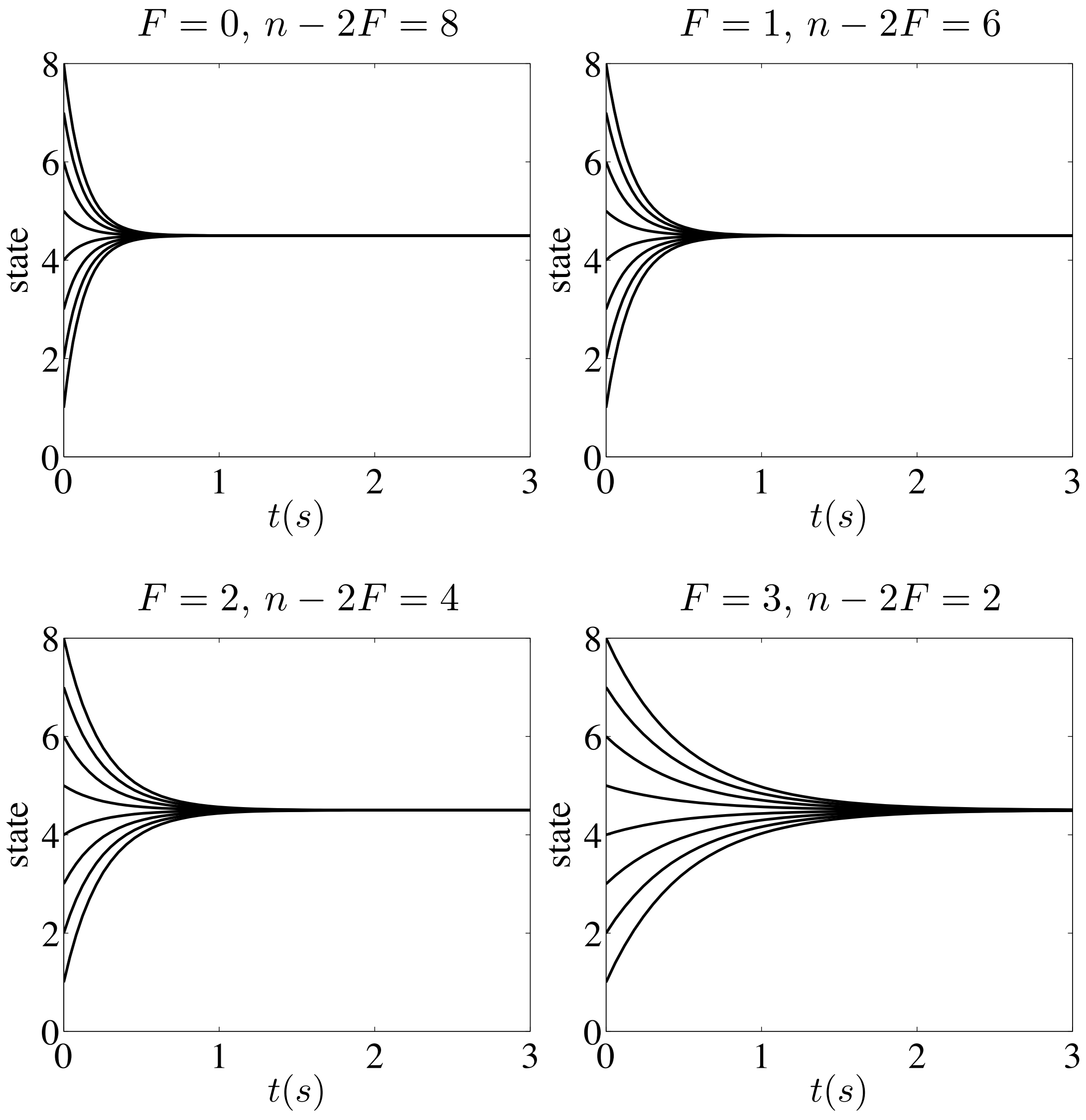


Figure 26: Performance tradeoff for resilience given ARC-P with parameter F

argument. Afterwards, we provide a necessary degree condition, and then generalize the results to the case of time-varying (or switching) topology by using Ψ as a common Lyapunov function.

4.6.1 Analysis of Degree Conditions in Fixed Topology

In this section, we assume that $\sigma(t) \equiv s$ and $\mathcal{D}_s$ belongs to $\Gamma_{M,F} \subset \Gamma_n$ or $\Gamma_{B,F} \subset \Gamma_n$ whenever there are at most $F' \leq F$ malicious or Byzantine adversaries, respectively, in the network (F'-total model). Whenever the digraph belongs to one of these sets, we show that ARC-P with parameter F achieves CTRAC. In particular, when dealing with malicious agents, we consider the following class of digraphs with restricted in-degrees or out-degrees, defined by

$$\Gamma_{M,F} = \{\mathcal{D}_k \in \Gamma_n | \text{ at least one of } M1_F \text{ and } M2_F \text{ holds}\}, \tag{45}$$

where

$M1_F$: $\delta^{\text{in}}(\mathcal{D}_k) \geq \lfloor (n+2F)/2 \rfloor$;

$M2_F$: $\exists \mathcal{S} \subseteq \mathcal{V}, |\mathcal{S}| \geq 2F+1,$ such that $d_i^{\text{out}} = n-1, \forall i \in \mathcal{S}$.

When dealing with Byzantine agents, we require stronger assumptions on the in-degrees and out-degrees. In this case, we define

$$\Gamma_{B,F} = \{\mathcal{D}_k \in \Gamma_n | \text{ at least one of } B1_F \text{ and } B2_F \text{ holds}\}, \tag{46}$$

where

$B1_F$: $\delta^{\text{in}}(\mathcal{D}_k) \geq \lfloor (n+3F)/2 \rfloor$;

$B2_F$: $\exists \mathcal{S} \subseteq \mathcal{V}, |\mathcal{S}| \geq 3F+1,$ such that $d_i^{\text{out}} = n-1, \forall i \in \mathcal{S}$.

It follows from these definitions that $\Gamma_{B,F} \subseteq \Gamma_{M,F}$. Additionally, the conditions in (45) and (46) implicitly bound the parameter F of ARC-P by a function of the total number of agents n.

Specifically, property $M1_F$ implies

$$n - 1 \geq \delta^{\text{in}}(\mathcal{D}_s) \geq \lfloor n/2 \rfloor + F \implies F \leq \lceil n/2 \rceil - 1.$$

Similarly, property $M2_F$ implies

$$n \geq |\mathcal{S}| \geq 2F + 1 \implies 2F \leq n - 1.$$

In either case, $n > 2F \geq 2F'$. Analogously, the properties $B1_F$ and $B2_F$ imply $n > 3F \geq 3F'$. Therefore, it follows that in the following analysis, we require $F \leq \lceil n/2 \rceil - 1$ whenever $\mathcal{D}_s \in \Gamma_{M,F}$, and $F \leq \lceil n/3 \rceil - 1$ whenever $\mathcal{D}_s \in \Gamma_{B,F}$. A consequence of this is that $\delta^{\text{in}}(\mathcal{D}_s) \geq 2F$ for all $\mathcal{D}_s \in \Gamma_{M,F}$ (or $\mathcal{D}_s \in \Gamma_{B,F}$). Therefore, the first case of (29) always holds under these assumptions. From this, (41) may be rewritten using (29) as

$$D^+\Psi(x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})}) = \sum_{m=F+1}^{d_{k_t}(t)+1-F} \left(\xi_{k_t}^m - x_{k_t}\right) - \sum_{l=F+1}^{d_{j_t}(t)+1-F} \left(\xi_{j_t}^l - x_{j_t}\right). \tag{47}$$

This equation is the basis of the agreement argument below. Here we combine Lemmas 4.20 and 4.21 with the assumption $\mathcal{D}_s \in \Gamma_{M,F}$ or $\mathcal{D}_s \in \Gamma_{B,F}$ for malicious or Byzantine adversaries, respectively, in order to show global exponential convergence of $x_{\mathcal{N}}$ to span$\{1_N\}$.

Theorem 4.24. *Consider a time-invariant network topology given by $\mathcal{D}$. Suppose each normal agent in $\mathcal{N}$ uses ARC-P with parameter F under the F'-total model, where $F' \leq F$. Assume that (i) $n > 2F$ and $\mathcal{D} \in \Gamma_{M,F}$ if the adversaries are malicious (or crash nodes), or (ii) $n > 3F$ and $\mathcal{D} \in \Gamma_{B,F}$ if the adversaries are Byzantine. Then $x_{\mathcal{N}}$ globally exponentially converges to the agreement space span$\{1_N\}$ and CTRAC is achieved. Moreover, the convergence to the agreement space is bounded by*

$$\mathit{dist}(x_{\mathcal{N}}(t), \mathit{span}\{1_N\}) \leq 2\sqrt{N}\, \mathit{dist}(x_{\mathcal{N}}(0), \mathit{span}\{1_N\}) e^{-t}. \tag{48}$$

Proof. (i) Fix $t \geq 0$ and consider (47). Since there are at most $F' \leq F$ malicious agents, each term in the first sum is nonpositive and each term in the second sum is nonnegative. If at least one of the sorted values in the second sum is greater than or equal to any of the values in the first, say

$\xi_{k_t}^{m'} \leq \xi_{j_t}^{l'}$, then

$$D^+\Psi(x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})}) \leq -\Psi(x_{\mathcal{N}}), \tag{49}$$

since, in this case,

$$\begin{aligned} D^+\Psi(x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})}) = & \sum_{\substack{m=F+1 \\ m\neq m'}}^{d_{k_t}(t)+1-F} (\xi_{k_t}^{m} - x_{k_t}) - \sum_{\substack{l=F+1 \\ l\neq l'}}^{d_{j_t}(t)+1-F} (\xi_{j_t}^{l} - x_{j_t}) \\ & + \left(\xi_{k_t}^{m'} - \xi_{j_t}^{l'}\right) - \Psi(x_{\mathcal{N}}) \leq -\Psi(x_{\mathcal{N}}). \end{aligned}$$

A sufficient condition for this to hold, given that *all* agents convey the same values to all neighbors, is to ensure there is a common value in the two sums, e.g., $\xi_{k_t}^{m'} = \xi_{j_t}^{l'}$. This is guaranteed if $|\mathcal{J}_{j_t}^{\text{in}} \cap \mathcal{J}_{k_t}^{\text{in}}| > 2F$, which is obviously true if property $M2_F$ holds. If only property $M1_F$ holds, it must also be the case, since otherwise, we reach the contradiction

$$\begin{aligned} n \geq |\mathcal{J}_{j_t}^{\text{in}} \cup \mathcal{J}_{k_t}^{\text{in}}| &= |\mathcal{J}_{j_t}^{\text{in}}| + |\mathcal{J}_{k_t}^{\text{in}}| - |\mathcal{J}_{j_t}^{\text{in}} \cap \mathcal{J}_{k_t}^{\text{in}}| \\ &\geq 2(\lfloor \frac{n}{2} \rfloor + F + 1) - 2F \geq n + 1. \end{aligned}$$

Therefore, (49) holds for all $t \geq 0$, and hence it can be shown that

$$\Psi(x_{\mathcal{N}}(t)) \leq \Psi(x_{\mathcal{N}}(0))e^{-t}.$$

Using (43), we conclude (48). Thus, we have shown global exponential convergence of $x_{\mathcal{N}}$ to span$\{1_N\}$. Finally, the normal values remain in $\mathcal{I}_0$ by Lemma 4.18, and it follows from Lemma 4.16 that $M_{\mathcal{N}}(t)$ is a nonincreasing function of time and $m_{\mathcal{N}}(t)$ is a nondecreasing function of time. Therefore, there is a common limit to which the normal values converge, and CTRAC is achieved.

(ii) The argument is identical to (i), except here to ensure there exists m' and l' such that $\xi_{k_t}^{m'} = \xi_{j_t}^{l'}$ and thereby guarantee (49), we need $|\mathcal{J}_{j_t}^{\text{in}} \cap \mathcal{J}_{k_t}^{\text{in}}| > 3F$. This is required if there are $F' = F$ Byzantine agents in the intersection because of the following argument. Suppose F of the normal agents' states are strictly greater than F other normal agents in the intersection. Then there are $3F$ agents in the intersection, and the adversaries may create $2F$ different values all strictly between these two sets of normal agent states. Thus, at least one more normal agent in the intersection is necessary to ensure a common value. Analogously to (i), property $B2_F$ guarantees $|\mathcal{J}_{j_t}^{\text{in}} \cap \mathcal{J}_{k_t}^{\text{in}}| > 3F$ by construction, and so does property $B1_F$. Otherwise, we reach the contradiction

$$n \geq 2(\lfloor (n+3F)/2 \rfloor + 1) - 3F \geq n+1.$$

□

Notice in the proof of Theorem 4.24 that it is not necessary that there exists a common in-neighbor in the reduced set of in-neighbors of j_t and k_t to show (49), and therefore (48). All that is required is that there exist $\xi_{k_t}^{m'}$ and $\xi_{j_t}^{l'}$ such that $\xi_{k_t}^{m'} \leq \xi_{j_t}^{l'}$. However, because this must hold globally (i.e., for all $x_{\mathcal{N}}(0) \in \mathbb{R}^N$ and $x_{(\mathcal{A},i)}(t) \in \mathbb{R}^M$ for $i \in \mathcal{N}$) and for all $t \geq 0$, it is untenable to depend on the values in those neighborhoods without insisting that there is a normal agent as a common in-neighbor.

4.6.1.1 Necessary Conditions

Next, we consider the following necessary conditions for ARC-P to achieve agreement in networks with fixed topology whenever the adversaries are crash nodes. Clearly, these necessary conditions also hold for malicious and Byzantine nodes.

Theorem 4.25 ($n > 2F$ and $\delta^{\text{in}}(\mathcal{D}) \geq 2F$ are Necessary Conditions)**.** *Consider a time-invariant network topology given by $\mathcal{D}$. Suppose each normal agent in $\mathcal{N}$ uses ARC-P with parameter F under the F'-total or F'-local crash model, where $F' \leq F$. If the agreement condition is satisfied, then $\delta^{in}(\mathcal{D}) \geq 2F$ and $n > 2F$.*

Proof. The case $F = 0$ is vacuously true, so assume $F \geq 1$. If $n \leq 2F$, then each normal node using ARC-P with parameter F removes all of its influence from its neighbors so that its input is always zero. Therefore, agreement is not achieved for any initial values not in span$\{1_N\}$. Therefore, assume $n > 2F$. Suppose $\exists i \in \mathcal{N}$ with $d_i < 2F$ and $\exists \epsilon > 0$ such that $x_j(0) - x_i(0) > \epsilon$ for all $j \in \mathcal{N} \setminus \{i\}$. If $d_i \geq F - 1$, let $F - 1$ of i's in-neighbors be crash adversaries with values smaller than $x_i(0)$ and assume they are compromised at time $t_0 = 0$. Then, $\dot{x}_i \equiv 0$ since both the normal and adversary values are removed. On the other hand, using Lemma 4.18 while treating i as an adversary, ensures $x_j(t) - x_i(t) > \epsilon \; \forall j \in \mathcal{N} \setminus \{i\}$ and $t \geq 0$. □

Recall that the sufficient conditions imply the necessary condition, $\delta^{\text{in}}(\mathcal{D}) \geq 2F$. However, the converse is clearly not true. The question then arises, are the sufficient conditions also necessary? The answer is no, but we delay further discussion of the conservativeness of the sufficient conditions until Section 4.6.3. Next, we study the sufficient conditions under time-varying network topologies.

4.6.2 Analysis of Degree Conditions in Time-Varying Network Topologies

Switching network topologies can arise from a number of factors: temporary removal of edges due to lossy communication channels, the addition or loss of edges caused by mobile agents, and so on. The results of the previous section may be extended to switching topologies in a straightforward manner by assuming $\mathcal{D}_{\sigma(t)} \in \Gamma_{M,F}$ or $\mathcal{D}_{\sigma(t)} \in \Gamma_{B,F}$ for $t \in \mathbb{R}_{\geq 0}$ whenever the adversaries are malicious or Byzantine, respectively. It is shown in Theorem 4.24 that Ψ is a Lyapunov function for each possible digraph $\mathcal{D} \in \Gamma_{M,F}$ or $\mathcal{D} \in \Gamma_{B,F}$. Further, the upper bound on convergence of $x_{\mathcal{N}}$ to span$\{1_N\}$ (48) holds globally and for each digraph $\mathcal{D} \in \Gamma_{M,F}$ or $\mathcal{D} \in \Gamma_{B,F}$. Therefore, Ψ is a common Lyapunov function, thus proving global exponential convergence of $x_{\mathcal{N}}$ to span$\{1_N\}$ for the switched system (27). On the other hand, Lemmas 4.16 and 4.18 hold for all network topologies. Therefore, the same argument used in the proof of Theorem 4.24 may be used for the case of switching topologies. Hence, we have the following result.

Corollary 4.26. *Suppose each normal agent in $\mathcal{N}$ uses ARC-P with parameter F under the F'-total model, where $F' \leq F$. Assume that (i) $n > 2F$ and $\mathcal{D}_{\sigma(t)} \in \Gamma_{M,F}$ for all $t \in \mathbb{R}_{\geq 0}$ if the adversaries are malicious (or crash nodes), or (ii) $n > 3F$ and $\mathcal{D}_{\sigma(t)} \in \Gamma_{B,F}$ for all $t \in \mathbb{R}_{\geq 0}$ if the adversaries are Byzantine. Then the agreement condition is satisfied with the convergence to the agreement space bounded by (48), and the safety condition is satisfied. Therefore, under these conditions, ARC-P achieves CTRAC.*

So far we have studied explicit switching in the network topology when the range of the switching signal is appropriately restricted (i.e., $\mathcal{D}_{\sigma(t)} \in \Gamma_{M,F}$ or $\mathcal{D}_{\sigma(t)} \in \Gamma_{B,F}$ for all $t \in \mathbb{R}_{\geq 0}$). But, even in fixed network topology, the algorithm ARC-P may be viewed as the linear consensus protocol of [152] with state-dependent switching. In ARC-P, the sort and reduce functions effectively remove the influence of a subset of neighbors based on the state values of those neighbors. The remaining relative states are summed as input to the integrator in the same manner as all of the neighbors are in the linear consensus protocol of [152], which justifies the analogy. Hence, the results of Section 4.6.1 provide new insight into the convergence of the protocol of [152] with state-dependent switching.

4.6.3 Examination of Degree Conditions

In this section, we examine the degree conditions $M1_F$ and $M2_F$ that define $\Gamma_{M,F}$ and $B1_F$ and $B2_F$ that define $\Gamma_{B,F}$. Important questions arise with regard to these properties:

(i) How do these conditions relate to known conditions on the maximum number of Byzantine processors in the network [110, 50];

(ii) How do they relate to conditions on the connectivity of the network when reaching agreement with Byzantine processors [50], or detecting and isolating malicious agents [159, 181];

(iii) How conservative are the conditions with respect to achieving the adversarial agreement problem using ARC-P;

(iv) How applicable are the conditions to networks of interest?

The first question has been answered in Section 4.6.1, where we showed that $B1_F$ and $B2_F$ imply $n > 3F$, which is a necessary condition when dealing with Byzantine behavior of finite automata in synchronous networks [110, 50]. The rest of this section is devoted to addressing the remaining questions.

To address (ii), we show that $M1_F$ and $M2_F$–and therefore also $B1_F$ and $B2_F$–imply $\kappa(\mathcal{G}) \geq 2F + 1$ whenever the network is bidirectional. This is a necessary and sufficient condition for the existence of an algorithm that can (a) ensure agreement of the nonfaulty agents in the presence of at most F Byzantine agents in synchronous networks [50], or (b) detect and isolate up to F malicious agents in linear consensus networks [159, 181]. However, the conditions $B1_F$ and $B2_F$ *do not* imply $2F + 1$-strong connectivity in digraphs. Indeed, conditions $M2_F$ and $B2_F$ do not even guarantee $\delta^{\text{out}}(\mathcal{D}) > 0$. In fact, an example of a network with $\delta^{\text{out}}(\mathcal{D}) = 0$ is given in the following section that illustrates our results. For such networks, detection of malicious agents in linear consensus networks is not possible [159, 181]. Therefore, CTRAC is achievable in cases where detection is not possible.

Theorem 4.27. *If $F \in \{0, 1, \ldots, \lfloor n/2 \rfloor - 1\}$ and the digraph satisfies (i) $M1_F$ or (ii) $M2_F$, then the underlying graph $\mathcal{G}$ is $2F + 1$-connected.*

Proof. (i) Fix $F \in \{0, 1, \ldots, \lfloor n/2 \rfloor - 1\}$ and consider the underlying graph $\mathcal{G}$, which must satisfy $\delta(\mathcal{G}) \geq \lfloor n/2 \rfloor + F$. By Menger's Theorem, $\kappa(\mathcal{G}) \geq 2F + 1$ is equivalent to $\mathcal{G}$ having at least $2F + 1$ vertex-disjoint paths between any distinct vertices $i, j \in \mathcal{V}$. Indeed, this is the case if $|\mathcal{J}_i \cap \mathcal{J}_j| \geq 2F + 2$ for all $i, j \in \mathcal{V}$. On the other hand, we know that $|\mathcal{J}_i \cap \mathcal{J}_j| \geq 2F + 1$ (c.f. the proof of Theorem 4.24). From this we conclude that if $(i, j) \notin \mathcal{E}_{\mathcal{G}}$ then there are at least $2F + 1$ vertex-disjoint paths between i and j. Therefore, assume there exists $i, j \in \mathcal{V}$ such that $(i, j) \in \mathcal{E}_{\mathcal{G}}$ and $|\mathcal{J}_i \cap \mathcal{J}_j| = 2F + 1$. In this case, there are $2F$ vertex-disjoint paths accounted for with vertices in $\mathcal{J}_i \cap \mathcal{J}_j$. But, because $F \leq \lfloor n/2 \rfloor - 1$, we know

$$|\mathcal{J}_i|, |\mathcal{J}_j| \geq \lfloor n/2 \rfloor + F + 1 \geq 2F + 2,$$

which means there exists $i' \in \mathcal{J}_i \setminus \mathcal{J}_i \cap \mathcal{J}_j$ and $j' \in \mathcal{J}_j \setminus \mathcal{J}_i \cap \mathcal{J}_j$. If $(i', j') \in \mathcal{E}_{\mathcal{G}}$, then i, i', j', j is the last vertex-disjoint path necessary to conclude $2F + 1$-connectivity. If $(i', j') \notin \mathcal{E}_{\mathcal{G}}$, then we know that $|\mathcal{J}_{i'} \cap \mathcal{J}_{j'}| \geq 2F + 1$, and there are at most $2F - 1$ vertices in $(\mathcal{J}_{i'} \cap \mathcal{J}_{j'}) \cap (\mathcal{J}_i \cap \mathcal{J}_j)$ because i and j cannot be in $\mathcal{J}_{i'} \cap \mathcal{J}_{j'}$. Hence, there exists $m \in \mathcal{J}_{i'} \cap \mathcal{J}_{j'} \setminus \mathcal{J}_i \cap \mathcal{J}_j$, so that i, i', m, j', j is the last vertex-disjoint path necessary to conclude $2F + 1$-connectivity.[29]

(ii) Any vertex cut must contain at least $2F + 1$ vertices, because otherwise a vertex remains in $\mathcal{S}$ adjacent to all other vertices. □

To address the conservativeness of the conditions with respect to convergence of ARC-P, we show that we can do no better using traditional metrics such as in-degree, out-degree, or connectivity (of the underlying graph). We do this by demonstrating that minimally relaxing these conditions leads to pathological examples with high connectivity in which ARC-P does not achieve agreement.

Example 4.5 (Relax $M1_F$ with $\delta^{\text{in}}(\mathcal{D}) = \lfloor (n + 2F)/2 \rfloor - 1$). [30] *Consider the network topology in Figure 27, in which $K_{\lceil n/2 \rceil}$ is the complete digraph on $\lceil n/2 \rceil$ vertices, and each vertex in X has exactly F neighbors in Y and each vertex in Y has either $F - 1$ or F neighbors in X. Now,*

[29]A similar result appears in Theorem 6 of [69], which is ascribed to Chartrand and Harary [32]. The theorem states that if $\delta(\mathcal{G}) \geq (n - 2 + q)/2$ where $1 \leq q \leq n - 1$, then $\kappa(\mathcal{G}) \geq q$.

[30]This pathological example was suggested by Sundaram in personal correspondence and later appeared in [210].

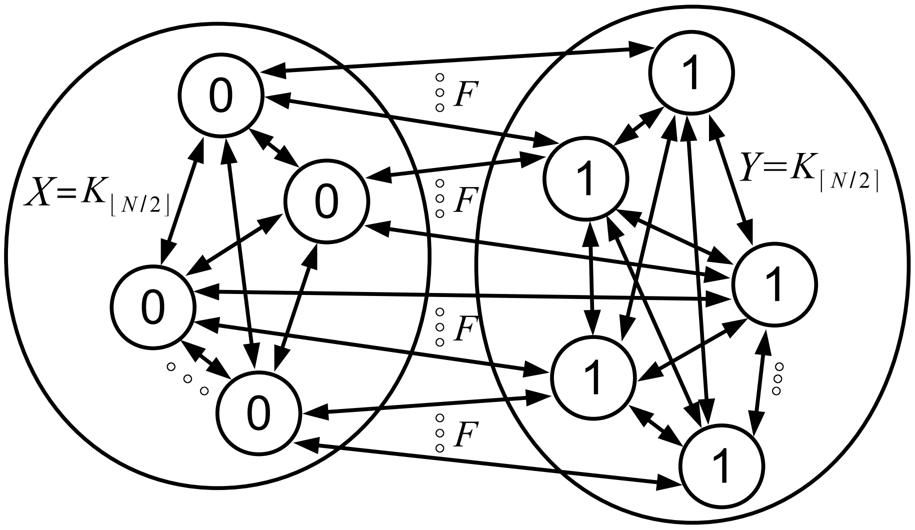


Figure 27: Relax $M1_F$: $\delta^{\text{in}}(\mathcal{D}) = \lfloor (n+2F)/2 \rfloor - 1$.

assume there are no adversaries and let all states in X have value 0 and all states in Y have value 1. Then, by (29), all agents in X will remove the influence of their neighbor in Y and vice versa. Hence, no consensus is reached, and no agent even changes its state. Furthermore, this graph is $(\lfloor n/2 \rfloor + F - 1)$-connected, which for large n may be much larger than $\kappa(\mathcal{G}) \geq 2F + 1$.

From this example, we see that reducing the minimum in-degree by just one from $M1_F$ is not sufficient for global convergence of $x_{\mathcal{N}}$ to span$\{1_N\}$. Additionally, in this example, the connectivity is very high. This suggests that the minimum in-degree and connectivity are not appropriate metrics to use in characterizing the network topologies in which ARC-P achieves agreement. The following example demonstrates that the minimum out-degree is also inadequate and further emphasizes the inadequacy of connectivity. Here, the number of nodes in $\mathcal{S}$ from $M2_F$ is reduced by one.

Example 4.6 (Relax $M2_F$ with $|\mathcal{S}| = 2F$ and $d_i^{\text{out}} = n - 2, \forall i \in \mathcal{V} \setminus \mathcal{S}$, so that $\delta^{\text{out}}(\mathcal{D}) = n - 2$)**.** *Consider the example of Figure 28, which has $|\mathcal{S}| = 2F$, with $\mathcal{S} = \mathcal{S}' \cup \{j\}$ and $d_i^{in} = n - 2$, $\forall i \in \mathcal{V} \setminus \mathcal{S}$, so that $\delta^{out}(\mathcal{D}) = n - 2$. Since $d_j^{in} = 2F - 1$, this example does not satisfy the necessary condition of Theorem 4.25. The argument in the proof shows that the agreement condition is not satisfied. Since the underlying graph is complete, this digraph is $(n-1)$-connected, which emphasizes the inadequacy of connectivity in characterizing the convergence properties of ARC-P.*

Example 4.7 (Relax $B1_F$ with $\delta^{\text{in}}(\mathcal{D}) = \lfloor (n+3F)/2 \rfloor - 1$)**.** *Consider the digraph shown in Figure 29. In the figure, the digraph is partitioned into 3 cliques (i.e., complete subdigraphs),*

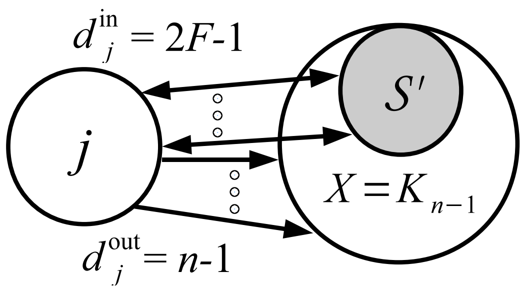

$d_j^{\text{in}} = 2F\text{-}1$
S'
j
$X = K_{n-1}$
$d_j^{\text{out}} = n\text{-}1$

Figure 28: Relax $M2_F$: $|\mathcal{S}| = 2F$ and $\delta^{\text{out}}(\mathcal{D}) = n - 2$.

$\mathcal{D} = X_1 \cup X_2 \cup X_3$, and each clique has $\lfloor n/2 \rfloor - \lfloor F/2 \rfloor$, F, and $\lceil n/2 \rceil - \lceil F/2 \rceil$ nodes, respectively. For clarity, we do not show edges internal to the cliques. We only show one representative node from the sets X_1 and X_3, but all nodes in each of these sets have F in-neighbors in each of the other two sets–which is possible since $n > 3F$. This leads to an in-degree of $d_i^{in} = \lfloor n/2 \rfloor + \lceil 3F/2 \rceil - 1$ for each $i \in X_1$, and an in-degree of $d_j^{in} = \lceil n/2 \rceil + \lfloor 3F/2 \rfloor - 1$ for each $j \in X_3$. On the other hand, the nodes in X_2 exchange information bidirectionally with all other nodes, so that $d_k^{in} = d_k^{out} = n - 1$ for all $k \in X_2$. Therefore, the minimum in-degree depends on the parity of n and F. If they have the same parity, $|X_1| = |X_3|$, and $\delta^{in}(\mathcal{D}) = \lfloor (n + 3F)/2 \rfloor - 1$. If n is odd and F is even, $|X_3| = |X_1| + 1$, and $\delta^{in}(\mathcal{D}) = \lfloor (n+3F)/2 \rfloor - 1$. Finally, if n is even and F is odd, $|X_3| = |X_1| - 1$, and $\delta^{in}(\mathcal{D}) = \lfloor (n + 3F)/2 \rfloor - 1$, which means, in any case, $B1_F$ is minimally relaxed.

To show that ARC-P may not achieve agreement in this digraph, let each node in X_1 and X_3 have initial value 1 and 3, respectively. Suppose all nodes in X_2 are Byzantine, and they transmit a constant trajectory of 1 to nodes in X_1 and 3 to nodes in X_3. Then nodes in X_1 remove the influence from their F neighbors in X_3 and vice versa, so that agreement fails.

Example 4.8 (Relax $B2_F$ with $|\mathcal{S}| = 3F$ and $\mathcal{S} = S_1 \cup S_2 \cup S_3$)**.** *Consider the digraph in Figure 30. In this example, $\mathcal{S} = S_1 \cup S_2 \cup S_3$, with $|S_i| = F$ for $i = 1, 2, 3$. The remaining nodes in $\mathcal{V} \setminus \mathcal{S}$ form a clique, K_{n-3F}. Nodes in S_1 and S_3 have value 1 and 3, respectively, and nodes in $\mathcal{V} \setminus \mathcal{S}$ have value 2. Nodes in S_2 are Byzantine and send values 1, 2, and 3, respectively, to nodes in S_1, $\mathcal{V} \setminus \mathcal{S}$, and S_3. Clearly, as in the previous examples, the normal nodes do not reach agreement, but remain fixed at their initial values.*

Although this section is replete with pathological examples in which ARC-P fails to achieve agreement–even when the networks have high minimum degrees and high connectivity–the news

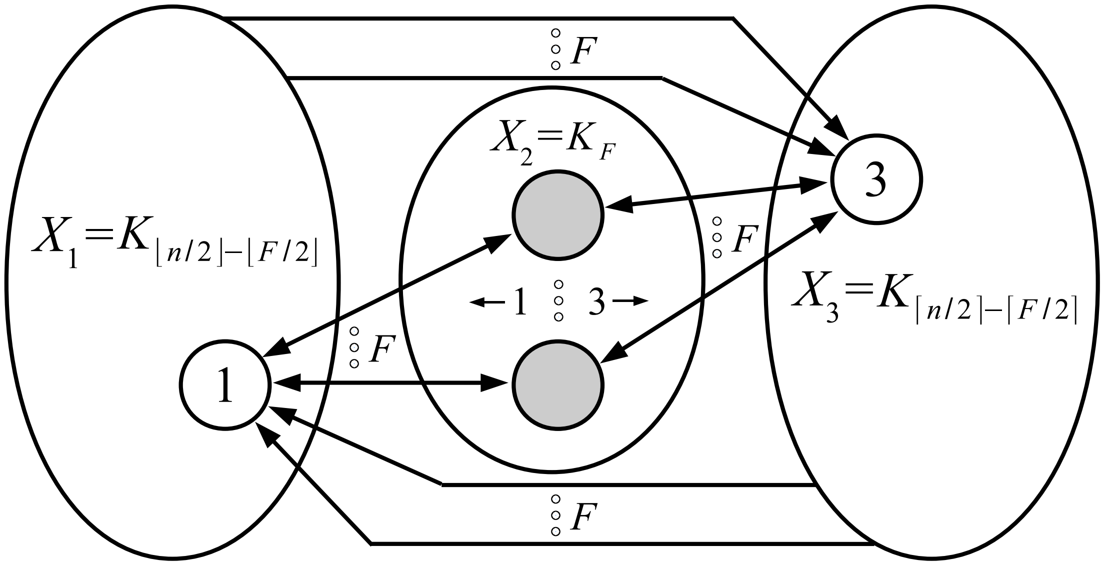

$X_1 = K_{\lfloor n/2 \rfloor - \lfloor F/2 \rfloor}$
$X_2 = K_F$
$X_3 = K_{\lceil n/2 \rceil - \lceil F/2 \rceil}$
1
3
F
F
F
F
← 1
3 →

Figure 29: Relax $B1_F$: $\delta^{\text{in}} = \lfloor (n+3F)/2 \rfloor - 1$.

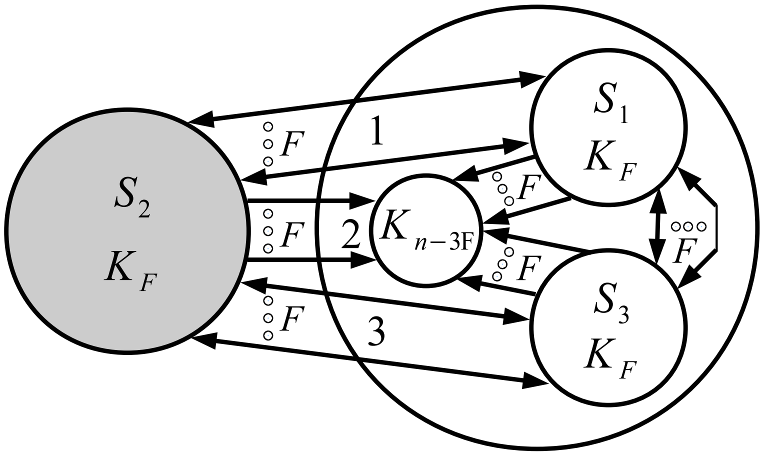


Figure 30: Relax $B2_F$: $|\mathcal{S}| = 3F$ and $\mathcal{S} = S_1 \cup S_2 \cup S_3$.

is not all bad. First, we now know that the sufficient conditions studied in Section 4.6.1 are the best we can have using minimum degrees and connectivity. Second, we can discern a pattern in the various examples. A common property is that there are pairs of subsets with high connectivity within the subsets, but nodes in each subset have relatively few in-neighbors outside of their subsets. Therefore, new topological conditions for digraphs that deal with (a) pairs of subsets of nodes and (b) the number of nodes with "enough" in-neighbors outside of their respective subset are essential to characterize under what conditions resilient distributed algorithms, such as ARC-P, converge.

4.7 Simulation Examples

In this section we present a couple of simulation examples to illustrate the results of this chapter. The first example looks at a time-invariant topology with a single Byzantine agent and the second example considers a malicious agent in time-varying networks.

Example 4.9 (Morale dynamics on fixed topology with single Byzantine agent)**.** *Consider a variation of the Byzantine generals problem in which the loyal generals attempt to improve the morale of their troops and reach consensus on the level of morale despite the influence of a subset of Byzantine generals. In addition, the troops have no knowledge of the goal of the generals. For the purposes of this example, the state value represents the level of morale. The sign of the value indicates either good (positive) or bad (negative) morale and the magnitude signifies the relative levels of morale. Here, we assume that the morale dynamics of each agent behave as an integrator with the input (influence) either given by ARC-P, as in (29), or simply by the sum of relative morale values:*

$$\dot{x}_i(t) = \sum_{j \in \mathcal{N}_i^{in}} \left(x_{(j,i)}(t) - x_i(t) \right), \; x_i(0) = x_{0_i},$$

where $x_i(t)$ is the morale value of agent i and x_{0_i} is the initial morale value of agent i. We refer to the influence rule above as the linear consensus protocol (LCP), which is a special case of the weighted sum of relative states studied extensively in the literature [152].

Each general is able to continuously influence all of the troops and the other generals, and the generals can provide different influence to different individuals. The influence network is shown in Figure 31, in which nodes 17 through 20 form a clique and are the generals (shown as squares). The other nodes are the troops (shown as circles). Troop i has initial morale $-i$, for $i = 1, \dots, 16$, and the generals have initial morale of 1, 2, 3, and 4, respectively, for nodes 17, 18, 19, and 20.

The central question of this example is whether either LCP or ARC-P can ensure that the troops reach asymptotic consensus on a positive morale given that it is possible that one of the generals is Byzantine (i.e., $F = 1$). Observe that the network of Figure 31 satisfies $B2_F$ whenever $F = 1$, with $\mathcal{S} = \{17, 18, 19, 20\}$, and can therefore sustain the compromise of a single agent as Byzantine whenever the troops and loyal generals use ARC-P. In this case, we choose node 20 to be the Byzantine general. In order to elude detection, the Byzantine general conveys a morale trajectory

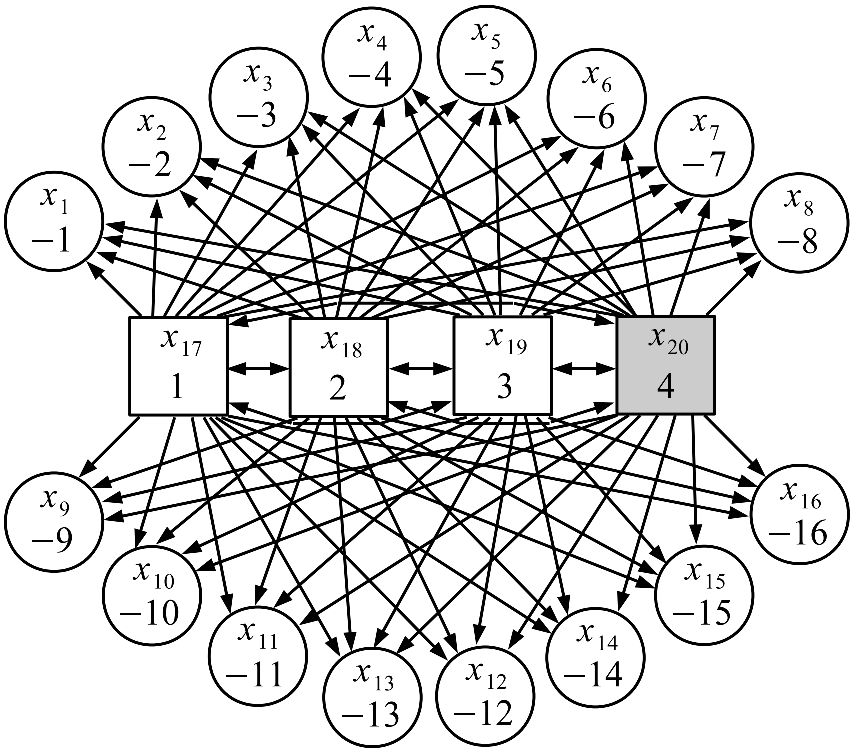
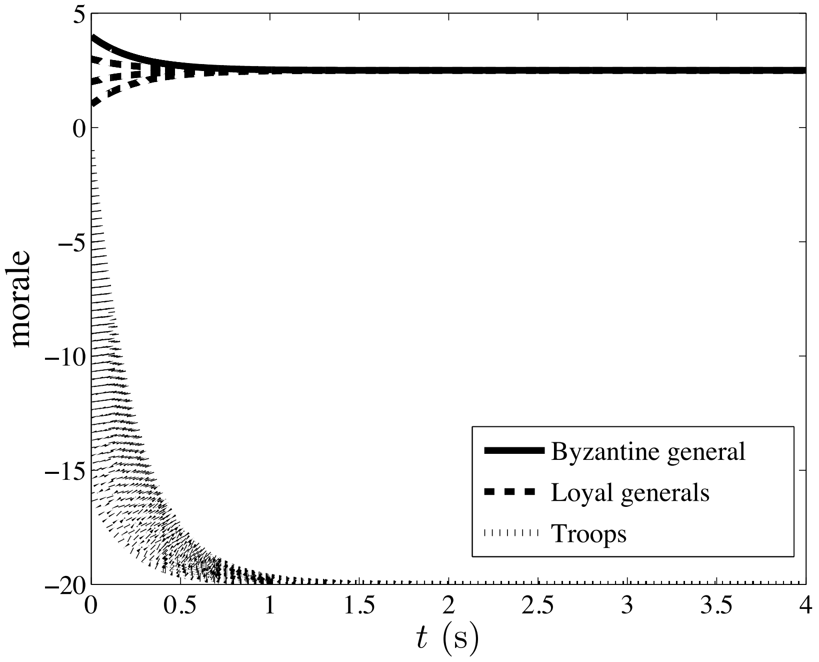


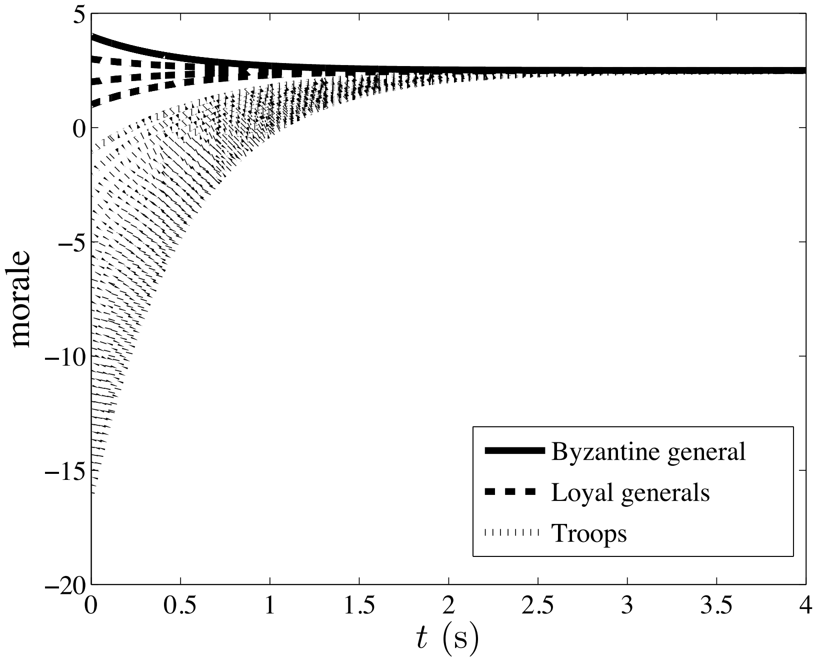
Figure 31: Influence network. Squares are generals and circles are troops. Node 20 is Byzantine.

that satisfies the preassigned strategy–either ARC-P or LCP–to the other generals. But, to the troops, the Byzantine general conveys a highly negative morale of -87.5. *The results for LCP and ARC-P are shown in Figure* 32. *The Byzantine morale trajectory shown in the figures is the one conveyed to the other generals. Using LCP, the troops reach consensus at a negative morale of* -20 *and the generals reach consensus at 2.5, whereas with ARC-P the troops reach consensus at the same value of the other generals at 2.5.*

This example illustrates an important property of ARC-P: *It only requires local information for resilience against adversaries.* In contrast, without nonlocal information, the detection and identification techniques of [157, 158, 160, 159, 179, 180, 181, 186] would not successfully detect the Byzantine general. This is because from the perspective of the loyal generals, the Byzantine general behaves as it should and they receive no feedback from the troops. From the perspective of the troops, the Byzantine general appears to be influenced by no other node. Hence, without prior knowledge of at least some nonlocal aspects of the network topology, the Byzantine general remains undetected.



(a) LCP.



(b) ARC-P.

Figure 32: Byzantine general attempts to reduce morale; it succeeds with LCP, fails with ARC-P.

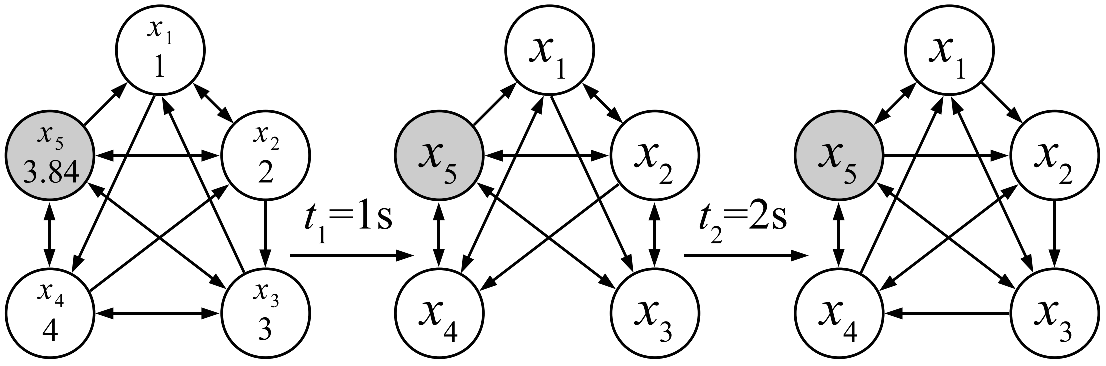


Figure 33: Communication network with switching topology. Agent 5 is malicious.

Example 4.10 (Malicious Damped Oscillator)**.** *In this example, we consider time-varying networks. We assume that the topology switches as illustrated in Figure 33, where node 5 is malicious. The minimum degree in each digraph in the sequence is 3; therefore, all topologies satisfy $M1_F$ with $F = 1$. Hence, Corollary 4.26 implies that whenever ARC-P with parameter $F \in \{1, 2\}$ is used by each normal node, and at most one node is malicious, then CTRAC is assured. For the sake of illustration, we demonstrate this result for the case of a malicious damped oscillator that has frequency 5 Hz. The malicious agent's trajectory decays at a slower rate than the consensus dynamics, as shown in Figure 34. As expected, the normal agents using ARC-P with parameter $F = 1$ achieve CTRAC with an exponential rate of convergence.*

4.8 Summary

In this chapter, we introduce a novel continuous-time consensus problem, the Continuous-Time Resilient Asymptotic Consensus (CTRAC), which is formulated in terms of compromised nodes (or adversaries). We define several adversary models and introduce a continuous-time consensus protocol. This protocol, referred to as the Adversarial Robust Consensus Protocol (ARC-P), demonstrates resilience to extreme influence from adversaries under sufficient topological conditions. We prove that the protocol is well formed in the sense that existence and uniqueness of solutions are assured. We analyze ARC-P for the special case of complete networks under all of the variations of adversary models, and show that for the specific class of complete networks, it is possible to terminate in finite time with an ϵ-approximate solution. Then, we analyze ARC-P in a class of networks characterized by their degree properties and show that ARC-P with parameter F achieves CTRAC whenever the

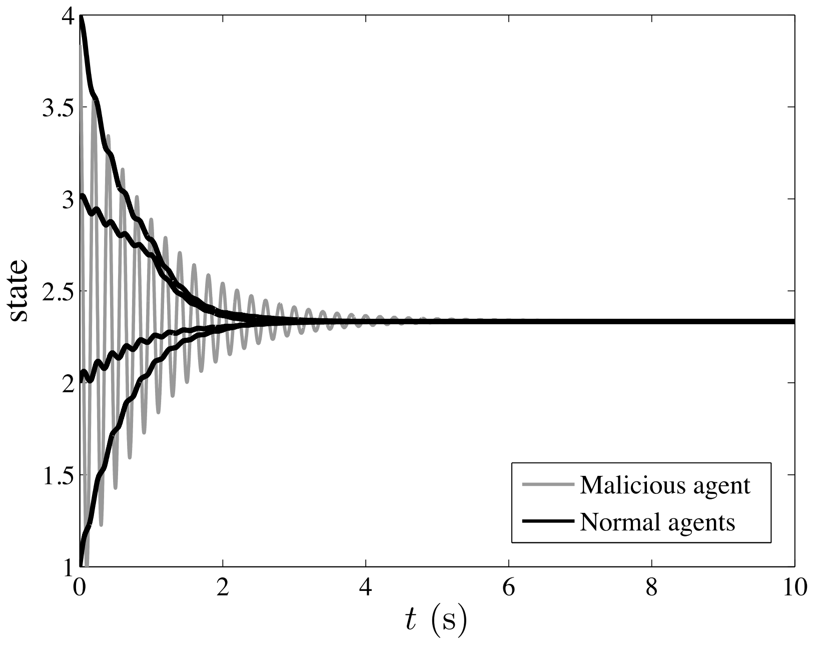


Figure 34: Malicious agent tries to prevent consensus in switching topology of Figure 33.

parameter is sufficiently large (but not too large) and the bound on the total number of adversaries is no more than F. These results apply also to time-varying networks, as long as the networks always satisfy the degree conditions. Finally, we demonstrate the need for new topological conditions that provide sufficient local redundancy so that sufficient information remains after the removal of extreme values in the neighborhood. The following chapter addresses this need with the introduction of network robustness.

CHAPTER V

CONTINUOUS-TIME RESILIENT ASYMPTOTIC CONSENSUS IN ROBUST NETWORKS

This chapter continues our study of Continuous-Time Resilient Asymptotic Consensus (CTRAC). In the previous chapter, we introduced the Adversarial Robust Consensus Protocol (ARC-P), which mitigates adversarial influence by eliminating the extreme values in the neighborhood of the node. We demonstrated the effectiveness of ARC-P under several different adversary models. However, the conditions that ensure CTRAC is achieved are quite conservative and do not capture the essential topological conditions required by ARC-P (or other algorithms that use only local information).

In this chapter, we introduce a novel graph theoretic property referred to as *network robustness*, or just *robustness* [210, 116, 115]. Network robustness formalizes the notion of sufficient local redundancy of information flow in the network. This property of sufficient local redundancy is important for algorithms such as ARC-P that remove local information in order to maintain resilience against faults or adversaries.

We also introduce a variant of ARC-P that is more selective in how it removes extreme values. Because this modification of ARC-P also uses weights, we refer to the algorithm as weighted ARC-P with selective reduce, or just ARC-P2 for brevity. Using robustness, we fully characterize the network topologies in which ARC-P2 is able to achieve CTRAC under the F-total malicious and crash models. The selective nature of ARC-P2 is crucial for this analysis, and therefore, the version of ARC-P studied in Chapter IV is *not* studied here. Robustness also enables the formulation of the tight necessary and sufficient conditions on the normal network for CTRAC to be achieved in the presence of Byzantine adversaries under any of the scope of threat assumptions. Separate necessary and sufficient conditions are presented for the F-local and f-fraction local malicious models.

The chapter is organized as follows. Section 5.1 motivates and defines the definition of ARC-P2. Section 5.2 revisits the issue of existence and uniqueness of solutions whenever ARC-P2 is used by the normal nodes. Section 5.3 defines the various notions of network robustness and demonstrates several useful properties of robust networks. The resilient consensus results are presented in Section 5.4. A simulation example is given in Section 5.5. Finally, Section 5.6 concludes the chapter.

5.1 Weighted ARC-P with Selective Reduce (ARC-P2)

Linear consensus algorithms have attracted significant interest in recent years [152, 168], due to their applicability in a variety of contexts. In such strategies, at time t, each node senses or receives information from its neighbors, and changes its value according to the Linear Consensus Protocol (LCP)

$$\dot{x}_i(t) = \sum_{j \in \mathcal{N}_i^{\text{in}}(t)} w_{(j,i)}(t) \left(x_{(j,i)}(t) - x_i(t) \right), \tag{50}$$

where $x_{(j,i)}(t) - x_i(t)$ is the relative state of agent j with respect to agent i and $w_{(j,i)}(t)$ is a piecewise continuous weight assigned to the relative state at time t.

Different conditions have been reported in the literature to ensure asymptotic consensus is reached [166, 143, 90]. It is common to assume that the weights are nonnegative, uniformly bounded, and piecewise continuous. That is, there exist constants $\alpha, \beta \in \mathbb{R}_{>0}$, with $\beta \geq \alpha$, such that the following conditions hold:

- $w_{(j,i)}(t) = 0$ whenever $j \notin \mathcal{N}_i^{\text{in}}(t), i \in \mathcal{N}, t \in \mathbb{R}_{\geq 0}$;
- $\alpha \leq w_{(j,i)}(t) \leq \beta, \forall j \in \mathcal{N}_i^{\text{in}}(t), i \in \mathcal{N}, t \in \mathbb{R}_{\geq 0}$;

One problem with LCP given in (50) is that it is not resilient to misbehaving nodes. In fact, it is shown in [90, 75] that a single ‘leader’ node can cause all agents to reach consensus on an arbitrary value of its choosing (potentially resulting in a dangerous situation in physical systems).

5.1.1 Informal Description of ARC-P2

As shown in the last chapter, ARC-P achieves CTRAC under appropriate topological conditions. Using ARC-P with parameter F, a normal node i always removes $2F$ values in its inclusive neighborhood $\mathcal{J}_i^{\text{in}}(t)$ (i.e., the F largest and smallest), regardless of the normal node’s value. In particular, node i’s *own value* may be removed, and if node i has one of the extreme normal values (i.e., $x_i(t) \in \{m_{\mathcal{N}}(t), M_{\mathcal{N}}(t)\}$ where $m_{\mathcal{N}}(t) = \min_{j \in \mathcal{N}}\{x_j(t)\}$ and $M_{\mathcal{N}}(t) = \max_{j \in \mathcal{N}}\{x_j(t)\}$), then potentially *more than* F values greater or less than $x_i(t)$ may be removed. This is unnecessary to ensure safety. Therefore, weighted ARC-P with selective reduce (ARC-P2) and parameter F modifies this behavior by using the fact that the normal node i is sure that its own value is good, and therefore should be kept. Instead of always removing the F largest and smallest values, only F

values strictly larger or smaller than its own value are removed. In cases where a normal node is an extreme value of its neighborhood, F or fewer values are removed with ARC-P2.

ARC-P2 also differs from ARC-P because ARC-P2 has time-varying, piecewise continuous, nonnegative, uniformly bounded weights $w_{(j,i)}(t) \in \mathbb{R}_{\geq 0}$ associated to each pair $(j, i) \in \mathcal{V} \times \mathcal{N}$. The weights are uniformly bounded above by $\beta \in \mathbb{R}_{>0}$. We define (without loss of generality) $w_{(j,i)}(t) \equiv 0$ for $j \notin \mathcal{N}_i^{\text{in}}(t)$. Whenever $j \in \mathcal{N}_i^{\text{in}}(t)$, there is the uniform lower bound $\alpha \in \mathbb{R}_{>0}$, so the weights are bounded as $0 < \alpha \leq w_{(j,i),k}(t) \leq \beta$.

As with ARC-P, we consider ARC-P2 with either parameter $F \in \mathbb{Z}_{\geq 0}$ or $f \in [0, 1]$. For consistency in notation, let $F_i(t) \equiv F$ if the parameter is F, and let $F_i(t) = \lfloor f d_i(t) \rfloor$ if the parameter is f (note that all normal nodes either use one parameter or the other, depending on the scope of threat model assumed). Whenever the normal nodes assume the F-total or F-local models, at most F of node i's neighbors may be compromised, and the parameter used is F.[31] Similarly, whenever the normal nodes assume the f-fraction local model, at most $\lfloor f d_i(t) \rfloor$ of node i's neighbors may be compromised, and the parameter used is f. Since node i is unsure of *which* neighbors may be compromised, it removes the extreme values with respect to its own value. The following steps describe ARC-P2.

1. At time t, each normal node i obtains the values of its in-neighbors, and forms a sorted list.

2. If there are less than $F_i(t)$ values strictly larger than its own value, $x_i(t)$, then normal node i removes all values that are strictly larger than its own. Otherwise, it removes precisely the largest $F_i(t)$ values in the sorted list.[32] Likewise, if there are less than $F_i(t)$ 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 $F_i(t)$ values.

3. Let $\mathcal{R}_i(t)$ denote the set of nodes whose values were removed by normal node i in step 2 at

[31]Of course, if the scope of threat model *assumed* (i.e., at design time) is not the *true* scope of threat, then ARC-P2 may fail to be resilient.

[32]Ties may be broken arbitrarily; however, it is required that the algorithm is able to match the correct weights to the values kept.

time t. Each normal node i applies the update[33]

$$\dot{x}_i(t) = \sum_{j \in \mathcal{N}_i^{\text{in}}(t) \backslash \mathcal{R}_i(t)} w_{(j,i)}(t) \left(x_{(j,i)}(t) - x_i(t) \right), \tag{51}$$

Observe that the set of nodes removed by normal node i, $\mathcal{R}_i(t)$, is possibly time-varying. Hence, even though the underlying network topology may be fixed, ARC-P effectively induces switching behavior, and can be viewed as the linear update of (50) with a specific rule for state-dependent switching (the rule given in step 2).

5.1.2 Formal Description of ARC-P2

The previous section outlined the steps taken in ARC-P2 to remove the influence of nodes with extreme values. In order to analyze (27) for existence and uniqueness of solutions whenever the normal agents use ARC-P2, it is useful to express ARC-P2 as a composition of functions. For this, we require the following definitions, which are analogous to the functions defined in Definition 4.9 on page 110.

Definition 5.1. *Let $k \in \mathbb{N}$ and $F_i \in \mathbb{Z}_{\geq 0}$. Denote the elements of vectors $\xi, w, z \in \mathbb{R}^k$ by ξ_l, w_l, and z_l, respectively, for $l = 1, 2, \ldots, k$. Then:*

*(i) The (ascending) **sorting function** on k elements, $\rho_k \colon \mathbb{R}^k \to \mathbb{R}^k$, is defined by $\xi = \rho_k(z)$ such that ξ is a permutation of z which satisfies*

$$\xi_1 \leq \xi_2 \leq \cdots \leq \xi_k;$$

*(ii) The **weighted zero-selective reduce function** with respect to F_i and k, $r^k_{0,F_i} \colon \mathbb{R}^k \times \mathbb{R}^k \to \mathbb{R}$, is defined by (52), where $1_{\geq 0}(\alpha)$ and $1_{<0}(\alpha)$ are indicator functions, and the weights are uniformly bounded by $0 < \alpha \leq w_l \leq \beta$, $\forall l$.*

[33]Note that if all neighboring values are removed, then $\dot{x}_i(t) = 0$.

$$r_{0,F_i}^k(z,w) = \begin{cases} \sum_{l=1}^{F_i} w_l 1_{\geq 0}(z_l) z_l + \sum_{l=F_i+1}^{k-F_i} w_l z_l + \sum_{l=k-F_i+1}^{k} w_l 1_{\leq 0}(z_l) z_l & k > 2F_i; \\ \sum_{l=1}^{k-F_i} w_l 1_{\geq 0}(z_l) z_l + \sum_{l=F_i+1}^{k} w_l 1_{\leq 0}(z_l) z_l & F < k \leq 2F_i; \\ 0 & k \leq F_i; \end{cases} \tag{52}$$

(iii) The composition of the sorting and weighted zero-selective reduce functions with respect to F_i and k is defined by $\phi_{0,F_i}^k \colon \mathbb{R}^k \times \mathbb{R}^k \to \mathbb{R}$, which is defined for all $z \in \mathbb{R}^k$ and $w \in \mathbb{R}^k$ such that $0 < \alpha \leq w_l \leq \beta$ by

$$\phi_{0,F_i}^k(z,w) = r_{0,F_i}^k(\rho_k(z), w).$$

The update rule of ARC-P2 with either parameter $F \in \mathbb{Z}_{\geq 0}$ or $f \in [0,1]$ determines

$$u_i = f_{i,\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},i)})$$

for each normal agent $i \in \mathcal{N}$ at each point in time $t \in \mathbb{R}_{\geq 0}$ by

$$f_{i,\sigma(t)}(t, x_{\mathcal{N}}, x_{\mathcal{A}}) = \phi_{0,F_i(t)}^{d_i(t)}\left(N_i(t)(x_{(\mathcal{V},i)}(t) - x_i(t)1_n), w_i(t)\right), \tag{53}$$

where $1_n \in \mathbb{R}^n$ is the vector of ones, $F_i(t) \equiv F$ if the parameter is F, $F_i(t) = \lfloor f d_i(t) \rfloor$ if the parameter is f, and $N_i(t) \in \mathbb{R}^{d_i(t) \times n}$ is a time-varying sparse matrix with each row corresponding to a distinct $j \in \mathcal{N}_i^{\text{in}}(t)$ such that each row has a single 1 in the j-th column. Thus, there is a one-to-one correspondence between $j \in \mathcal{N}_i^{\text{in}}(t)$ and rows in $N_i(t)$. The time-varying weight vector

$$w_i(t) = [w_{(i_1(t),i)}(t), w_{(i_2(t),i)}(t), \ldots, w_{(i_{d_i(t)}(t),i)}(t)]^\mathsf{T},$$

satisfies the bound $0 < \alpha \leq w_{(i_j(t),i)} \leq \beta$ for all $j = 1, 2, \ldots, d_i(t)$, where $i_1(t), i_2(t), \ldots, i_{d_i(t)}(t)$ are the node indices of the neighbors of node i in the order determined by the sorting function at time t (i.e., according to (28) such that the weights match the corresponding neighbor). These terms are defined so that (53) is equivalent to (51) for all $t \in \mathbb{R}_{\geq 0}$.

5.2 Existence and Uniqueness of Solutions

The argument for existence and uniqueness of solution of (27) whenever the normal nodes use ARC-P2 is identical to the one given for ARC-P in Section 4.3 on page 113. Therefore, we proceed by providing the results and illustrating the differences from ARC-P.

Lemma 5.2 (Analogue to Lemma 4.14)**.** *Each function $f_{i,\sigma(t)}(x_{\mathcal{N}}, x_{(\mathcal{A},i)})$ for $i \in \mathcal{N}$ defined in (53) (or (51)) with parameter $F \in \mathbb{Z}_{\geq 0}$ (or $f \in [0,1]$), satisfies a global Lipschitz condition in $x = [x_{\mathcal{N}}^{\mathsf{T}}, x_{(\mathcal{A},i)}^{\mathsf{T}}]^{\mathsf{T}}$, $x_{\mathcal{N}}$, and $x_{(\mathcal{A},i)}$.*

Proof. Because the weights are piecewise continuous, it follows that $f_{i,\sigma(t)}(\cdot)$ is piecewise continuous in time t just as it is with ARC-P. All that remains to be shown is that the weighted zero-selective reduce function is Lipschitz. To do this, fix $t \in \mathbb{R}_{\geq 0}$, $F_i(t) = F \in \mathbb{Z}_{\geq 0}$, $d_i(t) = k$, and $w_i(t) = w \in \mathbb{R}^k$. The argument to $\phi_{0,F}^k(\cdot, w)$ is linear and the sorting function is Lipschitz. Fix $z, y \in \mathbb{R}^k$. The key observation is that

$$1_{\geq 0}(z_l)z_l - 1_{\geq 0}(y_l)y_l \leq |z_l - y_l|,$$

for each $l = 1, 2, \ldots, k$, which is trivial to show by checking the four cases depending on the signs of z_l and y_l. Since $0 < \alpha \leq w_l \leq \beta$, it follows that

$$w_l 1_{\geq 0}(z_l)z_l - w_l 1_{\geq 0}(y_l)y_l \leq \beta|z_l - y_l|,$$

Likewise, the inequality holds when the indicator function is $1_{\leq 0}(\cdot)$ instead of $1_{\geq 0}(\cdot)$. Combining this with the triangle inequality, it is straightforward to show using the Manhattan norm that $r_{0,F}^k$ is Lipschitz with Lipschitz constant β. The remaining argument follows that of Lemma 4.14. □

Theorem 5.3. *The function $f_{\sigma(t)}(t, x_{\mathcal{N}}, x_{(\mathcal{A},\mathcal{N})})$ that defines the dynamics of the normal agents, with $f_{i,\sigma(t)}(\cdot)$ defined in (53) (or (51)) with parameter $F \in \mathbb{Z}_{\geq 0}$ (or $f \in [0,1]$) and any threat model such that $x_{(\mathcal{A},\mathcal{N})}$ is piecewise continuous, satisfies a global Lipschitz condition in $x_{\mathcal{N}}$.*

Proof. The proof follows the same argument as Theorem 4.15, using Lemma 5.2 and the fact that the weights are piecewise continuous. □

Next, we present the analogue to Lemma 4.16.

Lemma 5.4. *Consider the normal agent $i \in \mathcal{N}$ using ARC-P2 with parameter $F \in \mathbb{Z}_{\geq 0}$ under the F'-local or F'-total models where $F' \leq F$, or parameter $f \in [0,1]$ under the f'-fraction local model where $f' \leq f$. Define $F_i(t) \equiv F$ or $F_i(t) = \lfloor f d_i(t) \rfloor$ whenever the parameter is F or f, respectively. Then, for each $t \in \mathbb{R}_{\geq 0}$*

$$B(m_{\mathcal{N}}(t) - x_i(t)) \leq f_{i,\sigma(t)}(t, x_{\mathcal{N}}, x_{(\mathcal{A},i)}) \leq B(M_{\mathcal{N}}(t) - x_i(t)),$$

where $B = \beta(n - 1 - \min_{i \in \mathcal{N}}\{F_i(t)\})$.

Proof. If $d_i(t) \leq F_i(t)$, or if $F_i(t) < d_i(t) \leq 2F_i(t)$ and there are at most $F_i(t)$ neighbors with larger and smaller values than $x_i(t)$, then $f_{i,\sigma(t)}(t, x_{\mathcal{N}}, x_{(\mathcal{A},i)}) = 0$, and the result follows. Therefore, assume $d_i(t) > F_i(t)$ and at least one value not equal to $x_i(t)$ is used in the update at time t, say $x_{(j,i)}(t)$. As in the proof of Lemma 4.16, we can show that $m_{\mathcal{N}}(t) \leq x_{(j,i)}(t) \leq M_{\mathcal{N}}(t)$. Since there at most $n-1$ neighbors of i, at least $F_i(t)$ values are removed or equal to $x_i(t)$ (since $d_i(t) > F_i(t)$), and $w_{(j,i)}(t) \leq \beta$ for all $j \in \mathcal{N}_i^{\text{in}}(t)$, it follows that

$$B(m_{\mathcal{N}}(t) - x_i(t)) \leq \sum_{j \in \mathcal{N}_i^{\text{in}}(t) \setminus \mathcal{R}_i(t)} w_{(j,i)}(t)(x_{(j,i)}(t) - x_i(t)) \leq B(M_{\mathcal{N}}(t) - x_i(t)).$$

□

Next, we present the analogue to the invariant set result of Lemma 4.18. Just as Lemma 4.18 ensures the safety condition holds for ARC-P, Lemma 5.5 ensures the safety condition for ARC-P2.

Lemma 5.5 (Invariant Set)**.** *Suppose the normal agents in $\mathcal{N}$ use ARC-P2 with parameter $F \in \mathbb{Z}_{\geq 0}$ under the F'-local or F'-total models where $F' \leq F$, or parameter $f \in [0,1]$ under the f'-fraction local model where $f' \leq f$. Then, the hypercube $\mathcal{H}_0 = [m_{\mathcal{N}}(0), M_{\mathcal{N}}(0)]^N$ defined by*

$$\mathcal{H}_0 = \{y \in \mathbb{R}^N \colon m_{\mathcal{N}}(0) \leq y_i \leq M_{\mathcal{N}}(0),\ i = 1, 2, \ldots, N\},$$

is robustly positively invariant for the system of normal agents. Therefore, under these assumptions, the safety condition of the CTRAC problem is ensured.

Proof. The proof follows the argument of Lemma 4.18, using Lemma 5.4 instead of Lemma 4.16.

□

Remark 5.1. *Lemma 5.4 implies that $M_{\mathcal{N}}(t)$ is a nonincreasing function of time and $m_{\mathcal{N}}(t)$ is a nondecreasing function of time. Therefore, whenever agreement among the normal nodes is achieved, there is a common limit to which the normal values converge.*

It follows from these results and the argument presented in Section 4.3 that ARC-P2 ensures existence and uniqueness of solutions.

Corollary 5.6 (Existence and Uniqueness)**.** *Suppose the adversaries have piecewise continuous trajectories on $[0, \infty)$. Suppose each normal agent uses ARC-P2 with parameter $F \in \mathbb{Z}_{\geq 0}$ under the F'-local or F'-total models where $F' \leq F$, or parameter $f \in [0, 1]$ under the f'-fraction local model where $f' \leq f$. Then, the switched system (27) with component functions given in (53) (or (51)) has a unique Carathéodory solution on $[0, \infty)$ (given the choice of bounded, nonnegative, piecewise continuous weights, piecewise constant switching signal $\sigma(t)$, and $x_{(\mathcal{A},\mathcal{N})}(t)$ for $t \in \mathbb{R}_{\geq 0}$).*

5.3 Robust Network Topologies

Traditionally, network connectivity has been the key metric for studying robustness of distributed algorithms. This is because connectivity formalizes the notion of redundant information flow across the network through independent paths. Due to the fact that each independent path may include multiple intermediate nodes, network connectivity is well-suited for studying resilient distributed algorithms that assume such nonlocal information is available (for example, by explicitly relaying information across multiple hops in the network [130], by 'inverting' the dynamics on the network to recover the needed information [181, 159], or by resiliently encoding information along multiple paths [92]). However, when the nodes in the network use only local information (as in ARC-P), connectivity does not capture the notion of local redundancy needed for such algorithms to succeed. For this, we consider *network robustness* [210].

5.3.1 Network Robustness and Fractional Robustness

Network robustness formalizes the notion of sufficient local redundancy of information flow in the network. Therefore, this property is useful for the study of resilient distributed algorithms that use only local information. To motivate the technical details involved with defining robustness, recall the

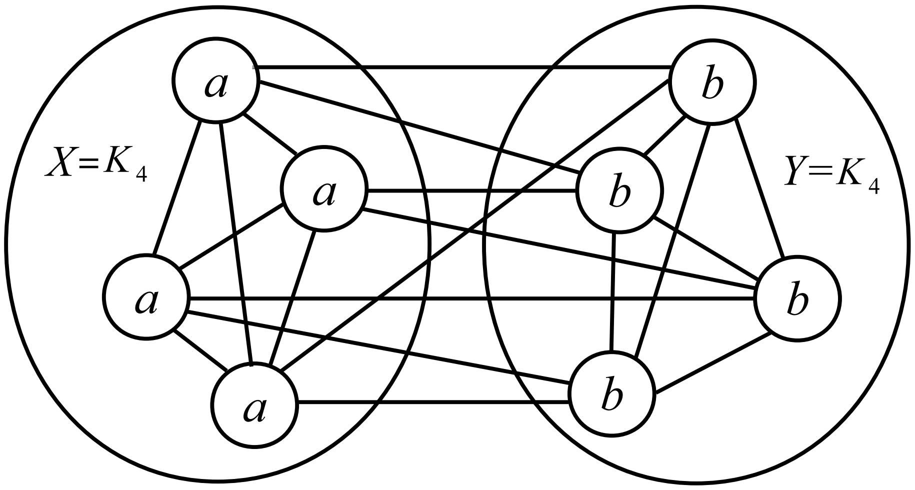


Figure 35: Graph in which ARC-P with parameter $F = 2$ fails to converge.

pathological examples studied in Section 4.6.3. In each of the examples, there are pairs of subsets of nodes with high connectivity within the subsets, but all nodes in each subset have relatively few neighbors outside of their respective subsets.

For concreteness, consider the graph of Figure 35, which illustrates a specific graph in the class of graphs studied in Example 4.5 of Section 4.6.3 (see Figure 27 on page 143 for the illustration of the class of graphs). In Figure 35, there are $n = 8$ nodes divided into two cliques (complete subgraphs), $X = K_4$ and $Y = K_4$, where K_n is the complete graph on n nodes. The graph is highly connected, with $\kappa(\mathcal{G}) = 5$. Each node in X has exactly $F = 2$ neighbors in Y, and vice versa. One can see that if the initial values of nodes in X and Y are $a \in \mathbb{R}$ and $b \in \mathbb{R}$, respectively, with $a < b$, then asymptotic consensus is not achieved whenever ARC-P is used with parameter F, even in the absence of adversaries. This is because each node views the values of its F neighbors from the opposing set as extreme, and removes all of these values in order to maintain safety under the F-total or F-local models. The only remaining values for each node are from its own set, and thus no node ever changes its value.

Now consider the digraph in Figure 36 and assume the nodes begin with initial values a and b, as in the graph of Figure 35. Suppose all nodes are normal and each node uses ARC-P with parameter $F = 2$. This digraph has one more directed edge than the graph of Figure 35 – the directed edge (5,2) – which in this case facilitates consensus. In informal terms, node 2 in X has three neighbors in Y, of which only two are removed from consideration. The additional outside influence causes

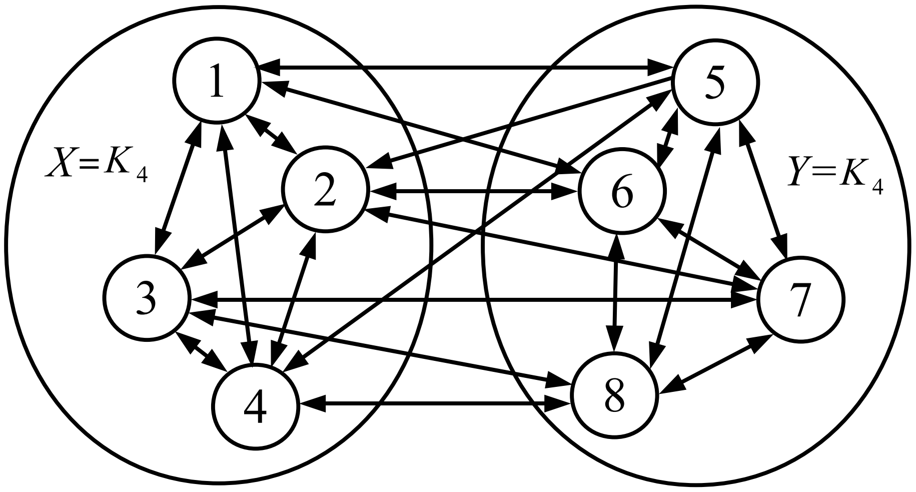


Figure 36: Digraph in which ARC-P with parameter $F = 2$ succeeds (without adversaries).

its value to increase toward b. This trend propagates to the other nodes in X through the influence of node 2, thereby facilitating asymptotic consensus. Therefore, existence of a node with sufficient outside influence is an important property for subsets of nodes to have whenever ARC-P is used (or other distributed algorithms that achieve resilience under the F-total and F-local models). This motivates the definition of r-edge reachable sets [210].[34]

Definition 5.7 (r-edge reachable set)**.** *Given a nontrivial digraph $\mathcal{D}$ and a nonempty subset $\mathcal{S}$ of nodes of $\mathcal{D}$, we say $\mathcal{S}$ is an r**-edge reachable set** if there exists $i \in \mathcal{S}$ such that $|\mathcal{N}_i^{in} \setminus \mathcal{S}| \geq r$, where $r \in \mathbb{Z}_{\geq 0}$.*

A set $\mathcal{S}$ is r-edge reachable if it contains a node that has at least r in-neighbors outside of $\mathcal{S}$. The parameter r quantifies the local redundancy of information from nodes outside $\mathcal{S}$ to *some* node inside $\mathcal{S}$. Intuitively, the r-edge reachability property captures the idea that some node inside the set is influenced by a sufficiently large number of nodes from outside the set. The fact that X is 3-edge reachable in Figure 36, yet only 2-edge reachable in Figure 35, is precisely the difference that enables convergence toward consensus in one, but not the other.

To further illustrate r-edge reachability, consider the sets S_1, S_2, and S_3 in Figure 37. S_1 is 3-edge reachable because node 3 has three neighbors outside of S_1 (nodes $4, 7,$ and 8). S_2 is 5-edge reachable because node 8 has five neighbors outside S_2 (nodes $3, 4, 5, 6,$ and 7). Lastly, S_3 is 5-edge

[34]In [210], edge reachability is defined as reachability. We modify the terminology to avoid confusion with reachability properties defined with respect to directed paths.

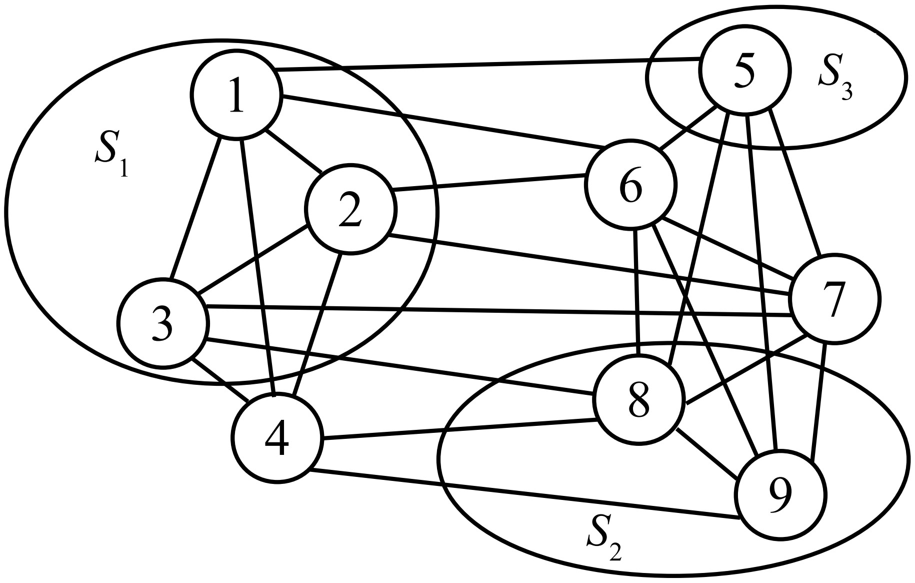


Figure 37: Graph for illustrating reachability properties.

reachable, because node 5 has five neighbors, all of which are outside the singleton S_3.

The fact that X is 3-edge reachable in Figure 36 is not the *only* reason that asymptotic consensus is achieved in this digraph (whenever ARC-P with parameter $F = 2$ is used). In fact, it is the cumulative effect of edge reachability properties of many subsets of nodes in the network that ensures consensus is *always* achievable.[35] If there is any nonempty, disjoint pair of subsets of nodes such that neither set is 3-edge reachable, then we could assign different initial values to nodes in each set (as in Figure 35) and prevent consensus (whenever the consensus algorithm is designed to be resilient under the F-total or F-local models with $F = 2$). Therefore, in order to generalize the notion of sufficient local redundancy of information in the network, we must consider every pair of nonempty and disjoint subsets of nodes and insist that at least one of the subsets has the r-edge reachability property. This is the definition of r-robustness [210].

Definition 5.8 (r-robustness)**.** *A nonempty, nontrivial digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ on n nodes ($n \geq 2$) is r-**robust**, with $r \in \mathbb{Z}_{\geq 0}$, if for every pair of nonempty, disjoint subsets of $\mathcal{V}$, at least one of the subsets is r-edge reachable. By convention, if $\mathcal{D}$ is empty or trivial ($n \leq 1$), then $\mathcal{D}$ is 0-robust. The trivial graph is also 1-robust.*[36]

[35]Consensus is always achievable (for all initial conditions) in the digraph of Figure 36 whenever ARC-P with parameter F is used *and all nodes are normal*. If adversaries are present, then more robustness is needed.

[36]The trivial graph is defined to be both 0-robust and 1-robust for consistency with properties shown to hold for larger digraphs in the sequel.

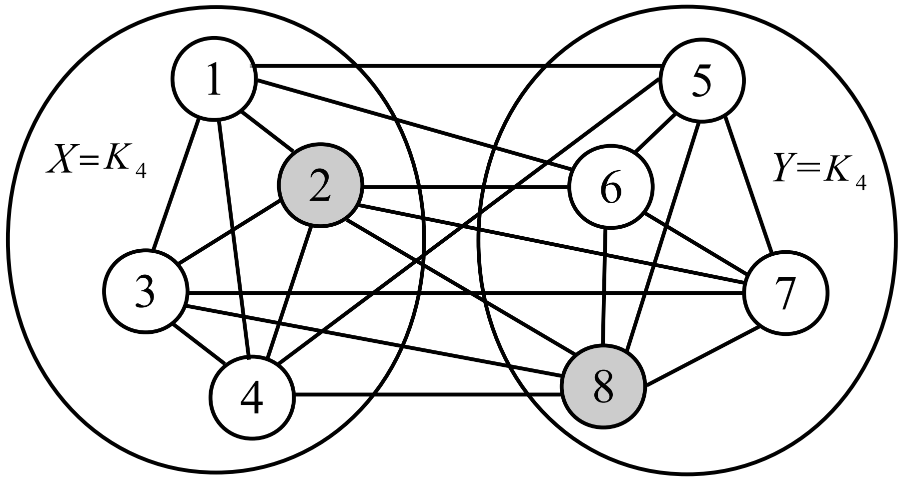


Figure 38: A 3-robust graph in which X and Y are 3-edge reachable. Nodes 2 and 8 are crash nodes.

However, if there are adversary nodes in the network, then the situation becomes more complex. For example, consider the network modeled by the graph in Figure 38. One can verify that the graph is 3-robust by checking every possible pair of disjoint subsets, and confirming that at least one of them is 3-edge reachable. Consider the disjoint subsets X and Y shown in the figure, and note that both of the sets are 3-edge reachable – nodes 2 and 8 each have three neighbors outside of their respective sets. However, no other nodes in those two sets have more than two neighbors outside their set. Therefore, nodes 2 and 8 are the *only* ones with access to sufficient information outside their set. Suppose these two nodes (2 and 8) are crash nodes and the initial values of nodes in X and Y are a and b, respectively. Then, by stubbornly maintaining their initial values, nodes 2 and 8 are able to prevent consensus whenever the normal nodes use ARC-P with parameter $F = 2$. One way to remedy this is to require the whole network to be more robust. Another way is to introduce another form of information redundancy by specifying a *minimum number of nodes* that are sufficiently influenced from outside their set. In order to capture this intuition, we define the following concept.

Definition 5.9 ((r,s)-edge reachable set)**.** *Given a nontrivial digraph $\mathcal{D}$ and a nonempty subset of nodes $\mathcal{S}$, we say that $\mathcal{S}$ is an (r,s)**-edge reachable set** if there are at least s nodes in $\mathcal{S}$ with at least r in-neighbors outside of $\mathcal{S}$, where $r, s \in \mathbb{Z}_{\geq 0}$; i.e., given $\mathcal{X}_{\mathcal{S}}^{r} = \{i \in \mathcal{S} \colon |\mathcal{N}_i^{in} \setminus \mathcal{S}| \geq r\}$, then $|\mathcal{X}_{\mathcal{S}}^{r}| \geq s$.*

A general illustration of an (r,s)-edge reachable set of nodes is shown in Figure 39. The pa-

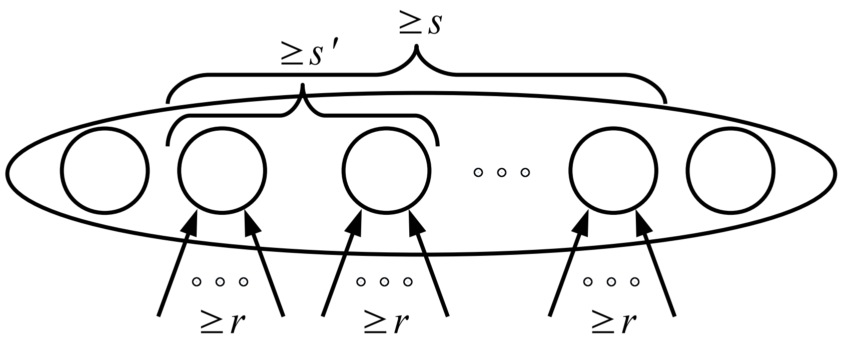


Figure 39: Illustration of an (r, s)-edge reachable set of nodes.

rameter s in the definition of (r, s)-edge reachability quantifies a lower bound on the number of nodes in the set with at least r in-neighbors outside $\mathcal{S}$. Observe that, in general, a set is (r, s')-edge reachable, for $s' \leq s$, if it is (r, s)-edge reachable. At one extreme, whenever there are no nodes in $\mathcal{S}$ with at least r in-neighbors outside of $\mathcal{S}$, then $\mathcal{S}$ is only $(r, 0)$-edge reachable. At the other extreme, $\mathcal{S}$ can be at most $(r, |\mathcal{S}|)$-edge reachable. Also note that r-edge reachability is equivalent to $(r, 1)$-edge reachability. Hence, (r, s)-edge reachability provides finer granularity than r-edge reachability by specifying not only the existence of a node with sufficient outside influence, but a lower bound on the number of such nodes.

For a more specific example, consider again the graph in Figure 37 (on page 161). Now we may characterize S_1 as $(3, 3)$-edge reachable (all nodes have three neighbors outside S_1). S_2 is both $(5, 1)$-edge reachable (due to node 8) and $(4, 2)$-edge reachable. Depending on the situation (in terms of the scope of threat model and algorithm used), one characterization may be preferred over the other. Again, S_3 is $(5, 1)$-edge reachable.

The notion of (r, s)-edge reachability characterizes the class of sets that are r-edge reachable with greater specificity on the number of nodes that have sufficient outside influence. This additional specificity is useful for defining (r, s)*-robustness*, which is analogous to r-robustness, but has greater specificity on the number of nodes *in the pair of subsets* that have enough neighbors outside their respective sets. Because s is a lower bound on the number of nodes in set $\mathcal{S}$ with at least r in-neighbors outside, the maximal s such that set $\mathcal{S}$ is (r, s)-edge reachable is *the exact* number of nodes with at least r in-neighbors outside $\mathcal{S}$. In the definition of (r, s)-robustness, we are interested in the exact number of such nodes.

Definition 5.10 ((r,s)-robustness)**.** *A nonempty, nontrivial digraph $\mathcal{D}=(\mathcal{V},\mathcal{E})$ on n nodes ($n \geq 2$) is (r,s)-**robust**, for nonnegative integers $r \in \mathbb{Z}_{\geq 0}$, $1 \leq s \leq n$, if for every pair of nonempty, disjoint subsets $\mathcal{S}_1$ and $\mathcal{S}_2$ of $\mathcal{V}$ at least one of the following holds (recall $\mathcal{X}^r_{\mathcal{S}_k} = \{i \in \mathcal{S}_k \colon |\mathcal{N}^{in}_i \setminus \mathcal{S}_k| \geq r\}$ for $k \in \{1,2\}$):*

(i) $|\mathcal{X}^r_{\mathcal{S}_1}| = |\mathcal{S}_1|$*;*

(ii) $|\mathcal{X}^r_{\mathcal{S}_2}| = |\mathcal{S}_2|$*;*

(iii) $|\mathcal{X}^r_{\mathcal{S}_1}| + |\mathcal{X}^r_{\mathcal{S}_2}| \geq s$*.*

By convention, if $\mathcal{D}$ is empty or trivial ($n \leq 1$), then $\mathcal{D}$ is (0,1)-robust. If $\mathcal{D}$ is trivial, $\mathcal{D}$ is also (1,1)-robust.[37]

A few remarks are in order with respect to this definition. The properties required of each nonempty, disjoint pair of subsets of nodes are codified by conditions (i)-(iii). Condition (iii) captures the main idea behind the definition. In an (r,s)-robust network, it is generally required that there are at least s nodes in the pair of nonempty, disjoint subsets $\mathcal{S}_1, \mathcal{S}_2 \subset \mathcal{V}$ that have at least r in-neighbors outside their respective sets. However, there are cases in which this condition cannot be satisfied. A simple example is when $|\mathcal{S}_1 \cup \mathcal{S}_2| < s$ (i.e., whenever the sets are small relative to s). Conditions (i) and (ii) are included to handle these special cases. Whenever condition (i) or (ii) is satisfied, it means that *all* nodes in the respective subset have at least r in-neighbors from outside. It follows that for large subsets relative to s, conditions (i) and (ii) are conservative, and it is sufficient for condition (iii) to hold. The downside to introducing conditions (i) and (ii), which do not depend on s, is that for some values of r there are digraphs such that *every* pair of nonempty, disjoint subsets satisfies either condition (i) or (ii). In these examples, there is no meaning behind the parameter s. For this reason, s is restricted to lie between 1 and n.[38]

An example of a graph that is $(2,s)$-robust, for all $1 \leq s \leq n$ is the graph of Figure 35. Indeed, it satisfies conditions (i) or (ii) for all pairs of nonempty, disjoint subsets of nodes. Therefore,

[37] The trivial graph is defined to be both (0,1)-robust and (1,1)-robust for consistency with properties shown to hold for larger digraphs in the sequel.

[38] Note that $s = 0$ is *not* allowed in (r,s)-robustness because in that case any digraph on $n \geq 2$ nodes satisfies the definition for any $r \in \mathbb{Z}_{\geq 0}$, which subverts the interpretation of the parameter r. At the other extreme, the maximal meaningful value of s is $s = n$ since condition (iii) can *never* be satisfied with $s > n$.

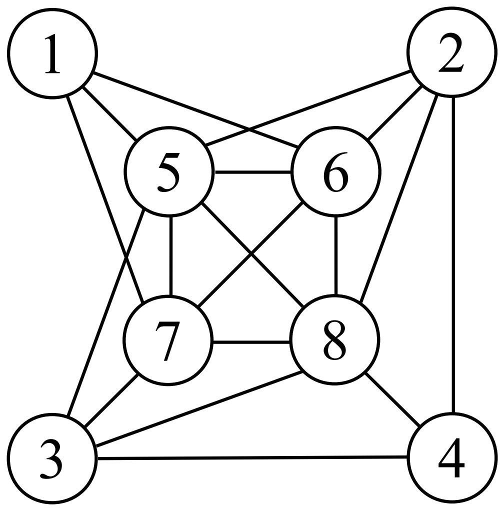


Figure 40: A 3-robust graph that is *not* (3,2)-robust.

condition (iii) in Definition 5.10 is *never* needed, and the definition is satisfied with $r = 2$ for all valid values of s.

The reader may then wonder whether the introduction of conditions (i) and (ii) in the definition of (r, s)-robustness prevents a meaningful comparison with r-robustness. On the contrary, if $s = 1$, then conditions $(i) - (iii)$ collapse to (iii) (since $\mathcal{S}_1$ and $\mathcal{S}_2$ are nonempty). In this case, condition (iii) holds if and only if one of the sets is r-edge reachable. Therefore, $(r, 1)$-robustness is equivalent to r-robustness. In more general terms, a digraph is (r, s')-robust and (r', s)-robust if it is (r, s)-robust, for $s' \leq s$ and $r' \leq r$ (see Property 5.13 in the sequel). Therefore, a digraph is r-robust whenever it is (r, s)-robust. The converse, however, is not true. Consider the graph in Figure 40. This graph is 3-robust, but is not $(3, 2)$-robust. For example, let $\mathcal{S}_1 = \{1, 3, 5, 6, 7\}$ and $\mathcal{S}_2 = \{2, 4\}$. Then, only node 2 has at least three nodes outside of its set, so all of the conditions $(i) - (iii)$ fail to hold for this pair.

Consider again the examples of Figures 35 and 40. The graph of Figure 35 is $(2, 8)$-robust, but is not 3-robust, which can be shown by selecting $\mathcal{S}_1 = X$ and $\mathcal{S}_2 = Y$. On the other hand, the graph in Figure 40 is 3-robust, but is not (2,5)-robust (e.g., let $\mathcal{S}_1 = \{1, 5, 6\}$ and $\mathcal{S}_2 = \{2, 3, 4\}$; then only nodes 2, 3, 5, and 6 have two or more neighbors outside their set). The question then arises, which graph is more robust? More generally, how does one compare relative robustness? Clearly, if digraph $\mathcal{D}_1$ is (r_1, s_1)-robust and digraph $\mathcal{D}_2$ is (r_2, s_2)-robust with maximal r_k and s_k for $k \in \{1, 2\}$, where $r_1 > r_2$ and $s_1 > s_2$, then one can conclude that $\mathcal{D}_1$ is more robust than

$\mathcal{D}_2$. We adopt the convention that r-robustness (with maximal r) takes precedence in the total order that determines relative robustness. The maximal s in (r, s)-robustness is then used for ordering the robustness of two r-robust digraphs with the same value of r. The reason for adopting this convention is twofold. As exemplified by the graph in Figure 35, there are digraphs in which the parameter s may take on any value, and in such cases s loses its precise meaning. On the other hand, there are properties shown in the following section that show the utility of the parameter r in characterizing the robustness of a given digraph. For example, there is an upper bound on the value of r for any digraph on n nodes (Property 5.19). Moreover, if a digraph is sufficiently robust, in terms of r-robustness, then this implies a minimum value of $s > 1$ for smaller values $r' < r$ in terms of (r', s)-robustness (Property 5.21). Property 5.20 indicates that parameter r may be traded for greater values of the parameter s. The converse is obviously not true, as demonstrated by the graph of Figure 35 (recall it is $(2, 8)$-robust, but is not 3-robust). Therefore, r is the more essential parameter in characterizing the robustness of a network.[39]

Given this total order on the robustness of networks, the graph in Figure 40, which is 3-robust, is more robust than the graph of Figure 35. Yet, the graph of Figure 40 is only 3-connected, whereas the graph of Figure 35 is 5-connected. Hence, it is possible that a digraph with *less* connectivity is *more* robust. In the sequel, we further explore the relationship between connectivity and robustness.

Fractional Edge Reachability and Robustness

As demonstrated by the motivating examples at the beginning of this section, the edge reachability and robustness properties described so far are useful to describe resilience properties under the F-local and F-total models. With these scope of threat assumptions, there is a fixed upper bound on the number of compromised nodes in any normal node's neighborhood. To maintain safety under these threat models requires the normal nodes to be skeptical of values different from their own. To facilitate convergence, enough neighbors with different values are needed to assure the node that it is safe to change its value. This requirement is codified by network robustness. However, for the f-fraction local model, there is no absolute bound on the number of neighbors that may be compromised. Rather, it stipulates a bound on the *fraction* of neighbors that may be compromised. Hence, for the f-fraction local model, we require a fractional notion of edge reachability and robustness.

[39]While Properties 5.20 and 5.21 do not fully justify this total order, it is convenient nonetheless to impose a total order.

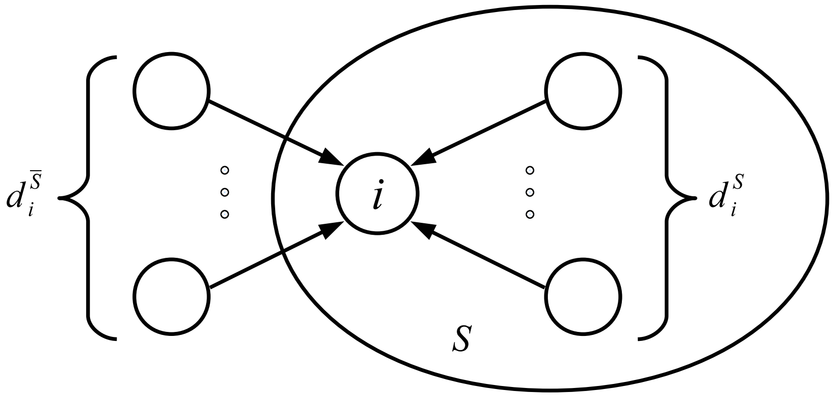


Figure 41: Illustration of a p-fraction edge reachable set of nodes.

First, we define a p-fraction edge reachable set.

Definition 5.11 (p-fraction edge reachable set)**.** *Given a nonempty digraph $\mathcal{D}$ and a nonempty subset $\mathcal{S}$ of nodes of $\mathcal{D}$, we say $\mathcal{S}$ is a p-**fraction edge reachable set** if there exists $i \in \mathcal{S}$ such that $|\mathcal{N}_i^{in}| > 0$ and $|\mathcal{N}_i^{in} \setminus \mathcal{S}| \geq \lceil p|\mathcal{N}_i^{in}|\rceil$, where $0 \leq p \leq 1$. If $|\mathcal{N}_i^{in}| = 0$ or $|\mathcal{N}_i^{in} \setminus \mathcal{S}| = 0$ for all $i \in \mathcal{S}$, then $\mathcal{S}$ is 0-fraction edge reachable.*

A set $\mathcal{S}$ is p-fraction edge reachable if it contains a non-isolated node i (i.e., $d_i > 0$) that has at least $\lceil pd_i \rceil$ neighbors outside of $\mathcal{S}$. The parameter p quantifies the ratio of influence from neighbors outside $\mathcal{S}$ to neighbors inside $\mathcal{S}$ for *at least* one node inside $\mathcal{S}$. To illustrate this definition, consider Figure 41 where node $i \in \mathcal{S}$ is a node with at least $\lceil pd_i \rceil$ neighbors outside of $\mathcal{S}$. In the figure, $d_i^{\mathcal{S}}$ and $d_i^{\bar{\mathcal{S}}}$ denote the neighbors of i inside and outside $\mathcal{S}$, respectively. Given this notation, node i satisfies the inequality

$$\frac{d_i^{\bar{\mathcal{S}}}}{d_i} \geq p.$$

Note that if no such $i \in \mathcal{S}$ (with $d_i > 0$) satisfies this inequality, then $\mathcal{S}$ is *not* p-fraction edge reachable.

To further illustrate p-fraction edge reachability, consider once again the sets S_1, S_2, and S_3 in Figure 37 (on page 161). Each node in S_1 has three-fifths of its neighbors outside S_1; so S_1 is $\frac{3}{5}$-fraction edge reachable. Node 8 has five-sixths of its neighbors outside of S_2, so S_2 is $\frac{5}{6}$-fraction edge reachable. Finally, S_3 is a non-isolated singleton; therefore, S_3 is 1-fraction edge reachable.

The p-fraction edge reachability property is defined with respect to a specific set $\mathcal{S} \subseteq \mathcal{V}$. As was the case with r-robustness and (r, s)-robustness, to demonstrate fractional robustness of the network requires checking fractional edge reachability properties of every nonempty, disjoint pair of subsets of nodes. In this case, at least one of the subsets in each pair must be p-fraction edge reachable. This is the definition of p-fraction robustness.

Definition 5.12 (p-fraction robustness)**.** *A nonempty, nontrivial digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ on n nodes ($n \geq 2$) is p-**fraction robust**, with $0 \leq p \leq 1$, if for every pair of nonempty, disjoint subsets of $\mathcal{V}$, at least one of the subsets is p-fraction edge reachable. If $\mathcal{D}$ is empty or trivial ($n \leq 1$), then $\mathcal{D}$ is 0-fraction robust.*

5.3.2 Properties of Robust Networks

In this section, we demonstrate some properties of robust and fractional robust digraphs, with greater focus on robustness. In particular, we explore relationships between robustness and other graph theoretic metrics such as connectivity and minimum degree. We also consider the implications of how directed edge removal and other modifications to a digraph relate to robustness. We begin by establishing an inheritance property of (r, s)-robust and p-fraction robust digraphs.

Property 5.13. *Every (r, s)-robust digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ is also (r', s')-robust when $0 \leq r' \leq r$, $1 \leq s' \leq s$. Every p-fraction robust digraph is also p'-fraction robust when $0 \leq p' \leq p$.*

Proof. If $\mathcal{D}$ is empty or trivial, there is nothing to prove, so assume $\mathcal{D}$ is nonempty and nontrivial. Fix any nonempty, disjoint pair $\mathcal{S}_1, \mathcal{S}_2 \subset \mathcal{V}$. If $\mathcal{D}$ is (r, s)-robust, then at least one of the three conditions (i)–(iii) of Definition 5.10 holds. Observe that $|\mathcal{X}_{\mathcal{S}_k}^{r'}| \geq |\mathcal{X}_{\mathcal{S}_k}^{r}|$ for $k = 1, 2$. Hence if (i) or (ii) hold, then $|\mathcal{X}_{\mathcal{S}_k}^{r'}| \geq |\mathcal{X}_{\mathcal{S}_k}^{r}| = |\mathcal{S}_k| \geq |\mathcal{X}_{\mathcal{S}_k}^{r'}|$, which implies $|\mathcal{X}_{\mathcal{S}_k}^{r'}| = |\mathcal{S}_k|$. If (iii) holds, then

$$|\mathcal{X}_{\mathcal{S}_1}^{r'}| + |\mathcal{X}_{\mathcal{S}_2}^{r'}| \geq |\mathcal{X}_{\mathcal{S}_1}^{r}| + |\mathcal{X}_{\mathcal{S}_2}^{r}| \geq s \geq s'.$$

Thus, any pair of nonempty, disjoint subsets of nodes in $\mathcal{D}$ satisfy Definition 5.10 with r and s replaced by r' and s'. Therefore, $\mathcal{D}$ is (r', s')-robust.

Similarly, if $\mathcal{D}$ is p-fraction robust, then at least one of $\mathcal{S}_1$ or $\mathcal{S}_2$ is p-fraction edge reachable. Whichever set is p-fraction edge reachable is also p'-fraction edge reachable, and the result follows. □

The next two results demonstrate the utility of robustness in analyzing linear consensus protocols. Recall that when there are no misbehaving nodes, the Linear Consensus Protocol given in (50) achieves consensus in time-invariant networks if and only if the network contains a rooted out-branching. The following result shows that 1-robustness is equivalent to the existence of a rooted out-branching. This result is proved in [210]. We include here a similar result for p-fraction robust digraphs. The proof of this property follows the same argument used in [210].

Property 5.14 ([210])**.** *A digraph $\mathcal{D}$ is 1-robust if and only if $\mathcal{D}$ contains a rooted-out branching.*

Property 5.15. *A digraph $\mathcal{D}$ is p-fraction robust for some $0 < p \leq 1$ if and only if $\mathcal{D}$ contains a rooted-out branching.*

Proof. If $\mathcal{D}$ is p-fraction robust for some $0 < p \leq 1$, we prove that $\mathcal{D}$ has a rooted out-branching by contradiction. Assume that $\mathcal{D}$ does not have a rooted out-branching. Decompose $\mathcal{D}$ into its strongly connected components, and note that since $\mathcal{D}$ does not have a rooted out-branching, there must be at least two components that have no incoming edges from any other components. However, this contradicts the assumption that $\mathcal{D}$ is p-fraction robust for some $p > 0$ (at least one of the two subsets must have a node with at least one neighbor outside its set). Hence, there exists a rooted out-branching.

Next, assume $\mathcal{D}$ contains a rooted out-branching, but is at most 0-fraction robust. Then we can find two subsets of nodes that do not have neighbors from outside, which contradicts the assumption that $\mathcal{D}$ contains a rooted out-branching. $\square$

Remark 5.2. *The proof of Property 5.14 given in [210] is a more direct version of the proof of Theorem 5 in [144].*

The next two results demonstrate how modifications to the digraph affect the robustness of the digraph. The first result deals with the amount of robustness guaranteed upon removal of directed edges from nodes in the network. Conversely, the second result considers the addition of directed edges. The first result is proven for r-robust digraphs in [210]. Here, we extend the result to (r, s)-robust and p-fraction robust digraphs.

Property 5.16 (Directed Edge Removal)**.** *Given an (r, s)-robust digraph $\mathcal{D}$, let $\mathcal{D}'$ be the digraph produced by removing up to k incoming edges from each node in $\mathcal{D}$, where $0 \leq k < r$. Then $\mathcal{D}'$ is*

$(r-k, s)$-robust. Similarly, suppose $\mathcal{D}$ is p-fraction robust and $\mathcal{D}'$ is produced by removing up to a q-fraction of incoming edges from each node in $\mathcal{D}$, with $0 \leq q < p \leq 1$. Then $\mathcal{D}'$ is $(p-q)$-fraction robust.

Proof. From the definition of (r, s)-edge reachable sets, we know that if a set is (r, s)-edge reachable, then by removing up to k incoming edges from each node in $\mathcal{D}$, where $0 \leq k < r$, the set is $(r-k, s)$-edge reachable. Then, the result follows from the definition of (r, s)-robustness. Similarly, if at most a q-fraction of incoming edges are removed from each node of a p-fraction edge reachable set, with $0 \leq q < p \leq 1$, then the set is still $(p-q)$-fraction edge reachable. The result then follows from the definition of p-fraction robustness. □

The following result formalizes the intuition that adding links to a robust network can never reduce the robustness of the network. The same does not hold true for fractional robustness.

Property 5.17 (Monotonicity). *Suppose $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ is an (r, s)-robust spanning subdigraph of $\mathcal{D}' = (\mathcal{V}, \mathcal{E}')$, where $\mathcal{E}' = \mathcal{E} \cup \mathcal{E}''$ and $|\mathcal{E}''| \geq 0$. Then $\mathcal{D}'$ is (r, s)-robust.*

Proof. Suppose $\mathcal{D}'$ is not (r, s)-robust. Then there exists a pair of nonempty, disjoint subsets $\mathcal{S}_1, \mathcal{S}_2 \subset \mathcal{V}$ such that all of the conditions (i)-(iii) in Definition 5.10 fail to hold with r and s. By removing directed edges in $\mathcal{E}''$, the number of nodes in $\mathcal{X}^r_{\mathcal{S}_1}$ and $\mathcal{X}^r_{\mathcal{S}_2}$ can only decrease, and therefore none of conditions (i)-(iii) hold for the pair $\mathcal{S}_1, \mathcal{S}_2$ in $\mathcal{D}$. Hence, $\mathcal{D}$ is not (r, s)-robust, which is a contradiction. □

Note that the monotonicity property above does not generally hold for fractional robustness. To construct a counterexample, we first show that all cycle graphs on $n > 3$ nodes are $\frac{1}{2}$-fraction robust. Our counterexample will be constructed from a cycle graph.

Property 5.18 (Cycle graphs are $\frac{1}{2}$-fraction robust). *Let $C_n = (\mathcal{V}, \mathcal{E})$ denote the cycle graph on n nodes. Then, C_n is $\frac{1}{2}$-fraction robust whenever $n > 3$ and 1-fraction robust for $n \in \{2, 3\}$.*

Proof. First, observe that C_2 and C_3 are 1-fraction robust because at least one of the subsets in any pair of nonempty, disjoint subsets is a non-isolated singleton. To show that C_n is $\frac{1}{2}$-fraction robust for $n > 3$, note that each node in C_n has degree 2. Given any nonempty subset of nodes $\mathcal{S} \subset \mathcal{V}$, if there is some node $i \in \mathcal{S}$ such that one of the neighbors of the node is inside its set, but the other

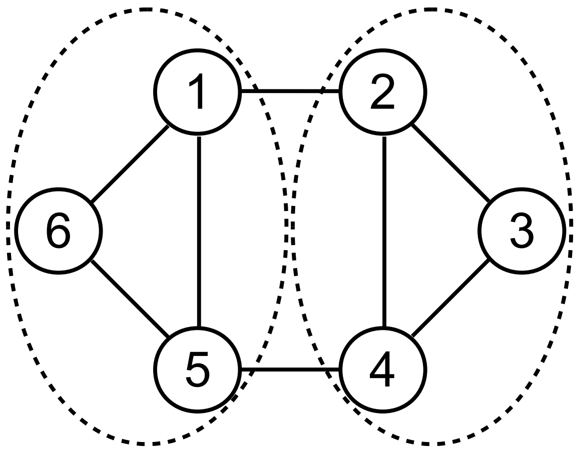


Figure 42: Counterexample to monotonicity property for fractional robustness.

is outside, then $\mathcal{S}$ is $\frac{1}{2}$-fraction edge reachable. On the other hand, if all neighbors are outside the set, then the set is 1-fraction edge reachable. Note that C_n cannot be (at most) 0-fraction robust because it is connected. Hence, there are two possibilities for C_n; it is either 1- or $\frac{1}{2}$–fraction robust (since every proper, nonempty subset of nodes in C_n must be either 1- or $\frac{1}{2}$-fraction edge reachable). Existence of a single nonempty, disjoint pair $\mathcal{S}_1, \mathcal{S}_2 \subset \mathcal{V}$ such that both $\mathcal{S}_1$ and $\mathcal{S}_2$ are $\frac{1}{2}$-fraction edge reachable then implies the result. Let $\mathcal{S}_1$ be constructed by taking some node i and one of its neighbors. From the remaining nodes construct $\mathcal{S}_2$ by selecting some node $j \in \mathcal{V} \setminus \mathcal{S}_1$ and one of j's neighbors in $\mathcal{V} \setminus \mathcal{S}_1$. Since $n \geq 4$, this construction is possible. It follows that both nodes in each set have one neighbor inside and one neighbor outside their set. Thus, both $\mathcal{S}_1$ and $\mathcal{S}_2$ are $\frac{1}{2}$-fraction edge reachable. $\square$

Given Property 5.18, we obtain a counterexample to the monotonicity property as follows. Construct $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$ from C_6 as shown in Figure 42. Each subset of $\mathcal{V}'$ shown in Figure 42 is $\frac{1}{3}$-fraction edge reachable (all nodes with neighbors outside their set have only one-third of their neighbors outside). Therefore, starting with C_6 and adding edges $\{1, 5\}$ and $\{2, 4\}$, actually reduces the fractional robustness of the graph.

Next, we look at the maximum amount of robustness one can expect from a network with n nodes. As one would expected, the complete digraph K_n is the most robust topology on n nodes.

Property 5.19 (Maximum robustness)**.** *No digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ on n nodes is $(\lceil n/2 \rceil + 1)$-robust. Conversely, the complete digraph, denoted $K_n = (\mathcal{V}, \mathcal{E}_{K_n})$, with $\mathcal{E}_{K_n} = \{(i, j) \in \mathcal{V} \times \mathcal{V} \colon i \neq j\}$,* *is $(\lceil n/2\rceil, s)$-robust, for $1 \le s \le n$. Furthermore, whenever $n > 1$ is odd, K_n is the only digraph on n nodes that is $(\lceil n/2\rceil, s)$-robust with $s \ge \lfloor n/2\rfloor$.*

Proof. Assume $\mathcal{D}$ is nonempty and nontrivial (otherwise, the result holds by definition). Pick $\mathcal{S}_1$ and $\mathcal{S}_2$ by taking any bipartition of $\mathcal{V}$ such that $|\mathcal{S}_1| = \lceil n/2\rceil$ and $|\mathcal{S}_2| = \lfloor n/2\rfloor$. Neither $\mathcal{S}_1$ nor $\mathcal{S}_2$ have $\lceil n/2\rceil + 1$ nodes; therefore, neither one is $(\lceil n/2\rceil+1)$-edge reachable. Hence, $\mathcal{D}$ is not $(\lceil n/2\rceil+1)$-robust. Now suppose $\mathcal{D} = K_n$. For any nonempty, disjoint $\mathcal{S}_1, \mathcal{S}_2 \subset \mathcal{V}$, $|\mathcal{V} \setminus \mathcal{S}_i| \ge \lceil n/2\rceil$ holds for at least one of $i \in \{1,2\}$. For whichever i this holds, $|\mathcal{X}_{\mathcal{S}_i}^{\lceil n/2\rceil}| = |\mathcal{S}_i|$, so that K_n is $(\lceil n/2\rceil, s)$-robust, for $1 \le s \le n$. For the last statement, we show that whenever $n > 1$ is odd, removing any directed edge from K_n causes the resulting digraph to lose $(\lceil n/2\rceil, \lfloor n/2\rfloor)$-robustness. Suppose $e = (i,j)$ is the directed edge removed from $\mathcal{E}_{K_n}$ to form $\mathcal{D}'' = (\mathcal{V}, \mathcal{E}'')$, with $\mathcal{E}'' = \mathcal{E}_{K_n} \setminus \{e\}$. Choose $\mathcal{S}_1$ and $\mathcal{S}_2$ by taking any bipartition of $\mathcal{V}$ in $\mathcal{D}''$ such that $|\mathcal{S}_1| = \lceil n/2\rceil$, $|\mathcal{S}_2| = \lfloor n/2\rfloor$, $i \in \mathcal{S}_1$, and $j \in \mathcal{S}_2$. Then, $|\mathcal{X}_{\mathcal{S}_1}^{\lceil n/2\rceil}| = 0$ and $|\mathcal{X}_{\mathcal{S}_2}^{\lceil n/2\rceil}| = \lfloor n/2\rfloor - 1 < |\mathcal{S}_2|$. Therefore, $\mathcal{D}''$ is not $(\lceil n/2\rceil, s)$-robust for $s \ge \lfloor n/2\rfloor$. This is sufficient to prove the statement because of the monotonicity result of Property 5.17, combined with the fact that any spanning subdigraph of K_n, $\mathcal{D}' = (\mathcal{V}, \mathcal{E}') \subset K_n$, can be obtained from a directed edge removal process starting with some directed edge $e = (i,j) \notin \mathcal{E}'$. $\square$

Now that we have shown the limit on the amount of robustness one can expect from a digraph on n nodes, we next explore the relationship between parameters r and s in (r,s)-robustness. The following property shows that robustness in terms of parameter r can be traded for larger values of parameter s.

Property 5.20 ((r,s)-robust implies $(r-1,s+1)$-robust)**.** *Suppose $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ is an (r,s)-robust digraph on n nodes with $r \in \mathbb{N}$, $1 \le s \le n$. Then $\mathcal{D}$ is $(r-1, s+1)$-robust.*

Proof. Fix any disjoint and nonempty subsets of nodes $\mathcal{S}_1, \mathcal{S}_2 \subset \mathcal{V}$. If $|\mathcal{X}_{\mathcal{S}_k}^{r}| = |\mathcal{S}_k|$ or $|\mathcal{X}_{\mathcal{S}_k}^{r-1}| = |\mathcal{S}_k|$ for $k \in \{1,2\}$, then the pair satisfies at least one of the conditions of $(r-1,s+1)$-robustness. So, assume $|\mathcal{X}_{\mathcal{S}_k}^{r}| < |\mathcal{S}_k|$ and $|\mathcal{X}_{\mathcal{S}_k}^{r-1}| < |\mathcal{S}_k|$ for $k \in \{1,2\}$. Then, we must have

$$|\mathcal{X}_{\mathcal{S}_1}^{r}| + |\mathcal{X}_{\mathcal{S}_2}^{r}| \ge s.$$

Since $s \ge 1$, at least one of $\mathcal{X}_{\mathcal{S}_1}^{r}$ or $\mathcal{X}_{\mathcal{S}_2}^{r}$ is nonempty. Without loss of generality, assume $\mathcal{X}_{\mathcal{S}_1}^{r}$ is nonempty, and fix $i \in \mathcal{X}_{\mathcal{S}_1}^{r}$. Define $\mathcal{S}_1' = \mathcal{S}_1 \setminus \{i\}$ and $\mathcal{S}_2' = \mathcal{S}_2$. Since $|\mathcal{X}_{\mathcal{S}_1}^{r}| < |\mathcal{S}_1|$, it follows that $\mathcal{S}_1'$ and $\mathcal{S}_2'$ are disjoint and nonempty. Observe that if $j \in \mathcal{X}_{\mathcal{S}_1'}^{r}$, then $j \in \mathcal{X}_{\mathcal{S}_1}^{r-1}$ (because node i is the only difference between $\mathcal{S}_1'$ and $\mathcal{S}_1$, and so even if $i \in \mathcal{N}_j^{\text{in}}$, j still has at least $r-1$ *other* in-neighbors from outside). Therefore, $|\mathcal{X}_{\mathcal{S}_1'}^{r}| < |\mathcal{S}_1'|$ (otherwise $|\mathcal{X}_{\mathcal{S}_1}^{r-1}| = |\mathcal{S}_1|$) and $|\mathcal{X}_{\mathcal{S}_2'}^{r}| < |\mathcal{S}_2'|$ (since $\mathcal{S}_2' = \mathcal{S}_2$). It follows that

$$|\mathcal{X}_{\mathcal{S}_1'}^{r}| + |\mathcal{X}_{\mathcal{S}_2'}^{r}| \geq s.$$

By including $i \in \mathcal{X}_{\mathcal{S}_1}^{r} \subseteq \mathcal{X}_{\mathcal{S}_1}^{r-1}$ back into the set (i.e., by considering $\mathcal{S}_1$ instead of $\mathcal{S}_1'$), we have

$$|\mathcal{X}_{\mathcal{S}_1}^{r-1}| + |\mathcal{X}_{\mathcal{S}_2}^{r-1}| \geq s+1.$$

□

Property 5.20 enables the following property that relates r-robustness to (r', s)-robustness whenever $r' \leq r$ and $s = r - r' + 1$.

Property 5.21 ($(r+s-1)$-robust implies (r,s)-robust)**.** *If $\mathcal{D}$ is $(r+s-1)$-robust with $r \in \mathbb{Z}_{\geq 0}$, $s \in \mathbb{N}$, and $1 \leq r+s-1 \leq \lceil n/2 \rceil$, then $\mathcal{D}$ is (r,s)-robust.*

Proof. The result follows from repeated application (exactly s times) of Property 5.20, since $(r+s-1)$-robust is equivalent to $(r+s-1, 1)$-robust. □

The remaining properties in this section examine the relationship between robustness and other graph theoretic metrics. In particular, we study degree properties and connectivity. The following result indicates some degree properties of robust digraphs.

Property 5.22 (Robustness Implies Degree Properties)**.** *Given an (r,s)-robust digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$, with $0 \leq r \leq \lceil n/2 \rceil$ and $1 \leq s \leq n$, the minimum in-degree of $\mathcal{D}$, $\delta^{in}(\mathcal{D})$, is at least*

$$\delta^{in}(\mathcal{D}) \geq \begin{cases} r+s-1 & \text{if } s < r; \\ 2r-2 & \text{if } s \geq r. \end{cases}$$

Furthermore, whenever $n > 1$ and $s \geq r$, at least one of the following holds for each $j \in \mathcal{V}$: (a) $d_j^{in} \geq 2r-1$; (b) $\exists k \in \mathcal{N}_j^{out}$ with $k \notin \mathcal{N}_j^{in}$; or (c) $d_j^{out} \geq r$.

Proof. For the first statement, whenever $r \in \{0, 1\}$, there is nothing to prove. Also, if $n \leq 2$, then $r \leq 1$. Therefore, assume $n \geq 3$ and $2 \leq r \leq \lceil n/2 \rceil$. Fix $j \in \mathcal{V}$. First, let $\mathcal{S}_1 = \{j\}$ and $\mathcal{S}_2 = \mathcal{V} \setminus \mathcal{S}_1$. Then, $|\mathcal{X}^r_{\mathcal{S}_2}| = 0$ so that $|\mathcal{X}^r_{\mathcal{S}_1}| = |\mathcal{S}_1|$. This proves $|\mathcal{N}^{\text{in}}_j| \geq r$. Next, whenever $s < r$, form $\mathcal{S}_1$ by choosing $s - 1$ of node j's in-neighbors along with j itself. Take $\mathcal{S}_2 = \mathcal{V} \setminus \mathcal{S}_1$ as before. Since $|\mathcal{S}_1| = s < r$, again $|\mathcal{X}^r_{\mathcal{S}_2}| = 0$ so that $|\mathcal{X}^r_{\mathcal{S}_1}| = |\mathcal{S}_1|$. This implies j has an additional r in-neighbors outside of $\mathcal{S}_1$, thereby guaranteeing $|\mathcal{N}^{\text{in}}_j| \geq r + s - 1$. On the other hand, whenever $s \geq r$, form $\mathcal{S}_1$ by choosing $r - 2$ of node j's in-neighbors along with j itself. Again, choose $\mathcal{S}_2 = \mathcal{V} \setminus \mathcal{S}_1$. Since $|\mathcal{S}_1| < r$ and $s \geq r$, again $|\mathcal{X}^r_{\mathcal{S}_2}| = 0$ so that $|\mathcal{X}^r_{\mathcal{S}_1}| = |\mathcal{S}_1|$. This implies j has an additional r in-neighbors outside of $\mathcal{S}_1$, thereby guaranteeing $|\mathcal{N}^{\text{in}}_j| \geq 2r - 2$. Since $j \in \mathcal{V}$ is arbitrary, the bound on $\delta^{\text{in}}(\mathcal{D})$ follows.

Whenever $n > 1$ and $r \leq s \leq n$, form $\mathcal{S}^0_1$ by choosing $r - 1$ of node j's in-neighbors along with j itself, and take $\mathcal{S}^0_2 = \mathcal{V} \setminus \mathcal{S}^0_1$. There are three cases. If $|\mathcal{X}^r_{\mathcal{S}^0_1}| = |S^0_1|$ or $j \in \mathcal{X}^r_{\mathcal{S}^0_1}$, then $d^{\text{in}}_j \geq 2r - 1$. If $|\mathcal{X}^r_{\mathcal{S}^0_2}| = |S^0_2|$, then $d^{\text{out}}_j \geq n - r \geq \lfloor n/2 \rfloor$. If n is odd and $r = \lceil n/2 \rceil$, then $\mathcal{D}$ must be complete (by Property 5.19), in which case (c) holds. Otherwise, $|\mathcal{X}^r_{\mathcal{S}^0_2}| = |S^0_2|$ implies $d^{\text{out}}_j \geq r$. Finally, suppose $|\mathcal{X}^r_{\mathcal{S}^0_1}| + |\mathcal{X}^r_{\mathcal{S}^0_2}| \geq s \geq r$ and $j \notin \mathcal{X}^r_{\mathcal{S}^0_1}$. In this case, $|\mathcal{X}^r_{\mathcal{S}^0_2}| \geq 1$, so there exists $k_1 \in \mathcal{S}^0_2$ such that $(j, k_1) \in \mathcal{E}$. If $k_1 \notin \mathcal{N}^{\text{in}}_j$, we are done. Otherwise, construct $\mathcal{S}^1_1$ by swapping k_1 with one of j's in-neighbors in $\mathcal{S}^0_1$, and take $\mathcal{S}^1_2 = \mathcal{V} \setminus \mathcal{S}^1_1$. By the same arguments, the first two cases imply at least one of (a), (b), or (c) hold. So, assume $|\mathcal{X}^r_{\mathcal{S}^1_1}| + |\mathcal{X}^r_{\mathcal{S}^1_2}| \geq s$ and $j \notin \mathcal{X}^r_{\mathcal{S}^1_1}$. Then, there exists $k_2 \in \mathcal{S}^1_2$ such that $(j, k_2) \in \mathcal{E}$. If $k_2 \notin \mathcal{N}^{\text{in}}_j$, we are done. Otherwise, we continue to construct $\mathcal{S}^m_1$ for $m = 2, \ldots, r - 2$ by swapping k_m with one of the nodes in $\mathcal{S}^{m-1}_1 \setminus \{j, k_1, k_2, \ldots, k_{m-1}\}$ and construct $\mathcal{S}^m_2 = \mathcal{V} \setminus \mathcal{S}^m_1$ until (a) or (c) holds (by one of the first two cases), or we find $k_m \notin \mathcal{N}^{\text{in}}_j$ with $(j, k_m) \in \mathcal{E}$ (i.e., (b) holds), or we reach $m = r - 1$, in which case we may construct $\mathcal{S}^{r-1}_1 = \{j, k_1, k_2, \ldots, k_{r-1}\}$ and $\mathcal{S}^{r-1}_2 = \mathcal{V} \setminus \mathcal{S}^{r-1}_1$. At this point j has $r - 1$ out-neighbors in $\mathcal{S}^{r-1}_1$, so if $|\mathcal{X}^r_{\mathcal{S}^{r-1}_2}| \geq 1$, then there is at least one more out-neighbor, so (c) holds. Otherwise, $|\mathcal{X}^r_{\mathcal{S}^{r-1}_1}| = |\mathcal{S}^{r-1}_1|$ and (a) holds. □

The previous result demonstrates some of the degree properties of a robust digraph. The next result shows that a sufficiently large minimum in-degree implies a certain amount of robustness.

Property 5.23 (Large Minimum In-Degree Implies Robustness). *If a nontrivial digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ has minimum in-degree $\delta^{in}(\mathcal{D}) \geq \lfloor n/2 \rfloor + r - 1$ with $0 \leq r \leq \lceil n/2 \rceil$, then $\mathcal{D}$ is (r, s)-robust, for* *all* $1 \leq s \leq n$.

Proof. Fix any disjoint and nonempty subsets of nodes $\mathcal{S}_1, \mathcal{S}_2 \subset \mathcal{V}$. The smaller of the two subsets must contain no more than $\lfloor n/2 \rfloor$ nodes. Without loss of generality, assume $|\mathcal{S}_1| \leq \lfloor n/2 \rfloor$. At most $\lfloor n/2 \rfloor - 1$ neighbors of any node in $\mathcal{S}_1$ are also in $\mathcal{S}_1$. Since $\delta^{\text{in}}(\mathcal{D}) \geq \lfloor n/2 \rfloor + r - 1$, there are at least r additional neighbors outside of $\mathcal{S}_1$ for every node in $\mathcal{S}_1$. Therefore, condition (i) of Definition 5.10 is satisfied, and $\mathcal{D}$ is (r, s)-robust. □

Finally, we relate the robustness of the underlying graph to its connectivity.

Property 5.24 (Connectivity of Robust Networks). *Suppose $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ is an r-robust digraph, with $0 \leq r \leq \lceil n/2 \rceil$. Then the underlying graph $\mathcal{G}_\mathcal{D}$ is at least r-connected. Furthermore, if $\mathcal{D}$ is (r, r)-robust, with $3 \leq r \leq \lceil n/2 \rceil$, then $\mathcal{G}_\mathcal{D}$ is at least $(\lceil 3r/2 \rceil - 1)$-connected.*

Proof. If $r = 0$, the first statement is vacuously true, and if $r = 1$, it holds by Property 5.14. Therefore, assume $r \geq 2$. By Property 5.17, the underlying graph $\mathcal{G}_\mathcal{D} = (\mathcal{V}, \mathcal{E}_\mathcal{G})$ is r-robust. By Property 5.13 and Property 5.14, the graph is connected. Suppose there is a vertex cut $\mathcal{K} \subset \mathcal{V}$ such that $|\mathcal{K}| < r$, and denote the $k \geq 2$ connected components remaining after the removal of $\mathcal{K}$ by $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k$. Let $\mathcal{S}_1 = \mathcal{C}_1$ and $\mathcal{S}_2 = \mathcal{C}_2$. Since $\mathcal{G}_\mathcal{D}$ is r-robust, either $\mathcal{S}_1$ or $\mathcal{S}_2$ is r-edge reachable, which contradicts the fact that $\mathcal{K}$ is a vertex cut. Hence, any vertex cut $\mathcal{K}$ must satisfy $|\mathcal{K}| \geq r$, so that $\mathcal{G}_\mathcal{D}$ is at least r-connected.

For the second statement, suppose there is a vertex cut $\mathcal{K} \subset \mathcal{V}$ such that $r \leq |\mathcal{K}| \leq \lceil 3r/2 \rceil - 2$, and denote the $k \geq 2$ connected components remaining after the removal of $\mathcal{K}$ by $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k$. Partition $\mathcal{K}$ into $\mathcal{K} = \mathcal{K}_1 \cup \mathcal{K}_2 \cup \mathcal{K}_3$ such that $|\mathcal{K}_1| = |\mathcal{K}_2| = \lceil r/2 \rceil - 1 > 0$ (since $r \geq 3$) and the remaining nodes go to $\mathcal{K}_3$; i.e., $1 \leq |\mathcal{K}_3| \leq \lfloor r/2 \rfloor$. Then form $\mathcal{S}_1 = \mathcal{C}_1 \cup \mathcal{K}_1$ and $\mathcal{S}_2 = \mathcal{C}_2 \cup \mathcal{K}_2$. Because $|\mathcal{K}| \leq \lceil 3r/2 \rceil - 2$ and $\delta(\mathcal{G}_\mathcal{D}) \geq 2r - 2$ by Property 5.22, it follows that $|\mathcal{C}_i| \geq \lfloor r/2 \rfloor + 1$ for all $1 \leq i \leq k$ (since there are at most $\lceil 3r/2 \rceil - 2$ neighbors in $\mathcal{K}$). Therefore, $|\mathcal{S}_1|, |\mathcal{S}_2| \geq r$. Combining this with the fact that $\mathcal{G}_\mathcal{D}$ is (r, r)-robust (by Property 5.17), we are guaranteed $|\mathcal{X}_{\mathcal{S}_1}^r| + |\mathcal{X}_{\mathcal{S}_2}^r| \geq r$.

Because $|\mathcal{K}_1 \cup \mathcal{K}_2| \leq r - 1$, there is $v \in \mathcal{C}_1 \cup \mathcal{C}_2$ such that v has at least r in-neighbors outside of its set. Without loss of generality, assume $v \in \mathcal{C}_1$. Since $|\mathcal{K}_2| + |\mathcal{K}_3| \leq r - 1$, $\exists j \in \mathcal{C}_2 \cup \cdots \cup \mathcal{C}_k$ such that $(j, v) \in \mathcal{E}$, which contradicts the fact that $\mathcal{K}$ is a vertex cut whose removal results in components $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k$. Hence, $\mathcal{G}_\mathcal{D}$ is at least $(\lceil 3r/2 \rceil - 1)$-connected. □

5.4 CTRAC Results

We demonstrate in this section that the (r, s)-robustness property is the key property for analyzing ARC-P2 with parameter F under the F-total model. We also show that r-robustness is useful for analyzing ARC-P2 with parameter F under the F-local model, and p-fraction robustness is useful under the f-fraction local model. More specifically, we show that $(F + 1, F + 1)$-robustness of the network is both necessary and sufficient for normal nodes using ARC-P2 with parameter F to achieve CTRAC in time-invariant networks in the presence of malicious nodes under the F-total model. In fact, $(F + 1, F + 1)$-robustness is a necessary condition in time-invariant networks under the F-total or F-local crash models (and therefore necessary for the malicious and Byzantine models). We show that $(2F + 1)$-robustness is sufficient under the F-local malicious model. For the f-fraction local model, f-fraction robust is necessary in the presence of crash adversaries and $2f$-fraction robust is sufficient in the presence of malicious adversaries. For Byzantine adversaries, it is necessary and sufficient for the normal network (i.e., the network containing only the normal nodes and directed edges whose head and tail are both normal) to be $(F + 1)$-robust under the F-total or F-local models. For the f-fraction local model, f-fraction robustness of the normal network is necessary and p'-fraction robustness is sufficient, if $p' > f$.

Recall that Lemma 5.5 shows that ARC-P2 with parameter F (or f) ensures the safety condition of CTRAC holds under the F-total and F-local models (or f-fraction local model). It follows from Lemma 5.4 that $M_{\mathcal{N}}(\cdot)$ is nonincreasing with time, and $m_{\mathcal{N}}(\cdot)$ is nondecreasing with time. Therefore, if agreement is achieved among the normal agents, then the values of the normal agents must converge to a common limit. For this reason, we focus on proving that the Lyapunov candidate $\Psi(t) = M_{\mathcal{N}}(t) - m_{\mathcal{N}}(t)$ asymptotically vanishes (i.e., agreement is achieved). In the following sections, we show that this Lyapunov function decreases over sufficiently large time intervals whenever the normal nodes update their values according to ARC-P2, provided the network is sufficiently robust.

5.4.1 Necessary Conditions

In this section, we present some necessary conditions for any of the adversary models whenever ARC-P2 is used in time-invariant networks. The first result shows that whenever the normal nodes use ARC-P2 with parameter F under the F-total or F-local model – with any of the threat models studied here – then $(F+1, F+1)$-robustness is a necessary condition. Moreover, $n > 2F$ is also necessary. The second result shows that in the case of parameter f under the f-fraction local model, f-robustness is necessary.

Theorem 5.25 (Necessary Conditions for ARC-P2 with Parameter F under the F-Total and F-Local Crash Model)**.** *Consider a time-invariant network modeled by digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ where each normal node updates its value according to ARC-P2 with parameter $F \in \mathbb{Z}_{\geq 0}$. If CTRAC is achieved under the F-total or F-local crash model then the network topology is $(F+1, F+1)$-robust and $n > 2F$.*

Proof. If $\mathcal{D}$ is not $(F+1, F+1)$-robust, then there are nonempty, disjoint $\mathcal{S}_1, \mathcal{S}_2 \subset \mathcal{V}$ such that none of the conditions $(i)-(iii)$ of Definition 5.10 hold (with $r = F+1$ and $s = F+1$). Suppose the initial value of each node in $\mathcal{S}_1$ is a and each node in $\mathcal{S}_2$ is b, with $a < b$. Let all other nodes have initial values taken from the interval $[a, b]$. Since $|\mathcal{X}_{\mathcal{S}_1}^{F+1}| + |\mathcal{X}_{\mathcal{S}_2}^{F+1}| \leq F$, suppose all nodes in $\mathcal{X}_{\mathcal{S}_1}^{F+1}$ and $\mathcal{X}_{\mathcal{S}_2}^{F+1}$ are crash nodes that compromise these nodes at $t_0 = 0$ (and therefore keep their values constant for all $t \geq 0$). With this assignment of adversaries, there is still at least one normal node in both $\mathcal{S}_1$ and $\mathcal{S}_2$ since $|\mathcal{X}_{\mathcal{S}_1}^{F+1}| < |\mathcal{S}_1|$ and $|\mathcal{X}_{\mathcal{S}_2}^{F+1}| < |\mathcal{S}_2|$, respectively. Therefore, each normal node in $\mathcal{S}_1$ removes the F or less values greater than a from outside and each normal node in $\mathcal{S}_2$ removes the F or less values less than b from outside. Therefore, the normal nodes in $\mathcal{S}_1$ maintain the value of a and the normal nodes in $\mathcal{S}_2$ maintain the value of b for all $t \geq 0$. Hence, consensus among normal nodes is not achieved, which contradicts the assumption.

Since $(F+1, F+1)$-robustness is a necessary condition, it follows from Property 5.19 that

$$n > 2(\lceil n/2 \rceil - 1) \geq 2F.$$

Therefore, $n > 2F$ is also necessary. □

Theorem 5.26 (Necessary Conditions for ARC-P2 with Parameter f under the f-Fraction Local Crash Model)**.** *Consider a time-invariant network modeled by digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ where each normal node updates its value according to ARC-P2 with parameter $f \in [0, 1]$ in the presence of crash adversaries under the f-fraction local model. If CTRAC is achieved, then $\mathcal{D}$ is f-fraction* *robust*.

Proof. Suppose that $\mathcal{D}$ is not f-fraction robust. Then, there exists nonempty, disjoint subsets $\mathcal{S}_1, \mathcal{S}_2 \subset \mathcal{V}$ such that neither $\mathcal{S}_1$ nor $\mathcal{S}_2$ is f-fraction edge reachable. This means that for every $i \in \mathcal{S}_k$, $|\mathcal{N}_i^{\text{in}} \setminus \mathcal{S}_k| < \lceil f d_i \rceil$, for $k \in \{1,2\}$. From this, it follows that $|\mathcal{N}_i^{\text{in}} \setminus \mathcal{S}_k| \leq \lfloor f d_i \rfloor$ for all $i \in \mathcal{S}_k$. Suppose the initial value of each node in $\mathcal{S}_1$ is a and each node in $\mathcal{S}_2$ is b, with $a < b$. Let all other nodes have initial values taken from the interval $[a, b]$. Assume all nodes are normal. Then, using ARC-P2 with parameter f, each node i in $\mathcal{S}_1$ removes the $\lfloor f d_i \rfloor$ or less values greater than a from outside $\mathcal{S}_1$. Likewise, each node j in $\mathcal{S}_2$ removes the $\lfloor f d_j \rfloor$ or less values less than b from outside $\mathcal{S}_2$. Therefore, as in Theorem 5.25, it follows that all nodes in $\mathcal{S}_1$ keep the value a and each node in $\mathcal{S}_2$ keeps the value b for all $t \geq 0$. Therefore, CTRAC is not achieved. ☐

5.4.2 F-Total Malicious and Crash Models

This section presents the main result for the F-total malicious and crash models in time-invariant and time-varying networks. We show that $(F+1, F+1)$-robustness is both necessary and sufficient for ARC-P2 to achieve CTRAC under the F-total malicious model in time-invariant networks under the additional continuity assumption requiring uniform continuity of the malicious adversaries' state trajectories. Afterwards, we show that in time-varying networks that satisfy a dwell time assumption, it is sufficient for the switching topologies to eventually switch between $(F+1, F+1)$-robust topologies for all $t \geq t_0 \geq 0$.

Theorem 5.27 (Tight Conditions for ARC-P2 under F-Total Malicious and Crash Model)**.** *Consider a time-invariant network modeled by digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ where the adversaries satisfy the F-total malicious or crash model and have uniformly continuous trajectories on* $[0, \infty)$. *Suppose each normal node updates its value according to ARC-P2 with parameter F. Then, CTRAC is achieved if and only if the network topology is* $(F+1, F+1)$*-robust*.

Proof. Necessity follows from Theorem 5.25. For sufficiency, we know from Lemmas 5.4 and 5.5 that both $M_{\mathcal{N}}(\cdot)$ and $m_{\mathcal{N}}(\cdot)$ are monotone and bounded functions of t. Therefore each of them has a limit, denoted by A_M and A_m, respectively. Note that if $A_M = A_m$, then CTRAC is achieved. We prove by contradiction that this must be the case. The main idea behind the proof is to use the gap between A_M and A_m and combine this with both the uniform continuity assumption on the malicious nodes' value trajectories and a careful selection of subsets of nodes to show that $\Psi(t)$ will shrink to be smaller than the gap $A_M - A_m$ in finite time (a contradiction). To this end, suppose that $A_M \neq A_m$ (note that $A_M > A_m$ by definition). Since $M_{\mathcal{N}}(t) \to A_M$ monotonically, we have $M_{\mathcal{N}}(t) \geq A_M$ for all $t \geq 0$. Similarly, $m_{\mathcal{N}}(t) \leq A_m$ for all $t \geq 0$. Moreover, for each $\epsilon > 0$ there exists $t_\epsilon > 0$ such that $M_{\mathcal{N}}(t) < A_M + \epsilon$ and $m_{\mathcal{N}}(t) > A_m - \epsilon$, $\forall t \geq t_\epsilon$. Next, define constant $\epsilon_0 = (A_M - A_m)/4 > 0$, which satisfies

$$M_{\mathcal{N}}(t) - \epsilon_0 \geq m_{\mathcal{N}}(t) + \epsilon_0 + (A_M - A_m)/2. \tag{54}$$

This inequality informs the choice of subsets of nodes to be defined shortly in order to limit the influence of the malicious nodes. Indeed, since the adversary trajectory[40] x_k is uniformly continuous on $[0, \infty)$ for $k \in \mathcal{A}$, it follows that for each $\nu > 0$, there exists $\delta_k(\nu) > 0$ such that $|x_k(t_1) - x_k(t_2)| < \nu$ whenever $|t_1 - t_2| < \delta_k(\nu)$. Define $\delta(\nu) = \min_{k \in \mathcal{A}}\{\delta_k(\nu)\}$.

Next, we define the sets of nodes that are vital to the proof. For any $t_0 \geq 0, t \geq t_0, \Delta > 0$, and $\eta > 0$, define

$$\mathcal{X}_M(t, t_0, \Delta, \eta) = \{i \in \mathcal{V} \colon \exists t' \in [t, t + \Delta] \text{ s.t. } x_i(t') > M_{\mathcal{N}}(t_0) - \eta\}$$

and

$$\mathcal{X}_m(t, t_0, \Delta, \eta) = \{i \in \mathcal{V} \colon \exists t' \in [t, t + \Delta] \text{ s.t. } x_i(t') < m_{\mathcal{N}}(t_0) + \eta\}.$$

Observe that if we choose $\eta \leq \epsilon_0 = (A_M - A_m)/4$, $\nu < (A_M - A_m)/2$, and $\Delta < \delta(\nu)$, then we are guaranteed by the uniform continuity assumption and (54) that for any $t_0 \geq 0$ and $t \geq t_0$, $\mathcal{X}_M(t, t_0, \Delta, \eta) \cap \mathcal{X}_m(t, t_0, \Delta, \eta) \cap \mathcal{A} = \emptyset$. That is, with these choices of η, ν, and Δ, no malicious node can be in both $\mathcal{X}_M(t, t_0, \Delta, \eta)$ and $\mathcal{X}_m(t, t_0, \Delta, \eta)$. This follows because otherwise there exists $t_1, t_2 \in [t, t + \Delta]$ and $k \in \mathcal{A}$ such that $x_k(t_1) > M_{\mathcal{N}}(t_0) - \eta$ and $x_k(t_2) < m_{\mathcal{N}}(t_0) + \eta$, from which we reach the contradiction to the uniform continuity assumption

$$x_k(t_1) - x_k(t_2) > M_{\mathcal{N}}(t_0) - m_{\mathcal{N}}(t_0) - 2\eta \geq \frac{A_M - A_m}{2} > \nu.$$

[40]Since the adversaries in this case are *not* deceptive, $x_{(k,i)} \equiv x_{(k,j)}$ for all $i, j \in \mathcal{N}$. Therefore, the trajectory of adversary $k \in \mathcal{A}$ is uniquely defined as x_k with no confusion.

We now proceed by showing that if we choose η, ν, and Δ small enough, then no *normal* node can be in both $\mathcal{X}_M(t, t_0, \Delta, \eta)$ and $\mathcal{X}_m(t, t_0, \Delta, \eta)$ for any $t_0 \geq 0$ and $t \geq t_0$. First, we require some generic bounds on the normal node trajectories. For $i \in \mathcal{N}$, we know from Lemma 5.4 that for $\tau \in [t', t]$,

$$\dot{x}_i(\tau) = \sum_{j \in \mathcal{N}_i^{\text{in}} \setminus \mathcal{R}_i(\tau)} w_{(j,i)}(\tau) \left(x_{(j,i)}(\tau) - x_i(\tau) \right) \leq B(M_{\mathcal{N}}(\tau) - x_i(\tau)) \leq B(M_{\mathcal{N}}(t') - x_i(\tau)),$$

whenever the derivative exists[41], where $B = (n - F - 1)\beta$. Using the integrating factor $e^{B(\tau - t')}$, and integrating in the sense of Lebesgue over the time interval $[t', t]$, we have

$$x_i(t) \leq x_i(t')e^{-B(t-t')} + M_{\mathcal{N}}(t')(1 - e^{-B(t-t')}), \;\; \forall t \geq t'. \tag{55}$$

By interchanging the roles of t and t', we have

$$x_i(t) \geq x_i(t')e^{B(t'-t)} + M_{\mathcal{N}}(t)(1 - e^{B(t'-t)}), \;\; \forall t \leq t'. \tag{56}$$

Similarly, we can show that for $i \in \mathcal{N}$,

$$x_i(t) \geq x_i(t')e^{-B(t-t')} + m_{\mathcal{N}}(t')(1 - e^{-B(t-t')}), \;\; \forall t \geq t', \tag{57}$$

and

$$x_i(t) \leq x_i(t')e^{B(t'-t)} + m_{\mathcal{N}}(t)(1 - e^{B(t'-t)}), \;\; \forall t \leq t', \tag{58}$$

Now fix $\eta \leq \epsilon_0 = (A_M - A_m)/4$, $\nu < (A_M - A_m)/2$, and $\Delta < \min\{\delta(\nu), \log(3)/B\}$, and suppose $i \in \mathcal{N} \cap \mathcal{X}_M(t, t_0, \Delta, \eta)$. Then $\exists t' \in [t, t + \Delta]$ such that $x_i(t') > M_{\mathcal{N}}(t_0) - \eta$. Combining

[41]Recall, the solutions of the normal nodes' trajectories are understood in the sense of Carathéodory. Hence, it is possible that the derivative of the solution does not exist on a set of points in time of Lebesgue measure zero.

this with (57), it follows that for $s \in [t', t+\Delta]$,

$$\begin{aligned}
x_i(s) &\geq x_i(t')e^{-B(s-t')} + m_{\mathcal{N}}(t')(1 - e^{-B(s-t')}) \\
&> (M_{\mathcal{N}}(t_0) - \eta)e^{-B(s-t')} + m_{\mathcal{N}}(t_0)(1 - e^{-B(s-t')}) \\
&\geq (A_M - \eta)e^{-B(s-t')} + m_{\mathcal{N}}(t_0) - A_m e^{-B(s-t')} \\
&\geq m_{\mathcal{N}}(t_0) + (A_M - A_m)e^{-B(s-t')} - \frac{A_M - A_m}{4}e^{-B(s-t')} \\
&\geq m_{\mathcal{N}}(t_0) + \frac{3}{4}(A_M - A_m)e^{-B\Delta} \\
&> m_{\mathcal{N}}(t_0) + \frac{A_M - A_m}{4} \\
&\geq m_{\mathcal{N}}(t_0) + \eta,
\end{aligned}$$

where we have used the fact that $\Delta < \log(3)/B$ in deriving the last line. Similarly, using (56), it follows that for $s \in [t, t']$,

$$\begin{aligned}
x_i(s) &\geq x_i(t')e^{B(t'-s)} + M_{\mathcal{N}}(s)(1 - e^{B(t'-s)}) \\
&> (M_{\mathcal{N}}(t_0) - \eta)e^{B(t'-s)} + M_{\mathcal{N}}(s)(1 - e^{B(t'-s)}) \\
&\geq M_{\mathcal{N}}(s) - \eta e^{B(t'-s)} \\
&\geq M_{\mathcal{N}}(s) - \frac{A_M - A_m}{4}e^{B\Delta} \\
&> A_M - \frac{3}{4}(A_M - A_m) \\
&\geq A_m + \frac{1}{4}(A_M - A_m) \\
&\geq m_{\mathcal{N}}(t_0) + \eta.
\end{aligned}$$

Therefore, $i \notin \mathcal{X}_m(t, t_0, \Delta, \eta)$.

Similarly, with the given choices for η, ν, and Δ, if $j \in \mathcal{N} \cap \mathcal{X}_m(t, t_0, \Delta, \eta)$, then $\exists t' \in [t, t+\Delta]$

such that $x_i(t') < m_{\mathcal{N}}(t_0) + \eta$. It follows from (55) that for $s \in [t', t + \Delta]$,

$$\begin{aligned}
x_j(s) &\le x_j(t')e^{-B(s-t')} + M_{\mathcal{N}}(t')(1 - e^{-B(s-t')}) \\
&< (m_{\mathcal{N}}(t_0) + \eta)e^{-B(s-t')} + M_{\mathcal{N}}(t_0)(1 - e^{-B(s-t')}) \\
&\le M_{\mathcal{N}}(t_0) - (M_{\mathcal{N}}(t_0) - m_{\mathcal{N}}(t_0))e^{-B(s-t')} + \eta e^{-B(s-t')} \\
&\le M_{\mathcal{N}}(t_0) - (A_M - A_m)e^{-B(s-t')} + \frac{A_M - A_m}{4}e^{-B(s-t')} \\
&\le M_{\mathcal{N}}(t_0) - \frac{3}{4}(A_M - A_m)e^{-B\Delta} \\
&< M_{\mathcal{N}}(t_0) - \frac{A_M - A_m}{4} \\
&\le M_{\mathcal{N}}(t_0) - \eta,
\end{aligned}$$

where we have used the fact that $\Delta < \log(3)/B$ in deriving the last line. Finally, using (58), it follows that for $s \in [t, t']$,

$$\begin{aligned}
x_j(s) &\le x_j(t')e^{B(t'-s)} + m_{\mathcal{N}}(s)(1 - e^{B(t'-s)}) \\
&< (m_{\mathcal{N}}(t_0) + \eta)e^{B(t'-s)} + m_{\mathcal{N}}(s)(1 - e^{B(t'-s)}) \\
&\le m_{\mathcal{N}}(s) + \eta e^{B(t'-s)} \\
&\le A_m + \frac{A_M - A_m}{4}e^{B\Delta} \\
&< A_m + \frac{3}{4}(A_M - A_m) \\
&\le M_{\mathcal{N}}(t_0) - \frac{A_M - A_m}{4} \\
&\le M_{\mathcal{N}}(t_0) - \eta.
\end{aligned}$$

Thus, $j \notin \mathcal{X}_M(t, t_0, \Delta, \eta)$. This shows that $\mathcal{X}_M(t, t_0, \Delta, \eta)$ and $\mathcal{X}_m(t, t_0, \Delta, \eta)$ are disjoint for appropriate choices of the parameters.

Next, we show that by choosing ϵ small enough, we can define a sequence of sets,

$$\{\mathcal{X}_M(t_\epsilon + k\Delta, t_\epsilon, \Delta, \epsilon_k)\}_{k=0}^{k=N}$$

and

$$\{\mathcal{X}_m(t_\epsilon + k\Delta, t_\epsilon, \Delta, \epsilon_k)\}_{k=0}^{k=N},$$

where $N = |\mathcal{N}|$, so that we are guaranteed that by the Nth step, at least one of the sets contains no normal nodes. This will be used to show that Ψ has shrunk below $A_M - A_m$. Toward this end, let $\epsilon_0 = (A_M - A_m)/4$, $\nu < (A_M - A_m)/2$, and $\Delta < \min\{\delta(\nu), \log(3)/B\}$. Then fix

$$\epsilon < \frac{1}{2}\left[\frac{\alpha}{B}(1 - e^{-B\Delta})e^{-B\Delta}\right]^{2N}\epsilon_0.$$

For $k = 0, 1, 2, \dots, N$, define $\epsilon_k = [\frac{\alpha}{B}(1 - e^{-B\Delta})e^{-B\Delta}]^{2k}\epsilon_0$, which results in

$$\epsilon_0 > \epsilon_1 > \cdots > \epsilon_N > 2\epsilon > 0.$$

For brevity, define

$$\mathcal{X}_M^k = \mathcal{X}_M(t_\epsilon + k\Delta, t_\epsilon, \Delta, \epsilon_k)$$

and

$$\mathcal{X}_m^k = \mathcal{X}_m(t_\epsilon + k\Delta, t_\epsilon, \Delta, \epsilon_k)$$

for $k = 0, 1, \dots, N$. Observe that by definition, there is at least one normal node (the ones with extreme values) in $\mathcal{X}_M^0$ and $\mathcal{X}_m^0$, and we have shown above that all of the $\mathcal{X}_M^k$ and $\mathcal{X}_m^k$ are disjoint. It follows from the fact that there are at most F malicious nodes in the network (F-total model) and $\mathcal{D}$ is $(F+1, F+1)$-robust, that either $\exists i \in \mathcal{X}_M^0 \cap \mathcal{N}$ or $\exists i \in \mathcal{X}_m^0 \cap \mathcal{N}$ (or both) such that i has at least $F+1$ neighbors outside of its set. That is, either i has at least $F+1$ neighbors $i_1, i_2, \dots, i_{F+1}$ such that $x_{i_k}(t) \leq M_{\mathcal{N}}(t_\epsilon) - \epsilon_0$ for all $t \in [t_\epsilon, t_\epsilon + \Delta]$ (if $i \in \mathcal{X}_M^0 \cap \mathcal{N}$), or $x_{i_k}(t) \geq m_{\mathcal{N}}(t_\epsilon) + \epsilon_0$ for all $t \in [t_\epsilon, t_\epsilon + \Delta]$ (if $i \in \mathcal{X}_m^0 \cap \mathcal{N}$). Assume $i \in \mathcal{X}_M^0 \cap \mathcal{N}$ and suppose that none of the $F+1$ (or more) neighbors outside of $\mathcal{X}_M^0$ are used in its update at some time $t' \in [t_\epsilon, t_\epsilon + \Delta]$ at which the derivative exists. Then, $x_i(t') \leq M_{\mathcal{N}}(t_\epsilon) - \epsilon_0$ (otherwise, it would use at least one of its $F+1$ neighbors' values outside of $\mathcal{X}_M^0$). It follows from (55) that

$$x_i(t_\epsilon + \Delta) \leq M_{\mathcal{N}}(t_\epsilon) - \epsilon_0 e^{-B\Delta}.$$

Using this with (55) to upper bound $x_i(t)$ for $t \in [t_\epsilon + \Delta, t_\epsilon + 2\Delta]$, we see that

$$x_i(t) \leq M_{\mathcal{N}}(t_\epsilon) - \epsilon_0 e^{-2B\Delta} \leq M_{\mathcal{N}}(t_\epsilon) - \epsilon_1.$$

Therefore, in this case $i \notin \mathcal{X}_M^1$. Alternatively, assume at least one of the values from its neighbors outside of $\mathcal{X}_M^0$ is used for almost all $t \in [t_\epsilon, t_\epsilon + \Delta]$. Then,

$$\begin{aligned} \dot{x}_i(t) &\leq \alpha(M_{\mathcal{N}}(t_\epsilon) - \epsilon_0 - x_i(t)) + (B - \alpha)(M_{\mathcal{N}}(t_\epsilon) - x_i(t)) \\ &\leq -Bx_i(t) + BM_{\mathcal{N}}(t_\epsilon) - \alpha\epsilon_0, \end{aligned}$$

for almost all $t \in [t_\epsilon, t_\epsilon + \Delta]$. Using this, we can show

$$\begin{aligned} x_i(t_\epsilon + \Delta) &\leq x_i(t_\epsilon)e^{-B\Delta} + (M_{\mathcal{N}}(t_\epsilon) - \tfrac{\alpha\epsilon_0}{B})(1 - e^{-B\Delta}) \\ &\leq M_{\mathcal{N}}(t_\epsilon) - \tfrac{\alpha}{B}(1 - e^{-B\Delta})\epsilon_0. \end{aligned}$$

Using this with (55) to upper bound $x_i(t)$ for $t \in [t_\epsilon + \Delta, t_\epsilon + 2\Delta]$, we see that for all $t \in [t_\epsilon + \Delta, t_\epsilon + 2\Delta]$,

$$\begin{aligned} x_i(t) &\leq M_{\mathcal{N}}(t_\epsilon) - \tfrac{\alpha}{B}(1 - e^{-B\Delta})e^{-B(t - t_\epsilon - \Delta)}\epsilon_0 \\ &\leq M_{\mathcal{N}}(t_\epsilon) - \tfrac{\alpha}{B}(1 - e^{-B\Delta})e^{-B\Delta}\epsilon_0 \\ &\leq M_{\mathcal{N}}(t_\epsilon) - \epsilon_1. \end{aligned}$$

Thus, in either case $i \notin \mathcal{X}_M^1$. The final step is to show that $j \notin \mathcal{X}_m^1$ whenever j is a normal node with $j \notin \mathcal{X}_m^0$. Whenever $j \notin \mathcal{X}_m^0$, it means that $x_j(t_\epsilon + \Delta) \geq m_{\mathcal{N}}(t_\epsilon) + \epsilon_0$. Using this with (57) to lower bound $x_j(t)$ for $t \in [t_\epsilon + \Delta, t_\epsilon + 2\Delta]$, we see that

$$x_j(t) \geq m_{\mathcal{N}}(t_\epsilon) + \epsilon_0 e^{-B\Delta} \geq m_{\mathcal{N}}(t_\epsilon) + \epsilon_1.$$

Hence, j is also not in $\mathcal{X}_m^1$, as claimed. Likewise, we can show using (55) that $j \notin \mathcal{X}_M^1$ whenever j is a normal node with $j \notin \mathcal{X}_M^0$. Therefore, if $i \in \mathcal{X}_M^0 \cap \mathcal{N}$ has at least $F + 1$ neighbors outside of its set, we are guaranteed that $|\mathcal{X}_M^1 \cap \mathcal{N}| < |\mathcal{X}_M^0 \cap \mathcal{N}|$ and $|\mathcal{X}_m^1 \cap \mathcal{N}| \leq |\mathcal{X}_m^0 \cap \mathcal{N}|$. Using a similar argument, we can show that if $i \in \mathcal{X}_m^0 \cap \mathcal{N}$ has at least $F + 1$ neighbors outside of its set, we are guaranteed that $|\mathcal{X}_m^1 \cap \mathcal{N}| < |\mathcal{X}_m^0 \cap \mathcal{N}|$ and $|\mathcal{X}_M^1 \cap \mathcal{N}| \leq |\mathcal{X}_M^0 \cap \mathcal{N}|$.

Now, if both $\mathcal{X}_M^1 \cap \mathcal{N}$ and $\mathcal{X}_m^1 \cap \mathcal{N}$ are nonempty, we can repeat the above argument to show that either $|\mathcal{X}_m^2 \cap \mathcal{N}| < |\mathcal{X}_m^1 \cap \mathcal{N}|$ or $|\mathcal{X}_M^2 \cap \mathcal{N}| < |\mathcal{X}_M^1 \cap \mathcal{N}|$, or both. It follows by induction that as long as both $\mathcal{X}_M^j \cap \mathcal{N}$ and $\mathcal{X}_m^j \cap \mathcal{N}$ are nonempty, then either $|\mathcal{X}_m^{j+1} \cap \mathcal{N}| < |\mathcal{X}_m^j \cap \mathcal{N}|$ or $|\mathcal{X}_M^{j+1} \cap \mathcal{N}| < |\mathcal{X}_M^j \cap \mathcal{N}|$ (or both), for $j = 1, 2, \ldots$. Since $|\mathcal{X}_m^0 \cap \mathcal{N}| + |\mathcal{X}_M^0 \cap \mathcal{N}| \leq N$, there exists $T < N$ such that at least one of $\mathcal{X}_M^T \cap \mathcal{N}$ and $\mathcal{X}_m^T \cap \mathcal{N}$ is empty. If $\mathcal{X}_M^T \cap \mathcal{N} = \emptyset$, then $M_\mathcal{N}(t_\epsilon + T\Delta) \leq M_\mathcal{N}(t_\epsilon) - \epsilon_T < M_\mathcal{N}(t_\epsilon) - 2\epsilon$. Similarly, if $\mathcal{X}_m^T \cap \mathcal{N} = \emptyset$, then $m_\mathcal{N}(t_\epsilon + T\Delta) \geq m_\mathcal{N}(t_\epsilon) + \epsilon_T > m_\mathcal{N}(t_\epsilon) + 2\epsilon$. In either case, $\Psi(t_\epsilon + T\Delta) < A_M - A_m$ and we reach the desired contradiction. $\square$

When the network is time-varying, one can state the following theorem.

Theorem 5.28. *Consider a time-varying network modeled by $\mathcal{D}(t) = (\mathcal{V}, \mathcal{E}(t))$ where the adversaries satisfy the F-total malicious or crash model and have uniformly continuous trajectories on $[0, \infty)$. Let $\{t_k\}$ denote the switching times of $\sigma(t)$ and assume that $t_{k+1} - t_k \geq \tau$ for all k. Suppose each normal node updates its value according to ARC-P2 with parameter F. Then, CTRAC is achieved if there exists $t_0 \geq 0$ such that $\mathcal{D}(t)$ is $(F+1, F+1)$-robust, $\forall t \geq t_0$.*

Proof. The proof follows the contradiction argument of the proof of Theorem 5.27, but here we use the dwell time assumption. In this case, let

$$\Delta < \min\{\delta(\nu), \log(3)/B, \tfrac{\tau}{N}\}.$$

Fix

$$\epsilon < \frac{1}{2}\left[\frac{\alpha}{B}(1 - e^{-B\Delta})e^{-B\Delta}\right]^{2N} \epsilon_0,$$

and let $t'_\epsilon \geq 0$ be a point in time such that $M_\mathcal{N}(t) < A_M + \epsilon$ and $m_\mathcal{N}(t) > A_m - \epsilon$ for all $t \geq t'_\epsilon$. Define $t' = \max\{t_0, t'_\epsilon\}$. Then, associated to the switching signal $\sigma(t)$, we define t_ϵ as the next switching instance after t', or t' itself if there are no switching instances after t'. Since $\Delta < \tau/N$, the same sequence of calculations can be used (as in the proof of Theorem 5.27) to show that $\Psi(t_\epsilon + T\Delta) < A_M - A_m$. $\square$

To illustrate these results on the examples of Section 5.3, the graphs in Figures 35 and 40 can withstand the compromise of at most 1 malicious node in the network using ARC-P2 with parameter

$F = 1$ (each graph is (2,2)-robust but not (3,3)-robust). This is not to say that it is impossible for the normal nodes to reach consensus if there are, for example, two nodes that are compromised. Instead, these results say that it is not possible that *any* two nodes can be compromised and still guarantee CTRAC using ARC-P2 with parameter $F = 2$.

5.4.3 F-Local Malicious and Crash Models

Theorem 5.29 (Sufficient Conditions for ARC-P2 under F-Local Malicious and Crash Model)**.** *Consider a time-invariant network modeled by digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ where the adversaries satisfy the F-local malicious or crash model. Suppose each normal node updates its value according to ARC-P2 with parameter F. Then, CTRAC is achieved if the network topology is $(2F+1)$-robust.*

Proof. The proof follows the same line as that of Theorem 5.27. The main difference is that the sets of nodes $\mathcal{X}_M$ and $\mathcal{X}_m$ include only normal nodes. That is, for any $t_0 \geq 0, t \geq t_0, \Delta > 0$, and $\eta > 0$, define

$$\mathcal{X}_M(t, t_0, \Delta, \eta) = \{i \in \mathcal{N} \colon \exists t' \in [t, t+\Delta] \text{ s.t. } x_i(t') > M_{\mathcal{N}}(t_0) - \eta\}$$

and

$$\mathcal{X}_m(t, t_0, \Delta, \eta) = \{i \in \mathcal{N} \colon \exists t' \in [t, t+\Delta] \text{ s.t. } x_i(t') < m_{\mathcal{N}}(t_0) + \eta\}.$$

Likewise, for $k = 1, 2, \ldots, N$, the definitions of $\mathcal{X}_M^k$ and $\mathcal{X}_m^k$ are modified to include only normal nodes. The analysis showing that $\mathcal{X}_M^k$ and $\mathcal{X}_m^k$ are disjoint still holds, and in this case does not require the uniform continuity assumption (since adversaries are excluded from the sets). By definition, it follows that $\mathcal{X}_M^0$ and $\mathcal{X}_m^0$ are nonempty. Since the network is $(2F+1)$-robust, either $\exists i \in \mathcal{X}_M^0$ or $\exists i \in \mathcal{X}_m^0$, or both, such that i has at least $2F+1$ neighbors outside of its set. If such i is in $\mathcal{X}_M^0$, then at most F of the neighbors are malicious (F-local model) and the others are normal with value $x_j(t) \leq M_{\mathcal{N}}(t_\epsilon) - \epsilon_0$ for $t \in [t_\epsilon, t_\epsilon + \Delta]$. The remaining argument follows the same line as that of Theorem 5.27. □

As with the F-total model, we have the following result for time-varying networks (whose proof follows the same line as that of Theorem 5.28).

Theorem 5.30 (Time-Varying Sufficient Condition for F-Local Malicious or Crash Model)**.** *Consider a time-varying network modeled by $\mathcal{D}(t) = (\mathcal{V}, \mathcal{E}(t))$ where the adversaries satisfy the F-local malicious or crash model. Let $\{t_k\}$ denote the switching times of $\sigma(t)$ and assume that $t_{k+1} - t_k \geq \tau$ for all k. Suppose each normal node updates its value according to ARC-P2 with parameter F. Then, CTRAC is achieved if there exists $t_0 \geq 0$ such that $\mathcal{D}(t)$ is $(2F+1)$-robust, $\forall t \geq t_0$.*

To illustrate these results, consider the 3-robust graph of Figure 40. Recall that this graph cannot generally sustain 2 malicious nodes as specified by the 2-total model; it is not (3,3)-robust. However, under the 1-local model, it can sustain two malicious nodes if the *right* nodes are compromised. For example, nodes 1 and 4 may be compromised under the 1-local model and the normal nodes will still reach consensus. This example illustrates the advantage of the F-local model, where there is no concern about global assumptions. If a digraph is $(2F+1)$-robust, then up to F nodes may be compromised in any node's neighborhood, possibly resulting in more than F malicious nodes in the network (as in this example).

5.4.4 f-Fraction Local Malicious and Crash Models

We proved in Theorem 5.26 that f-fraction robustness is a necessary condition for ARC-P2 with parameter f to achieve CTRAC in time-invariant networks under the f-fraction local crash model. We now show that p-fraction robustness, with $p > 2f$, is sufficient.

Theorem 5.31 (Sufficient Condition for ARC-P2 under the f-Fraction Local Malicious and Crash Model)**.** *Consider a time-invariant network modeled by digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ where the adversaries satisfy the f-fraction local malicious or crash model. Suppose each normal node updates its value according to ARC-P2 with parameter f. Then, CTRAC is achieved if the network topology is p-fraction robust, where $2f < p \leq 1$.*

Proof. The proof follows the same argument as the proof of Theorems 5.27 and 5.29. In this case, the definitions of $\mathcal{X}_M^k$ and $\mathcal{X}_m^k$ are modified to include only normal nodes. Therefore, as in Theorem 5.29, the uniform continuity assumption is not needed. Then, the p-fraction robust assumption (with $p > 2f$) ensures that there exists a (normal) node in either $\mathcal{X}_M^k$ or $\mathcal{X}_m^k$ with at least $\lceil p d_i \rceil$ neighbors outside of either $\mathcal{X}_M^k$ or $\mathcal{X}_m^k$, respectively. Here, at most $2\lfloor f d_i \rfloor$ of these values are thrown away (with at most $\lfloor f d_i \rfloor$ of them as adversaries, under the f-fraction local model, and at most $\lfloor f d_i \rfloor$ of these strictly smaller, or larger, than node i's value). Because $p > 2f$, it follows that $\lceil p d_i \rceil - 2\lfloor f d_i \rfloor \geq 1$. Therefore, at least one normal value outside of i's set (either $\mathcal{X}_M^k$ or $\mathcal{X}_m^k$) is used. The rest of the analysis is identical to the proof of Theorem 5.29 (which follows Theorem 5.27). □

As with the other adversary models, we may state the following result for time-varying networks.

Theorem 5.32 (Sufficiency with f-Fraction Local Malicious Model in Time-Varying Networks)**.** *Consider a time-varying network modeled by $\mathcal{D}(t) = (\mathcal{V}, \mathcal{E}(t))$ where the adversaries satisfy the f-fraction local malicious or crash model. Let $\{t_k\}$ denote the switching times of $\sigma(t)$ and assume that $t_{k+1} - t_k \geq \tau$ for all k. Suppose each normal node updates its value according to ARC-P2 with parameter F. Then, CTRAC is achieved if there exists $t_0 \geq 0$ such that $\mathcal{D}(t)$ is p-fraction robust, where $2f < p \leq 1$, for all $t \geq t_0$.*

5.4.5 F-Total, F-Local and f-Fraction Local Byzantine Models

The results so far have focused on malicious and crash adversaries. In this section, we consider Byzantine adversaries. Here, we state the results with regard to the normal network, defined as follows.

Definition 5.33 (Normal Network)**.** *For a network $\mathcal{D} = (\mathcal{V}, \mathcal{E})$, define the normal network of $\mathcal{D}$, denoted by $\mathcal{D}_\mathcal{N}$, as the network induced by the normal nodes, i.e., $\mathcal{D}_\mathcal{N} = (\mathcal{N}, \mathcal{E}_\mathcal{N})$, where $\mathcal{E}_\mathcal{N}$ contains those directed edges whose tail and head are both normal nodes.*

Theorem 5.34 (Necessary and Sufficient Condition with F-Total Byzantine Model)**.** *Consider a time-invariant network modeled by digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ where the adversaries satisfy the F-total Byzantine model. Suppose each normal node updates its value according to ARC-P2 with parameter F. Then, CTRAC is achieved if and only if the topology of the normal network is $(F+1)$-robust.*

Proof. Necessity follows from an analogous argument as the proof of Theorem 5.25 whenever there are no Byzantine nodes present (i.e., the normal network *is* the network).

Sufficiency follows the same argument of Theorem 5.29, where $\mathcal{X}_M^k$ and $\mathcal{X}_m^k$ are modified to include only normal nodes. Until one of these sets is empty, there will always be a normal value used by one of the nodes in each, which facilitates an argument following that of Theorem 5.27. As in Theorem 5.29, uniform continuity is not needed since $\mathcal{X}_M^k$ and $\mathcal{X}_m^k$ include only normal nodes. □

The following results are straightforward extensions of the above result to the local models and time-varying networks.

Theorem 5.35 (Necessary and Sufficient Condition with F-Local Byzantine Model)**.** *Consider a time-invariant network modeled by digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ where the adversaries satisfy the F-local Byzantine model. Suppose each normal node updates its value according to ARC-P2 with parameter F. Then, CTRAC is achieved if and only if the topology of the normal network is $(F+1)$-robust.*

Proof. The proof is identical to the proof of Theorem 5.34. □

Theorem 5.36 (Necessary and Sufficient Condition with f-Fraction Local Byzantine Model)**.** *Consider a time-invariant network modeled by digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ where the adversaries satisfy the f-fraction local Byzantine model. Suppose each normal node updates its value according to ARC-P2 with parameter f. Then, CTRAC is achieved if the normal network is p-fraction robust, where $p > f$, and a necessary condition is for the normal network to be f-fraction robust.*

Proof. Necessity follows from Theorem 5.26. For sufficiency, the definitions of $\mathcal{X}_M^k$ and $\mathcal{X}_m^k$ are modified from the proof of Theorem 5.27 to include only normal nodes. Therefore, as in Theorem 5.29, the uniform continuity assumption is not needed. Then, the p-fraction robust assumption (with $p > f$) ensures that there exists a normal node in either $\mathcal{X}_M^k$ or $\mathcal{X}_m^k$ with at least $\lceil pd_i \rceil$ normal neighbors outside of either $\mathcal{X}_M^k$ or $\mathcal{X}_m^k$, respectively. Since $\lceil pd_i \rceil - \lfloor fd_i \rfloor \geq 1$, we can use a similar argument as in the proof of Theorem 5.27 to show that regardless of whether this normal value is used, either $\mathcal{X}_M^{k+1}$ or $\mathcal{X}_M^{k+1}$ must decrease by at least one. Thus, the same induction argument holds to show the contradiction. □

Theorem 5.37 (Sufficiency with F-Total, F-Local, and f-Fraction Local Byzantine Model in Time–Varying Networks)**.** *Consider a time-varying network modeled by $\mathcal{D}(t) = (\mathcal{V}, \mathcal{E}(t))$ where the adversaries satisfy either the (i) F-total or F-local, or (ii) f-fraction local Byzantine model. Let $\{t_k\}$ denote the switching times of $\sigma(t)$ and assume that $t_{k+1} - t_k \geq \tau$ for all k. Suppose each* *normal node updates its value according to ARC-P2 with parameter (i) F or (ii) f. Then, CTRAC is achieved if there exists $t_0 \geq 0$ such that $\mathcal{D}(t)$ is (i) $(2F+1)$-robust or (ii) p-fraction robust, where $2f < p \leq 1$, for all $t \geq t_0$.*

Proof. The time-varying arguments required in Theorems 5.30 and 5.32 are sufficient for the Byzantine case as well. $\square$

5.5 Simulation Example

This section presents a numerical example to illustrate the F-total crash model. In this example, the network is given by the (2,2)-robust graph shown in Figure 43. To verify that this graph is (2,2)-robust one must exhaustively check every nonempty, disjoint pair of subsets of nodes to make sure that either every node in one of the sets has at least 2 neighbors outside of its set, or that there are at least 2 nodes in the union of the subsets that have 2 or more neighbors outside of their respective sets. For example, the pair of sets $\{6\}$ and $\mathcal{V} \setminus \{6\}$ passes this test since all nodes in the first set (just node 6) has at least 2 neighbors outside of its set (in this case just node 6's neighbors). For another example, the pair of sets $\{1, 2, 11, 12\}$ and $\{5, 6\}$ passes since node 11 and node 5 each have 2 or more neighbors outside of their respective sets.

Since the network is (2,2)-robust, it can sustain a single crash node in the network under the 1-total model. Suppose that the node with the largest degree, node 14, is compromised at $t_0 = 0$ so that its value remains constant at $x_{14}(0) = 2$ for all $t \geq 0$. The nodes have continuous dynamics and the normal nodes use either the Linear Consensus Protocol (LCP) given in (50) or ARC-P2 with parameter $F = 1$ for their control input. In either case, the weights are selected to be unity for all neighboring nodes that are kept. The initial values of the nodes are shown in Figure 43 beneath the label of the node's value. The goal of the crash adversary is to drive the values of the normal nodes to its value.

The results for this example are shown in Figure 44. It is clear in Figure 44(a) that the crash node is able to drive the values of the normal nodes to its value of 2 whenever LCP is used. On the other hand, the crash node is unable to achieve its goal whenever ARC-P2 is used. Note that due to the large degree of the crash node, it has the potential to drive the consensus process to any value in the interval $[0, 1]$ by choosing the desired value as its initial value and remaining constant. However,

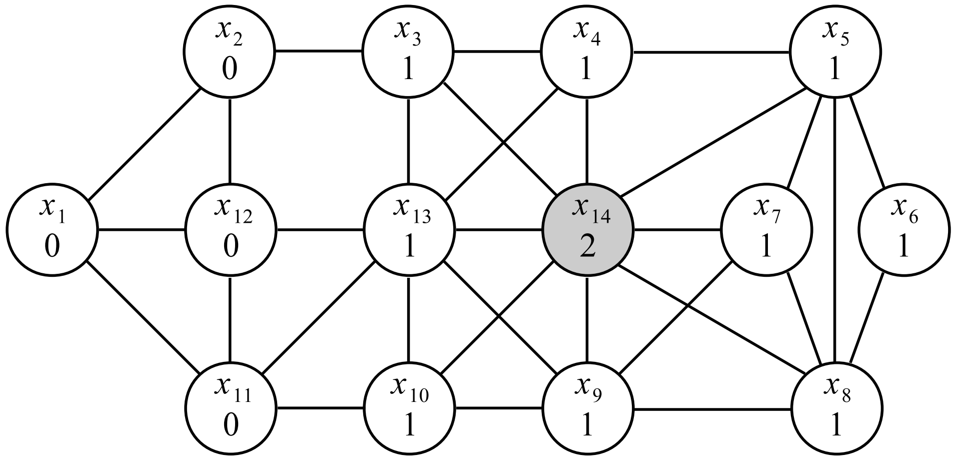
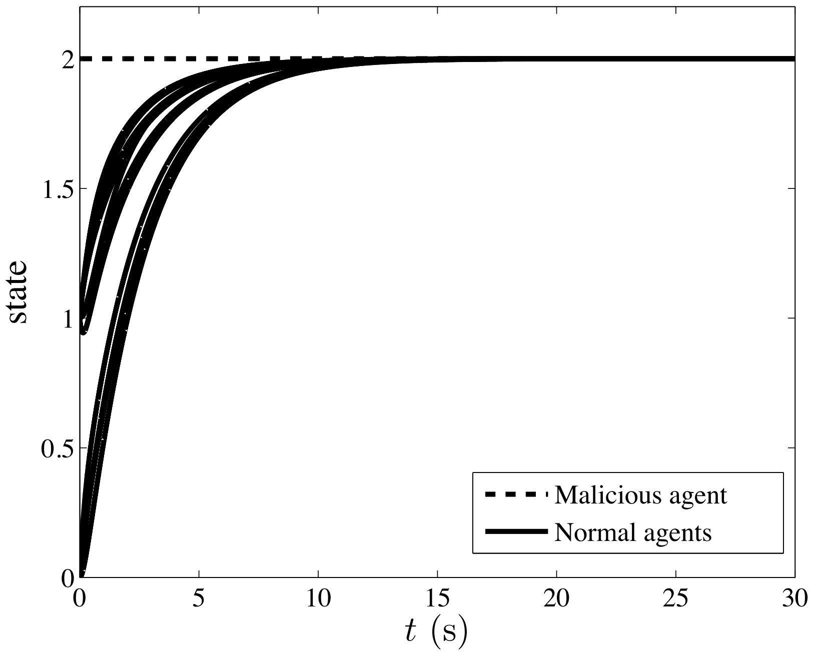


Figure 43: (2,2)-Robust network topology used in CTRAC simulation.

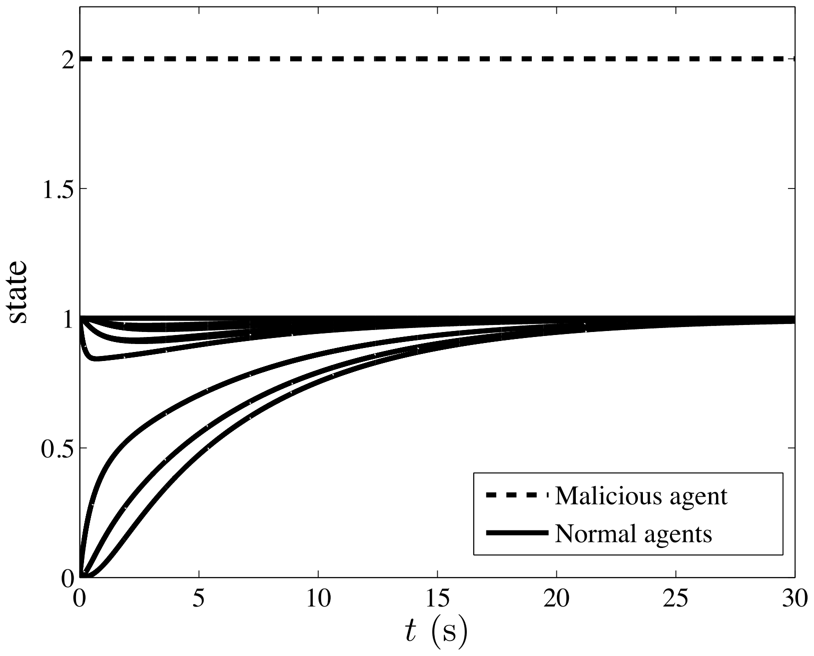
this is allowed with CTRAC (because the consensus value is within the range of the initial values held by normal nodes). Another observation is that the consensus process in the case of ARC-P2 is slower than LCP; this is to be expected, due to the fact that ARC-P2 effectively removes several edges from the network at each time instance, thereby reducing the algebraic connectivity of the Laplacian of the mirror graph [156].

5.6 Summary

The notion of graph connectivity has long been the backbone of investigations into fault tolerant and secure distributed algorithms. Indeed, under the assumption of full knowledge of the network topology, connectivity is *the key* metric in determining whether a fixed number of malicious or Byzantine adversaries can be overcome. However, in large scale systems and complex networks, it is not practical for the various nodes to obtain knowledge of the global network topology. This necessitates the development of algorithms that allow the nodes to operate on purely local information. This chapter continues and extends the work started in [51, 97, 8, 9, 113, 114, 210, 116, 194], and represents a step in this direction for the particular application of distributed consensus. In this chapter, we have presented a modification to ARC-P, which we refer to as ARC-P2. The version of ARC-P studied in this chapter (ARC-P2) has time-varying, piecewise continuous, bounded weights and is more selective in the local filtering process (by using a node's own value as a control on how many values to remove).



(a) LCP.



(b) ARC-P2.

Figure 44: CTRAC simulation of a time-invariant network under the F-total malicious model.

We have also studied the property of network robustness introduced in [210], and we have introduced two extensions to the notion of robustness. The first, referred to as (r, s)-robustness, provides finer granularity in specifying the amount of redundancy of nodes with sufficient local redundancy of information (through the parameter s). The second is a fractional version of robustness (p-fraction robustness) that is well suited to the f-fraction local scope of threat model. Using these various definitions of network robustness, we provide necessary/sufficient conditions for the normal nodes in large-scale networks to mitigate the influence of adversaries. We show that the various versions of network robustness are the appropriate analogues to graph connectivity when considering purely local filtering rules at each node in the network. Just as connectivity has played a central role in the existing analysis of reliable distributed algorithms with global topological knowledge, we believe that robust digraphs (and its variants) will play an important role in the investigation of purely local algorithms.

CHAPTER VI

DISCRETE-TIME RAC IN SYNCHRONOUS NETWORKS

Engineering system design has witnessed a paradigm shift from centralized to distributed, which has been propelled by advances in networking and low-cost, high performance embedded systems. These advances have enabled a transition from end-to-end routing of information in large-scale networked systems to *in-network computation* of aggregate quantities of interest [72]. In-network computing offers certain performance advantages, including reduced latency, less communication overhead, and greater robustness to node and link failures.

A fundamental challenge of in-network computation is that the quantities of interest must be calculated using only *local information*, i.e., information obtained by each node through sensor measurements, calculations, or communication only with neighbors in the network. Another important challenge is that large-scale distributed systems have many potential vulnerable points for failures or attacks. To obtain the desired computational result, it is important to design the in-network algorithms to be able to withstand the compromise of a subset of the nodes and *still ensure some notion of correctness* (possibly at a degraded level of performance). We refer to such a networked system as being *resilient* to adversaries. Given the growing threat of malicious attacks in large-scale cyber-physical systems, this is an important and challenging problem [30].

One of the most important objectives in networked systems is to reach consensus on a quantity of interest [130, 152, 191, 178]. Consensus is fundamental to diverse applications such as data aggregation [95], distributed estimation [172], distributed optimization [192], distributed classification [66], and flocking [90]. Reaching consensus (and more generally, transmitting information) resiliently in the presence of faulty or misbehaving nodes has been studied extensively in distributed computing [110, 130, 165, 16, 61, 17], communication networks [82, 92], and mobile robotics [1, 46, 27]. Among other things, it has been shown that given F (worst-case) adversarial nodes, there exists a strategy for these nodes to disrupt consensus if the network connectivity[42] is $2F$ or less. Conversely, if the network connectivity is at least $2F + 1$, then there exist strategies

[42]The network connectivity is defined as the smaller of the two following values: (i) the size of a minimal vertex cut and (ii) $n - 1$, where n is the number of nodes in the network.

for the *normal* nodes to use that ensure consensus is reached (under the local broadcast model of communication) [130, 181, 159]. However, these consensus algorithms either require that normal nodes have at least some nonlocal information (e.g., knowledge of multiple independent paths in the network between themselves and other nodes) or assume that the network is *complete*, i.e., all-to-all communication or sensing [110, 113, 1, 46, 27]. Moreover, these algorithms tend to be computationally expensive. Therefore, there is a need for resilient consensus algorithms that are *low complexity* and *operate using only local information* (i.e., without knowledge of the network topology and the identities of non-neighboring nodes). A key challenge is to characterize fundamental topological properties that allow the normal nodes to compute an appropriate consensus value in an in-network manner, despite the influence of misbehaving nodes.

The faulty or misbehaving nodes can be characterized by threat models and scope of threat assumptions. Examples of fault or threat models include *non-colluding* [159], *malicious* [159, 181, 113], *Byzantine* [110, 1, 114, 194], or *crash* [1, 46] nodes. Typically, the scope of the faults or threats is assumed to be bounded by a constant, i.e., at most F out of n nodes fail or are compromised. We refer to this as the *F-total model*. Alternatively, the scope may be local; e.g., at most F neighbors of any normal node fail (*F-local model*), or at most a fraction f of neighbors are compromised (*f-fraction local model*).

6.1 Previous Work on Resilient Consensus With Only Local Information

In [51], the authors introduced the *Approximate Byzantine Consensus* problem, in which the normal nodes are required to achieve *approximate agreement*[43] (i.e., they should converge to a relatively small convex hull contained in their initial values) in the presence of F-total Byzantine faults in finite time. They consider only complete networks (where there is a direct connection between every pair of nodes), and they propose the following algorithm: each node disregards the largest and smallest F values received from its neighbors and updates its state to be the average of a carefully chosen subset of the remaining values. This algorithm was extended to a family of algorithms, named the *Mean-Subsequence-Reduced (MSR)* algorithms, in [98]. Although the research on Approximate Byzantine Consensus for complete networks is mature, there are few papers that have

[43]If the network is synchronous, and if one allows $t \to \infty$, then approximate agreement is equivalent to asymptotic consensus.

attempted to analyze this algorithm in more general topologies [97], and even then, only certain special networks have been investigated.

Recently, we have studied resilient algorithms in the presence of misbehaving nodes. In [113], we propose a continuous-time variation of the MSR algorithms, named the *Adversarial Robust Consensus Protocol (ARC-P)*, to solve asymptotic consensus under the F-total malicious model. The results of [113] are extended to both malicious and Byzantine threat models in networks with constrained information flow and dynamic network topology in [114]. The sufficient conditions studied in [114] are stated in terms of in-degrees and out-degrees of nodes in the network and are shown to be sharp; i.e., if the conditions are relaxed, even minimally, then there are examples in which the relaxed conditions are not sufficient. In [210], we generalize the MSR algorithm as the *Weighted-Mean-Subsequence-Reduced (W-MSR)* algorithm and study general distributed algorithms with F-local malicious adversaries.

In a recent paper, developed independently of our work, Vaidya *et al.* have characterized tight conditions for resilient consensus using the MSR algorithm whenever the threat model is Byzantine and the scope is F-total [194]. The network constructions used in [194] are very similar to the robust digraphs presented here. In particular, the networks in [194] also require redundancy of direct information exchange between subsets of nodes in the network.

In contrast to the deterministic approach taken here, gossip algorithms have been studied for in-network computation of aggregate functions such as sums, averages, and quantiles [95]. In such algorithms, each node chooses at random a single neighbor to communicate with in each round. This scheme limits the required computational, communication, and energy resources, and provides some robustness against time-varying topologies and random node and link failures [28]. However, we are not aware of any work that studies the resilience of gossip-based algorithms to malicious attacks.

This chapter continues our study of resilient consensus. Specifically, we study the resilient asymptotic consensus problem in discrete-time synchronous networks, referred to as DTRAC. For DTRAC, we study W-MSR and demonstrate similar results for discrete time as we showed in Chapter V for continuous time.

Organization

The rest of the chapter is organized as follows. Section 6.2 introduces the model of the synchronous multi-agent network, the update model for the class of synchronous algorithms considered, the problem statement, and the adversary models within the synchronous framework. Section 6.3 presents the W-MSR algorithm, and compares it to related algorithms in the literature. The main results are given in Section 6.4. A simulation example is presented in Section 6.5. Finally, a summary is given in Section 6.6.

6.2 System Model and Problem Statement

6.2.1 Synchronous Multi-Agent Network Model

Consider a time-varying network modeled by $\mathcal{D}(t) = (\mathcal{V}, \mathcal{E}(t))$, where $\mathcal{V} = \{1, 2, \ldots, n\}$ is the *node set* and $\mathcal{E}(t) \subset \mathcal{V} \times \mathcal{V}$ is the *directed edge set* at time t. Each directed edge $(j, i) \in \mathcal{E}(t)$ models *information flow* and indicates that node i can be influenced by (or receive information from) node j at time t. Without loss of generality, the node set is comprised of a nonempty set of N *normal nodes*, $\mathcal{N} = \{1, 2, \ldots, N\}$, and a set of M *adversary nodes*, $\mathcal{A} = \{N+1, N+2, \ldots, n\}$, such that $\mathcal{V} = \mathcal{N} \cup \mathcal{A}$ and $M = n - N$. Let $\Gamma_n = \{\mathcal{D}_1, \ldots, \mathcal{D}_d\}$ denote the set of all digraphs on n nodes, which is of course a finite set. Note that $\mathcal{D}(t) \in \Gamma_n$ for all $t \in \mathbb{Z}_{\geq 0}$.

The time-varying topology of the network is governed by a piecewise constant switching signal $\sigma(\cdot)$, which is defined on $\mathbb{Z}_{\geq 0}$ and takes values in $\{1, \ldots, d\}$. In order to emphasize the role of the switching signal, we denote $\mathcal{D}_{\sigma(t)} = \mathcal{D}(t)$. Note that time-invariant networks are represented by defining $\mathcal{D}_{\sigma(t)} \equiv \mathcal{D}_s$, or by simply dropping the dependence on time t.

Suppose that each node $i \in \mathcal{N}$ begins with some scalar value $x_i(0) \in \mathbb{R}$ called the initial value of node i. The execution of the nodes proceeds in a sequence of *rounds*, or *time steps* $t \in \mathbb{Z}_{\geq 0}$, in which each normal node conveys its current value (or state) $x_i(t) \in \mathbb{R}$ to its out-neighbors, perceives the values of its in-neighbors, and updates its value. The network is synchronous in the sense that all normal nodes execute rounds at a uniform rate and they are perfectly synchronized (which justifies the common notion of time $t \in \mathbb{Z}_{\geq 0}$). The network is assumed to be reliable (meaning all conveyed values are perceived without distortion or noise), and values are conveyed instantaneously. Each node must convey at most one value (but not necessarily the same value) to each of its out-neighbors in any given round. Each normal node $i \in \mathcal{N}$ conveys the same value $x_i(t)$ to each of its out-neighbors in round t.

The value $x_i(t) \in \mathbb{R}$ of node $i \in \mathcal{V}$ at time $t \in \mathbb{Z}_{\geq 0}$ defines unambiguously the state of node i at time t for any node that is not deceptive (e.g., a normal node or malicious adversary). However, in order to handle the deceptive Byzantine adversaries, we let $x_{(j,i)}(t)$ denote the state of node j *intended* for node i at time t. Note that even if $(j,i) \notin \mathcal{E}(t)$, $x_{(j,i)}(t)$ is still defined. In the case that $j \in \mathcal{N}$ is normal, we define $x_{(j,i)}(t) \equiv x_j(t)$. On the other hand, if $j \in \mathcal{A}$ is an adversary, then $x_{(j,i)}(t)$ is the state trajectory that adversary j would like to convey to node i, but the topological constraints on the network prevent it from doing so. With this terminology, we denote the collective states of all nodes in $\mathcal{N}$, $\mathcal{A}$, and $\mathcal{V}$ intended for agent i by

$$x_{(\mathcal{N},i)}(t) = [x_{(1,i)}(t), \ldots, x_{(N,i)}(t)]^{\mathsf{T}} = [x_1(t), \ldots, x_N(t)]^{\mathsf{T}} \in \mathbb{R}^N,$$

$$x_{(\mathcal{A},i)}(t) = [x_{(N+1,i)}(t), \ldots, x_{(n,i)}(t)]^{\mathsf{T}} \in \mathbb{R}^M,$$

and

$$x_{(\mathcal{V},i)}(t) = [x_{(1,i)}(t), \ldots, x_{(n,i)}(t)]^{\mathsf{T}} \in \mathbb{R}^n,$$

respectively. Since $x_{(\mathcal{N},i)}(t) \equiv x_{(\mathcal{N},j)}(t)$ for all $i, j \in \mathcal{V}$, we unambiguously define $x_{\mathcal{N}}(t) = x_{(\mathcal{N},i)}(t)$ for any $i \in \mathcal{V}$. Finally, we denote the vector containing all adversary states intended for normal nodes by $x_{(\mathcal{A},\mathcal{N})}(t) = [x^{\mathsf{T}}_{(\mathcal{A},1)}(t), \ldots, x^{\mathsf{T}}_{(\mathcal{A},N)}(t)]^{\mathsf{T}} \in \mathbb{R}^{MN}$.

6.2.2 Synchronous Update Model

At the beginning of each time step $t \in \mathbb{Z}_{\geq 0}$, each normal node i conveys its value to its out-neighbors in the network. Once each normal node i conveys its value, it then collects the values perceived into a multiset. If the network is time invariant, then we assume the nodes are aware of their in-degrees. Therefore, if an adversary k chooses not to transmit a value to a normal out-neighbor i in round t, the normal node i selects some (possibly predetermined) value to use for $x_{(k,i)}(t)$ in the update of round t. If the network is time varying, then the nodes are not aware of their in-neighbors for round t prior to the round. Hence, if an adversary k chooses not to transmit a value to a normal node i in round t (but otherwise could), then by convention we say $i \notin \mathcal{N}_k^{\text{out}}(t)$ and $k \notin \mathcal{N}_i^{\text{in}}(t)$. The multiset of perceived (and possibly fabricated) values is then $\{x_{(j,i)}(t)\}$, $j \in \mathcal{N}_i^{\text{in}}(t)$. At the end of round $t \in \mathbb{Z}_{\geq 0}$, normal node i updates its values for round $t+1$ according to the prescribed rule

$$x_i(t+1) = f_{i,\sigma(t)}(t, x_i(t), \{x_{(j,i)}(t)\}), \;\; i \in \mathcal{N}, j \in \mathcal{N}_i^{\text{in}}(t), t \in \mathbb{Z}_{\geq 0}, \mathcal{D}_{\sigma(t)} \in \Gamma_n. \tag{59}$$

The update rule $f_{i,\sigma(t)}(\cdot)$ can be an arbitrary function, and may be different for each node, depending on its role in the network. These functions are designed *a priori* with the intent that the normal nodes reach consensus (perhaps on a function of interest). However, some of the nodes may not follow the prescribed strategy if they are compromised by an adversary. Such adversary nodes threaten the group objective, and it is important to design the $f_{i,\sigma(t)}(\cdot)$'s in such a way that the influence of such nodes can be eliminated or reduced without prior knowledge about their identities.

6.2.3 Adversaries in Discrete-Time Synchronous Networks

The discrete-time synchronous adversary model consists of a threat model (either Byzantine, malicious, or crash model) and scope of threat assumption (either F-total, F-local, or f-fraction local model). The scope of threat models defined in discrete time are completely analogous to the definitions in continuous time, so they are not repeated here.[44] However, there are some nontrivial differences in the definitions of the threat models, which are discussed here. Recall that for continuous time, we require that if $k \in \mathcal{J}_i^{\text{in}}(t)$, then agent k conveys its state to agent i at time t. This requirement implies that if the network is time invariant and $k \in \mathcal{J}_i^{\text{in}}$, then agent k *must* convey its state to agent i for all $t \in \mathbb{R}_{\geq 0}$.

In time-invariant,[45] discrete-time networks we allow for the possibility that adversary node k does not convey its value to node i whenever $k \in \mathcal{J}_i^{\text{in}}$ in any round of its choosing. This possibility is handled in the update rule of normal node i by substituting a value in place of the absent one in its multiset $\{x_{(j,i)}(t)\}$, $j \in \mathcal{N}_i^{\text{in}}(t)$, of time step t. On the other hand, we do impose the restriction that at most one value may be conveyed to each out-neighbor in a given round. This restriction distinguishes the threat models studied here from flooding attacks (e.g., denial-of-service attacks). The assumption may be reasonable if, for instance, the time length of the round is too short to

[44]The F-total model is identical. The only difference in the F-local and f-fraction local models is that $t \in \mathbb{Z}_{\geq 0}$ instead of $t \in \mathbb{R}_{\geq 0}$.

[45]Observe that with time-varying networks, the time-varying nature of the directed edges encapsulates the ability of an adversary node to avoid conveying its value to would-be out-neighbors.

reliably convey multiple values to a single out-neighbor.[46]

6.2.4 Problem Statement

The *discrete-time resilient asymptotic consensus* (DTRAC) problem in synchronous networks is similar to its continuous-time counterpart and is defined as follows. Let $M_{\mathcal{N}}(t)$ and $m_{\mathcal{N}}(t)$ be the *maximum* and *minimum* values of the normal nodes at time t, respectively. As in continuous time, we also consider the Lyapunov certificate $\Psi(t) = M_{\mathcal{N}}(t) - m_{\mathcal{N}}(t)$.

Definition 6.1 (Discrete-Time Resilient Asymptotic Consensus in Synchronous Networks)**.** *The normal nodes are said to achieve* ***discrete-time resilient asymptotic consensus*** *(DTRAC) in the presence of adversary nodes (given a particular adversary model) if*

(i) $lim_{t\to\infty}\Psi(t) = 0,$ *and*

(ii) $x_i(t+1) \in \mathcal{I}_t = [m_{\mathcal{N}}(t), M_{\mathcal{N}}(t)]$ *for all* $t \in \mathbb{Z}_{\geq 0}$, $i \in \mathcal{N}$,

for any choice of initial values $x_i(0) \in \mathbb{R}$ *for* $i \in \mathcal{N}$.

A distributed algorithm that follows an update rule of the form given by (59) is said to be a *successful* DTRAC algorithm with respect to $\mathcal{D}_{\sigma(t)}$ if it achieves DTRAC in the sequence of digraphs determined by $\sigma(t)$.

The two conditions of the DTRAC problem have several important implications. First, condition (i) implies that the normal nodes reach consensus asymptotically. This is a condition on *agreement*. Condition (ii) requires that a (successful) update rule $f_{i,\sigma(t)}(\cdot)$ must select the next value so that the interval containing the normal nodes at time t, denoted $\mathcal{I}_t$, does not grow. Observe that if this condition is satisfied, then $\mathcal{I}_{t+1} \subseteq \mathcal{I}_t$. By recursively applying this property, one obtains the condition $\mathcal{I}_t \subseteq \mathcal{I}_0$ for all $t \in \mathbb{Z}_{\geq 0}$. Therefore, the interval $\mathcal{I}_0$ containing the initial values of the normal nodes is an invariant set for $t \geq 0$. This invariance property is a *safety* condition on the values of the normal nodes. It is important when the current estimate of the consensus value is used in a safety critical process and the interval $\mathcal{I}_0$ is known to be safe.

[46]Alternatively, if the directed edges are assumed to be authenticated (meaning the node conveying its value is identifiable to the perceiving node), then under the synchronous network model defined here, an adversary that conveys more than one value could easily be detected. In such case, the adversary is effectively reduced to a flooding node.

Also, observe that if condition (ii) is satisfied then $m_{\mathcal{N}}(t)$ is a nondecreasing function of time, and $M_{\mathcal{N}}(t)$ is a nonincreasing function of time. Together with the agreement condition (i), this implies that the values of the normal nodes converge to a common limit $C \in \mathcal{I}_0$ (called the *consensus value*). The condition $C \in \mathcal{I}_0$ establishes *validity* of the consensus value C. Furthermore, this validity condition (i.e., $C \in \mathcal{I}_0$) establishes the DTRAC problem, like the CTRAC problem, as a partially constrained consensus problem.

The validity condition is reasonable in applications where the consensus value must lie in the range given by the initial values of the normal nodes. For example, consider a sensor network in which real-valued measurements are taken of the environment, and the nodes must reach agreement on the measurement value. If the range of measurements taken by normal sensors is relatively small, it may be the case that any value selected within the range of normal sensors is reasonable to choose as the measurement value. On the other hand, if an adversary node is able to arbitrarily bias the estimate, then an arbitrarily large error may be introduced into the estimation process.

Alternatively, suppose the nodes seek to minimize some (convex) objective function $\sum h_i(\theta)$, where each of the h_i's is a local convex function and θ is the optimization variable. Further, suppose the initial value of each node i is the value of θ that minimizes h_i. Then, any value of θ in the interval $\mathcal{I}_0$ is a convex combination of the estimates, and represents an estimate of the optimal value of θ, within bounded error (where the bound on the error is a function of the size of $\mathcal{I}_0$). However, if the adversary nodes are able to drive the consensus value outside of $\mathcal{I}_0$ at an arbitrary distance from $\mathcal{I}_0$, then the value of the objective function will also be arbitrarily far from its minimum.

6.3 Resilient Synchronous Algorithm

While there are many algorithms that facilitate consensus, a class of *linear iterative algorithms* has attracted significant interest in recent years [152, 168]. In such strategies, each node senses or receives information from its neighbors at each time step, and updates its value according to

$$x_i(t+1) = \sum_{j \in \mathcal{J}_i^{\text{in}}(t)} w_{(j,i)}(t) x_{(j,i)}(t), \tag{60}$$

where $w_{(j,i)}(t)$ is the weight assigned to node j's value received by node i at time step t, and $w_{(i,i)(t)}$ is the weight assigned to its own value at time step t. The above strategy is the so-called *Linear*

Iterative Consensus Protocol (LICP) (see (7) on page 49).

Different conditions have been reported in the literature to ensure asymptotic consensus is reached [205, 166, 144, 90, 191]. In discrete time, it is common to assume that there exists a constant $\alpha \in \mathbb{R}, 0 < \alpha < 1$ such that all of the following conditions hold:

- $w_{(j,i)}(t) = 0$ whenever $j \notin \mathcal{J}_i^{\text{in}}(t), i \in \mathcal{N}, t \in \mathbb{Z}_{\geq 0}$;
- $w_{(j,i)}(t) \geq \alpha, \forall j \in \mathcal{J}_i^{\text{in}}(t), i \in \mathcal{N}, t \in \mathbb{Z}_{\geq 0}$;
- $\sum_{j=1}^{n} w_{(j,i)}(t) = 1, \forall i \in \mathcal{N}, t \in \mathbb{Z}_{\geq 0}$;

where it is assumed that all nodes are normal (i.e., $\mathcal{N} = \mathcal{V}$).

Given these conditions, a necessary and sufficient condition for reaching asymptotic consensus in time-invariant networks is that the digraph has a *rooted out-branching*, also called a *rooted directed spanning tree* [168]. In light of Property 5.14, 1-robustness is an equivalent necessary and sufficient condition. The case of dynamic networks is not quite as straightforward. In this case, under the conditions stated above, a sufficient condition for reaching asymptotic consensus is that there exists a uniformly bounded sequence of contiguous time intervals such that the union of digraphs across each interval has a rooted out-branching [166]. Recently, a more general condition referred to as the *infinite flow property* has been shown to be both necessary and sufficient for asymptotic consensus for a class of discrete-time stochastic models [189]. Finally, the lower bound on the weights is needed because there are examples of asymptotically vanishing weights in which consensus is not reached [127]. Later in this chapter, we show that consensus is reached for a class of networks where only *some* of the weights have a lower bound.

The values of the weights in the update rule affect the rate of consensus. Given a fixed, bidirectional network topology, the selection of the optimal weights in (60) with respect to the speed of the consensus process can be done by solving a semidefinite program (SDP) [205]. However, this SDP is solved at design time with global knowledge of the network topology. A simple suboptimal choice of weights that requires only local information is to let $w_{(j,i)}(t) = 1/(1 + d_i(t))$ for $j \in \mathcal{J}_i^{\text{in}}(t)$.

One problem with the linear update given in (60) is that it is not resilient to adversary nodes. In fact, it was shown in [90, 75] that a single 'leader' node can cause all agents to reach consensus on an arbitrary value of its choosing (potentially resulting in a dangerous situation in physical systems) simply by holding its value constant. Thus the dynamics given by (60) do not facilitate DTRAC for any of the fault models. We now describe a simple modification to the update rule, and then provide a comprehensive characterization of network topologies in which DTRAC is achieved under such dynamics.

6.3.1 Description of the W-MSR Algorithm

At every time step $t \in \mathbb{Z}_{\geq 0}$, each normal node i obtains the values of other nodes in its neighborhood. Given the scope of threat models, there may be at most F_i adversary nodes in the neighborhood of normal node i (where $F_i = F$ with the F-total or F-local models and $F_i = \lfloor f d_i(t) \rfloor$ with the f-fraction local model); however, node i is unsure of which neighbors may be compromised. To ensure that node i updates its value in a safe manner, we consider a protocol where each node removes the extreme values with respect to its own value. More specifically:

1. At each time step t, each normal node i obtains the values of its neighbors. Any value not received (from adversary neighbors in time-invariant networks) is replaced by the normal node's own value, $x_i(t)$. Node i forms a sorted list from the multiset $\{x_{(j,i)}(t)\}, j \in \mathcal{N}_i^{\text{in}}(t)$.

2. If there are less than F_i values strictly larger than its own value, $x_i(t)$, then normal 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 (breaking ties arbitrarily). 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.

3. Let $\mathcal{R}_i(t)$ denote the set of nodes whose values were removed by normal node i in step 2 at time step t. Each normal node i applies the update

$$x_i(t+1) = w_{(i,i)}(t)x_i(t) + \sum_{j \in \mathcal{N}_i^{\text{in}}(t) \setminus \mathcal{R}_i(t)} w_{(j,i)}(t)x_{(j,i)}(t), \tag{61}$$

where the weights $w_{(j,i)}(t)$ satisfy the conditions[47]

[47]In this case, a simple choice for the weights is to let $w_{(j,i)}(t) = 1/(1 + d_i(t) - |\mathcal{R}_i(t)|)$ for $j \in \mathcal{J}_i^{\text{in}}(t) \setminus \mathcal{R}_i(t)$.

- $w_{(j,i)}(t) = 0$ whenever $j \notin \mathcal{J}_i^{\text{in}}(t)$ or $j \in \mathcal{R}_i(t)$, and for all $i \in \mathcal{N}$, $t \in \mathbb{Z}_{\geq 0}$;
- $w_{(j,i)}(t) \geq \alpha$, whenever $j \in \mathcal{J}_i^{\text{in}}(t) \setminus \mathcal{R}_i(t)$, and for all $i \in \mathcal{N}$, $t \in \mathbb{Z}_{\geq 0}$;
- $\sum_{j=1}^{n} w_{(j,i)}(t) = 1, \forall i \in \mathcal{N}, t \in \mathbb{Z}_{\geq 0}$;

As a matter of terminology, we refer to the bound on the number (or fraction) of larger or smaller values that could be thrown away as the *parameter* of the algorithm. The parameter of W-MSR with the F-local and F-total models is F, whereas the parameter with the f-fraction local model is f, and the meaning of the parameter will be clear from the context. The above algorithm is extremely lightweight, and does not require any normal node to have any knowledge of the network topology or of the identities of nodes. The only knowledge assumed is that each normal node is aware of its in-degree whenever the network is fixed (so that the node knows if it needs to substitute its value for missing values in the multiset $\{x_{(j,i)}(t)\}$, $j \in \mathcal{N}_i^{\text{in}}(t)$).

Observe that the set of nodes removed by normal node i, $\mathcal{R}_i(t)$, is possibly time-varying. Hence, even if the underlying network topology is fixed, the W-MSR algorithm effectively induces switching behavior, and can be viewed as the linear update of (60) with a specific rule for state-dependent switching (the rule given in step 2).

6.3.2 Use of Related Algorithms

The W-MSR algorithm is introduced in [210] by Zhang and Sundaram, where it is studied in the presence of F-local malicious nodes. The W-MSR algorithm is a discrete-time version of weighted ARC-P with selective reduce, and is in fact the inspiration for this modification of ARC-P.

The use of other algorithms that remove extreme values and then form an average from a subset of the remaining values have been studied for decades. In [51], functions that perform this type of operation are referred to as *approximation functions*, and both synchronous and asynchronous algorithms are studied that use these approximation functions in complete networks for resilience to F-total Byzantine faults. These approximation functions are used in the family of *Mean-Subsequence-Reduced (MSR)* algorithms [98]. There are a few key differences between the operations used in the W-MSR algorithm and the MSR algorithm of [98]. First, W-MSR does not always remove the largest and smallest F values as in the MSR algorithm [98] (and as in the original form of ARC-P). Instead, only the extreme values that are strictly larger or strictly smaller than the given node's value are removed (as in weighted ARC-P with selective reduce). Since the node's own value may be one of the F extreme values, the MSR algorithm may throw away this useful (correct) information. Second, W-MSR uses all values retained after removing the extreme values. MSR, on the other hand, may select only a subsequence of the remaining values to use in the update. However, because the lower bound on the weights, $\alpha > 0$, may be arbitrarily small, W-MSR can come arbitrarily close to selecting only a subsequence of remaining values by setting the appropriate weights to α (instead of 0 as would be done in MSR). Finally, MSR averages the remaining values instead of allowing for weighted averages as in W-MSR.

MSR algorithms have also been used for Byzantine point convergence of mobile robots in complete networks [27]. Besides Byzantine faults, some works also consider other threat models and a combination of these faults [98]. However, few papers have addressed the convergence of MSR algorithms in less restrictive (non-complete) networks. Some exceptions include [97, 8, 9]. In [97], the authors studied *local convergence* (convergence of a subset of nodes) in undirected regular graphs[48]; the results are extended to asynchronous networks in [9] and global convergence of a class of undirected regular graphs, named *Partially Fully Connected Networks (PFCN)*, in [8]. More recently, [194] provides necessary and sufficient conditions on the network topology required for a special case of the MSR algorithm (which retains all of the values after removing the extreme ones) to achieve consensus in the presence of F-total Byzantine faults.

Finally, it is worth noting the relationship between the W-MSR algorithm and robust consensus algorithms designed to withstand outliers [119, 142]. The problem of robust consensus to outliers does not assume a threat model, such as malicious or Byzantine nodes. Instead, some measurements may be statistical outliers caused by noisy measurements and the goal is to reach consensus on the measurements in a manner that reduces the error introduced by the outliers. In these works the nodes with outlier measurements are cooperative in the consensus process. Therefore, such techniques are not designed to work in the presence of adversary nodes.

6.4 Resilient Consensus Analysis

We start with the following result showing that W-MSR *always* satisfies the safety condition for DTRAC under any of the adversary models considered here. Because the crash and malicious

[48]A regular graph is a graph where each vertex has the same number of neighbors.

adversaries are special cases of the Byzantine adversary, the only threat model we need to consider is the Byzantine model. Recall that $M_{\mathcal{N}}(t)$ and $m_{\mathcal{N}}(t)$ are the maximum and minimum values of the *normal* nodes at time step t, respectively.

Theorem 6.2 (Safety Condition Always Satisfied by W-MSR)**.** *Suppose each normal node updates its value according to the W-MSR algorithm with parameter F under the F-total or F-local Byzantine model, or with parameter f under the f-fraction local Byzantine model. Then, for each node $i \in \mathcal{N}$, $x_i(t+1) \in \mathcal{I}_t = [m_{\mathcal{N}}(t), M_{\mathcal{N}}(t)]$, for all $t \in \mathbb{Z}_{\geq 0}$, regardless of the network topology.*

Proof. Since the update of W-MSR forms a convex combination of the values in $\mathcal{J}_i^{\text{in}}(t) \setminus \mathcal{R}_i(t)$ (which is *never* empty since node i always uses its own value), it is sufficient to show that all of the values in $\mathcal{J}_i^{\text{in}}(t) \setminus \mathcal{R}_i(t)$ must lie in $\mathcal{I}_t$. Define $F_i = F$ if the scope of threat is F-total or F-local, and define $F_i = \lfloor f d_i(t) \rfloor$ if the scope of threat is f-fraction local. Fix $x_{(j,i)}(t) \in \mathcal{J}_i^{\text{in}}(t) \setminus \mathcal{R}_i(t)$ and suppose that $x_{(j,i)}(t) > M_{\mathcal{N}}(t)$. Then, by definition j must be an adversary and $x_{(j,i)}(t) > x_i(t)$. Since i uses $x_{(j,i)}(t)$ in round t, there must be at least F_i more nodes in the neighborhood of i with values at least as large as $x_{(j,i)}(t)$ (otherwise, it would be removed). Hence, these nodes must also be adversaries, which contradicts the assumption that there are at most F_i adversary nodes in the neighborhood of i at time t. Thus, $x_{(j,i)}(t) \leq M_{\mathcal{N}}(t)$. Similarly, we can show that if $x_{(j,i)}(t) \in \mathcal{J}_i^{\text{in}}(t) \setminus \mathcal{R}_i(t)$, then $x_{(j,i)}(t) \geq m_{\mathcal{N}}(t)$. □

Having guaranteed the safety condition, we now provide a characterization of networks in which the agreement condition (and thus, the validity condition) is satisfied for each of the adversary models. First we provide necessary conditions for time-invariant networks formulated in terms of network robustness for the existence of a successful algorithm that solves DTRAC. We then show that the necessary conditions are also sufficient for some of the adversary models by showing that W-MSR achieves DTRAC under the necessary conditions. For the remaining adversary models we provide weaker sufficient conditions.

6.4.1 Necessary Conditions for Time-Invariant Synchronous Networks

In this section, we present a couple of necessary conditions. The first result provides a necessary condition on the network topology for *any* successful DTRAC algorithm with respect to $\mathcal{D}$ under the F-total or F-local crash model. The second result is a necessary condition for the success of W-MSR with parameter f under the f-fraction local model.

In more detail, the following result shows that $(F+1, F+1)$-robustness is a necessary condition for any synchronous distributed algorithm to achieve DTRAC under the F-total or F-local crash model in time-invariant synchronous networks. Since Byzantine and malicious nodes are strictly more expressive than crash adversaries, the result is also necessary with Byzantine and malicious adversaries.

Theorem 6.3 (Necessity with F-Total and F-Local Crash Adversaries)**.** *Consider a time-invariant synchronous network modeled by a digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ in the presence of crash adversaries under the F-total or F-local model. Suppose each normal node i updates its value according to a successful DTRAC algorithm with respect to $\mathcal{D}$. Then, $\mathcal{D}$ is $(F+1, F+1)$-robust.*

Proof. Suppose $\mathcal{D}$ is not $(F+1, F+1)$-robust. Then, there are nonempty, disjoint $\mathcal{S}_1, \mathcal{S}_2 \subset \mathcal{V}$ such that none of the conditions $(i) - (iii)$ in Definition 5.10 hold (with $r = F+1$ and $s = F+1$). Consider two scenarios. In the first scenario (which is the primary scenario of interest), suppose the initial value of each node in $\mathcal{S}_1$ is a and each node in $\mathcal{S}_2$ is b, with $a < b$. Let all other nodes have initial values taken from the interval $[a, b]$. Since $|\mathcal{X}_{\mathcal{S}_1}^{F+1}| + |\mathcal{X}_{\mathcal{S}_2}^{F+1}| \leq F$, suppose all nodes in $\mathcal{X}_{\mathcal{S}_1}^{F+1}$ and $\mathcal{X}_{\mathcal{S}_2}^{F+1}$ are crash nodes (which is allowed under both the F-total and F-local models). The crash adversaries in $\mathcal{X}_{\mathcal{S}_1}^{F+1}$ and $\mathcal{X}_{\mathcal{S}_2}^{F+1}$ keep their values constant at a and b, respectively, and choose to always convey their values. With this assignment of adversaries, there is still at least one normal node in both $\mathcal{S}_1$ and $\mathcal{S}_2$ since $|\mathcal{X}_{\mathcal{S}_1}^{F+1}| < |\mathcal{S}_1|$ and $|\mathcal{X}_{\mathcal{S}_2}^{F+1}| < |\mathcal{S}_2|$, respectively. Therefore, consider a specific normal node $i \in \mathcal{S}_1 \setminus \mathcal{X}_{\mathcal{S}_1}^{F+1}$.

For scenario 2, suppose all of normal node i's neighbors outside of $\mathcal{S}_1$ are crash adversaries (there are at most F such neighbors), they have the same initial values as scenario 1, and they all convey their values (so that all neighbors of i convey their values to i in both scenarios). Let all other nodes (including i) have initial value a. Therefore, scenarios 1 and 2 are identical from the perspective of node i. Furthermore, in scenario 2, the normal nodes are in agreement. Hence, the update rule must select $x_i(1) = a$ in either scenario to ensure the safety condition.[49] In a similar

[49]Note that even a nondeterministic algorithm must choose $x_i(1) = a$ in scenario 1 because if it can choose $x_i(1) \neq a$ in scenario 1, it may do so in scenario 2 (because its inputs and state are identical in each case), which would violate the safety condition.

manner, one can argue that any normal node $j \in \mathcal{S}_2 \setminus \mathcal{X}_{\mathcal{S}_2}^{F+1}$ in scenario 1 must select $x_j(1) = b$. Finally, any normal node k in $\mathcal{V} \setminus (\mathcal{S}_1 \cup \mathcal{S}_2)$ in scenario 1 must set its value $x_k(1) \in [a, b]$ to ensure safety (since $[m_{\mathcal{N}}(0), M_{\mathcal{N}}(0)] = [a, b]$). Therefore, round 1 has the same distribution of values as round 0. By induction, we conclude that for each round $r \in \mathbb{Z}_{\geq 0}$ each node in $\mathcal{S}_1$ has value a, each node in $\mathcal{S}_2$ has value b, and all other nodes have values in $[a, b]$. Therefore, no consensus is achieved, which contradicts the assumption that DTRAC is achieved. Therefore, the network must be $(F + 1, F + 1)$-robust. □

The necessary condition we would like to show for the f-fraction local model is the following. If *any* successful DTRAC algorithm achieves DTRAC in $\mathcal{D}$, then $\mathcal{D}$ is f-robust. However, this necessary condition is not provable using the construction of scenarios 1 and 2 as in the proof of Theorem 6.3. This is because there is no absolute bound on the number of adversaries in any node's neighborhood (rather the bound is fractional, and therefore depends on the degrees of the nodes). It is possible that the bound $\lfloor f d_i \rfloor$ on the number of adversaries in node i's neighborhood may be much larger than the bound allowed for some other node j. If $d_j < d_i$ and j shares the $\lfloor f d_i \rfloor$ or less nodes we would like to assign as adversaries, then it is possible that this assignment of adversaries violates the f-fraction local model. However, for W-MSR with parameter f, we have no problem using a similar argument to Theorem 6.3. This is because each node i will remove up to $\lfloor f d_i \rfloor$ values in its neighborhood that are strictly larger or smaller than its own value.

Theorem 6.4 (Necessary Condition for W-MSR with Parameter f under the f-Fraction Local Crash Model)**.** *Consider a time-invariant synchronous network modeled by a digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ in the presence of crash adversaries under the f-fraction local model. Suppose each normal node i updates its value according to W-MSR with parameter f. If DTRAC is achieved, then $\mathcal{D}$ is f-fraction robust.*

Proof. Suppose that $\mathcal{D}$ is not f-fraction robust. Then, there exists nonempty, disjoint subsets $\mathcal{S}_1, \mathcal{S}_2 \subset \mathcal{V}$ such that neither $\mathcal{S}_1$ nor $\mathcal{S}_2$ is f-fraction edge reachable. This means that for every $i \in \mathcal{S}_k$, $|\mathcal{N}_i^{\text{in}} \setminus \mathcal{S}_k| < \lceil f d_i \rceil$, for $k \in \{1, 2\}$. From this, it follows that $|\mathcal{N}_i^{\text{in}} \setminus \mathcal{S}_k| \leq \lfloor f d_i \rfloor$ for all $i \in \mathcal{S}_k$. Suppose the initial value of each node in $\mathcal{S}_1$ is a and each node in $\mathcal{S}_2$ is b, with $a < b$. Let all other nodes have initial values taken from the interval $[a, b]$. Assume all nodes are normal. Then, using W-MSR with parameter f, each node i in $\mathcal{S}_1$ removes the $\lfloor f d_i \rfloor$ or less values greater than

a from outside $\mathcal{S}_1$. Likewise, each node j in $\mathcal{S}_2$ removes the $\lfloor fd_j \rfloor$ or less values less than b from outside $\mathcal{S}_2$. The remaining nodes select values in $[a, b]$ by Theorem 6.2. By induction, it follows that all nodes in $\mathcal{S}_1$ keep the value a and each node in $\mathcal{S}_2$ keeps the value b for all time steps. Therefore, DTRAC is not achieved. $\square$

6.4.2 F-Total Malicious and Crash Models

The following result is one of the major contributions of the chapter. It provides, for the first time, a *necessary and sufficient* condition on the time-invariant network topology for the W-MSR algorithm to succeed under the F-total malicious model. In light of Theorem 6.3, this result shows that $(F+1, F+1)$-robustness is both necessary and sufficient for the existence of a synchronous algorithm that achieves DTRAC in time-invariant synchronous networks. Note that we use the Lyapunov function $\Psi(t) = M_{\mathcal{N}}(t) - m_{\mathcal{N}}(t)$ in the analysis.

Theorem 6.5 (Sufficiency with F-Total Malicious and Crash Model)**.** *Consider a time-invariant network modeled by a digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ where each normal node updates its value according to the W-MSR algorithm with parameter F. Under the F-total malicious model, DTRAC is achieved if and only if the network topology is $(F+1, F+1)$-robust.*

Proof. Necessity follows from Theorem 6.3. For sufficiency, recall that Ψ is a nonincreasing function of t. Therefore, whenever the normal nodes are in agreement at time t_k, then consensus is maintained for $t \geq t_k$. Hence, fix $t_k \geq 0$ and assume $\Psi(t_k) > 0$.

For $t \geq t_k$ and $\eta > 0$, define $\mathcal{X}_M(t, t_k, \eta) = \{j \in \mathcal{V} \colon x_j(t) > M_{\mathcal{N}}(t_k) - \eta\}$ and $\mathcal{X}_m(t, t_k, \eta) = \{j \in \mathcal{V} \colon x_j(t) < m_{\mathcal{N}}(t_k) + \eta\}$. Define $\epsilon_0 = \Psi(t_k)/2$ and let $\epsilon_j = \alpha\epsilon_{j-1}$ for $j = 1, 2, \ldots, N-1$, where $N = \mathcal{N}$. It follows that $\epsilon_j = \alpha^j \epsilon_0 > 0$. By definition, the sets $\mathcal{X}_M(t_k, t_k, \epsilon_0)$ and $\mathcal{X}_m(t_k, t_k, \epsilon_0)$ are nonempty and disjoint. Because $\mathcal{D}$ is $(F+1, F+1)$-robust and there are at most F malicious nodes in the network (F-total model), it follows that either there exists $i \in \mathcal{X}_M(t_k, t_k, \epsilon_0) \cap \mathcal{N}$ or there exists $i \in \mathcal{X}_m(t_k, t_k, \epsilon_0) \cap \mathcal{N}$, or there exists such i in both, such that i has at least $F+1$ neighbors outside of its set. Therefore, if $i \in \mathcal{X}_M(t_k, t_k, \epsilon_0) \cap \mathcal{N}$ (with at least

$F+1$ neighbors outside its set), then

$$\begin{aligned} x_i(t_k+1) &= w_{(i,i)}(t_k)x_i(t_k) + \sum_{j\in\mathcal{N}_i^{\text{in}}\setminus\mathcal{R}_i(t_k)} w_{(j,i)}(t_k)x_{(j,i)}(t_k) \\ &\le \alpha(M_{\mathcal{N}}(t_k)-\epsilon_0) + (1-\alpha)M_{\mathcal{N}}(t_k) \\ &\le M_{\mathcal{N}}(t_k) - \alpha\epsilon_0 = M_{\mathcal{N}}(t_k) - \epsilon_1. \end{aligned}$$

Note that for any normal node not in $\mathcal{X}_M(t_k,t_k,\epsilon_0)$, the above inequality holds because any normal node always uses its own value in the update. From this, we conclude

$$|\mathcal{X}_M(t_k+1,t_k,\epsilon_1)\cap\mathcal{N}| < |\mathcal{X}_M(t_k,t_k,\epsilon_0)\cap\mathcal{N}|.$$

Similarly, if $i\in\mathcal{X}_m(t_k,t_k,\epsilon_0)\cap\mathcal{N}$ (with at least $F+1$ neighbors outside its set), then

$$\begin{aligned} x_i(t_k+1) &= w_{(i,i)}(t_k)x_i(t_k) + \sum_{j\in\mathcal{N}_i^{\text{in}}\setminus\mathcal{R}_i(t_k)} w_{(j,i)}(t_k)x_{(j,i)}(t_k) \\ &\ge \alpha(m_{\mathcal{N}}(t_k)+\epsilon_0) + (1-\alpha)m_{\mathcal{N}}(t_k) \\ &\ge m_{\mathcal{N}}(t_k) + \alpha\epsilon_0 = m_{\mathcal{N}}(t_k) + \epsilon_1. \end{aligned}$$

Similarly as above, this inequality holds for any normal node not in $\mathcal{X}_m(t_k,t_k,\epsilon_0)$. From this, we conclude

$$|\mathcal{X}_m(t_k+1,t_k,\epsilon_1)\cap\mathcal{N}| < |\mathcal{X}_m(t_k,t_k,\epsilon_0)\cap\mathcal{N}|.$$

By repeating this analysis, we can show by induction that as long as both $\mathcal{X}_M(t_k+j,t_k,\epsilon_j)\cap\mathcal{N}$ and $\mathcal{X}_m(t_k+j,t_k,\epsilon_j)\cap\mathcal{N}$ are both nonempty, then either

$$|\mathcal{X}_M(t_k+j+1,t_k,\epsilon_{j+1})\cap\mathcal{N}| < |\mathcal{X}_M(t_k+j,t_k,\epsilon_j)\cap\mathcal{N}|,$$

or

$$|\mathcal{X}_m(t_k+j+1,t_k,\epsilon_{j+1})\cap\mathcal{N}| < |\mathcal{X}_m(t_k+j,t_k,\epsilon_j)\cap\mathcal{N}|,$$

or both. Since $|\mathcal{X}_M(t_k,t_k,\epsilon_0)\cap\mathcal{N}|+|\mathcal{X}_m(t_k,t_k,\epsilon_0)\cap\mathcal{N}| \le |\mathcal{N}| = N$, there exists $T<N$ such that

one of the sets $\mathcal{X}_M(t_k+T, t_k, \epsilon_T) \cap \mathcal{N}$, $\mathcal{X}_m(t_k+T, t_k, \epsilon_T) \cap \mathcal{N}$, or both, is empty. It follows that in the former case, $M_{\mathcal{N}}(t_k+T) \leq M_{\mathcal{N}}(t_k) - \epsilon_T$, and in the latter case, $m_{\mathcal{N}}(t_k+T) \geq m_{\mathcal{N}}(t_k) + \epsilon_T$. Since

$$\epsilon_0 > \epsilon_1 > \cdots > \epsilon_T \geq \epsilon_{N-1} > 0,$$

we have

$$\begin{aligned}
\Psi(t_k+N-1) - \Psi(t_k) &\leq \Psi(t_k+T) - \Psi(t_k) \\
&\leq (M_{\mathcal{N}}(t_k+T) - M_{\mathcal{N}}(t_k)) + (m_{\mathcal{N}}(t_k) - m_{\mathcal{N}}(t_k+T)) \\
&\leq -\epsilon_T \\
&\leq -\epsilon_{N-1}.
\end{aligned}$$

Therefore, $\Psi(t_k+N-1) \leq \Psi(t_k)(1-\alpha^{N-1}/2)$. Define $\gamma = (1-\alpha^{N-1}/2)$. Since γ is not a function of t_k, and t_k was chosen arbitrarily, it follows that

$$\Psi(t_k + j(N-1)) \leq \gamma^j \Psi(t_k),$$

for all $j \in \mathbb{Z}_{\geq 0}$. Because $\gamma < 1$, it follows that $\Psi(t) \to 0$ as $t \to \infty$. ☐

This result establishes the notion of network robustness introduced in Definition 5.10 as the appropriate metric for reasoning about purely local distributed algorithms, supplanting the traditional metric of connectivity. When the network is time-varying, one can state the following theorem.

Theorem 6.6 (Sufficiency with F-total malicious model in time-varying networks)**.** *Consider a time-varying network modeled by $\mathcal{D}(t) = (\mathcal{V}, \mathcal{E}(t))$ where each normal node updates its value according to the W-MSR algorithm with parameter F. Let $\{t_k\}$ denote the set of time steps in which $\mathcal{D}(t_k)$ is $(F+1, F+1)$-robust. Then, under the F-total malicious model, DTRAC is achieved if*

$$|\{t_k\}| = \infty \quad \textit{and} \; |t_{k+1} - t_k| \leq c, \;\; \forall k,$$

for some $c \in \mathbb{N}$.

Proof. Fix $t_k \in \{t_k\}$ such that $\mathcal{D}(t_k)$ is $(F+1, F+1)$-robust. Assume $\Psi(t_k) > 0$ (otherwise, consensus is maintained for all $t \geq t_k$). Define $\mathcal{X}_M(t, t_k, \eta)$, $\mathcal{X}_m(t, t_k, \eta)$, and $\epsilon_0 = \Psi(t_0)/2$ as in the proof of Theorem 6.5. In this case, define $\epsilon_j = \alpha^c \epsilon_{j-1}$ for $j = 1, 2, \ldots, N-1$, where $N = \mathcal{N}$. Because $\mathcal{D}(t_k)$ is $(F+1, F+1)$-robust and there are at most F malicious nodes in the network (F-total model), it follows that either there exists $i \in \mathcal{X}_M(t_k, t_k, \epsilon_0) \cap \mathcal{N}$ or there exists $i \in \mathcal{X}_m(t_k, t_k, \epsilon_0) \cap \mathcal{N}$, or there exists such i in both, such that i has at least $F+1$ neighbors outside of its set. As in the proof of Theorem 6.5, we can show in the former case ($i \in \mathcal{X}_M(t_k, t_k, \epsilon_0) \cap \mathcal{N}$ with at least $F+1$ neighbors outside its set) that

$$x_i(t_k + 1) \leq M_{\mathcal{N}}(t_k) - \alpha\epsilon_0.$$

Because node i always uses its own value in its update, we can show that if $x_i(t_k + j) \leq M_{\mathcal{N}}(t_k + j) - \alpha^j \epsilon_0$, then

$$\begin{aligned} x_i(t_k + j + 1) &= w_{(i,i)}(t_k + j) x_i(t_k + j) + \sum_{j \in \mathcal{N}_i^{\text{in}} \setminus \mathcal{R}_i(t_k+j)} w_{(j,i)}(t_k + j) x_{(j,i)}(t_k + j) \\ &\leq \alpha(M_{\mathcal{N}}(t_k + j) - \alpha^j \epsilon_0) + (1 - \alpha) M_{\mathcal{N}}(t_k + j) \\ &\leq M_{\mathcal{N}}(t_k) - \alpha^{j+1}\epsilon_0, \end{aligned}$$

even in time steps when the network is not $(F+1, F+1)$-robust. By induction, it follows that $x_i(t_k + j) \leq M_{\mathcal{N}}(t_k) - \alpha^j \epsilon_0$ for all $j \in \mathbb{Z}_{\geq 0}$. Note that this inequality holds as well for any normal node $i \notin \mathcal{X}_M(t_k, t_k, \epsilon_0)$ (since it uses its own value). In a similar manner, we can show in the latter case ($i \in \mathcal{X}_m(t_k, t_k, \epsilon_0) \cap \mathcal{N}$ with at least $F+1$ neighbors outside its set) that $x_i(t_k + j) \geq m_{\mathcal{N}}(t_k) + \alpha^j \epsilon_0$ for all $j \in \mathbb{Z}_{\geq 0}$ (and this holds as well for $i \notin \mathcal{X}_m(t_k, t_k, \epsilon_0)$).

By hypothesis, there exists $t_{k+1} \in \{t_k+1, t_k+2, \ldots, t_k+c\}$ such that $\mathcal{D}(t_{k+1})$ is $(F+1, F+1)$-robust. Since $|t_{k+1} - t_k| \leq c$, at least one of the inequalities

$$|\mathcal{X}_M(t_{k+1}, t_k, \epsilon_1) \cap \mathcal{N}| < |\mathcal{X}_M(t_k, t_k, \epsilon_0) \cap \mathcal{N}|,$$

or

$$|\mathcal{X}_m(t_{k+1}, t_k, \epsilon_1) \cap \mathcal{N}| < |\mathcal{X}_m(t_k, t_k, \epsilon_0) \cap \mathcal{N}|$$

holds. By repeating this analysis, we can show by induction that as long as both $\mathcal{X}_M(t_{k+j}, t_k, \epsilon_j) \cap \mathcal{N}$ and $\mathcal{X}_m(t_{k+j}, t_k, \epsilon_j) \cap \mathcal{N}$ are both nonempty, then either

$$|\mathcal{X}_M(t_{k+j+1}, t_k, \epsilon_{j+1}) \cap \mathcal{N}| < |\mathcal{X}_M(t_{k+j}, t_k, \epsilon_j) \cap \mathcal{N}|,$$

or

$$|\mathcal{X}_m(t_{k+j+1}, t_k, \epsilon_{j+1}) \cap \mathcal{N}| < |\mathcal{X}_m(t_{k+j}, t_k, \epsilon_j) \cap \mathcal{N}|,$$

or both. Since $|\mathcal{X}_M(t_k, t_k, \epsilon_0) \cap \mathcal{N}| + |\mathcal{X}_m(t_k, t_k, \epsilon_0) \cap \mathcal{N}| \leq |\mathcal{N}| = N$, there exists $T < N$ such that one of the sets $\mathcal{X}_M(t_{k+T}, t_k, \epsilon_T) \cap \mathcal{N}$, $\mathcal{X}_m(t_{k+T}, t_k, \epsilon_T) \cap \mathcal{N}$, or both, is empty. It follows that in the former case, $M_{\mathcal{N}}(t_{k+T}) \leq M_{\mathcal{N}}(t_k) - \epsilon_T$, and in the latter case, $m_{\mathcal{N}}(t_{k+T}) \geq m_{\mathcal{N}}(t_k) + \epsilon_T$. Since

$$\epsilon_0 > \epsilon_1 > \cdots > \epsilon_T \geq \epsilon_{N-1} > 0,$$

we have

$$\begin{aligned}
\Psi(t_{k+N-1}) - \Psi(t_k) &\leq \Psi(t_{k+T}) - \Psi(t_k) \\
&\leq (M_{\mathcal{N}}(t_{k+T}) - M_{\mathcal{N}}(t_k)) + (m_{\mathcal{N}}(t_k) - m_{\mathcal{N}}(t_{k+T})) \\
&\leq -\epsilon_T \\
&\leq -\epsilon_{N-1}.
\end{aligned}$$

Therefore, $\Psi(t_{k+N-1}) \leq \Psi(t_k)(1 - \alpha^{c(N-1)}/2)$. Define $\gamma = (1 - \alpha^{c(N-1)}/2)$. Since γ is not a function of t_k, $|\{t_k\}| = \infty$, $|t_{k+1} - t_k| \leq c$ for any $t_k \in \{t_k\}$, and t_k was chosen arbitrarily in $\{t_k\}$, it follows that

$$\Psi(t_k + jc(N-1)) \leq \Psi(t_{k+j(N-1)}) \leq \gamma^j \Psi(t_k),$$

for all $j \in \mathbb{Z}_{\geq 0}$. Because $\gamma < 1$, it follows that $\Psi(t) \to 0$ as $t \to \infty$. □

To illustrate these results consider the graphs in Figures 35 and 40 (on pages 159 and 165, respectively). These graphs can withstand the compromise of $F = 1$ malicious node in the network using the W-MSR algorithm with parameter $F = 1$ (each graph is (2,2)-robust but not (3,3)-robust). This is not to say that it is impossible for the normal nodes to reach consensus if there are, for example, two nodes that are compromised. Instead, these results say that there are two *specific* nodes that can be compromised by an adversary to prevent consensus (e.g., nodes 5 and 6 in Figure 40).

6.4.3 F-Local Malicious and Crash Models

The previous results (Theorems 6.3 and 6.5) fully characterize those time-invariant synchronous network topologies that are able to handle F-total malicious (or crash) adversaries. In order to deal with the possibility that the total number of adversaries is quite large, we now consider the F-local and f-fraction local malicious models. Since there is no upper bound on the total number of adversaries under these models, we cannot rely on a "sufficient" number of nodes in each set having enough neighbors outside. Therefore, the parameter s in the definition of (r, s)-robust has no impact. Instead, we use r-robustness (and p-fraction robustness), and increase the lower bound on the number of neighbors in the edge reachability properties (i.e., the value of r or p).

The following result shows that $(2F+1)$-robustness is a sufficient condition under the F-local malicious model (and therefore also with crash adversaries) to ensure DTRAC is achieved in time-invariant synchronous networks. This result is proved by Zhang and Sundaram in [210] using a contradiction argument. We provide here an alternative proof that provides a minimum guaranteed performance bound (i.e., geometric decay of the Lyapunov certificate $\Psi(t)$ over large enough time intervals).

Theorem 6.7 ([210], Sufficiency with F-Local Malicious Model)**.** *Consider a time-invariant network modeled by a digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ where each normal node updates its value according to the W-MSR algorithm with parameter F. Under the F-local malicious model, DTRAC is achieved if the topology of the network is $(2F+1)$-robust.*

Proof. The proof follows the same argument as the proof of Theorem 6.5. In this case, the sets $\mathcal{X}_M(t, t_k, \eta)$ and $\mathcal{X}_m(t, t_k, \eta)$ are defined to include only normal nodes. Then, the $(2F+1)$-robust assumption ensures that there exists a node in either $\mathcal{X}_M$ or $\mathcal{X}_m$ with at least $2F+1$ neighbors outside of either $\mathcal{X}_M$ or $\mathcal{X}_m$, respectively. Since at most $2F$ of these values are thrown away (with at most F of them as adversaries, under the F-local model, and at most F of these strictly smaller, or larger, than node i's value), it follows that at least one normal value outside of i's set (either $\mathcal{X}_M$ or $\mathcal{X}_m$) will be used in the update. The rest of the analysis is identical to the proof of Theorem 6.5.

Therefore, we can show that for $\gamma = (1 - \alpha^{N-1}/2)$ and any $t_k \in \mathbb{Z}_{\geq 0}$

$$\Psi(t_k + j(N-1)) \leq \gamma^j \Psi(t_k),$$

for all $j \in \mathbb{Z}_{\geq 0}$. Because $\gamma < 1$, it follows that $\Psi(t) \to 0$ as $t \to \infty$. □

Following the argument of Theorem 6.6 with the modifications described in the proof of Theorem 6.7, we can show the following result for time-varying networks.

Theorem 6.8 ([210], Sufficiency with F-Local Malicious Model in Time-Varying Networks)**.** *Consider a time-varying network modeled by $\mathcal{D}(t) = (\mathcal{V}, \mathcal{E}(t))$ where each normal node updates its value according to the W-MSR algorithm with parameter F. Let $\{t_k\}$ denote the set of time steps in which $\mathcal{D}(t)$ is $(2F+1)$-robust. Then, under the F-local malicious model, DTRAC is achieved if*

$$|\{t_k\}| = \infty \quad \textit{and} \; |t_{k+1} - t_k| \leq c, \;\; \forall k,$$

for some $c \in \mathbb{N}$.

The sufficient condition of Theorem 6.7 does not match the necessary condition of Theorem 6.3. However, according to Property 5.21 (on page 173), the sufficient condition *implies* the necessary one. The following result from [210] indicates that $(2F+1)$-robustness is *sharp* for W-MSR in time-invariant synchronous networks under the F-local malicious model. This indicates (as was the case for the topological conditions Γ_M and Γ_B studied in Chapter IV) that a new topological property (other than r-robustness or (r,s)-robustness) is needed to characterize the tight conditions for a successful DTRAC algorithm under the F-local malicious model.

Proposition 6.9 ([210])**.** *For every $F \in \mathbb{Z}_{>0}$, there exists a $2F$-robust network that fails to reach consensus using the W-MSR algorithm with parameter F.*

To illustrate the results of this section, consider the 3-robust graph of Figure 40 (on page 165). Recall that this graph cannot generally sustain 2 malicious nodes as specified by the 2-total model (it is not (3,3)-robust). However, under the 1-local model, it can sustain two malicious nodes if the *right* nodes are compromised. For example, nodes 1 and 4 may be compromised under the 1-local model and the normal nodes will still reach consensus. This example illustrates the advantage of the

F-local model, where there is no concern about global assumptions. If a digraph is $(2F+1)$-robust, then up to F nodes may be compromised in any node's neighborhood, possibly resulting in more than F malicious nodes in the network (as in the previous example).

6.4.4 f-Fraction Local Malicious and Crash Models

We proved in Theorem 6.4 that f-fraction robustness is a necessary condition for the W-MSR algorithm with parameter f to achieve DTRAC in time-invariant networks under the f-fraction local malicious model. We now show that p-fraction robustness, with $p > 2f$, is sufficient.

Theorem 6.10 (Sufficiency with f-Fraction Local Malicious and Crash Models)**.** *Consider a time-invariant network modeled by a digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ where each normal node updates its value according to the W-MSR algorithm with parameter f. Under the f-fraction local malicious model, DTRAC is achieved if the topology of the network is p-fraction robust, where $2f < p \leq 1$.*

Proof. The proof follows the same argument as the proof of Theorem 6.5. In this case, the sets $\mathcal{X}_M(t, t_k, \eta)$ and $\mathcal{X}_m(t, t_k, \eta)$ are defined to include only normal nodes. Then, the p-fraction robust assumption ensures that there exists a (normal) node in either $\mathcal{X}_M$ or $\mathcal{X}_m$ with at least $\lceil pd_i \rceil$ neighbors outside of either $\mathcal{X}_M$ or $\mathcal{X}_m$, respectively. Here, at most $2\lfloor fd_i \rfloor$ of these values are thrown away (with at most $\lfloor fd_i \rfloor$ of them as adversaries, under the f-fraction local model, and at most $\lfloor fd_i \rfloor$ of these strictly smaller, or larger, than node i's value). Because $p > 2f$, it follows that $\lceil pd_i \rceil - 2\lfloor fd_i \rfloor \geq 1$. Therefore, at least one normal value outside of i's set (either $\mathcal{X}_M$ or $\mathcal{X}_m$) will be used in the update . The rest of the analysis is identical to the proof of Theorem 6.5. Therefore, we can show that for $\gamma = (1 - \alpha^{N-1}/2)$ and any $t_k \in \mathbb{Z}_{\geq 0}$

$$\Psi(t_k + j(N-1)) \leq \gamma^j \Psi(t_k),$$

for all $j \in \mathbb{Z}_{\geq 0}$. Because $\gamma < 1$, it follows that $\Psi(t) \to 0$ as $t \to \infty$. $\square$

As with the other adversary models, we may state the following result for time-varying networks.

Theorem 6.11 (Sufficiency with f-Fraction Local Malicious Model in Time-Varying Networks)**.** *Consider a time-varying network modeled by $\mathcal{D}(t) = (\mathcal{V}, \mathcal{E}(t))$ where each normal node updates* *its value according to the W-MSR algorithm with parameter f. Let $\{t_k\}$ denote the set of time steps in which $\mathcal{D}(t)$ is p-fraction robust, where $2f < p \leq 1$. Then, under the f-fraction local malicious model, DTRAC is achieved if*

$$|\{t_k\}| = \infty \quad \text{and} \; |t_{k+1} - t_k| \leq c, \;\; \forall k,$$

for some $c \in \mathbb{N}$.

6.4.5 F-Total, F-Local and f-Fraction Local Byzantine Models

The results so far have focused on malicious adversaries. In this section, we consider Byzantine adversaries. The recent paper [194] investigates a similar algorithm in the context of F-total Byzantine adversaries, and provides necessary and sufficient conditions for the algorithm to succeed. While their proof techniques are different, the main result can be stated succinctly using robustness as follows. This formulation and proof of the necessary and sufficient conditions is due to Zhang and Sundaram. The result uses the normal network, as given in Definition 5.33 on page 188.

Theorem 6.12 ([194], Necessary and Sufficient Condition with F-Total Byzantine Model)**.** *Consider a time-invariant network modeled by a digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ where each normal node updates its value according to the W-MSR algorithm with parameter F. Under the F-total Byzantine model, DTRAC is achieved if and only if the topology of the normal network is $(F + 1)$-robust.*

Proof. To prove sufficiency, besides the method used in [194, 193], we can also use the approach proposed in the proof of Theorem 6.5. Note that when the original network is $(2F + 1)$-robust, the normal network is $(F + 1)$-robust.

To prove necessity, if the normal network is not $(F + 1)$-robust, we can assign the two disjoint sets that are not $(F + 1)$-edge reachable the maximum and minimum values, respectively. Since the Byzantine nodes can send different values to different neighbors, suppose they send the maximum and minimum values to the maximum and minimum sets, respectively. Then, nodes in these two sets never use any values from outside their own sets and consensus is not reached. □

The following results are straightforward extensions of the above result from [194] to the local models and time-varying networks.

Theorem 6.13 (Necessary and Sufficient Condition with F-Local Byzantine Model)**.** *Consider a time-invariant network modeled by a digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ where each normal node updates its value according to the W-MSR algorithm with parameter F. Under the F-local Byzantine model, DTRAC is achieved if and only if the topology of the normal network is $(F + 1)$-robust.*

Proof. The proof is identical to the proof of Theorem 6.12. ☐

Theorem 6.14 (Necessary and Sufficient Condition with f-Fraction Local Byzantine Model)**.** *Consider a time-invariant network modeled by a digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ where each normal node updates its value according to the W-MSR algorithm with parameter f. Under the f-fraction local Byzantine model, DTRAC is achieved if the normal network is p-fraction robust, where $p > f$, and a necessary condition is for the normal network to be f-fraction robust.*

Proof. Necessity follows from Theorem 6.4. For sufficiency, the definitions of $\mathcal{X}_M$ and $\mathcal{X}_m$ are modified to include only normal nodes. Then, the p-fraction robust assumption (with $p > f$) ensures that there exists a normal node in either $\mathcal{X}_M^k$ or $\mathcal{X}_m^k$ with at least $\lceil pd_i \rceil$ normal neighbors outside of either $\mathcal{X}_M$ or $\mathcal{X}_m$, respectively. Since $\lceil pd_i \rceil - \lfloor fd_i \rfloor \geq 1$, a similar argument to the proof of Theorem 6.5 may be used to prove the result. ☐

Theorem 6.15 (Sufficiency with F-Total, F-Local, and f-Fraction Local Byzantine Model in Time–Varying Networks)**.** *Consider a time-varying network modeled by $\mathcal{D}(t) = (\mathcal{V}, \mathcal{E}(t))$ where each normal node updates its value according to the W-MSR algorithm with parameter F (or parameter f for the f-fraction local model). Let $\{t_k\}$ denote the set of time steps in which $\mathcal{D}(t)$ is either (i) $(2F+1)$-robust, or (ii) p-fraction robust, where $2f < p \leq 1$. Then, under (i) the F-total or F-local Byzantine model, or (iii) the f-fraction local Byzantine model, respectively, DTRAC is achieved if*

$$|\{t_k\}| = \infty \quad \text{and } |t_{k+1} - t_k| \leq c, \ \ \forall k,$$

for some $c \in \mathbb{N}$.

Proof. The time-varying arguments required in Theorems 6.8 and 6.11 are sufficient for the Byzantine case as well. ☐

6.5 Simulation Examples

This section presents a numerical example to illustrate our results. In this example, the network is given by the (2,2)-robust graph shown in Figure 45, in which the node set is $\mathcal{V} = \{1, 2, \ldots, 14\}$ and node $i \in \mathcal{V}$ has initial value $x_i(0)$ shown in the circle representing the node. This is the same network topology of Figure 43 used in the simulation example of Section 5.5. Instead of considering a crash adversary, as is done in the previous example, we now consider a single malicious node, and look at both time-invariant and time-varying networks. Since the network is (2,2)-robust, Theorem 6.5 indicates it can generally sustain at most a single malicious node in the network. Suppose that the node with the largest degree, node 14, is compromised and turns malicious. The normal nodes use either LICP given in (60) or W-MSR with parameter $F = 1$ for their update rule. Each normal node $i \in \mathcal{N}$ uses the weights $w_{(j,i)}(t) = |\mathcal{J}_i^{\text{in}}(t)|^{-1}$ for each $j \in \mathcal{J}_i^{\text{in}}(t)$ with LICP and $w_{(j,i)}(t) = (|\mathcal{J}_i^{\text{in}}(t) \setminus \mathcal{R}_i(t)|)^{-1}$ for each $j \in \mathcal{J}_i^{\text{in}}(t) \setminus \mathcal{R}_i(t)$ with W-MSR. The malicious node's objective is to prevent the normal nodes from reaching consensus and to drive the normal node values outside of the range of their initial values.

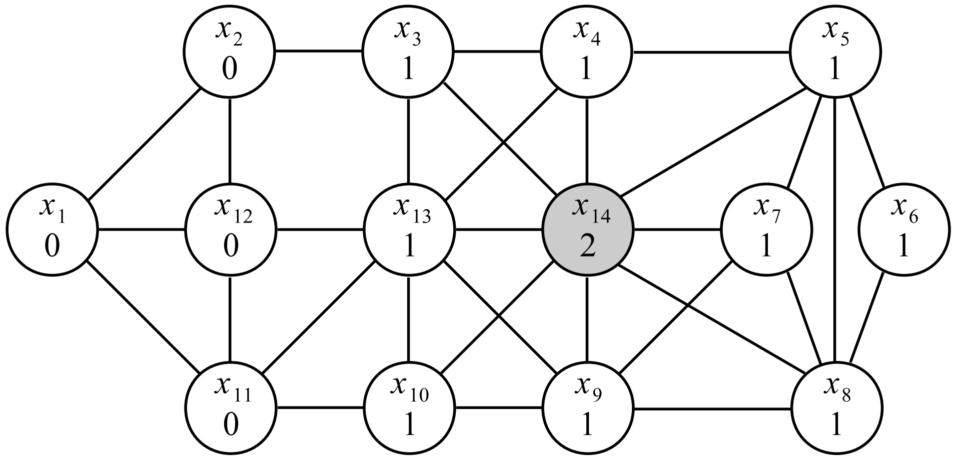


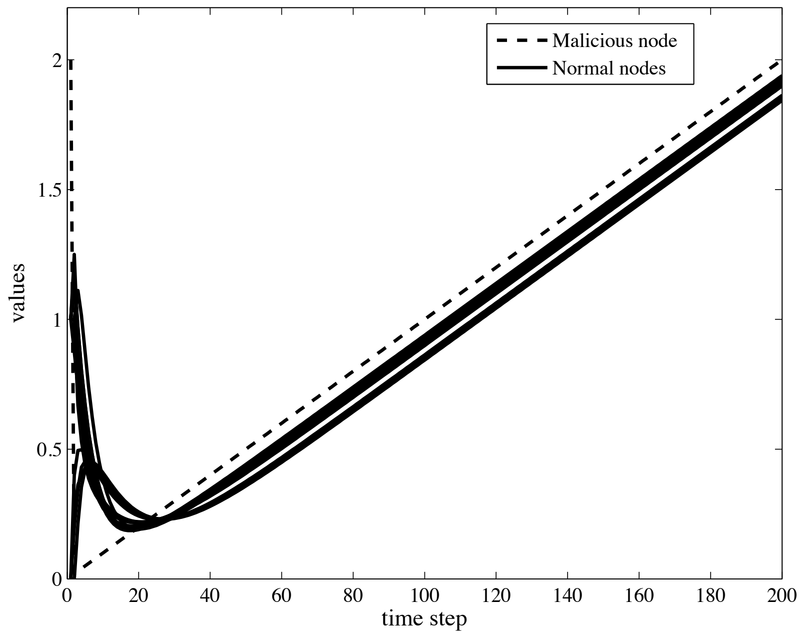
Figure 45: (2,2)-Robust network topology used in DTRAC simulations.

The results for the time-invariant network of Figure 45 are shown in Figure 46. It is clear in Figure 46(a) that the malicious node is able to drive the values of the normal nodes outside of the range of initial values and prevent consensus whenever LICP is used. On the other hand, the malicious node is unable to achieve its goal whenever W-MSR is used. Note that although consensus can be reached, the malicious node still has the potential to drive the consensus process to any value in the interval $[0, 1]$ by choosing the desired value as its initial value and remaining constant. However, this is allowed with resilient asymptotic consensus (because the consensus value is within the range of the initial values held by normal nodes).

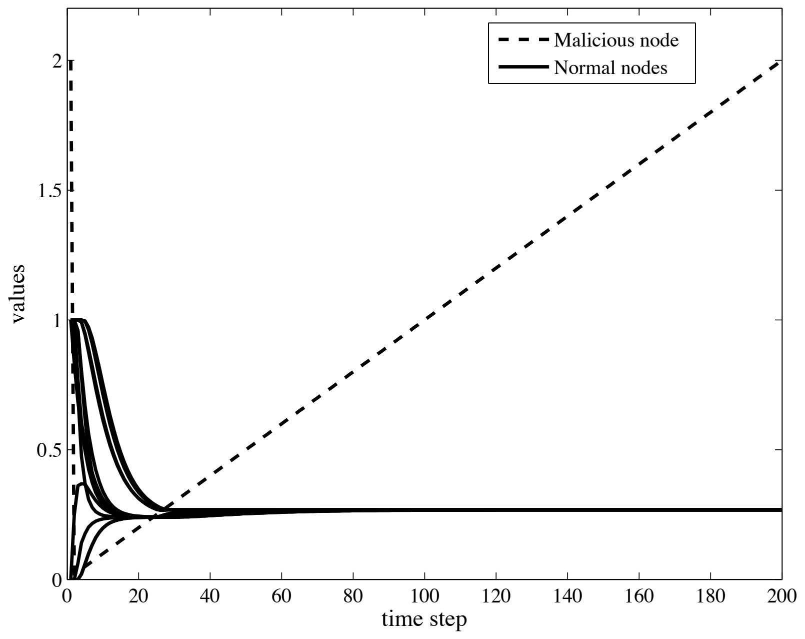
Finally, we illustrate the time-varying network result for the 1-total malicious model by removing approximately half of the directed edges in 9 out of every 10 consecutive time steps. To do this, we check whether the time step is equal to 0 modulo 10. If it is not, then we model directed edge removal by a Bernoulli process with parameter $p = 0.5$, so that approximately half of the directed edges are removed in these time steps. The results are illustrated in Figure 47, and show that only the speed of convergence is affected when using W-MSR.

6.6 Summary

This chapter extends the Resilient Asymptotic Consensus (RAC) problem to discrete-time synchronous networks, called DTRAC. Analogous results are presented as was shown for the continuous-time case.

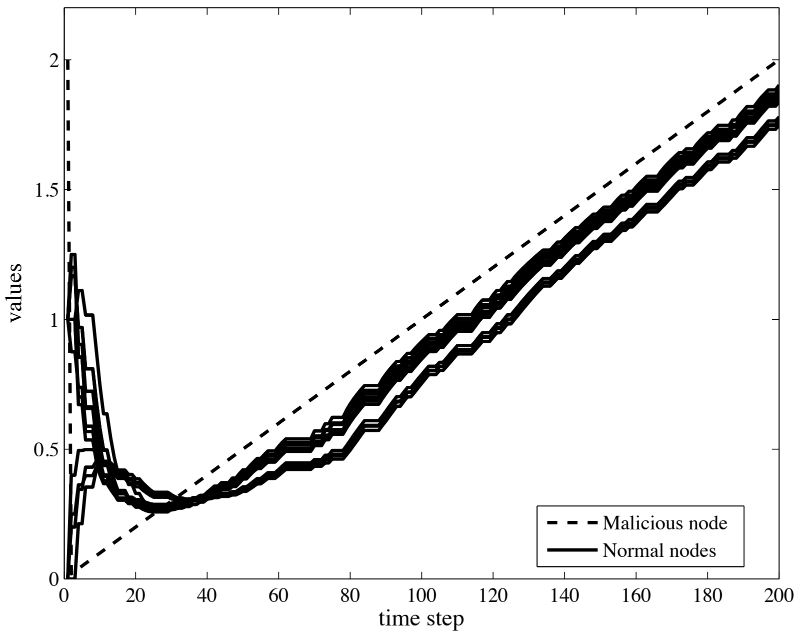


(a) LICP.

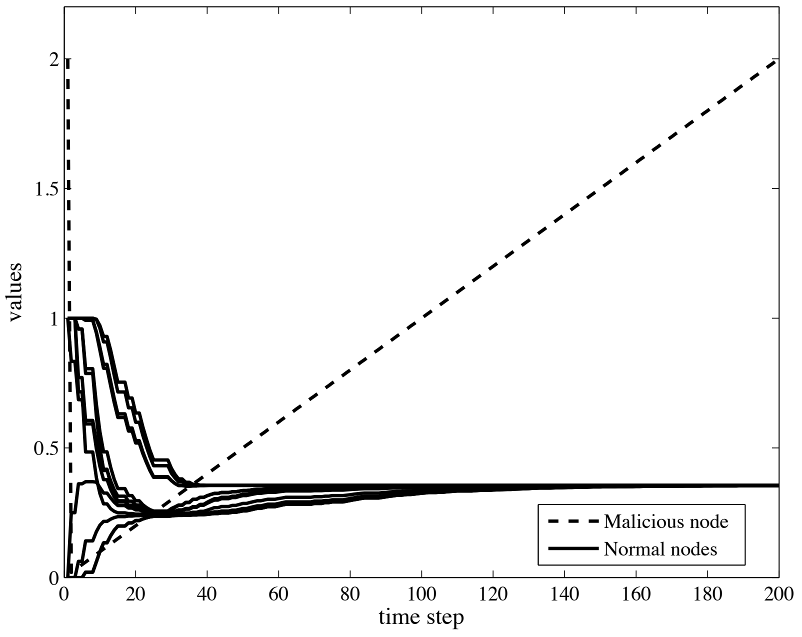


(b) W-MSR.

Figure 46: Simulation of W-MSR & LICP in a fixed network under the F-total malicious model.



(a) LICP.



(b) W-MSR.

Figure 47: Simulation of W-MSR & LICP in a dynamic network under the F-total malicious model.

CHAPTER VII

DISCRETE-TIME RAC IN ASYNCHRONOUS NETWORKS

Consensus problems have a rich history in distributed computing [130] and communication [82]. More recently, consensus has become an active area of control research [152, 168]. This is because reaching agreement is a fundamental task in distributed and multi-agent systems, and arises in diverse applications such as agent flocking [90, 151], synchronized path following [89], distributed estimation [172], and load balancing for parallel processors [19]. A major concern in large-scale distributed systems is whether the group objectives can be achieved in the presence of uncertainties such as communication delays, data loss, or node failures. While researchers in control have studied consensus algorithms that have been shown to be resilient to communication delays [145], data loss [156], and quantization [94], the robustness of such algorithms to node failures has been shown to be lacking [75].

Of course, consensus algorithms that are robust to node failures have been studied in distributed computing [161, 130], communication networks [92], and mobile robotics [1, 46, 27]. In these works, the faulty nodes may be characterized by *fault models* and *scope of fault* assumptions. Two common fault models are the *crash fault* [1, 46] and the *Byzantine fault* [110, 51, 1, 194]. Whenever a node suffers a crash fault, it simply stops somewhere in its execution (this is also referred to as a stopping failure [130]). A Byzantine faulty node, on the other hand, may behave in an arbitrary manner. Therefore, worst case executions must be considered. The scope of the faults is usually assumed to satisfy some global bound; for example, at most F out of n nodes fail. We refer to this as the F-total model. A local bound on the number of faulty nodes has been considered in fault-tolerant broadcasting [162, 86, 210], where it is assumed that at most F of any normal node's neighbors fail. We refer to this as the F-local model.

Another important concern in large-scale networked systems is the issue of malicious attacks and security breaches [30]. Attacks on the network may include jamming [20], denial-of-service [4], false data injection [138], replay [139], or deception [212]. In jamming and denial-of-service attacks, the attacker reduces (or even destroys) the availability of data from the communication network. False data injection and deception attacks affect the integrity of the data. Likewise, replay attacks inject incorrect and outdated information by repeating previously transmitted data.

In the context of security, it is also important to consider security breaches in which a subset of nodes are compromised and behave as adversaries. In this case, the compromised nodes may be characterized by threat models and scope of threat assumptions. For such a scenario, the Byzantine model is a suitable threat model. But, depending on the communication realization, the full generality of the Byzantine model may not be necessary. In general, Byzantine nodes may simultaneously send different information to different neighbors in the network [130]. However, if the nodes broadcast their information to neighbors, then duplicity (of this type) is not possible. We refer to the Byzantine node under the local broadcast model as a *malicious node* [181, 210, 114, 116, 115].

There are two general approaches to designing distributed algorithms that are resilient to compromised nodes. Either the compromised nodes may be detected and identified so that their influence may be removed, or the algorithms must filter the information received from neighbors in such a way that ensures resilience. The problem of detecting misbehavior and identifying compromised nodes in linear consensus networks has been studied in [181, 159]. The paper [181] considers the problem of distributed function calculation in the presence of up to F malicious nodes (i.e., F-total model). On the other hand, [159] focuses on the special case of consensus, but considers both malicious nodes and non-colluding nodes under the F-total model. The focus of these works is to characterize the conditions on the network topology under which the compromised nodes can be identified from calculations performed by an iterative linear consensus protocol, and then design algorithms that can overcome the adversaries.

While identifying compromised nodes is clearly an interesting and important approach, these detection and identification techniques require each node to have information of the network topology beyond its local neighborhood. This requirement of *nonlocal information* may not be suitable for large-scale networks. Furthermore, the detection algorithms are computationally expensive. On the other hand, algorithms that filter the information received from neighbors to ensure resilience only require local information and may be computationally efficient [113, 210, 114, 116, 115]. A class of such algorithms were originally used in the study of the approximate agreement problem [51], and were extended to a family of algorithms, called the *Mean-Subsequence-Reduced (MSR)* algorithms [98]. In [115], we generalize the MSR algorithm as the *Weighted-Mean-Subsequence-Reduced (W-MSR)* algorithm, and study its convergence properties in synchronous networks under various threat models and scope of threat assumptions.

This chapter considers a consensus problem, referred to as resilient asymptotic consensus, which is closely related to the approximate agreement problem. The main difference is that in this case we ignore the requirement of finite termination. Instead, we consider the asymptotic performance of the networked system. Another minor difference lies in semantics. Instead of viewing some of the nodes as suffering faults, we make the interpretation that the nodes have been compromised in a security breach and behave as adversaries. Note that classical results [51] as well as very recent results in approximate agreement [194, 193, 195] may also be interpreted in this manner.

The contributions of this chapter are as follows. First, we formulate the resilient asymptotic consensus problem in an asynchronous framework, with a local broadcast model of communication.[50] Then, we characterize, for the first time, the necessary and sufficient condition on the network topology for the existence of an algorithm that achieves resilient asymptotic consensus in time-invariant asynchronous networks under the local broadcast model of communication, and in the presence of up to F malicious nodes (F-local model). To show sufficiency, we adapt the W-MSR algorithm to an asynchronous setting and prove that the algorithm succeeds under the necessary condition. Finally, we provide a sufficient condition for the existence of an algorithm that achieves resilient asymptotic consensus in the asynchronous network model under the F-local model. Note that the results can readily be extended to the other adversary models, similarly as was done in previous chapters.

The rest of the chapter is organized as follows. Section 7.1 recalls the meaning of a malicious adversary, and discusses details and assumptions placed on the threat model in asynchronous networks. Section 7.3 describes how the W-MSR algorithm is adapted to an asynchronous setting. Section 7.4 contains the main results. Section 7.5 provides context for how these results relate to the literature. Finally, Section 7.6 summarizes the chapter.

7.1 Malicious Nodes in Asynchronous Networks

The assumptions on the malicious nodes need to be clarified for the asynchronous framework. Malicious nodes are still omniscient adversaries. In particular, they know all other values and the

[50] We do not consider a point-to-point model of communication because this problem is recently solved in [195]. See Section 7.5 for more details.

full network topology; they are aware of the update rules $f_i(\cdot)$, $\forall i \in \mathcal{N}$; they are aware of which other nodes are adversaries; and they know the plans of the other adversaries.[51] Although malicious nodes have complete knowledge, their ability to affect other nodes is limited. Specifically, a malicious node $k \in \mathcal{A}$ may choose whether or not to broadcast its value to its out-neighbors in any round $r \in \mathbb{Z}_{\geq 0}$, but if it does, it must send at most one value $x_k(r)$ tagged with round r (otherwise, the normal nodes receiving more than one value from node k in round r would know that k is an adversary). Additionally, in any finite time interval, a malicious node must send a finite number of messages. Finally, a malicious node may update its value in an arbitrary fashion. Since it is omniscient, one must assume that the update is one that causes the most disruption to the normal nodes (perhaps with respect to some disruption objective function). Observe that malicious nodes are Byzantine nodes restricted to the local broadcast model. Byzantine nodes differ in that they are capable of sending different messages to different out-neighbors, which is possible under a point-to-point communication model [195].

7.2 System Model and Problem Statement

The digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ models the communication subsystem of the networked system. The nodes communicate under a *local broadcast model*, in which the out-neighbors of each node i in $\mathcal{D}$ are precisely those nodes capable of receiving messages from i. The communication is assumed to be *reliable* (meaning all transmitted messages are eventually delivered successfully), but the messages may incur different (arbitrary) delays in transmission to different nodes and messages may be received out of order. It is further assumed that the messages are signed and that the signature of each normal node is unforgeable.

For the class of algorithms studied in this work, the execution of each node proceeds in a sequence of rounds $r \in \mathbb{Z}_{\geq 0}$ that consists of *transmit*, *receive*, and *update stages*. A node (or subset of nodes) that is in the process of executing its algorithm (i.e., it is in either the transmit, receive, or update stage of some round $r \in \mathbb{Z}_{\geq 0}$) is said to be *active*. The networked system is *asynchronous*, meaning the nodes do not necessarily execute rounds at the same rate and the nodes are *not* synchronized. Therefore, at any point in real time $t \in \mathbb{R}_{\geq 0}$, the nodes may be in different stages of

[51]One may take the viewpoint that a centralized omniscient adversary informs and directs the behavior of the malicious nodes.

different rounds. The reference time $t = 0$ is defined as the point in time at which the first subset of normal nodes becomes active. Because there is no synchrony among the nodes, it is possible that at some time $t > 0$, some of the nodes may have not yet become active. To handle this situation, we assume there exists a *dormant stage* of round $r = 0$, in which nodes may accumulate messages from in-neighbors but do not act upon them until becoming active. Note that we place no limitations on the amount of storage available at each node.

In order to keep track of the messages corresponding to a given round, each message is tagged by the round $r \in \mathbb{Z}_{\geq 0}$ in which it is sent. Every node (including adversary nodes) sends at most one message to its out-neighbors in each round, and each normal node sends exactly one message per round. Since the normal nodes follow this protocol, any adversary node that sends multiple messages tagged by a single round would be easily detected as an adversary. However, an adversary may skip rounds without detection (in finite time) because the messages may be received out of order and with arbitrary delay. Moreover, an adversary may decide at some point in time to stop sending messages altogether.

7.2.1 Update Model

Suppose that each node $i \in \mathcal{V}$ maintains a scalar value $x_i(r) \in \mathbb{R}$ called the *state* of node i in round r. In particular, each node begins with the private value $x_i(0)$ (which could represent a measurement, optimization variable, etc.). At the beginning of each round $r \in \mathbb{Z}_{\geq 0}$ (once the node becomes active), each normal node i broadcasts its value to its out-neighbors in the network (transmit stage). The value sent by node j in round $r \in \mathbb{Z}_{\geq 0}$ is denoted $x_j(r)$. Once each normal node i transmits its value, it then waits[52] to receive $d_i^* < d_i$ values (receive stage) from its in-neighbors (d_i^* depends on the adversary model, scope of threat assumptions, and size of $\mathcal{N}_i^{\text{in}}$). Observe that d_i^* should be small enough so as to avoid deadlock. Once node $i \in \mathcal{N}$ collects d_i^* values from in-neighbors, it updates its value for round $r + 1$ according to the prescribed rule

$$x_i(r+1) = f_i\left(x_i(r), \{x_j(r)\}\right), \quad i \in \mathcal{N}, j \in \mathcal{V}_i^*(r),$$

[52]If node i is one of the last normal nodes to become active, it is possible that it may have already received at least $d_i - F$ messages from in-neighbors for multiple rounds. In such a case, there is no need for node i to wait.

where $\mathcal{V}_i^*(r)$ is the set of nodes corresponding to the d_i^* values received from node i's in-neighbors in round r. The update rule $f_i(\cdot)$ can be an arbitrary function, and may be different for each node, depending on its role in the network. These functions are designed *a priori* so that the normal nodes reach consensus. However, some of the nodes may not follow the prescribed strategy if they are compromised by an adversary. Such misbehaving nodes threaten the group objective, and it is important to design the $f_i(\cdot)$'s in such a way that the influence of such nodes can be eliminated or reduced without prior knowledge about their identities.

7.2.2 Asynchronous DTRAC Problem Statement

The *discrete-time resilient asymptotic consensus problem in asynchronous networks* differs slightly in its formulation from the synchronous version of the problem. In this case, let $M_{\mathcal{N}}(r)$ and $m_{\mathcal{N}}(r)$ be the *maximum* and *minimum* values of the normal nodes in round r, respectively.

Definition 7.1 (Discrete-Time Resilient Asymptotic Consensus in Asynchronous Networks)**.** *The normal nodes are said to achieve* ***resilient asymptotic consensus*** *in the presence of adversary nodes (given a particular adversary model) if*

(i) $m_{\mathcal{N}}(r+1) \geq m_{\mathcal{N}}(r)$ *and* $M_{\mathcal{N}}(r+1) \leq M_{\mathcal{N}}(r)$*, for* $r \in \mathbb{Z}_{\geq 0}$*, and*

(ii) $\lim_{r\to\infty} M_{\mathcal{N}}(r) - m_{\mathcal{N}}(r) = 0$,

for any choice of initial values.

The RAC problem consists of two conditions. The first (i) is a *validity* or *safety* condition. If it is satisfied, then the states of the normal nodes remain inside the initial interval $[m_{\mathcal{N}}(0), M_{\mathcal{N}}(0)]$ (safety), and any value selected (i.e., through termination) is guaranteed to lie in this interval (validity). The second (ii) is a *convergence* condition on agreement. Observe that any asynchronous algorithm that achieves resilient asymptotic consensus in the presence of F-totally bounded malicious nodes must wait for no more than $d_i^* = d_i - F$ values in its receive stage in order to avoid deadlock.

7.2.3 Quantized RAC

In the case of quantized consensus, we establish the same network model as described above. The model still allows infinite precision computation and infinite-rate transmission; however the consensus value at each node is quantized. To describe the quantization scheme, we require the following definition of a δ-set [142].

Definition 7.2. *A set $\Delta \subset \mathbb{R}$ is a **δ-set** for $\delta \in \mathbb{R}$ if $\forall v, v' \in \Delta$,*

$$\frac{|v - v'|}{\delta} \in \mathbb{Z}_{\geq 0}.$$

Let $q_\Delta \colon \mathbb{R} \to \Delta$ be a quantizer that rounds a real number x to its closest value in δ-set Δ, and rounds up whenever x lies at the midpoint between two values in Δ. The quantizer is given by

$$q_\Delta(x) = \max\{\arg\min_{v \in \Delta} |v - x|\}. \tag{62}$$

7.2.3.1 Update Model with Quantized State

Suppose that each node $i \in \mathcal{V}$ maintains a scalar value $x_i(r) \in \mathbb{R}$ for all rounds $r \in \mathbb{Z}_{\geq 0}$ and a quantized value $v_i(r)$ for rounds $r \in \mathbb{Z}_{>0}$. In particular, each node begins with the private value $x_i(0)$. At the beginning of each round $r \in \mathbb{Z}_{\geq 0}$ (once the node becomes active), each normal node i broadcasts its value to its out-neighbors in the network (transmit stage). The value sent by node j in round $r \in \mathbb{Z}_{\geq 0}$ is denoted $x_j(r)$. Once each normal node i transmits its value, it then waits to receive $d_i^* < d_i$ values (receive stage) from its in-neighbors (d_i^* depends on the adversary model, scope of threat assumptions, and size of $\mathcal{N}_i^{\text{in}}$). Observe that d_i^* should be small enough so as to avoid deadlock. Once node $i \in \mathcal{N}$ collects d_i^* values from in-neighbors, it updates its values for round $r + 1$ according to the prescribed rules

$$\begin{aligned} x_i(r+1) &= f_i\left(x_i(r), \{x_j(r)\}\right), \quad i \in \mathcal{N}, j \in \mathcal{V}_i^*(r), \\ v_i(r+1) &= q_\Delta(x_i(r+1)). \end{aligned}$$

where $\mathcal{V}_i^*(r)$ is the set of nodes corresponding to the d_i^* values received from node i's in-neighbors in round r. The update rule $f_i(\cdot)$ can be an arbitrary function, and may be different for each node, depending on its role in the network. These functions are designed *a priori* so that the normal nodes reach consensus. However, some of the nodes may not follow the prescribed strategy if they are compromised by an adversary. Such misbehaving nodes threaten the group objective, and it is important to design the $f_i(\cdot)$'s in such a way that the influence of such nodes can be eliminated or reduced without prior knowledge about their identities.

7.2.3.2 Problem Statement

The *discrete-time quantized resilient asymptotic consensus (QRAC) problem in asynchronous networks* differs slightly in its formulation from the version without quantization. In this case, we require that the quantized values agree with a maximum error given by the precision of the quantizer (i.e., no values differ by more than δ).

Definition 7.3 (Discrete-Time Quantized Resilient Asymptotic Consensus in Asynchronous Networks)**.** *The normal nodes are said to achieve* ***quantized resilient asymptotic consensus*** *(QRAC) in the presence of adversary nodes (given a particular adversary model) if*

(i) $q_\Delta(m_\mathcal{N}(r+1)) \geq q_\Delta(m_\mathcal{N}(r))$ *and* $q_\Delta(M_\mathcal{N}(r+1)) \leq q_\Delta(M_\mathcal{N}(r))$*, for* $r \in \mathbb{Z}_{\geq 0}$*, and*

(ii) $\lim_{r\to\infty} q_\Delta(M_\mathcal{N}(r)) - q_\Delta(m_\mathcal{N}(r)) \leq \delta$*,*

for some δ*-set* Δ *and for any choice of initial values* $x_i(0)$*,* $i \in \mathcal{N}$*.*

7.3 Asynchronous Resilient Algorithms

In this section, we modify the Weighted Mean-Subsequence-Reduce (W-MSR) algorithm that we studied in synchronous networks [115, 116] (described in the previous chapter). The modifications are made to accommodate asynchrony and are similar to the modifications done for similar algorithms [51, 195], and consist of the following two changes: (i) the messages are tagged with the corresponding round index, and (ii) each normal node j waits to receive only $d_j^* = d_j - F$ messages from in-neighbors for a given round r before updating its value. After this, we describe another variation of the algorithm in which each node maintains a quantized state value along with the unquantized one.

7.3.1 Asynchronous W-MSR Algorithm

Asynchronous W-MSR with parameter F:

In each round $r \in \mathbb{Z}_{\geq 0}$, once active, normal node i performs the following steps:

1. *Transmit step*: Send current value $x_i(r)$ to outgoing neighbors, along with round index r.

2. *Receive step*: Wait[53] to receive exactly $d_i^* = d_i - F$ messages from different nodes tagged by round index r, and break ties arbitrarily. Sort the $d_i - F$ values in ascending order. If there are less than F values strictly larger than its own value, $x_i(r)$, 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.

3. *Update step*: Let $\mathcal{R}_i(r)$ denote the set of nodes whose values were removed or disregarded by normal node i in step 2 of round r. Each normal node i applies the update

$$x_i(r+1) = \sum_{j \in \mathcal{J}_i^{\text{in}} \setminus \mathcal{R}_i(r)} w_{ij}(r) x_j(r), \tag{63}$$

   where the weights $w_{ij}(r)$ satisfy the following conditions for all rounds $r \in \mathbb{Z}_{\geq 0}$ and for some $0 < \alpha << 1$.

   - $w_{ij}(r) = 0$ whenever $j \notin \mathcal{J}_i^{\text{in}}$ or $j \in \mathcal{R}_i(r)$;
   - $w_{ij}(r) \geq \alpha, \forall j \in \mathcal{J}_i^{\text{in}} \setminus \mathcal{R}_i(r), i \in \mathcal{N}$;
   - $\sum_{j=1}^{n} w_{ij}(r) = 1, \forall i \in \mathcal{N}$.

   Together, these conditions imply that the updated value is a convex combination of values in $\mathcal{J}_i^{\text{in}} \setminus \mathcal{R}_i(r)$ with a uniform lower bound on the weights given by α.

[53]If node i is one of the last normal nodes to become active, it is possible that it may have already received at least $d_i - F$ messages from in-neighbors for multiple rounds. In such a case, there is no need for node i to wait.

7.3.2 Quantized Asynchronous W-MSR Algorithm

Let $q_\Delta \colon \mathbb{R} \to \Delta$ be a quantizer that rounds a real number x to its closest value in δ-set Δ (recall the definition of a δ-set given on page 229), and rounds up whenever x lies at the midpoint between two values in Δ. The quantizer is given by

$$q_\Delta(x) = \max\{\arg\min_{v \in \Delta} |v - x|\}.$$

The quantized version of the algorithm is described as follows.

Quantized Asynchronous W-MSR with parameter F:

Each node $i \in \mathcal{V}$ begins with value $x_i(0) \in \mathbb{R}$ and quantized value $v_i(0) = q_\Delta(x_i(0)) \in \Delta$. Once active, normal node i performs the following steps in each round $r \in \mathbb{Z}_{\geq 0}$:

1. *Transmit step*: Send current value $x_i(r)$ to outgoing neighbors, along with round index r.

2. *Receive step*: Wait to receive exactly $d_i^* = d_i - F$ messages from different nodes tagged by round index r, and break ties arbitrarily. Sort the $d_i - F$ values in ascending order. If there are less than F values strictly larger than its own value, $x_i(r)$, 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.

3. *Update step*: Let $\mathcal{R}_i(r)$ denote the set of nodes whose values were removed or disregarded by normal node i in step 2 of round r. Each normal node i applies the update of (63) and then quantizes The value of $x_i(r+1)$ is then quantized as

$$v_i(r+1) = q_\Delta(x_i(r+1)).$$

The rest of the chapter is concerned with analyzing necessary and sufficient conditions on the network topology for the normal nodes using Asynchronous W-MSR (or Quantized Asynchronous W-MSR) with parameter F to achieve RAC (or QRAC).

7.4 Resilient Consensus Analysis

We start with the following result showing that W-MSR always satisfies the validity condition for resilient asymptotic consensus. We then provide the definition of robustness used in the analysis. Recall that $M_{\mathcal{N}}(r)$ and $m_{\mathcal{N}}(r)$ are the maximum and minimum values of the *normal* nodes in round r, respectively.

Lemma 7.4. *Suppose each normal node updates its value according to the Asynchronous W-MSR algorithm with parameter F under the F-total or F-local model. Then, for each normal node $i \in \mathcal{N}$, $x_i(r+1) \in [m_{\mathcal{N}}(r), M_{\mathcal{N}}(r)]$, regardless of the network topology. From this we conclude $m_{\mathcal{N}}(r+1) \geq m_{\mathcal{N}}(r)$ and $M_{\mathcal{N}}(r+1) \leq M_{\mathcal{N}}(r)$.*

Proof. Suppose that one value, say $x_j(r)$, used in the update (63) satisfies $x_j(r) > M_{\mathcal{N}}(r)$. Then, by definition of $M_{\mathcal{N}}(r)$, j must be an adversary and $x_j(r) > x_i(r)$. Since i uses $x_j(r)$ in round r, there must be at least F more nodes in the neighborhood of i with values at least as large as $x_j(r)$. Hence, these nodes must also be adversaries, which contradicts the assumption that at most F in-neighbors of i are adversary nodes. Thus, $x_j(r) \leq M_{\mathcal{N}}(r)$. Similarly, we can show that $x_j(r) \geq m_{\mathcal{N}}(r)$. The result follows since $x_i(r+1)$ in (63) is a convex combination of values in $[m_{\mathcal{N}}(r), M_{\mathcal{N}}(r)]$. □

7.4.1 Necessary Condition for F-Total or F-Local Malicious Model

The following is the major contribution of the chapter and provides, for the first time, a *necessary and sufficient* condition for there to exist an algorithm that can achieve resilient asymptotic consensus in asynchronous networks with a local broadcast communication model under the F-total malicious model. First, we prove necessity. Then we show sufficiency by demonstrating that Asynchronous W-MSR achieves resilient asymptotic consensus with this condition.

Theorem 7.5. *If an asynchronous algorithm achieves resilient asymptotic consensus under the F-total or F-local malicious model in a nontrivial ($n \geq 2$) time-invariant asynchronous network under the local broadcast model, then the network is $(2F+1, F+1)$-robust.*

Proof. Suppose there exists an asynchronous algorithm that achieves resilient asymptotic consensus in a nontrivial network that is not $(2F+1, F+1)$-robust. Then, there are nonempty, disjoint

$\mathcal{S}_1, \mathcal{S}_2 \subset \mathcal{V}$ such that none of the conditions $(i) - (iii)$ in Definition 5.10 hold (with $r = 2F + 1$ and $s = F + 1$). Suppose the initial value of each node in $\mathcal{S}_1$ is a and each node in $\mathcal{S}_2$ is b, with $a < b$. Let all other nodes have initial values taken from the interval $[a, b]$. Since $|\mathcal{X}_{\mathcal{S}_1}^{2F+1}| + |\mathcal{X}_{\mathcal{S}_2}^{2F+1}| \leq F$, suppose all nodes in $\mathcal{X}_{\mathcal{S}_1}^{2F+1}$ and $\mathcal{X}_{\mathcal{S}_2}^{2F+1}$ are malicious (which is allowed under both the F-total and F-local models) and keep their values constant for all rounds. With this assignment of adversaries, there is still at least one normal node in both $\mathcal{S}_1$ and $\mathcal{S}_2$ since $|\mathcal{X}_{\mathcal{S}_1}^{2F+1}| < |\mathcal{S}_1|$ and $|\mathcal{X}_{\mathcal{S}_2}^{2F+1}| < |\mathcal{S}_2|$, respectively.

Fix any normal node $i \in \mathcal{S}_1$ (and therefore, $i \in \mathcal{S}_1 \setminus \mathcal{X}_{\mathcal{S}_1}^{2F+1}$), and note that $|\mathcal{N}_i^{\text{in}} \setminus \mathcal{S}_1| \leq 2F$. Suppose the delays for messages from $q_i = \min\{F, |\mathcal{N}_i^{\text{in}} \setminus \mathcal{S}_1|\}$ nodes in $\mathcal{N}_i^{\text{in}} \setminus \mathcal{S}_1$ are arbitrarily large compared to all the other incoming messages to node i in round 0 (and the delays are large enough so that node i has become active). Then $\mathcal{N}_i^{\text{in}} \setminus \mathcal{R}_i(0)$ includes at most $|\mathcal{N}_i^{\text{in}} \setminus \mathcal{S}_1| - q_i \leq F$ values outside of $\mathcal{S}_1$ (which from the perspective of the update rule could all be adversary values). Other values used by the update rule for node i are from inside $\mathcal{S}_1$ (including node i's own value), so they have value a. Therefore, the update rule must set $x_i(1) = a$ to ensure the validity condition (more specifically, to ensure $M_{\mathcal{N}}(1) \leq M_{\mathcal{N}}(0)$). In a similar manner, one can argue that any normal node $j \in \mathcal{S}_2 \setminus \mathcal{X}_{\mathcal{S}_2}^{2F+1}$ must select $x_j(1) = b$. Finally, since $[m_{\mathcal{N}}(0), M_{\mathcal{N}}(0)] = [a, b]$, any normal node k in $\mathcal{V} \setminus (\mathcal{S}_1 \cup \mathcal{S}_2)$ must set its value $x_k(1) \in [a, b]$ to ensure the validity condition. Therefore, round 1 has the same distribution of values as round 0. By induction, we conclude that for each round $r \in \mathbb{Z}_{\geq 0}$ each node in $\mathcal{S}_1$ has value a, each node in $\mathcal{S}_2$ has value b, and all other nodes have values in $[a, b]$. Therefore, no consensus is achieved, which contradicts the assumption that there exists an asynchronous algorithm that achieves resilient asymptotic consensus in a network that is not $(2F + 1, F + 1)$-robust. $\square$

7.4.2 Sufficiency for F-Total Malicious Model

Theorem 7.6 (Sufficiency)**.** *Consider a time-invariant asynchronous network under the local broadcast model. Suppose the communication is described by a digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$, where each normal node uses the Asynchronous W-MSR algorithm with parameter F. Then, under the F-total malicious model, resilient asymptotic consensus is achieved if the network topology is $(2F + 1, F + 1)$-robust.*

Proof. Define $\Psi(r) = M_{\mathcal{N}}(r) - m_{\mathcal{N}}(r)$, which is a nonincreasing function of r by Lemma 7.4. Whenever the normal nodes are in agreement at some round $r_0 \in \mathbb{Z}_{\geq 0}$, then consensus is maintained in future rounds $r \geq r_0$. In the analysis that follows, recall that r is the round index and does not correspond to a common point in real time among the nodes. The difference in real time between when any two nodes actually execute round r may be quite large. The main point is that eventually each node will execute round $r \in \mathbb{Z}_{>0}$ because the network delay is finite and normal nodes only wait for at most $d_i - F$ incoming messages from neighbors. With this in mind, fix $r_0 \geq 0$ and assume $\Psi(r_0) > 0$. For $r \geq r_0$ and $\eta > 0$, define $\mathcal{X}_M(r, r_0, \eta) = \{j \in \mathcal{V} \colon x_j(r) > M_{\mathcal{N}}(r_0) - \eta\}$ and $\mathcal{X}_m(r, r_0, \eta) = \{j \in \mathcal{V} \colon x_j(r) < m_{\mathcal{N}}(r_0) + \eta\}$. Define $\epsilon_0 = \Psi(r_0)/2$ and define $\epsilon_j = \alpha\epsilon_{j-1}$ for $j = 1, 2, \ldots, N-1$, where $N = \mathcal{N}$. It follows that $\epsilon_j = \alpha^j \epsilon_0 > 0$. By definition, the sets $\mathcal{X}_M(r_0, r_0, \epsilon_0)$ and $\mathcal{X}_m(r_0, r_0, \epsilon_0)$ are nonempty and disjoint. Because $\mathcal{D}$ is $(2F+1, F+1)$-robust and there are at most F malicious nodes in the network (F-total model), it follows that either there exists $i \in \mathcal{X}_M(r_0, r_0, \epsilon_0) \cap \mathcal{N}$ or there exists $i \in \mathcal{X}_m(r_0, r_0, \epsilon_0) \cap \mathcal{N}$, or there exists such i in both, such that i has at least $2F+1$ neighbors outside of its set. Suppose first that $i \in \mathcal{X}_M(r_0, r_0, \epsilon_0) \cap \mathcal{N}$ has at least $2F+1$ neighbors outside its set. Since at most $2F$ of these values will be ignored or removed (up to F ignored due to delays and F removed for being the smallest values in the in-neighborhood of node i), it follows that

$$\begin{aligned} x_i(r_0+1) &= \sum_{j \in \mathcal{J}_i^{\text{in}} \setminus \mathcal{R}_i(r_0)} w_{ij}(r_0) x_j(r_0) \\ &\leq \alpha(M_{\mathcal{N}}(r_0) - \epsilon_0) + (1-\alpha) M_{\mathcal{N}}(r_0) \\ &\leq M_{\mathcal{N}}(r_0) - \alpha\epsilon_0 = M_{\mathcal{N}}(r_0) - \epsilon_1. \end{aligned}$$

Note that for any normal node not in $\mathcal{X}_M(r_0, r_0, \epsilon_0)$, the above inequality holds as well because any normal node always uses its own value in the update. From this, we conclude

$$|\mathcal{X}_M(r_0+1, r_0, \epsilon_1) \cap \mathcal{N}| < |\mathcal{X}_M(r_0, r_0, \epsilon_0) \cap \mathcal{N}|.$$

Similarly, if $i \in \mathcal{X}_m(r_0, r_0, \epsilon_0) \cap \mathcal{N}$ has at least $2F+1$ neighbors outside its set, then

$$\begin{aligned} x_i(r_0+1) &= \sum_{j \in \mathcal{J}_i^{\text{in}} \setminus \mathcal{R}_i(r_0)} w_{ij}(r_0) x_j(r_0) \\ &\geq \alpha(m_{\mathcal{N}}(r_0) + \epsilon_0) + (1-\alpha) m_{\mathcal{N}}(r_0) \\ &\geq m_{\mathcal{N}}(r_0) + \alpha\epsilon_0 = m_{\mathcal{N}}(r_0) + \epsilon_1. \end{aligned}$$

Similarly as above, this inequality holds for any normal node not in $\mathcal{X}_m(r_0, r_0, \epsilon_0)$. From this, we conclude

$$|\mathcal{X}_m(r_0+1, r_0, \epsilon_1) \cap \mathcal{N}| < |\mathcal{X}_m(r_0, r_0, \epsilon_0) \cap \mathcal{N}|.$$

By repeating this analysis, we can show by induction that as long as $\mathcal{X}_M(r_0+j, r_0, \epsilon_j) \cap \mathcal{N}$ and $\mathcal{X}_m(r_0+j, r_0, \epsilon_j) \cap \mathcal{N}$ are both nonempty, then either

$$|\mathcal{X}_M(r_0+j+1, r_0, \epsilon_{j+1}) \cap \mathcal{N}| < |\mathcal{X}_M(r_0+j, r_0, \epsilon_j) \cap \mathcal{N}|,$$

or

$$|\mathcal{X}_m(r_0+j+1, r_0, \epsilon_{j+1}) \cap \mathcal{N}| < |\mathcal{X}_m(r_0+j, r_0, \epsilon_j) \cap \mathcal{N}|,$$

or both hold. Since

$$|\mathcal{X}_M(r_0, r_0, \epsilon_0) \cap \mathcal{N}| + |\mathcal{X}_m(r_0, r_0, \epsilon_0) \cap \mathcal{N}| \leq |\mathcal{N}| = N,$$

there exists $T < N$ such that one of the sets

$$\begin{aligned} &\mathcal{X}_M(r_0+T, r_0, \epsilon_T) \cap \mathcal{N}, \\ &\mathcal{X}_m(r_0+T, r_0, \epsilon_T) \cap \mathcal{N}, \end{aligned}$$

or both, is empty. It follows that in the former case,

$$M_{\mathcal{N}}(r_0+T) \leq M_{\mathcal{N}}(r_0) - \epsilon_T,$$

and in the latter case,

$$m_{\mathcal{N}}(r_0 + T) \geq m_{\mathcal{N}}(r_0) + \epsilon_T.$$

Since

$$\epsilon_0 > \epsilon_1 > \cdots > \epsilon_T \geq \epsilon_{N-1} > 0,$$

we have

$$\begin{aligned}
\Psi(r_0{+}N - 1) - \Psi(r_0) &\leq \Psi(r_0 + T) - \Psi(r_0) \\
&\leq (M_{\mathcal{N}}(r_0 + T) - M_{\mathcal{N}}(r_0)) \\
&\qquad + (m_{\mathcal{N}}(r_0) - m_{\mathcal{N}}(r_0 + T)) \\
&\leq -\epsilon_T \\
&\leq -\epsilon_{N-1}.
\end{aligned}$$

Therefore,

$$\Psi(r_0 + N - 1) \leq \Psi(r_0)(1 - \alpha^{N-1}/2).$$

Define $c = (1 - \alpha^{N-1}/2)$. Since c is not a function of r_0 and r_0 was chosen arbitrarily, it follows that

$$\Psi(r_0 + k(N - 1)) \leq c^k \Psi(r_0),$$

for all $k \in \mathbb{Z}_{\geq 0}$. Because $c < 1$, it follows that $\Psi(r) \to 0$ as $r \to \infty$. □

7.4.3 Sufficient Condition for F-Local Malicious Model

Theorem 7.7 (Sufficient Condition, F-Local)**.** *Consider a time-invariant asynchronous network under the local broadcast model. Suppose the communication is described by a digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$, where each normal node uses the Asynchronous W-MSR algorithm with parameter F. Then, under the F-local malicious model, resilient asymptotic consensus is achieved if the network topology is $(3F + 1)$-robust.*

Proof. In this case, the sets $\mathcal{X}_M$ and $\mathcal{X}_m$ are defined to include only normal nodes. Then, the $(3F + 1)$-robust assumption under the F-local model ensures at least one normal value outside of either $\mathcal{X}_M$ or $\mathcal{X}_m$ will be used in the update (at most up to F values could be from adversaries and have values greater than $M_{\mathcal{N}}(r)$, F smallest or largest values are removed, and F values are ignored due to time delays, leaving still one normal value outside that is used). The rest of the analysis is identical to the proof of Theorem 7.6. $\square$

7.4.4 RAC in Asynchronous Networks With Quantization

In this section, we show that if the processor can compute the exact values used in update (63), then by rounding this estimate, the quantized values will converge in finite time to values that are approximately equal, up to the quantization precision of the quantizer. In the special case where the limit to which the unquantized consensus process converges is equal to its quantized value, exact consensus is shown to hold. In order to analyze this situation, we require the following definitions. Let

$$\mathcal{I}_\Delta \triangleq [\min \Delta, \max \Delta],$$

and for $x \in \mathcal{I}_\Delta$ define $\lceil x \rceil_\Delta = \arg\min\{v \in \Delta \colon v \geq x\}$, $\lfloor x \rfloor_\Delta = \arg\max\{v \in \Delta \colon v \leq x\}$, and $\mu(x) = (\lceil x \rceil_\Delta + \lfloor x \rfloor_\Delta)/2$. For $x \notin \mathcal{I}_\Delta$, define $\lceil x \rceil_\Delta = \lfloor x \rfloor_\Delta = \mu(x) = \max \Delta$ whenever $x > \max \Delta$, and $\lceil x \rceil_\Delta = \lfloor x \rfloor_\Delta = \mu(x) = \min \Delta$ whenever $x < \min \Delta$. Note that $\lceil x \rceil_\Delta - \lfloor x \rfloor_\Delta = 0$ if $x \in \Delta$ or $x \notin \mathcal{I}_\Delta$, and $\lceil x \rceil_\Delta - \lfloor x \rfloor_\Delta = \delta$ otherwise. The following lemma is important in the main result.

Lemma 7.8. *Given a known δ-set Δ, with quantizer q_Δ, then for any $x \in \mathcal{I}_\Delta$, the following inequality holds:*

$$|q_\Delta(x) - x| + |x - \mu(x)| \leq \delta/2.$$

Proof. Observe that $q_\Delta(x) \in \{\lfloor x \rfloor_\Delta, \lceil x \rceil_\Delta\}$, so that $|q_\Delta(x) - \mu(x)| = \delta/2$ if $x \neq q_\Delta(x)$ and $|q_\Delta(x) - \mu(x)| = 0$ if $x = q_\Delta(x)$. Thus, $|q_\Delta(x) - x| + |x - \mu(x)| = 0$ whenever $x = q_\Delta(x)$; so, assume $x \neq q_\Delta(x)$ (and therefore $x \neq \lceil x \rceil_\Delta$ and $x \neq \lfloor x \rfloor_\Delta$). If $x \in (\lfloor x \rfloor_\Delta, \mu(x))$, then

$q_\Delta(x) = \lfloor x \rfloor_\Delta$, so that

$$\begin{aligned} |q_\Delta(x) - x| + |x - \mu(x)| &= |\lfloor x \rfloor_\Delta - x| + |x - \mu(x)| \\ &= x - \lfloor x \rfloor_\Delta + \mu(x) - x \\ &= \mu(x) - \lfloor x \rfloor_\Delta \\ &= \delta/2. \end{aligned}$$

Likewise, if $x \in [\mu(x), \lceil x \rceil_\Delta)$, then $q_\Delta(x) = \lceil x \rceil_\Delta$, so that

$$\begin{aligned} |q_\Delta(x) - x| + |x - \mu(x)| &= |\lceil x \rceil_\Delta - x| + |x - \mu(x)| \\ &= \lceil x \rceil_\Delta - x + x - \mu(x) \\ &= \lceil x \rceil_\Delta - \mu(x) \\ &= \delta/2. \end{aligned}$$

□

Theorem 7.9. *Consider a time-invariant asynchronous network under the local broadcast model and in the presence of malicious adversaries under the F-total model. Suppose the communication is described by a digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$, where each normal node uses the Quantized Asynchronous W-MSR algorithm with parameter F. Let Δ be some δ-set. Then, whenever $\mathcal{D}$ is ($2F+1, F+1$)-robust there exists a round $r' \in \mathbb{Z}_{>0}$ such that for all $r \geq r'$, QRAC is achieved. Moreover, if L is not a midpoint of two values in Δ, then for all $r \geq r'$, $|v_i(r) - v_j(r)| = 0$ for $i, j \in \mathcal{N}$, where L denotes the limit to which the values in $x_\mathcal{N}(r)$ converge (guaranteed by Theorem 7.6).*

Proof. The values $x_i(r)$ converge to a common limit by Theorem 7.6, which we call L. Hence, for all $\epsilon > 0$, there exists $r_\epsilon \in \mathbb{Z}_{>0}$ such that for all $r \geq r_\epsilon$, $|x_i(r) - L| < \epsilon$. First, suppose that $L \notin \Delta$. There are two cases to consider: $L \in \mathcal{I}_\Delta = [\min \Delta, \max \Delta]$ or $L \notin \mathcal{I}_\Delta$. Suppose first that $L \in \mathcal{I}_\Delta$ and that $L = \mu(L)$. Let $\epsilon = \delta/2$, which implies that for $r \geq r_{\delta/2}$, $x_i(r) \in (\lfloor L \rfloor_\Delta, \lceil L \rceil_\Delta) \subset \mathcal{I}_\Delta$ so that $\mu(x_i(r)) = \mu(L) = L$ for all $i \in \mathcal{N}$. It follows that for all $r \geq r_{\delta/2}$ and all $i, j \in \mathcal{N}$,

$$\begin{aligned}
|v_i(r) - v_j(r)| &= |q_\Delta(x_i(r)) - q_\Delta(x_j(r))| \\
&\leq |q_\Delta(x_i(r)) - L| + |L - q_\Delta(x_j(r))| \\
&\leq |q_\Delta(x_i(r)) - x_i(r)| + |x_i(r) - L| + |L - x_j(r)| + |x_j(r) - q_\Delta(x_j(r))| \\
&\leq \delta/2 - |x_i(r) - \mu(x_i(r))| + |x_i(r) - L| + |L - x_j(r)| + \delta/2 - |x_j(r) - \mu(x_j(r))| \\
&\leq \delta - |x_i(r) - L| + |x_i(r) - L| + |L - x_j(r)| - |x_j(r) - L| \\
&\leq \delta
\end{aligned}$$

where we have used Lemma 7.8 in going from the third to the fourth line, and the fact that $\mu(x_i(r)) = \mu(L) = L$ in going from the fourth to the fifth line.

Next, suppose $L \in \mathcal{I}_\Delta$, $L \notin \Delta$, and $L \neq \mu(L)$. Then, let $\epsilon = \min(|L - \mu(L)|, |L - q_\Delta(L)|)$, which implies that for $r \geq r_{\delta/2}$, $x_i(r)$ is not only in $(\lfloor L \rfloor_\Delta, \lceil L \rceil_\Delta) \subset \mathcal{I}_\Delta$, but in the same $\delta/2$ length subinterval as L for all $i \in \mathcal{N}$ (i.e., either in $(\lfloor L \rfloor_\Delta, \mu(L))$ if $L \in (\lfloor L \rfloor_\Delta, \mu(L))$ or in $(\mu(L), \lceil L \rceil_\Delta)$ if $L \in (\mu(L), \lceil L \rceil_\Delta)$). It follows that $q_\Delta(x_i(r)) = q_\Delta(L)$ for all $i \in \mathcal{N}$, and hence $|v_i(r) - v_j(r)| = 0$ for $i, j \in \mathcal{N}$ whenever $r \geq r_\epsilon$.

If $L \notin \mathcal{I}_\Delta$, then fix $\epsilon < \min_{v \in \Delta} |v - L| = \text{dist}(L, \Delta)$. Then, for all $r \geq r_\epsilon$, it follows from the definition of q_Δ that $q_\Delta(x_i(r)) = q_\Delta(x_j(r))$ for $i, j \in \mathcal{N}$.

Finally, if $L \in \Delta$, then set $\epsilon = \delta/2$, which implies that for all $r \geq r_\epsilon$, $x_i(r) \in (L - \delta/2, L + \delta/2)$ for $i \in \mathcal{N}$. Suppose $L \neq q_\Delta(x_i(r))$ for some $i \in \mathcal{N}$ and some $r \geq r_\epsilon$. Then there exists $v \in \Delta$ with $v \neq L$ such that $|v - x_i(r)| \leq |L - x_i(r)| < \delta/2$. Then,

$$|v - L| \leq |v - x_i(r)| + |x_i(r) - L| < \delta,$$

which contradicts the fact that Δ is a δ-set. Therefore, $L = q_\Delta(x_i(t))$ for all $i \in \mathcal{N}$ and $r \geq r_\epsilon$, so that $|v_i(r) - v_j(r)| = 0$ for $i, j \in \mathcal{N}$ whenever $r \geq r_\epsilon$. $\square$

Note that we cannot do better than the error of δ with this approach because if L is the midpoint of two values $v_1, v_2 \in \Delta$ with $v_2 - v_1 = \delta$, then values converging to L from below are rounded to v_1 while values converging to L from above are rounded to v_2, for all $r \in \mathbb{Z}_{>0}$.

7.5 Point-to-Point Communication Model in Related Work

Although the Byzantine approximate agreement problem was posed more than twenty-five years ago, the necessary and sufficient topological condition on a time-invariant network for a successful iterative algorithm to exist in the presence of up to F Byzantine nodes has been an open problem (for both synchronous and asynchronous networks) until very recently [194, 193, 195]. Synchronous networks under the F-total Byzantine model are studied in [194, 193], and both synchronous and asynchronous networks are studied in [195]. In [194], Vaidya et al. provide the tight condition required in synchronous directed networks to ensure convergence (instead of finite termination) of any iterative consensus algorithm in the presence of up to F Byzantine faulty nodes. In order to state the condition, we require the following definition, which provides a common notation from the definitions considered separately in [194] and [195].

Definition 7.10. *For nonempty, disjoint sets of nodes $A, B \subset \mathcal{V}$, $A \overset{r}{\Rightarrow} B$ if and only if there exists a node $v \in B$ that has at least r in-neighbors in A; i.e., $|\mathcal{N}_v^{in} \cap A| \geq r$. $A \overset{r}{\not\Rightarrow} B$ if and only if $A \overset{r}{\Rightarrow} B$ is not true.*

Given the relation of Definition 2.1, the tight condition may be stated as follows. For all quadruples of sets of nodes $\mathcal{F}, L, C, R$ that form a partition[54] of $\mathcal{V}$ such that $0 \leq |\mathcal{F}| \leq F$, $|L| > 0$, and $|R| > 0$, at least one of the two following conditions must hold true: (i) $R \cup C \overset{F+1}{\Rightarrow} L$ or (ii) $L \cup C \overset{F+1}{\Rightarrow} R$. Observe that this condition requires sufficient redundancy of directed edges between subsets of normal nodes in the network. Note that the condition can be restated in terms of robustness; i.e., the subdigraph induced by the normal nodes must be $(F+1)$-robust.

The necessary and sufficient condition for time-invariant asynchronous networks with a point-to-point communication model in the presence of up to F Byzantine nodes is given in [195]. The condition can also be stated using the relation of Definition 2.1. For all quadruples of sets of nodes $\mathcal{F}, L, C, R$ that form a partition of $\mathcal{V}$ such that $0 \leq |\mathcal{F}| \leq F$, $|L| > 0$, and $|R| > 0$, at least one of the two following conditions must hold true: (i) $R \cup C \overset{2F+1}{\Rightarrow} L$ or (ii) $L \cup C \overset{2F+1}{\Rightarrow} R$. Note that the condition can be restated in terms of robustness; i.e., the subdigraph induced by the normal nodes must be $(2F+1)$-robust.

[54]Here, sets $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_p \subseteq \mathcal{S}$ are said to form a **partition** of set $\mathcal{S}$ if $\cup_{i=1}^{p} \mathcal{S}_i = \mathcal{S}$ and $\mathcal{S}_i \cap \mathcal{S}_j = \emptyset$ for $i \neq j$. Note that in this context, some of the sets in the partition may be empty.

The concept of robust networks is introduced by Zhang and Sundaram in [210], where it is shown to be a useful property in studying the resilience of distributed algorithms (including consensus and broadcast algorithms) in the presence of F-local adversaries. A refined definition (which is the one presented here, with finer granularity through the introduction of parameter s) is given in [116, 115], in order to formulate the necessary and sufficient condition to achieve resilient asymptotic consensus in time-invariant synchronous networks under the F-total malicious model. Note that robust networks are quite common. In [210, 116, 115], it is shown that the robustness of the seed graph in the well-known preferential attachment model for scale-free networks [2] is maintained throughout the growth of the network. Moreover, it is shown in [209] that random networks also exhibit robustness properties.

7.6 Summary

In this chapter, we studied the problem of reaching consensus asymptotically in the presence of adversary nodes whenever the network is asynchronous under a local broadcast model of communication. The type of adversary considered is omniscient and may collude with other adversaries to achieve the goal of disrupting consensus among the normal nodes. The main limitation on the behavior of the adversary nodes is that whenever the adversary nodes communicate with neighbors, they must broadcast their messages so that all neighbors receive the same information. The asynchronous consensus algorithm studied here uses local strategies to ensure resilience against the adversary nodes. The class of topologies studied are those that are *robust*. Network robustness formalizes the notion of redundancy of direct information exchange between subsets of nodes in the network, and is an important property for analyzing the behavior of resilient distributed algorithms that use only local information. We also considered a quantization scheme to terminate in finite time while ensuring approximate agreement with the maximum error given by the precision of the quantizer.

CHAPTER VIII

RESILIENT ASYMPTOTIC SYNCHRONIZATION OF LINEAR SYSTEMS

Synchronization, like consensus, is a group objective where the agents seek to agree on their state values. Synchronization differs from consensus in the fact that the state values *dynamically change in the absence of influence from neighboring agents in the network*. Whereas consensus is an agreement process on *values*, synchronization is an agreement process on *the underlying dynamics*. Most often, synchronization phenomena arise in physical or biological systems where complex interactions in the coupled dynamics of the underlying physical processes cause the agents to synchronize. At times this behavior is undesirable and even harmful, such as the synchronization that occurs in the brain of an epileptic patient. Other times, synchronization is explicitly sought, and appropriate control laws or algorithms are designed to drive the agents to synchrony [171, 211].

A major challenge in the synchronization objective in multi-agent networks is the design of local coupling rules (controllers) that facilitate synchronization of the agents' dynamics. One aspect to this challenge is the possibility of complex and dynamic interaction topologies. Of course, this is a common issue with consensus objectives. A second aspect, unique to synchronization, is the complication of nonlinear, possibly chaotic, agent dynamics [176]. Further, if the agents are not identical, then synchronization to common dynamics may not be feasible, e.g., if the individual agents have no common equilibrium and a synchronization manifold does not exist [211]. Instead, in these cases, the error dynamics of each agent with respect to the *average dynamics* should be bounded near the origin [211].

Another major challenge is achieving synchronization resiliently in the presence of compromised nodes, or adversaries. As far as the author is aware, this problem has not been previously studied.[55] However, the need for resilient synchronization algorithms increases with the increasing number of applications in which synchronization is required. It is not difficult to imagine the damage that could be caused by a Stuxnet-like worm[56] insinuated into a networked multi-agent system with

[55]Resilient clock synchronization has been studied [129, 109, 120]. However, these techniques achieve agreement resiliently on logical clock values, instead of agreement on the oscillator dynamics. In the jargon of this manuscript, this type of clock synchronization is physically dependent consensus, but *not* synchronization.

[56]See http://en.wikipedia.org/wiki/Stuxnet and the references therein (accessed July 26, 2012).

synchronization algorithms that are not resilient. Instead of only destroying the infected nodes, the entire networked system could be destroyed. Therefore, the formulation of resilient synchronization problems and the design of controllers that ensure resilience are of utmost importance.

In this chapter, we study resilient asymptotic synchronization (RAS) of linear time-invariant (LTI) systems. The normal agents in the network are identical LTI systems. The goal is for each normal agent to asymptotically synchronize to a common open-loop trajectory of the system (i.e., a trajectory of the unforced system, with zero input) despite the influence of adversary agents. It is assumed that each LTI system is weakly stable, stabilizable, and detectable. Resilient synchronization controllers are designed for both full-state and output feedback. In the case of output feedback, a resilient Luenberger observer is introduced to construct an estimate of the full state. The weighted Adversarial Robust Consensus Protocol with selective reduce (ARC-P2) is used in the resilient synchronization controller in order to ensure resilience to adversaries. This chapter focuses on the F-total malicious model, and it is shown that RAS is achieved in the same robust network topologies as the consensus results for synchronous networks as shown in Chapters V and VI. The results of the chapter may readily be extended to the other adversary models as was done in previous chapters.

8.1 Background in Matrix Theory and LTI Systems

In this section we recall pertinent definitions, terminologies, and results from matrix theory [80] and LTI system theory [6]. We use the following notations common in numerical analysis [177, 196, 140]. The set of eigenvalues of matrix $A \in \mathbb{R}^{m \times m}$ is $\lambda(A) = \{\lambda \in \mathbb{C} \colon \det(A - \lambda I) = 0\}$. The largest real part of any eigenvalue of A is denoted $\alpha(A) = \max\{\mathrm{Re}(\lambda) \colon \lambda \in \lambda(A)\}$. The dimension of the eigenspace of A corresponding to a particular eigenvalue is referred to as the *geometric multiplicity* of the eigenvalue. On the other hand, the multiplicity of an eigenvalue as a root of the characteristic polynomial of A is called the *algebraic multiplicity* of the eigenvalue. If the geometric multiplicity of any eigenvalue of A is strictly less than its algebraic multiplicity, then A is said to be *defective*. Such an eigenvalue (with smaller geometric multiplicity than algebraic multiplicity) is said to be *defect* [177].

The state-space form of an LTI system is given by

$$\dot{x} = Ax + Bu,$$
$$y = Cx + Du,$$

where $x \in \mathbb{R}^m$ is the state, $u \in \mathbb{R}^r$ is the control input, $y \in \mathbb{R}^s$ is the output, $A \in \mathbb{R}^{m\times m}$, $B \in \mathbb{R}^{m\times r}$, $C \in \mathbb{R}^{s\times m}$, and $D \in \mathbb{R}^{s\times r}$. We assume from here on that $D \equiv 0$. For each initial condition $x(0) = x_0 \in \mathbb{R}^m$ and for any given input $u\colon \mathbb{R} \to \mathbb{R}^r$ the LTI system has a unique solution for all $t \in \mathbb{R}$; i.e., $x\colon \mathbb{R} \to \mathbb{R}^m$ exists and is unique. The exponential of matrix $A \in \mathbb{R}^{m\times m}$ is defined by

$$e^{At} = \sum_{k=0}^{\infty}(At)^k/k! \quad \text{for } t \in \mathbb{R}_{\geq 0}.$$

For an LTI system, the matrix exponential of A commutes with A. That is,

$$Ae^{At} = e^{At}A.$$

Further, e^{At} is the state transition matrix that characterizes the solution of the system [6]; i.e.,

$$x(t) = e^{At}x(0) + \int_0^t e^{A(t-\tau)}Bu_i(\tau)d\tau, \quad \text{for } t \in \mathbb{R}_{\geq 0},$$

and, in particular, if the system is unforced (i.e., $u \equiv 0$)

$$x(t) = e^{At}x(0), \quad \text{for } t \in \mathbb{R}_{\geq 0}.$$

Therefore, bounds on the matrix exponential can be useful for bounding the trajectory of the linear system.

An important bound, derived by Dahlquist [44], is

$$||e^{At}||_2 \leq e^{\mu(A)t}, \ \ \forall t \in \mathbb{R}_{\geq 0}, \tag{65}$$

where $||\cdot||_2$ is the spectral norm and

$$\mu(A) = \max\{\mu \colon \mu \in \lambda((A^{\mathsf{H}} + A)/2)\}.$$

is the *logarithmic norm* of A [177]. Note that A^{H} is the conjugate transpose, or *Hermitian transpose*, of A, so that $(A^{\mathsf{H}} + A)/2$ is a real-symmetric matrix and therefore has real eigenvalues [80]; hence $\mu(A) \in \mathbb{R}$.

Next, we define what it means for a matrix to be *stable*, or *weakly stable*.

Definition 8.1 ([177])**.** *The matrix $A \in \mathbb{R}^{m\times m}$ is a **stable matrix** if $\alpha(A) < 0$. The matrix A is said to be **weakly stable** if it is stable or if $\alpha(A) = 0$ but no eigenvalue λ with Re$(\lambda) = \alpha(A)$ is defect.*

We refer to an eigenvalue $\lambda \in \lambda(A)$ as *stable* if $\mathrm{Re}(\lambda) < 0$, *weakly stable* if $\mathrm{Re}(\lambda) = 0$ but λ is *non-defect*, and *unstable* otherwise. We present a couple of examples to explicate these definitions.

Example 8.1. *Consider the double integrator with state $x = [x_1, x_2]^{\mathsf{T}}$ and input $u \in \mathbb{R}$ described by the system*

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = Ax + Bu = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u.$$

The solution of the unforced system (i.e., $u \equiv 0$) is

$$\begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} = \begin{bmatrix} x_1(0) + x_2(0)t \\ x_2(0) \end{bmatrix}.$$

which grows unbounded for nontrivial initial condition $x_2(0) \neq 0$. The matrix A has a single defect eigenvalue 0 with algebraic multiplicity 2 and geometric multiplicity 1. Thus, A is not weakly stable and 0 is an unstable eigenvalue.

Example 8.2. *Consider the linear oscillator with frequency $\omega/(2\pi)$, where $\omega \in \mathbb{R}_{>0}$, and state $x = [x_1, x_2]^{\mathsf{T}}$. The system is given by*

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = Ax = \begin{bmatrix} 0 & -\omega \\ \omega & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix},$$

and has solution

$$\begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} = \begin{bmatrix} x_1(0)\cos(\omega t) - x_2(0)\sin(\omega t) \\ x_1(0)\sin(\omega t) + x_2(0)\cos(\omega t) \end{bmatrix}.$$

In this case, A has eigenvalues $\pm\omega\mathsf{i}$ (where $\mathsf{i} = \sqrt{-1}$), so that $\alpha(A) = 0$. Note that A is nondefective. Therefore, A is weakly stable and both eigenvalues are weakly stable.

To complete our review of the relationships between the stability of an LTI system, the matrix A, and its matrix exponential, we recall that e^{At} is bounded only if A is weakly stable and approaches zero asymptotically only if A is stable [177]. This fact may readily be deduced from (65) and the following theorem.

Theorem 8.2 ([177])**.** *A matrix $A \in \mathbb{R}^{m\times m}$ is stable (weakly stable) if and only if $\mu(A) < 0$ ($\mu(A) \leq 0$).*

We now recall some facts about stabilizability and detectability of LTI systems. If the pair (A, B) is stabilizable, then the unstable eigenvalues of A can be arbitrarily assigned (and therefore made stable) in the matrix $A + BK$ for some gain matrix $K \in \mathbb{R}^{r\times m}$ [6]. If the pair (A, C) is detectable, then the unobservable eigenvalues are stable. This means there exists an observability matrix $H \in \mathbb{R}^{m\times s}$ such that the estimated state $\hat{x}$ of the Luenberger observer

$$\dot{\hat{x}} = A\hat{x} + Bu + H(\hat{y} - y),$$

with estimate output $\hat{y} = C\hat{x}$ converges asymptotically to the true state x [6]. Thus, the error $e(t) = x(t) - \hat{x}(t)$ vanishes asymptotically.

In the remainder of this section, we review the Jordan canonical form [80] and demonstrate a related block diagonal form that proves useful in the design of our resilient synchronization control law. The Jordan decomposition theorem states that any square complex-valued matrix is similar to a block diagonal matrix containing Jordan blocks. A *Jordan block* $J_j(\lambda)$ is a $j \times j$ upper triangular matrix of the form [80]

$$J_j(\lambda) = \begin{bmatrix} \lambda & 1 & \dots & 0 & 0 \\ 0 & \lambda & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & \lambda & 1 \\ 0 & 0 & \dots & 0 & \lambda \end{bmatrix}. \tag{66}$$

Thus, for any square matrix $A \in \mathbb{C}^{m \times m}$ there exists an invertible $m \times m$ matrix P such that

$$J = P^{-1}AP,$$

where J is a Jordan matrix given by

$$J = \begin{bmatrix} J_{m_1}(\lambda_1) & 0 & \dots & 0 \\ 0 & J_{m_2}(\lambda_2) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & J_{m_k}(\lambda_k) \end{bmatrix}, \tag{67}$$

such that $m_1 + m_2 + \cdots + m_k = m$. The orders of the m_i may not be distinct and the eigenvalues λ_i need not be distinct [80]. The number of Jordan blocks corresponding to eigenvalue λ_i is equal to the geometric multiplicity λ_i. The sum of the sizes of the Jordan blocks with eigenvalue λ_i is the algebraic multiplicity of λ_i. These facts imply that if A is weakly stable, then the Jordan blocks corresponding to eigenvalues on the imaginary axis must be 1×1 matrices containing only the corresponding eigenvalue. This fact is useful in obtaining a block diagonal form related to the Jordan form.

Lemma 8.3. *For any square, real-valued, weakly stable matrix $A \in \mathbb{R}^{m \times m}$, there exists an invertible $m \times m$ matrix Q such that*

$$R = Q^{-1}AQ,$$

where R is a block diagonal matrix given by

$$R = \begin{bmatrix} J_{m_1}(\lambda_1) & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & J_{m_2}(\lambda_2) & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & J_{m_p}(\lambda_p) & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & R_2(\lambda_{p+1}, \lambda_{p+2}) & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & \dots & R_2(\lambda_{p+2q-1}, \lambda_{p+2q}) \end{bmatrix}.$$

Each $J_{m_i}(\lambda_i)$, for $i = 1, \dots, p$, is a Jordan block of the form (66) corresponding to eigenvalue λ_i either with negative real part or with $\lambda_i = 0$. If $\lambda_i = 0$, then $m_i = 1$ and $J_{m_i}(\lambda_i)$ is a 1×1 zero matrix. The first z Jordan blocks (with $0 \le z \le p$) account for the zero eigenvalues. The remaining $p - z$ Jordan blocks correspond to eigenvalues with negative real part. Each $R_2(\lambda_{p+2j-1}, \lambda_{p+2j})$, for $j = 1, 2, \dots, q$, is a 2×2 matrix of the form

$$R_2(\lambda_{p+2j-1}, \lambda_{p+2j}) = \begin{bmatrix} 0 & -\omega_j \\ \omega_j & 0 \end{bmatrix}, \tag{68}$$

where $\lambda_{p+2j-1} = -\omega_j \mathrm{i}$ and $\lambda_{p+2j} = \omega_j \mathrm{i}$, with $\omega_j \neq 0$ and $\mathrm{i} = \sqrt{-1}$.

Proof. Since A is weakly stable, all eigenvalues have nonpositive real parts and any eigenvalues with zero real part are non-defect. This implies that any zero eigenvalue corresponds to a 1×1 Jordan block. Further, because A is real-valued, any complex eigenvalues occur in conjugate pairs. By the Jordan decomposition theorem, there exists a Jordan matrix $J = P^{-1}AP$ of the form of (67) such that

(a) $J_{m_i}(\lambda_i)$, for $i = z + 1, \dots, p$, is a Jordan block of the form (66) corresponding to eigenvalue λ_i with negative real part;

(b) If $z \ge 1$ (i.e., A has at least one zero eigenvalue), then $J_{m_i}(\lambda_i) = 0$ is a 1×1 zero matrix for $i = 1, \dots, z$;

(c) $J_{m_l}(\lambda_l)$, for $l = p + 1, p + 2, \dots, k$, is a 1×1 Jordan block containing a nonzero eigenvalue on the imaginary axis. The λ_l's are ordered so that $\lambda_{p+2j-1} = -\omega_j \mathrm{i}$ and $\lambda_{p+2j} = \omega_j \mathrm{i}$, for $j = 1, 2, \ldots, q$, where $k = p + 2q$. Note that the ω_j's in different pairs need not be distinct.

For each 2×2 matrix defined in (68) there exists a 2×2 invertible matrix Q_{ω_j} such that

$$\begin{aligned} J_{R_2}(\lambda_{p+2j-1}, \lambda_{p+2j}) &= Q_{\omega_j}^{-1} R_2(\lambda_{p+2j-1}, \lambda_{p+2j}) Q_{\omega_j} \\ &= \begin{bmatrix} -\omega_j i & 0 \\ 0 & \omega_j i \end{bmatrix} = \begin{bmatrix} J_{m_{p+2j-1}}(\lambda_{p+2j-1}) & 0 \\ 0 & J_{m_{p+2j}}(\lambda_{p+2j}) \end{bmatrix}. \end{aligned}$$

Define

$$P' = \begin{bmatrix} I_{m_1} & 0 & \ldots & 0 & 0 & \ldots & 0 \\ 0 & I_{m_2} & \ldots & 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & I_{m_p} & 0 & \ldots & 0 \\ 0 & 0 & \ldots & 0 & Q_{\omega_1}^{-1} & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 0 & 0 & \ldots & Q_{\omega_q}^{-1}) \end{bmatrix},$$

where I_{m_j} is the identity matrix in $\mathbb{R}^{m_j \times m_j}$. Then a straightforward calculation shows that $Q = PP'$ yields the result. □

Using the modified Jordan form of Lemma 8.3, we can change the coordinate system of an LTI system with weakly stable $A \in \mathbb{R}^{m \times m}$ so that it consists of decoupled LTI subsystems.[57] To see this, suppose $R = Q^{-1}AQ$ is the modified Jordan form of A given in Lemma 8.3, and consider the coordinate transformation

$$x = Q\bar{x}.$$

Then, the dynamics in terms of $\bar{x}$ are

$$\dot{\bar{x}} = R\bar{x} + B'u, \tag{69a}$$

$$y = C'\bar{x} + Du, \tag{69b}$$

[57]Of course, any LTI system may be decoupled using the Jordan form. The utility of the decomposition described here is demonstrated in subsequent sections.

where $B' = Q^{-1}B$ and $C' = CQ$. Let

$$\bar{x} = \begin{bmatrix} \bar{x}_{m_1} \\ \bar{x}_{m_2} \\ \vdots \\ \bar{x}_{m_p} \\ \bar{x}_{m_{p+1}} \\ \vdots \\ \bar{x}_{m_{p+q}} \end{bmatrix} \quad \text{and} \quad B' = \begin{bmatrix} B'_{m_1} \\ B'_{m_2} \\ \vdots \\ B'_{m_p} \\ B'_{m_{p+1}} \\ \vdots \\ B'_{m_{p+q}} \end{bmatrix},$$

where $\bar{x}_{m_i} \in \mathbb{R}^{m_i}$ and $B'_{m_i} \in \mathbb{R}^{m_i \times r}$ for $i = 1, 2, \ldots, p$, and $\bar{x}_{m_{p+j}} \in \mathbb{R}^2$ and $B'_{m_{p+j}} \in \mathbb{R}^{2\times 2}$ for $j = 1, 2, \ldots, q$. Each of these components corresponds to a block in the matrix R. Using this notation, we may rewrite the state equation (69a) above as $p+q$ decoupled state equations, where

$$\dot{\bar{x}}_{m_i} = J_{m_i}(\lambda_i)\bar{x}_{m_i} + B'_{m_i}u, \;\; i = 1, 2, \ldots, p, \tag{70a}$$

$$\dot{\bar{x}}_{m_{p+j}} = R_2(\lambda_{p+2j-1}, \lambda_{p+2j})\bar{x}_{m_{p+j}} + B'_{m_{p+j}}u, \;\; j = 1, 2, \ldots, q. \tag{70b}$$

Note that if A has at least one zero eigenvalue (i.e., $z \geq 1$), then $\bar{x}_{m_i} \in \mathbb{R}$ in (70a) has integrator dynamics for $i = 1, \ldots, z$. On the other hand, if A has at least one eigenvalue with negative real part, then $\bar{x}_{m_j}$ has exponentially stable dynamics for $j = z+1, \ldots, p$. This can be shown using the matrix exponential of a Jordan block $J_{m_j}(\lambda)$, which has the form [6]

$$e^{J_{m_j}(\lambda)t} = e^{\lambda t} \begin{bmatrix} 1 & t & \ldots & \frac{t^{m_j-1}}{(m_j-1)!} \\ 0 & 1 & \ldots & \frac{t^{m_j-2}}{(m_j-2)!} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 1 \end{bmatrix}. \tag{71}$$

In general, if λ is a stable eigenvalue, then any norm of the matrix exponential $e^{J_{m_j}(\lambda)t}$ converges exponentially to zero. It follows that each subsystem in (70a) with stable eigenvalue converges exponentially to zero whenever $u \equiv 0$.

The matrix exponential of $R_2(\lambda_{p+2j-1}, \lambda_{p+2j})$ where $\lambda_{p+2j-1} = -\omega_j \mathrm{i}$ and $\lambda_{p+2j} = \omega_j \mathrm{i}$, with

$\omega_j \neq 0$ and $\mathsf{i} = \sqrt{-1}$, is given by

$$e^{R_2(\lambda_{p+2j-1}, \lambda_{p+2j})t} = \begin{bmatrix} \cos(\omega_j t) & -\sin(\omega_j t) \\ \sin(\omega_j t) & \cos(\omega_j t) \end{bmatrix}. \tag{72}$$

This matrix exponential is a time-varying rotation matrix with angular frequency ω_j. The unforced solution ($u \equiv 0$) of the weakly stable subsystems in (70b) has the form

$$\bar{x}_{m_{p+j}}(t) = \begin{bmatrix} \cos(\omega_j t) & -\sin(\omega_j t) \\ \sin(\omega_j t) & \cos(\omega_j t) \end{bmatrix} \bar{x}_{m_{p+j}}(0), \quad j = 1, 2, \ldots, q.$$

8.2 Multi-Agent Network Model and Problem Statement

Consider a time-varying network modeled by the (finite, simple) *digraph*, $\mathcal{D}(t) = (\mathcal{V}, \mathcal{E}(t))$, where $\mathcal{V} = \{1, ..., n\}$ is the *node set* and $\mathcal{E}(t) \subset \mathcal{V} \times \mathcal{V}$ is the *directed edge set* at time t. Each directed edge $(j, i) \in \mathcal{E}(t)$ models *information flow* and indicates that node i can be influenced by (or receive information from) node j at time t. The node set is partitioned into a set of N *normal nodes* $\mathcal{N} = \{1, 2, \ldots, N\}$ and a set of M *adversary nodes* $\mathcal{A} = \{N+1, N+2, \ldots, n\}$, with $M = n - N$. Let $\Gamma_n = \{\mathcal{D}_1, \ldots, \mathcal{D}_d\}$ denote the set of all digraphs on n nodes, which is of course a finite set. Note that $\mathcal{D}(t) \in \Gamma_n$ for all $t \in \mathbb{R}_{\geq 0}$.

The time-varying topology of the network is governed by a piecewise constant switching signal $\sigma(\cdot)$, which is defined on $\mathbb{R}_{\geq 0}$ and takes values in $\{1, \ldots, d\}$. In order to emphasize the role of the switching signal, we denote $\mathcal{D}_{\sigma(t)} = \mathcal{D}(t)$. Note that time-invariant networks are represented by defining $\mathcal{D}_{\sigma(t)} \equiv \mathcal{D}_s$, or by simply dropping the dependence on time t.

8.2.1 Normal Agent Dynamics

The agents are assumed to be *identical*. Each normal agent $i \in \mathcal{N}$ has state $x_i \in \mathbb{R}^m$, control input $u_i \in \mathbb{R}^r$, and output $y_i \in \mathbb{R}^s$. The dynamics of each normal agent $i \in \mathcal{N}$ is given by the linear time-invariant (LTI) system

$$\dot{x}_i = Ax_i + Bu_i, \tag{73a}$$

$$y_i = Cx_i. \tag{73b}$$

We assume (i) A is weakly stable, (ii) (A, B) is stabilizable, and (iii) (A, C) is detectable.

The state $x_i(t) \in \mathbb{R}^m$ of normal agent $i \in \mathcal{N}$ at time t has components $x_{i,1}, x_{i,2}, \dots, x_{i,m}$. Similarly, its output $y_i(t) \in \mathbb{R}^s$ has components $y_{i,1}, y_{i,2}, \dots, y_{i,s}$. This notation defines unambiguously the state and output of node i at time t for any node that is not deceptive (e.g., a normal node or malicious adversary). However, in order to handle deceptive adversaries, we let $x_{(j,i),k}(t)$ and $y_{(j,i),k}(t)$ denote, respectively, the k-th component of the state and output of agent j *intended* for agent i at time t. Whenever $(j, i) \in \mathcal{E}(t)$, $y_{(j,i),k}(t)$ is the k-th component of the output of agent j *conveyed* to agent i at time t. Note that even if $(j, i) \notin \mathcal{E}(t)$, $y_{(j,i),k}(t)$ and $x_{(j,i),k}(t)$ are still defined for all k. In the case that $j \in \mathcal{N}$ is normal, we define $y_{(j,i),k}(t) \equiv y_{j,k}(t)$ and $x_{(j,i),k}(t) \equiv x_{j,k}(t)$. On the other hand, if $j \in \mathcal{A}$ is an adversary, then $x_{(j,i),k}(t)$ and $y_{(j,i),k}(t)$ is the k-th component of the state and output trajectory, respectively, that adversary j would like to convey to agent i, but the topological constraints on the network prevent it from doing so. Observe that either agent i receives $y_{(j,i),k}(t)$ for all components $k \in \{1, 2, \dots, s\}$ (if $j \in \mathcal{N}_i^{\text{in}}(t)$), or for none of the components (if $j \notin \mathcal{N}_i^{\text{in}}(t)$). With this terminology, we denote the vector containing the k-th component of the states of all nodes in $\mathcal{N}$, $\mathcal{A}$, or $\mathcal{V}$ intended for agent i by

$$x_{(\mathcal{N},i),k}(t) = [x_{(1,i),k}(t), \dots, x_{(N,i),k}(t)]^\mathsf{T} = [x_{1,k}(t), \dots, x_{N,k}(t)]^\mathsf{T} \in \mathbb{R}^N,$$

$$x_{(\mathcal{A},i),k}(t) = [x_{(N+1,i),k}(t), \dots, x_{(n,i),k}(t)]^\mathsf{T} \in \mathbb{R}^M,$$

or

$$x_{(\mathcal{V},i),k}(t) = [x_{(1,i),k}(t), \dots, x_{(n,i),k}(t)]^\mathsf{T} \in \mathbb{R}^n,$$

respectively. The overall state vector of $\mathcal{N}$, $\mathcal{A}$, or $\mathcal{V}$ intended for agent i is denoted

$$x_{\mathcal{N}}(t) = x_{(\mathcal{N},i)}(t) = [x^\mathsf{T}_{(\mathcal{N},i),1}(t), \dots, x^\mathsf{T}_{(\mathcal{N},i),m}(t)]^\mathsf{T} \in \mathbb{R}^{Nm},$$

$$x_{(\mathcal{A},i)}(t) = [x^{\mathsf{T}}_{(\mathcal{A},i),1}(t), \dots, x^{\mathsf{T}}_{(\mathcal{A},i),m}(t)]^{\mathsf{T}} \in \mathbb{R}^{Mm},$$

or

$$x_{(\mathcal{V},i)}(t) = [x^{\mathsf{T}}_{(\mathcal{V},i),1}(t), \dots, x^{\mathsf{T}}_{(\mathcal{V},i),m}(t)]^{\mathsf{T}} \in \mathbb{R}^{nm},$$

respectively. Since $x_{(\mathcal{N},i)}(t) \equiv x_{(\mathcal{N},j)}(t)$ for all $i, j \in \mathcal{N}$, we unambiguously define $x_{\mathcal{N}}(t) = x_{(\mathcal{N},i)}(t)$ for any $i \in \mathcal{V}$. Finally, we denote the vector containing all adversary states intended for normal nodes by $x_{(\mathcal{A},\mathcal{N})}(t) = [x^{\mathsf{T}}_{(\mathcal{A},1)}(t), \dots, x^{\mathsf{T}}_{(\mathcal{A},N)}(t)]^{\mathsf{T}} \in \mathbb{R}^{MNm}$. In a similar manner, we define the k-th component of the collective *outputs* of $\mathcal{N}$, $\mathcal{A}$, or $\mathcal{V}$ intended for agent i, and so on.

8.2.2 Resilient Asymptotic Synchronization (RAS)

For Resilient Asymptotic Synchronization (RAS), we consider the intervals $\mathcal{I}_{t,k}$ defined by the k-th component of the states of the normal nodes at time t as follows. Let $\mathcal{I}_{t,k} = [m_{\mathcal{N},k}(t), M_{\mathcal{N},k}(t)]$, where $m_{\mathcal{N},k}(t) = \min_{i\in\mathcal{N}}\{x_{i,k}(t)\}$ and $M_{\mathcal{N},k}(t) = \max_{i\in\mathcal{N}}\{x_{i,k}(t)\}$ are the minimum and maximum values, respectively, of the k-th component of the states of the normal nodes at time t. Then, for each t we define the m-dimensional *orthotope* (or *hyperrectangle*) $\mathcal{H}_{t,\mathcal{N}}$ constructed from the intervals $\mathcal{I}_{t,k}$ by

$$\mathcal{H}_{t,\mathcal{N}} = \mathcal{I}_{t,1} \times \mathcal{I}_{t,2} \times \cdots \times \mathcal{I}_{t,m}.$$

In RAS there is no restriction on the initial states of the normal nodes; i.e., it is allowed that $x_i(0) \in \mathbb{R}^m$ for all $i \in \mathcal{N}$. However, it is implicitly assumed that the normal states are *safe* in the sense that they lie in some safe set $\mathcal{S}_{0,\mathcal{N}} \subset \mathbb{R}^m$. It is further assumed that the safe set $\mathcal{S}_{0,\mathcal{N}}$ contains the orthotope defined by the initial states of the normal nodes, $\mathcal{H}_{0,\mathcal{N}}$; i.e., $\mathcal{H}_{0,\mathcal{N}} \subseteq \mathcal{S}_{0,\mathcal{N}}$. With these concepts, the RAS problem is defined as follows.

Definition 8.4 (Resilient Asymptotic Synchronization)**.** *Suppose the normal agents are identical LTI systems described by (73) and have initial states $x_i(0) \in \mathbb{R}^m$ for $i \in \mathcal{N}$. Let $\mathcal{S}_{0,\mathcal{N}} \subset \mathbb{R}^m$ be a safe set that contains the orthotope $\mathcal{H}_{0,\mathcal{N}}$; i.e., $\mathcal{H}_{0,\mathcal{N}} \subseteq \mathcal{S}_{0,\mathcal{N}}$. Then the normal agents are said to achieve* ***resilient asymptotic synchronization*** *(RAS) in the presence of adversary nodes (given a particular adversary model) if there exists an (open-loop) trajectory $x_0(t)$ that satisfies $\dot{x}_0(t) = Ax_0(t)$ for all $t \in \mathbb{R}_{\geq 0}$ with $x_0(0) \in \mathcal{S}_{0,\mathcal{N}}$, such that the normal states asymptotically* *converge to x_0. That is,*

$$\lim_{t\to\infty} ||x_i(t) - x_0(t)||_2 = 0, \ \forall i \in \mathcal{N}. \tag{74}$$

A few remarks are in order with respect to the RAS problem. First, because the normal agents converge to an "open-loop" trajectory of the system, it is important that the "open-loop" system has no unstable modes. However, it is possible that the matrix A in the LTI system of (73) is a result local stabilization through an appropriate feedback controller, so that the "open-loop" system is in fact a closed-loop feedback control system. Regardless of whether (73) defines the dynamics of a plant or a feedback control system, the control input u_i is viewed as the feedback control input from the multi-agent network.

The RAS problem is related to the resilient asymptotic consensus (RAC) problem. Instead of converging to a common limit as in RAC, the states of the normal agents must converge to the open-loop trajectory $x_0(t)$ (and therefore common dynamics). Also, observe that the the open-loop trajectory $x_0(t)$ to which the normal agents must converge satisfies the safety condition $x_0(0) \in \mathcal{S}_{0,\mathcal{N}}$. The safety condition requires that the adversaries are not able to drive the normal agents to follow an open-loop trajectory with extreme initial values.

8.3 Resilient Asymptotic Synchronization Analysis

In this section, we analyze resilient control laws that achieve RAS in the presence of up to F malicious nodes (F-total model). The control input uses weighted ARC-P with selective reduce (ARC-P2) as one component of the control law. Therefore, we first revisit resilient asymptotic consensus with multi-dimensional signals. After this, we study RAS in the case of full-state feedback. Finally, we examine RAS with only output feedback.

8.3.1 Resilient Asymptotic Consensus with Vector States

It is shown in Chapter 5 that network robustness is the key property for characterizing ARC-P2 in time-invariant networks with scalar states under the F-total malicious model. It turns out the results of Chapter 5 are easily extended to the case of multi-dimensional states. The intuition is as follows. Each component of the state vector may be handled independently of the others by applying the resilient scalar consensus algorithm on each component of the vector state. To be more precise, we present a generalization of RAC for vector states. For this definition, as in Chapter 5, we assume that the nodes have access to the states in their inclusive neighborhood and that the nodes are integrators; i.e., $\dot{x}_i = u_i$. Further, let $M_{\mathcal{N},k}(t)$ and $m_{\mathcal{N},k}(t)$ be the *maximum* and *minimum* values of the k-th component ($k \in \{1, \dots, m\}$) of the states of the normal nodes at time t, respectively.

Definition 8.5 (Vector-Valued Continuous-Time Resilient Asymptotic Consensus (VVCTRAC))**.** *The normal agents are said to achieve* ***vector-valued continuous-time resilient asymptotic consensus*** *(VVCTRAC) in the presence of adversary nodes (given a particular adversary model) if for each* $k \in \{1, \dots, m\}$

(i) $\exists L_k \in \mathbb{R}$ *such that* $\lim_{t\to\infty} x_{i_k}(t) = L_k$ *for all* $i \in \mathcal{N}$*, and*

(ii) $x_{i_k}(t) \in \mathcal{I}_{0,k} = [m_{\mathcal{N},k}(0), M_{\mathcal{N},k}(0)]$ *for all* $t \in \mathbb{R}_{\geq 0}$*,* $i \in \mathcal{N}$ *(i.e.,* $\mathcal{I}_{0,k}$ *is a positively invariant set for each component* $k \in \{1, 2, \dots, m\}$ *of the normal nodes' states),*

for any choice of initial states $x_i(0) \in \mathbb{R}^m$*, for* $i \in \mathcal{N}$*, and any choice of adversary trajectories,* $x_{(\mathcal{A},\mathcal{N})}(t)$ *for* $t \in \mathbb{R}_{\geq 0}$.

To achieve VVCTRAC, we apply ARC-P2 on each component of the state vectors. To define the weights in the protocol, we assume there is a nonnegative, bounded, and piecewise continuous weight vector $w_{(j,i)}(t)$ associated to each pair $(j, i) \in \mathcal{V} \times \mathcal{N}$. The k-th component of the weight vector $w_{(j,i)}(t)$ is denoted $w_{(j,i),k}(t)$ and is uniformly bounded above by $\beta \in \mathbb{R}_{>0}$. We define (without loss of generality) $w_{(j,i)}(t) \equiv 0$ for $j \notin \mathcal{N}_i^{\text{in}}(t)$. Whenever $j \in \mathcal{N}_i^{\text{in}}(t)$, there is the uniform lower bound $\alpha \in \mathbb{R}_{>0}$, so the weights are bounded as $0 < \alpha \leq w_{(j,i),k}(t) \leq \beta$.

Recall the notation for the ascending sorting function, $\rho_k(z)$ for $z \in \mathbb{R}^k$, the weighted zero-selective reduce function with respect to $F \in \mathbb{Z}_{\geq 0}$ and $k \in \mathbb{Z}_{>0}$, $r^k_{0,F}(z, w)$ for $z \in \mathbb{R}^k$, $w \in \mathbb{R}^k$, and composition of the two, $\phi^k_{0,F}(z, w) = r^k_{0,F}(\rho_k(z), w)$, given in Definition 5.1 on page 154. Extend this definition to the composite vectors $z, w \in \mathbb{R}^{km}$ with $z = [z_1^\mathsf{T}, \dots, z_m^\mathsf{T}]^\mathsf{T}$ and $w = [w_1^\mathsf{T}, \dots, w_m^\mathsf{T}]^\mathsf{T}$ such that $z_i, w_i \in \mathbb{R}^k$ for $i = 1, \dots, m$, by defining $\Phi^{k,m}_{0,F} \colon \mathbb{R}^{km} \to \mathbb{R}^m$ such that

$$\Phi^{k,m}_{0,F} \colon \mathbb{R}^{km}(z, w) = \begin{bmatrix} \phi^k_{0,F}(z_1, w_1) \\ \vdots \\ \phi^k_{0,F}(z_m, w_m) \end{bmatrix}. \tag{75}$$

Also, recall the definition of the Kronecker product [80]. Given $B \in \mathbb{R}^{m\times n}$ and $C \in \mathbb{R}^{p\times q}$, the *Kronecker product* $B \otimes C \in \mathbb{R}^{mp\times nq}$ is defined as

$$B \otimes C \triangleq \begin{bmatrix} b_{11}C & \dots & b_{1n}C \\ \vdots & \ddots & \vdots \\ b_{m1}C & \dots & b_{mn}C \end{bmatrix}.$$

Then, the update rule of the vector valued version of weighted ARC-P with selective reduce and parameter F is given by

$$\begin{aligned} \dot{x}_i = f_{i,\sigma(t)}(t, x_{\mathcal{N}}, x_{(\mathcal{A},i)}) &= \Phi_{0,F}^{d_i(t),m}\left([I_m \otimes N_i(t)]x_{(\mathcal{V},i)}(t) - [x_i(t) \otimes 1_{d_i(t)}], w_i(t)\right) \\ &= \begin{bmatrix} \phi_{0,F}^{d_i(t)}\left(N_i(t)x_{(\mathcal{V},i),1}(t) - x_{i,1}(t)1_{d_i(t)}, w_{i,1}(t)\right) \\ \vdots \\ \phi_{0,F}^{d_i(t)}\left(N_i(t)x_{(\mathcal{V},i),m}(t) - x_{i,m}(t)1_{d_i(t)}, w_{i,m}(t)\right) \end{bmatrix}, \end{aligned} \tag{76}$$

for each normal node $i \in \mathcal{N}$ for $t \in \mathbb{R}_{\geq 0}$. The time-varying weight vector

$$w_i(t) = [w_{i,1}^\mathsf{T}(t), \dots, w_{i,m}^\mathsf{T}(t)]^\mathsf{T}$$

with

$$w_{i,k}(t) = [w_{(i_{1,k}(t),i),k}(t), w_{(i_{2,k}(t),i),k}(t), \dots, w_{(i_{d_i(t),k}(t),i),k}(t)]^\mathsf{T},$$

satisfies the bound $0 < \alpha \leq w_{(i_{j,k}(t),i),k}(t) \leq \beta$ for all $j = 1, 2, \dots, d_i(t)$ and $k = 1, 2, \dots, m$, where $i_{1,k}(t), i_{2,k}(t), \dots, i_{d_i(t),k}(t)$ are the node indices of the neighbors of node i in the order determined by the sorting function of component k at time t. For example, if the k-th component of neighbor $j \in \mathcal{N}_i^{\text{in}}(t)$ has the second smallest value in the neighborhood of i at time t (as determined by the sorting function), then the weight $w_{(i_{2,k}(t),i),k}(t)$ is equal to $w_{(j,i),k}(t)$. Finally, recall that $N_i(t) \in \mathbb{R}^{d_i(t)\times n}$ is a sparse matrix with each row corresponding to a distinct $j \in \mathcal{N}_i^{\text{in}}(t)$ such that each row has a single 1 in the j-th column. Thus, there is a one-to-one correspondence between $j \in \mathcal{N}_i^{\text{in}}(t)$ and rows in $N_i(t)$.

With these definitions and notation, it is straightforward to recreate all of the results of Chapter 5 for the vector case by applying the scalar results as m independent consensus problems. Thus, we may state the following results.

Corollary 8.6 (VVTRAC Safety with F-Total or F-Local Byzantine Model)**.** *Suppose each normal node updates its value according to the vector-value ARC-P2 with parameter F under the F'-total or F'-local Byzantine model, where $F' \leq F$. Then, for each normal node $i \in \mathcal{N}$, $x_i(t) \in \mathcal{H}_{0,\mathcal{N}}$ for all $t \in \mathbb{R}_{\geq 0}$, regardless of the network topology.*

Corollary 8.7 (VVTRAC Agreement with F-Total Malicious Model)**.** *Consider a time-invariant network modeled by a directed graph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ where each normal node updates its value according to ARC-P2 with parameter F. Then, VVCTRAC is achieved under the F-total uniformly continuous malicious model if and only if the network topology is $(F+1, F+1)$-robust.*

8.3.2 RAS with Full State Feedback

In this section, we study RAS whenever full-state feedback is available; i.e., $C = I_m$ so that $y_i = x_i$ for all $i \in \mathcal{N}$. The resilient control law uses the modified Jordan form of Lemma 8.3. Specifically, for A in (73a), there exists invertible Q such that

$$R = Q^{-1}AQ$$

has the form of Lemma 8.3. The main idea of the resilient control law is to decouple the dynamics and allow the stable components to converge to zero unforced. The weakly stable components are the only ones that explicitly require synchronization. If $\lambda_i = 0$, for $i = 1, \ldots, z$, then these weakly stable components have integrator dynamics, and correspond to the first z blocks in R. The remaining weakly stable components correspond to blocks $R_2(\lambda_{p+2j-1}, \lambda_{p+2j})$ in R, as defined in (68). For these components, we consider its matrix exponential (given in (72)), and the inverse of its matrix exponential. We define the following modified exponential matrices:

$$E_R(t) = \begin{bmatrix} E_1 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & E_2 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & E_p & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & e^{R_2(\lambda_{p+1},\lambda_{p+2})t} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & \dots & e^{R_2(\lambda_{p+2q-1},\lambda_{p+2q})t} \end{bmatrix} \quad (77)$$

and

$$F_R(t) = \begin{bmatrix} E_1 & 0 & \dots & 0 & 0 & \dots & 0 \\ 0 & E_2 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & E_p & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & e^{-R_2(\lambda_{p+1},\lambda_{p+2})t} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & \dots & e^{-R_2(\lambda_{p+2q-1},\lambda_{p+2q})t} \end{bmatrix}, \quad (78)$$

where $E_i = 1$ if $\lambda_i = 0$, and $E_i = 0$ otherwise (in the latter case, $m_i > 1$ is possible so E_i is an $m_i \times m_i$ zero matrix). In either case,

$$E_R(t)F_R(t) = F_R(t)E_R(t) = \begin{bmatrix} E_1 & 0 & \dots & 0 & 0 \\ 0 & E_2 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & E_p & 0 \\ 0 & 0 & \dots & 0 & I_{2q} \end{bmatrix}.$$

Moreover, both $E_R(t)$ and $F_R(t)$ commute with R. That is,

$$RE_R(t) = E_R(t)R$$

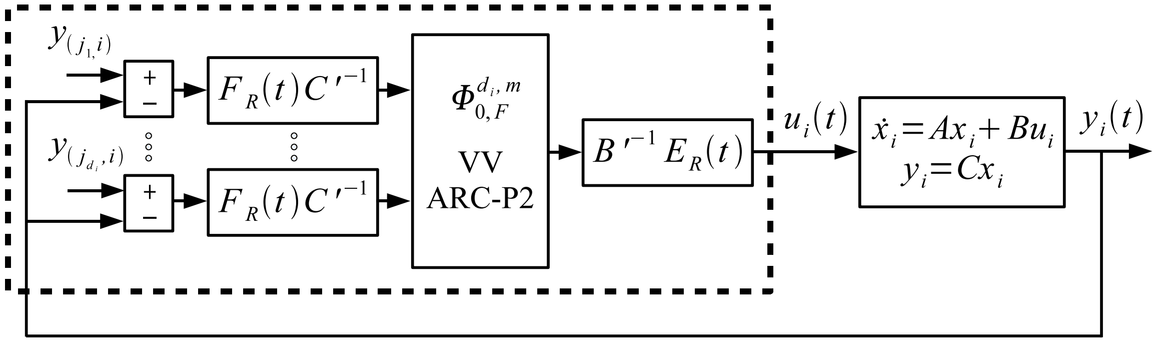


Figure 48: Resilient synchronization control law of Lemma 8.8 for agent i.

and

$$RF_R(t) = F_R(t)R.$$

The modified exponential matrices $E_R(t)$ and $F_R(t)$ are the keys to reducing resilient synchronization to resilient consensus. The following lemma demonstrates how the modified exponential matrices may be combined with vector-valued ARC-P2 to achieve RAS whenever the matrices B and C are invertible. The resilient synchronization controller introduced in the lemma is shown for normal agent i in Figure 48.

Lemma 8.8. *Suppose each agent $i \in \mathcal{N}$ is an LTI system as in (73) and there are at most F uniformly continuous malicious agents (F-total model). Let $B, C \in \mathbb{R}^{m\times m}$ be nonsingular matrices and suppose that A is weakly stable. Assume that the multi-agent network is time-invariant and modeled by digraph $\mathcal{D}$ that is $(F+1, F+1)$-robust. Further, suppose each normal agent $i \in \mathcal{N}$ uses the following control law for all $t \in \mathbb{R}_{\geq 0}$:*

$$u_i(t) = B'^{-1}E_R(t)\Phi_{0,F}^{d_i,m}\left(\tilde{N}_i[I_n \otimes (F_R(t)C'^{-1})]y_{(\mathcal{V},i)}(t) - [(F_R(t)C'^{-1}y_i(t)) \otimes 1_{d_i}], w_i(t)\right), \tag{79}$$

where $\tilde{N}_i = I_m \otimes N_i$, $R = Q^{-1}AQ$ is given as in Lemma 8.3, $B' = Q^{-1}B$, $C' = CQ$, $E_R(t)$ is defined by (77), $F_R(t)$ is given in (78), and $\Phi_{0,F}^{d_i,m}(\cdot,\cdot)$ is the vector-valued ARC-P2 function defined in (75) with $k = d_i$ and parameter F. Then RAS is achieved.

Proof. The closed-loop system for normal node i is given by

$$\dot{x}_i(t) = Ax_i(t) + QE_R(t)\Phi_{0,F}^{d_i,m}\left(\tilde{N}_i[I_n \otimes F_R(t)Q^{-1}]x_{(\mathcal{V},i)}(t) - [(F_R(t)Q^{-1}x_i(t)) \otimes 1_{d_i}], w_i(t)\right) \tag{80}$$

Make the change in coordinates for all $i \in \mathcal{V}$

$$x_i = Q\bar{x}_i.$$

The closed-loop system may be rewritten as

$$\dot{\bar{x}}_i(t) = R\bar{x}_i(t) + E_R(t)\Phi_{0,F}^{d_i,m}\left(\tilde{N}_i[I_n \otimes F_R(t)]\bar{x}_{(\mathcal{V},i)}(t) - [(F_R(t)\bar{x}_i(t)) \otimes 1_{d_i}], w_i(t)\right). \tag{81}$$

To facilitate analysis of the decoupled system in the $\bar{x}_i$ coordinates, let

$$\bar{x}_i = \begin{bmatrix} \bar{x}_i^{m_1} \\ \bar{x}_i^{m_2} \\ \vdots \\ \bar{x}_i^{m_p} \\ \bar{x}_i^{m_{p+1}} \\ \vdots \\ \bar{x}_i^{m_{p+q}} \end{bmatrix}, \ i \in \mathcal{N}$$

with elements $\bar{x}_i^{m_j} = [\bar{x}_{i,1}^{m_j}, \bar{x}_{i,2}^{m_j}, \ldots, \bar{x}_{i,m_j}^{m_j}]^\mathsf{T} \in \mathbb{R}^{m_j}$. Notice in (81) that the stable components of $\bar{x}_i$ (i.e., $\bar{x}_i^{m_j}$ for $j = z+1, \ldots, p$) are allowed to evolve freely, and only the weakly stable components are affected by the control law. In the following analysis, we show that each of the components $\bar{x}_i^{m_j}$ asymptotically synchronize to a corresponding component of an open-loop solution of $\dot{\bar{x}}_0 = R\bar{x}_0$ such that the initial condition of each element $\bar{x}_{0,k}$ satisfies

$$\bar{x}_{0,k}(0) \in \left[\min_{i\in\mathcal{N}}\{\bar{x}_{i,k}(0)\}, \max_{i\in\mathcal{N}}\{\bar{x}_{i,k}(0)\}\right].$$

To facilitate this analysis, denote

$$\bar{x}_0 = \begin{bmatrix} \bar{x}_0^{m_1} \\ \bar{x}_0^{m_2} \\ \vdots \\ \bar{x}_0^{m_p} \\ \bar{x}_0^{m_{p+1}} \\ \vdots \\ \bar{x}_0^{m_{p+q}} \end{bmatrix}.$$

For each stable component $\bar{x}_i^{m_j} \in \mathbb{R}^{m_j}$, for $j = z+1, \ldots, p$, the Jordan block $J_{m_j}(\lambda_j)$ contains a stable eigenvalue. The matrix exponential is given by (71), which implies that these components exponentially converge to zero. Thus, for *any* open-loop trajectory of

$$\dot{\bar{x}}_0^{m_j} = J_{m_j}(\lambda_j)\bar{x}_0^{m_j},$$

there exists $\kappa_j \colon \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ with $\kappa_j(t) \to 0$ as $t \to \infty$, such that for all $t \geq 0$

$$||\bar{x}_i^{m_j}(t) - e^{J_{m_j}(\lambda_j)t}\bar{x}_0^{m_j}(0)||_2 \leq \kappa_j(t)||\bar{x}_i^{m_j}(0) - \bar{x}_0^{m_j}(0)||_2, \;\; \forall i \in \mathcal{N}.$$

In particular, there exists an open-loop solution that satisfies the above inequality with initial values $\bar{x}_{0,k}^{m_j}(0)$, for $k = 1, 2, \ldots, m_j$, such that

$$\bar{x}_{0,k}^{m_j}(0) \in \left[\min_{i \in \mathcal{N}}\{\bar{x}_{i,k}^{m_j}(0)\}, \max_{i \in \mathcal{N}}\{\bar{x}_{i,k}^{m_j}(0)\}\right].$$

If A has $z \geq 1$ (non-defect) zero eigenvalues, then each component $\bar{x}_i^{m_k}(t) \in \mathbb{R}$, $k = 1, \ldots, z$, evolves as an integrator using ARC-P2. It follows from Corollary 8.6 and 8.7 that there exists $\bar{x}_0^{m_k}(0) \in [\min_{i \in \mathcal{N}}\{\bar{x}_i^{m_k}(0)\}, \max_{i \in \mathcal{N}}\{\bar{x}_i^{m_k}(0)\}]$ and $\kappa_k \colon \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ with $\kappa_k(t) \to 0$ as $t \to \infty$, such that for all $t \geq 0$

$$||\bar{x}_i^{m_k}(t) - \bar{x}_0^{m_k}(0)||_2 \leq \kappa_k(t)||\bar{x}_i^{m_k}(0) - \bar{x}_0^{m_k}(0)||_2, \;\; \forall i \in \mathcal{N}.$$

Next, consider the remaining weakly stable components $\bar{x}_i^{m_{p+j}} \in \mathbb{R}^2$, for $j = 1, 2, \ldots, q$, and denote $R_2(\lambda_{p+2j-1}, \lambda_{p+2j}) = R_{2,j}$ for brevity. The closed-loop system for this component is given by

$$\begin{aligned}\dot{\bar{x}}_i^{m_{p+j}} &= R_{2,j}\bar{x}_i^{m_{p+j}} \\ &\quad + e^{R_{2,j}t}\Phi_{0,F}^{d_i,2}\left([I_2 \otimes N_i][I_n \otimes e^{-R_{2,j}t}]\bar{x}_{(\mathcal{V},i)}^{m_{p+j}}(t) - [(e^{-R_{2,j}t}\bar{x}_i^{m_{p+j}}(t)) \otimes 1_{d_i}], w_i(t)\right).\end{aligned}$$

Consider the change of variable

$$z_i^j(t) = e^{-R_{2,j}t}\bar{x}_i^{m_{p+j}}(t), \;\; i \in \mathcal{V}.$$

Then, for all $i \in \mathcal{N}$

$$\begin{aligned}\dot{z}_i^j &= -R_{2,j}e^{-R_{2,j}t}\bar{x}_i^{m_{p+j}} \\ &\quad + e^{-R_{2,j}t}\left(R_{2,j}\bar{x}_i^{m_{p+j}} + e^{R_{2,j}t}\Phi_{0,F}^{d_i,2}\left([I_2 \otimes N_i]z_{(\mathcal{V},i)}^j(t) - [z_i^j(t) \otimes 1_{d_i}], w_i(t)\right)\right) \\ &= \Phi_{0,F}^{d_i,2}\left([I_2 \otimes N_i]z_{(\mathcal{V},i)}^j(t) - [z_i^j(t) \otimes 1_{d_i}], w_i(t)\right)\end{aligned}$$

where we have used the fact that $R_{2,j}e^{-R_{2,j}t} = e^{-R_{2,j}t}R_{2,j}$ [6] in going from the first to second equality. It follows from Corollary 8.7 that the z_i^j's asymptotically converge to a common value $\bar{x}_0^{m_{p+j}}(0) \in \mathbb{R}^2$. Since $z_i^j(0) = \bar{x}_i^{m_{p+j}}(0)$ for all $i \in \mathcal{N}$, Corollary 8.6 implies that the common limit of the consensus process $\bar{x}_0^{m_{p+j}}(0)$ satisfies for each element $k \in \{1, 2\}$,

$$\bar{x}_{0,k}^{m_{p+j}}(0) \in \left[\min_{i\in\mathcal{N}}\{\bar{x}_{i,k}^{m_{p+j}}(0)\}, \max_{i\in\mathcal{N}}\{\bar{x}_{i,k}^{m_{p+j}}(0)\}\right].$$

Because the z_i^j's asymptotically converge to $\bar{x}_0^{m_{p+j}}(0)$, there exists a positive-definite function $\kappa_{p+j}\colon \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ that satisfies $\kappa_{p+j}(t) \to 0$ as $t \to \infty$, such that for all $t \geq 0$

$$||z_i^j(t) - \bar{x}_0^{m_{p+j}}(0)||_2 \leq \kappa_{p+j}(t)||z_i^j(0) - \bar{x}_0^{m_{p+j}}(0)||_2, \;\; \forall i \in \mathcal{N}.$$

By multiplying each side of the inequality by $||e^{R_{2,j}t}||_2$ and using the submultiplicative property of matrix norms [80], it follows that

$$||\bar{x}_i^{m_{p+j}}(t) - e^{R_{2,j}t}\bar{x}_0^{m_{p+j}}(0)||_2 \leq \kappa_{p+j}(t)||e^{R_{2,j}t}||_2||\bar{x}_i^{m_{p+j}}(0) - \bar{x}_0^{m_{p+j}}(0)||_2, \ \ \forall i \in \mathcal{N},$$

where we have also used the fact that $z_i^j(0) = \bar{x}_i^{m_{p+j}}(0)$. It follows from Theorem 8.2 and (65) that there exists $\delta_{p+j} \geq 0$ such that

$$||\bar{x}_i^{m_{p+j}}(t) - e^{R_{2,j}t}\bar{x}_0^{m_{p+j}}(0)||_2 \leq \kappa_{p+j}(t)e^{-\delta_{p+j}t}||\bar{x}_i^{m_{p+j}}(0) - \bar{x}_0^{m_{p+j}}(0)||_2, \ \ \forall i \in \mathcal{N}$$

Since $\kappa_{p+j}(t) \to 0$ as $t \to \infty$, resilient asymptotic synchronization is achieved for each weakly stable component $\bar{x}_i^{m_{p+j}} \in \mathbb{R}^2$ for $j = 1, 2, \ldots, q$ in the $\bar{x}_i$ coordinates.

Combining the above inequalities, it follows that there exists $\kappa\colon \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ with $\kappa(t) \to 0$ as $t \to \infty$, such that for all $t \geq 0$

$$||\bar{x}_i(t) - e^{Rt}\bar{x}_0(0)||_2 \leq \kappa(t)||\bar{x}_i(0) - \bar{x}_0(0)||_2, \ \ \forall i \in \mathcal{N},$$

where the elements $\bar{x}_{0,k}$ for $k = 1, 2, \ldots, m$ have initial values that satisfy

$$\bar{x}_{0,k}(0) \in \left[\min_{i\in\mathcal{N}}\{\bar{x}_{i,k}(0)\}, \max_{i\in\mathcal{N}}\{\bar{x}_{i,k}(0)\}\right],$$

so that

$$x_{0,k}(0) \in \left[m_{\mathcal{N},k}(t), M_{\mathcal{N},k}(t)\right],$$

where $x_0 = Q\bar{x}_0$. Finally, multiplying each side of the above inequality by $||Q||_2$, using again the submultiplicative property of matrix norms, and substituting $\bar{x}_0(t) = e^{Rt}\bar{x}_0(0)$, it follows that

$$||x_i(t) - e^{At}x_0(0)||_2 \leq \kappa(t)||Q||_2||Q^{-1}||_2||x_i(0) - x_0(0)||_2, \ \ \forall i \in \mathcal{N}.$$

Since $\kappa(t) \to 0$ as $t \to \infty$, RAS is achieved.

□

Theorem 8.9. *Suppose each agent $i \in \mathcal{N}$ is an LTI system as in (73) with full-state feedback (i.e., $C = I_m$) and there are at most F malicious nodes (F-total model) with uniformly continuous state*

x_k and controller state η_k for $k \in \mathcal{A}$. Assume that A is weakly stable, the pair (A, B) is stabilizable, and let $K \in \mathbb{R}^{m\times r}$ be a stabilizing matrix such that $A + BK$ is Hurwitz. Assume that the multi-agent network is modeled by digraph $\mathcal{D}$ that is $(F+1, F+1)$-robust. Further, suppose each normal agent $i \in \mathcal{N}$ executes the dynamic control law with controller state η_i that is initially relaxed (i.e., $\eta_i(0) = 0$ for all $i \in \mathcal{N}$), and is given by

$$\begin{aligned}\dot{\eta}_i &= (A + BK)\eta_i - QE_R\Phi_{0,F}^{d_i,m}\left(\tilde{N}_i[I_n \otimes F_RQ^{-1}]s_{(\mathcal{V},i)} - [(F_RQ^{-1}s_i) \otimes 1_{d_i^{\text{in}}}], w_i\right)\\ u_i &= K\eta_i,\end{aligned} \tag{82}$$

where $\tilde{N}_i = I_m \otimes N_i$, $s_j = x_j - \eta_j$ for $j \in \mathcal{V}$, $s_{(\mathcal{V},i)} = [s_1^\mathsf{T}, s_2^\mathsf{T}, \ldots, s_n^\mathsf{T}]^\mathsf{T}$, $E_R(t)$ is defined by (77), $F_R(t)$ is given in (78), and $\Phi_{0,F}^{d_i,m}(\cdot,\cdot)$ is the filter applied in vector-valued ARC-P2 and defined in (75) with $k = d_i$. Then RAS is achieved.

Proof. The dynamics of x_i and s_i for each normal agent $i \in \mathcal{N}$ may be rewritten as

$$\dot{x}_i(t) = (A + BK)x_i(t) - BKs_i(t) \tag{83a}$$

$$\dot{s}_i(t) = As_i(t) + QE_R(t)\Phi_{0,F}^{d_i,m}\left(\tilde{N}_i[I_n \otimes F_R(t)Q^{-1}]s_{(\mathcal{V},i)}(t) - [(F_R(t)Q^{-1}s_i(t)) \otimes 1_{d_i^{\text{in}}}], w_i(t)\right) \tag{83b}$$

Observe that (83b) is decoupled from (83a) and matches (80) in the proof of Lemma 8.8. Note that since $\eta_i(0) = 0$, it follows that $s_i(0) = x_i(0)$ for all $i \in \mathcal{N}$. Therefore, Lemma 8.8 implies that the solutions of (83b) converge to a solution of $\dot{s}_0 = As_0$ such that $s_0(0) \in \mathcal{S}_{0,\mathcal{N}}$. Because the s_i's synchronize, it follows that the consensus term in (82) converges to zero. Combining this with the fact that $A + BK$ is Hurwitz, implies that $\eta_i \to 0$ as $t \to \infty$. Thus, for any $\epsilon > 0$ there exists $T \in \mathbb{R}_{>0}$ such that for all $t > T$,

$$||s_i(t) - e^{At}s_0(0)||_2 < \epsilon/2 \quad and \quad ||\eta_i(t)|| < \epsilon/2 \quad \text{for all } i \in \mathcal{N}.$$

Therefore, for $t > T$

$$||x_i(t) - e^{At}s_0(0)||_2 = ||s_i(t) + \eta_i(t) - e^{At}s_0(0)||_2 \leq ||s_i(t) - e^{At}s_0(0)||_2 + ||\eta_i(t)||_2 < \epsilon \;\; \forall i \in \mathcal{N},$$

so that RAS is achieved. □

8.3.3 RAS with Output Feedback

Theorem 8.10. *Suppose each agent $i \in \mathcal{N}$ is an LTI system as in (73) with output feedback and there are at most F malicious nodes (F-total model) with uniformly continuous observer state $\hat{x}_k$ and controller state η_k for $k \in \mathcal{A}$. Assume that A is weakly stable, the pair (A, B) is stabilizable, the pair (A, C) is detectable, and let $K \in \mathbb{R}^{m\times r}$ and $H \in \mathbb{R}^{m\times s}$ be a stabilizing and observer matrix, respectively, such that $A + BK$ and $A + HC$ are Hurwitz. Assume that the multi-agent network is modeled by digraph $\mathcal{D}$ that is $(F+1, F+1)$-robust. Further, suppose each normal agent $i \in \mathcal{N}$ executes the dynamic control law given below with controller state η_i that is initially relaxed (i.e., $\eta_i(0) = 0$ for all $i \in \mathcal{N}$) and Luenberger observer with observer states $\hat{x}_i$ for $i \in \mathcal{N}$ that are contained in some orthotope within the safe set $\mathcal{S}_{0,\mathcal{N}}$. Let*

$$\begin{aligned}\dot{\eta}_i &= (A + BK)\eta_i + H(\hat{y}_i - y_i) \\ &\quad - QE_R(t)\Phi_{0,F}^{d_i,m}\Big(\tilde{N}_i[I_n \otimes F_R(t)Q^{-1}]\hat{s}_{(\mathcal{V},i)}(t) - [(F_R(t)Q^{-1}\hat{s}_i(t)) \otimes 1_{d_i}], w_i(t)\Big)\end{aligned} \tag{84}$$

$$\dot{\hat{x}}_i = A\hat{x}_i + Bu_i + H(\hat{y}_i - y_i) \tag{85a}$$

$$u_i = K\eta_i \tag{85b}$$

$$\hat{y}_i = C\hat{x}_i, \tag{85c}$$

where $\tilde{N}_i = I_m \otimes N_i$, $\hat{s}_j = \hat{x}_j - \eta_j$ for $j \in \mathcal{V}$, $\hat{s}_{(\mathcal{V},i)} = [\hat{s}_1^{\mathsf{T}}, \hat{s}_2^{\mathsf{T}}, \dots, \hat{s}_n^{\mathsf{T}}]^{\mathsf{T}}$, $E_R(t)$ is defined by (77), $F_R(t)$ is given in (78), and $\Phi_{0,F}^{d_i,m}(\cdot,\cdot)$ is the filter applied in vector-valued ARC-P2 and defined in (75) with $k = d_i$. Then RAS is achieved.

Proof. Define $e_i = x_i - \hat{x}_i$. Then, the dynamics of x_i, $\hat{s}_i$, and e_i for each normal agent $i \in \mathcal{N}$ may be rewritten as

$$\dot{x}_i(t) = (A+BK)x_i(t) - BK(e_i(t)\hat{s}_i(t)) \tag{86a}$$

$$\dot{\hat{s}}_i(t) = A\hat{s}_i(t) + QE_R(t)\Phi_{0,F}^{d_i,m}\left(\tilde{N}_i[I_n \otimes F_R(t)Q^{-1}]\hat{s}_{(\mathcal{V},i)}(t) - [(F_R(t)Q^{-1}\hat{s}_i(t)) \otimes 1_{d_i}], w_i(t)\right) \tag{86b}$$

$$\dot{e}_i(t) = (A+HC)e_i(t) \tag{86c}$$

Observe that (86b) and (86c) are decoupled from each other and from (86a). Note that (86b) matches (80) in the proof of Lemma 8.8. Since $\eta_i(0) = 0$, it follows that $\hat{s}_i(0) = \hat{x}_i(0)$ for all $i \in \mathcal{N}$. Therefore, Lemma 8.8 implies that the solutions of (83b) converge to a solution of $\dot{\hat{s}}_0 = A\hat{s}_0$ such that $\hat{s}_0(0) \in \mathcal{S}_{0,\mathcal{N}}$ (since the $\hat{x}_i(0)$'s are in some orthotope within $\mathcal{S}_{0,\mathcal{N}}$). The e_i's converge to zero because $A+HC$ is Hurwitz. Therefore, the observer error term in (84) converges to zero. Because the $\hat{s}_i$'s synchronize, it follows also that the consensus term in (84) converges to zero. Combining these with the fact that $A+BK$ is Hurwitz, implies that $\eta_i \to 0$ as $t \to \infty$. Thus, for any $\epsilon > 0$ there exists $T \in \mathbb{R}_{>0}$ such that for all $t > T$,

$$||\hat{s}_i(t) - e^{At}\hat{s}_0(0)||_2 < \epsilon/3, \quad ||e_i(t)||_2 < \epsilon/3, \quad and \quad ||\eta_i(t)|| < \epsilon/3 \ \text{ for all } i \in \mathcal{N}.$$

Therefore, for $t > T$

$$\begin{aligned} ||x_i(t) - e^{At}\hat{s}_0(0)||_2 &= ||\hat{x}_i(t) + e_i(t) - e^{At}\hat{s}_0(0)||_2 \\ &= ||\hat{s}_i(t) + \eta_i(t) + e_i(t) - e^{At}\hat{s}_0(0)||_2 \\ &\le ||\hat{s}_i(t) - e^{At}\hat{s}_0(0)||_2 + ||e_i(t)||_2 + ||\eta_i(t)||_2 \\ &< \epsilon \ \forall i \in \mathcal{N}, \end{aligned}$$

so that RAS is achieved. □

8.4 Summary

In this chapter, we demonstrate how resilient consensus may be used in more complex group objectives. In particular, we focus on resilient synchronization and show that weighted ARC-P with selective reduce may be used as part of a dynamics control law to ensure asymptotic synchronization of the states of the normal agents. To do this, we first formulate the resilient asymptotic consensus problem on vector states and show how to extend weighted ARC-P with selective reduce to the vector case. The adversary model considered in this chapter is the malicious agent under the F-total model with the additional restriction that the variables shared with the normal nodes are uniformly continuous. This technical assumption enables the reuse of the consensus results of previous chapters to ensure resilient asymptotic consensus in networks that are $(F+1, F+1)$-robust. The results of this chapter may be easily extended to time-varying networks and other threat models as was done in previous chapters on consensus.

CHAPTER IX

ALGORITHMS FOR DETERMINING NETWORK ROBUSTNESS

Network connectivity has long been the key metric in the analysis of fault-tolerant and secure distributed algorithms [50]. This is because (strong) connectivity formalizes the notion of redundant information flow across a network through independent paths. Thus, for algorithms that seek to relay or encode information across multiple hops in the network, connectivity precisely captures the necessary property for analysis [50, 92]. More generally, for tasks that require nonlocal information, such as detection of adversary nodes, connectivity is the central property for characterizing the necessary topologies [181, 159]. However, whenever purely local strategies are employed, connectivity is no longer the key metric.

For algorithms that use purely local information, the agents make decisions and act based on their sensor measurements, calculations, dynamics, and direct interactions with neighbors in the network. No global information is shared or assumed to be known. Instead, information is disseminated within components of the network in an iterative or diffusive manner, rather than being relayed or routed across the network. Therefore, purely local algorithms are well suited to large-scale dynamic networks. Indeed, purely local strategies appear to be the strategies favored in biology and nature [169]. Flocking of birds and fish is postulated to arise from local interaction rules [169]. From an engineering perspective, one example of a class of algorithms that are explicitly designed to use purely local strategies is the iterative function calculation and consensus algorithms [178].

Edge reachability and network robustness are important properties for analyzing algorithms that use purely local strategies. It has been shown that any nontrivially robust network has a directed spanning tree [210]. In fact, 1-robustness is equivalent to existence of a directed spanning tree [210]. Moreover, for iterative consensus algorithms in a fixed network, existence of a directed spanning tree is a necessary and sufficient condition for achieving agreement among the nodes [168]. However, the full utility of edge reachability and network robustness is realized only when considering fault-tolerant and resilient dissemination of information in a network through purely local strategies. This is because edge reachability – which is defined for a nonempty set – captures the requirement that enough nodes inside the set are sufficiently influenced from outside the set. There are two forms of redundancy present in the definition of edge reachability: redundancy of ingoing edges from outside and redundancy of such nodes with redundant ingoing edges from outside. Robustness is a network-wide property that stipulates a lower bound on the edge reachability properties of a sufficiently large number of subsets of nodes.

Throughout this work, we have demonstrated the utility of edge reachability and network robustness as metrics for analysis of resilient distributed algorithms that use only local information. In this chapter we propose algorithms that determine the robustness of a given network, and we extend ways to explicitly construct robust networks. In particular, two centralized algorithms are introduced. The first algorithm checks for a given amount of robustness and the second one determines the robustness of any network, regardless of its connectedness properties. These algorithms assume the topology of the network is given as input to the algorithm (encoded by the adjacency matrix). A decentralized algorithm is proposed that enables the individual nodes of an undirected, connected network to compute the robustness of the network in a decentralized manner by broadcasting information about their neighborhood in order to locally reconstruct the network topology. The centralized algorithm is then used at each node to determine the overall robustness. A modification to this decentralized algorithm is proposed in which each individual node only checks the edge reachability conditions for subsets in which it is *not* included, thus resulting in a truly distributed algorithm. For these algorithms, we analyze their complexity and examine the improvement gained by the distributed algorithm. Finally, we extend a method of Zhang and Sundaram for constructing r-robust networks [210] to the case of (r, s)-robustness. This result demonstrates that many complex networks are robust because the robust network growth model entails the preferential attachment growth model of scale-free networks [2].

9.1 Centralized Algorithms for Checking and Determining Robustness

In this section, we present centralized algorithms for checking and determining robustness. The direct manner to check a digraph to determine whether it is (r, s)-robust is a combinatorial problem. Because the sets in each pair considered are required to be nonempty and disjoint, the total number of pairs $R(n)$ that must be checked is

$$R(n) = \sum_{k=2}^{n} \binom{n}{k} \left(2^{k-1} - 1\right), \tag{87}$$

where

- $n = |\mathcal{V}|$ is the number of nodes;
- each $k = 2, 3, \ldots, n$ in the sum is the size of the k-subsets of $\mathcal{V} = \{1, 2, \ldots, n\}$. Each k-subset of $\mathcal{V}$ is partitioned into exactly two nonempty parts, $\mathcal{S}_1$ and $\mathcal{S}_2$;
- $\binom{n}{k}$ is the number of k-subsets of $\{1, 2, \ldots, n\}$;
- $2^{k-1} - 1 = S(k, 2)$ is a Stirling number of the second kind, and is the number of ways to partition a k-set into exactly two nonempty unlabeled subsets (swapping the labels $\mathcal{S}_1$ and $\mathcal{S}_2$ results in the same pair)[58].

The form of (87) implies an algorithm for checking (r, s)-robustness of a digraph, *CheckRobustness*, which is shown in Algorithm 9.1. *CheckRobustness* takes as input values of r and s[59] and the adjacency matrix of the digraph, $A(\mathcal{D})$. It returns a Boolean variable indicating whether the digraph is (r, s)-robust along with a pair of sets with which the conditions fail. If the digraph is (r, s)-robust, the sets returned are empty sets. The algorithm iterates through all possible pairs of nonempty disjoint subsets, $\mathcal{S}_1, \mathcal{S}_2 \subset \mathcal{V}$, and checks conditions (i)-(iii) of Definition 5.10 (on page 163) using the adjacency matrix. To improve the performance on digraphs that fail the test, the algorithm returns the first pair that fails.

The second algorithm, called *DetermineRobustness* (and given in Algorithm 9.2), determines (r, s)-robustness of *any* digraph, regardless of the number of components of the digraph. To do this, it requires the adjacency matrix $A(\mathcal{D})$ as input. *DetermineRobustness* first initializes r and s to the maximum values possible for any digraph on n nodes (see Property 5.19 on page 171). Then, as in *CheckRobustness*, *DetermineRobustness* iterates through all possible pairs of nonempty disjoint

[58]The quantity may be argued directly by noticing that for each of the k elements, we have two choices: $\mathcal{S}_1$ or $\mathcal{S}_2$. But, we have to subtract the two sequences of choices resulting in $\mathcal{S}_1 = \emptyset$ or $\mathcal{S}_2 = \emptyset$. Finally, because the labels on the sets $\mathcal{S}_1$ and $\mathcal{S}_2$ are unimportant to the uniqueness of the nonempty partition, we divide by 2.

[59]Since all digraphs are 0-robust, and by Property 5.19 no digraph is r-robust with $r > \lceil n/2 \rceil$, it follows that one should only check $1 \leq r \leq \lceil n/2 \rceil$, $1 \leq s \leq n$.

subsets, $\mathcal{S}_1, \mathcal{S}_2 \subset \mathcal{V}$, and checks conditions (i)-(iii) of Definition 5.10. In this case, instead of terminating upon a failing condition, the value of s is first decremented. Once all values of s are checked for the given value of r, the algorithm decrements r and restores s to its maximal value of n. If the digraph has no directed spanning tree (i.e., if it is not 1-robust), then *DetermineRobustness* returns $(r = 0, s = n)$. Using this approach, *DetermineRobustness* returns the maximal values r and s such that $\mathcal{D}$ is (r, s)-robust.[60]

[60]Recall from Section 5.3.1 that the partial order on robustness compares the value of r first, and then compares the value of s for networks with the same r.

Algorithm 9.1: $\textsc{CheckRobustness}(A(\mathcal{D}), r, s)$

```
procedure ROBUSTNESSHOLDS $(A(\mathcal{D}), \mathcal{S}_1, \mathcal{S}_2, r, s)$
  $isRSRobust \leftarrow$ false , $s_{r,1} \leftarrow 0, s_{r,2} \leftarrow 0$
  for each $k \in \{1, 2\}$
    do for each $i \in \mathcal{S}_k$
         do if $\sum_{j \in \mathcal{N}_i^{\text{in}} \setminus \mathcal{S}_k} a_{ij} \geq r$
              then $s_{r,k} \leftarrow s_{r,k} + 1$
  if $(s_{r,1} == |\mathcal{S}_1|)$ or $(s_{r,2} == |\mathcal{S}_2|)$ or $(s_{r,1} + s_{r,2} \geq s)$
    then $isRSRobust \leftarrow$ true
  return $(isRSRobust)$

main
  $isRSRobust \leftarrow$ true
  for $k \leftarrow 2$ to $n$
    do for each $K_i \in \mathcal{K}_k$ $(i = 1, 2, \ldots, \binom{n}{k})$
       comment: $\mathcal{K}_k$ is the set of $\binom{n}{k}$ unique $k$-subsets of $\mathcal{V}$
       do for each $P_j \in \mathcal{P}_{K_i}$ $(j = 1, 2, \ldots, 2^{k-1} - 1)$
          comment: $\mathcal{P}_{K_i}$ is the set of partitions of $K_i$ with exactly two nonempty parts
          do comment: $P_j = \{\mathcal{S}_1, \mathcal{S}_2\}$
             if not ROBUSTNESSHOLDS $(A(\mathcal{D}), \mathcal{S}_1, \mathcal{S}_2, r, s)$
               then $isRSRobust \leftarrow$ false
                    return $(isRSRobust, \mathcal{S}_1, \mathcal{S}_2)$
  $\mathcal{S}_1 \leftarrow \emptyset, \mathcal{S}_2 \leftarrow \emptyset$
  return $(isRSRobust, \mathcal{S}_1, \mathcal{S}_2)$
```

Algorithm 9.2: DETERMINEROBUSTNESS($A(\mathcal{D})$)

```
$r \leftarrow \min\{\delta^{\text{in}}(\mathcal{D}), \lceil n/2 \rceil\}$
$s \leftarrow n$
for $k \leftarrow 2$ to $n$
  do
    for each $K_i \in \mathcal{K}_k$ $(i = 1, 2, \ldots, \binom{n}{k})$
    comment: $\mathcal{K}_k$ is the set of $\binom{n}{k}$ unique $k$-subsets of $\mathcal{V}$
      do
        for each $P_j \in \mathcal{P}_{K_i}$ $(j = 1, 2, \ldots, 2^{k-1} - 1)$
        comment: $\mathcal{P}_{K_i}$ is the set of partitions of $K_i$ with exactly two nonempty parts
          do
            comment: $P_j = \{\mathcal{S}_1, \mathcal{S}_2\}$
            $isRSRobust \leftarrow$ ROBUSTNESSHOLDS($A(\mathcal{D}), \mathcal{S}_1, \mathcal{S}_2, r, s$)
            if ($isRSRobust ==$ false) and ($s > 0$)
              then $s \leftarrow s - 1$
            while ($isRSRobust ==$ false) and ($r > 0$)
              do
                while ($isRSRobust ==$ false) and ($s > 0$)
                  do
                    $isRSRobust \leftarrow$ ROBUSTNESSHOLDS($A(\mathcal{D}), \mathcal{S}_1, \mathcal{S}_2, r, s$)
                    if not $isRSRobust$
                      then $s \leftarrow s - 1$
                if ($isRSRobust ==$ false)
                  then
                    $r \leftarrow r - 1$
                    $s \leftarrow n$
            if $r == 0$
              then return $(r, s)$
return $(r, s)$
```

It is clear from the form of $R(n)$ in (87) that these algorithms are not efficient. In fact, the algorithms are exponential in the square root of the size of the input (in this case, the adjacency matrix, which has size n^2). To analyze the complexity of the algorithms, we recall the following definition.

Definition 9.1 (Big-O Notation)**.** *Given $f, g \colon \mathbb{R} \to \mathbb{R}$, then $f \in \mathcal{O}(g(x))$ if there exists $c \in \mathbb{R}_{>0}$ and $x_0 \in \mathbb{R}$ such that $|f(x)| \leq c|g(x)|$ for all $x \geq x_0$.*

We define the worst-case complexity of an algorithm as the maximal number of steps $T(m)$ required to complete the algorithm whenever the input is of size m. We say the worst-case complexity of an algorithm is $\mathcal{O}(g(m))$ if $T(m) \in \mathcal{O}(g(m))$.

Proposition 9.2. *Algorithms 9.1 and 9.2 have worst-case complexity $\mathcal{O}(m3^{\sqrt{m}})$ where $m = n^2$ is the size of the input (the adjacency matrix).*

Sketch of Proof: The procedure *RobustnessHolds* requires $\mathcal{O}(n^2)$ steps because $\mathcal{S}_k$ contains $\mathcal{O}(n)$ elements and the summation in the if statement requires $\mathcal{O}(n)$ steps. In worst-case, there will be $R(n)$ calls to *RobustnessHolds* in Algorithm 9.1 and $R(n) + g(n)$ in Algorithm 9.2, where $g(n) \in \mathcal{O}(n^2)$ (since in Algorithm 9.2 there will be at most an additional $(\lceil n/2 \rceil - r)(n) + n - s$ calls to *RobustnessHolds* in an (r, s)-robust digraph caused by decrementing the values of r and s from their initial values). Therefore, in either case, there are $\mathcal{O}(R(n))$ calls to *RobustnessHolds*, and hence, $\mathcal{O}(n^2 R(n))$ steps in the worst case. Finally, to bound $R(n)$, we use the Binomial Theorem to obtain

$$\begin{aligned} R(n) &= \sum_{k=2}^{n} \binom{n}{k} (2^{k-1} - 1) \\ &\leq \sum_{k=2}^{n} \binom{n}{k} 2^k 1^{n-k} \\ &\leq 3^n. \end{aligned}$$

Therefore, Algorithms 9.1 and 9.2 are $\mathcal{O}(m3^{\sqrt{m}})$, where $m = n^2$.

9.2 Network Model

The remaining algorithms operate in a decentralized manner in a time-invariant network. To model the network, we consider the undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, ..., n\}$ is the *node set* and $\mathcal{E} \subset \binom{\mathcal{V}}{2}$ is the *edge set*. Each edge $\{i, j\} \in \mathcal{E}$ indicates that nodes i and j can exchange information. Each node $i \in \mathcal{V}$ is aware of its own identifier $i \in \mathcal{V}$ and its neighbor set $\mathcal{N}_i$. The diameter of the graph is denoted $diam$. Additionally, all nodes are normal; i.e., $\mathcal{V} = \mathcal{N}$. The network is assumed to be connected and fully synchronous with reliable communication. The execution of distributed algorithms in the synchronous network progresses in rounds mapped to the nonnegative integers, $\mathbb{Z}_{\geq 0}$. We assume multiple messages may be sent in a given round and that messages may be of arbitrary size.

9.3 A Decentralized Algorithm

In this section, we present a decentralized algorithm for determining the robustness of a connected network. The main idea of the algorithm is for the nodes to share information about their neighborhood in such a way so that each node obtains the topological information about the network. Once this information is obtained, the centralized algorithm, *DetermineRobustness*, may be used to determine the robustness. The decentralized algorithm, *DecentralDetermineRobustness* is shown in Algorithm 9.3. The algorithm uses several procedures: *LeaderElectBFSTree*, *InitiateConvergecast*, *ParticipateConvergecast*, *Broadcast*, *ParticipateBroadcast*, and *DetermineRobustness*.

Algorithm 9.3: DECENTRALDETERMINEROBUSTNESS($node\mathrm{ID}_i, \mathcal{N}_i$)

```
$(Leader, Parent, Children) \leftarrow$ LEADERELECTBFSTREE($node\mathrm{ID}_i, \mathcal{N}_i$)
if $(Leader == nodeID_i)$
  then { $A(\mathcal{G}) \leftarrow$ INITIATECONVERGECAST($node\mathrm{ID}_i, Children$)
       { BROADCAST($A(\mathcal{G}), Children$)
  else { PARTICIPATECONVERGECAST($node\mathrm{ID}_i, \mathcal{N}_i, Parent, Children$)
       { $A(\mathcal{G}) \leftarrow$ PARTICIPATEBROADCAST($Parent, Children$)
$(r, s) \leftarrow$ DETERMINEROBUSTNESS($A(\mathcal{G})$)
return $(r, s)$
```

The first procedure is *LeaderElectBFSTree*. *LeaderElectBFSTree* takes as input the node's ID and its neighbor set, and outputs a *leader* ID, a *parent* node, and a set of *children* nodes. *LeaderElectBFSTree* elects a leader in the network using parallel executions of Breadth-First Searches (BFSs) [130]. Each node initiates a modification of the *SynchBFS* algorithm of [130]. To do this, a calculation of the maximum ID is bootstrapped to the Breadth-First Search (BFS) tree construction.

The node with the maximum ID is declared the leader, and the BFS tree with the leader as the root node is the BFS tree used in the subsequent convergecast and broadcast procedures.

In *LeaderElectBFSTree* all variables are associated to the initiating node's ID because n parallel executions run simultaneously. There are three types of messages involved in the BFS tree construction: *search*, *respond*, and *propagate* messages. The *search* message, with node i as the initiating node, contains the sending node's ID (initially i), the maximum ID seen so far (initially i), and the initiating node's ID (also i in the first round). Each node (other than i) is initially unmarked. Whenever an unmarked node receives a *search* message (or possibly multiple *search* messages from different neighbors), it becomes marked. The receiving node sets the maximum ID variable as the max of the received maximum IDs and chooses one of the senders as its *parent*. It then sends a *respond* message to each neighbor from which it received a *search* message. The *respond* message contains its ID, a binary variable indicating whether the node was selected as *parent*, and the ID of the initiating node i. The next round, the marked node sends a *search* message to all of its neighbors to continue the construction of the BFS tree. Whenever a marked node receives a *respond* message, it checks to see if it is selected as the node's parent, and if so, it adds the node's ID to its *children* list. After each node sends its *search* message, it waits to receive all *respond* messages from neighbors. Once it receives all of its *respond* messages, it knows whether it is a leaf node in the BFS tree (i.e., if none of its neighbors selects it as *parent*). If a node is *not* a leaf node, it waits to receive *propagate* messages from all of its *children*. If it is a leaf node, it sends a *propagate* message to its *parent*, which contains the maximum ID it has seen. Once a non-leaf node receives *propagate* messages from all of its *children*, it takes the max of the maximum IDs and sends a *propagate* messages to its *parent*. Eventually, the initiating node i receives *propagate* messages from all of its *children* and then knows the maximum ID in the network. The node with the maximum ID asserts itself as the *leader*. The *parent* and *children* variables returned by *LeaderElectBFSTree* correspond to the node's *parent* and *children* in the BFS tree with the *leader* as the initiating node. If i is the leader, it selects itself as the parent (or the null symbol). *LeaderElectBFSTree* requires $\mathcal{O}(diam)$ rounds and $\mathcal{O}(diam|\mathcal{E}|)$ messages for each of the n parallel executions [130]. Hence, in total, *LeaderElectBFSTree* requires $\mathcal{O}(diam)$ rounds and $\mathcal{O}(n \times diam|\mathcal{E}|)$ messages

Once *LeaderElectBFSTree* terminates, the leader node is determined and its BFS tree is constructed, which provides an efficient mechanism for convergecast and broadcast. In *DecentralDe-*

termineRobustness, if node i is the leader, it initiates a convergecast using *InitiateConvergecast*. If node i is not the leader it participates in the convergecast using *ParticipateConvergecast*. *InitiateConvergecast* takes as input the leader's own node ID and the *children* list (which is just the neighbor set of the leader node). *ParticipateConvergecast* takes as input the node's own ID and its neighbor set, as well as the *parent* and *children* determined by *LeaderElectBFSTree*. In *InitiateConvergecast* and *ParticipateConvergecast*, there are two types of messages: *downstream* and *upstream* messages. *Downstream* messages are sent in the direction of leaf nodes, and *upstream* messages are sent in the direction of the *leader* (root) node. The leader node starts the convergecast by sending a *downstream* message containing its node ID and neighbor set to its children of the BFS tree constructed by *LeaderElectBFSTree*. Each node is initially unmarked and waits to receive a *downstream* message from its parent. Each *downstream* message contains a list of pairs, each containing a node ID and the neighbors of the node ID. Once a *downstream* message is received from its parent, the node becomes marked. The node adds it own node ID and neighbor set to the list of pairs, and sends this list in its *downstream* message to its children. Once the *downstream* message is sent, the node waits to receive *upstream* messages from all of its children. Whenever a leaf node receives its *downstream* message, it similarly adds its pair and sends the *upstream* message to its parents. The *upstream* messages are created by consolidating the neighbor lists in the *upstream* messages received from all of the node's children. Once the leader (root) node receives the *upstream* messages from its children, it can construct the adjacency matrix $A(\mathcal{G})$, which is the quantity returned in *InitiateConvergecast*. Once each non-leader node sends its *upstream* message, it begins its *ParticipateBroadcast* procedure. The convergecast procedure requires $\mathcal{O}(diam)$ rounds and $\mathcal{O}(|\mathcal{E}|)$ messages [130].

After the convergecast procedure terminates, the leader node has the adjacency matrix $A(\mathcal{G})$. It then initiates a broadcast using the BFS tree to provide the other nodes with the adjacency matrix. Each non-leader node waits for the adjacency matrix to arrive from its parent, and then relays the information to its children. Upon sending its message, the node then calls *DetermineRobustness* to obtain the values of r and s. The broadcast operation requires $\mathcal{O}(diam)$ rounds and $\mathcal{O}(|\mathcal{E}|)$ messages [130]. By combining the message and round complexity of the procedures in *DecentralDetermineRobustness*, it follows that *DecentralDetermineRobustness* requires $\mathcal{O}(diam)$ rounds and $\mathcal{O}(n \times diam|\mathcal{E}|)$ messages. Of course, *DecentralDetermineRobustness* also inherits the worst-case complexity of *DetermineRobustness* given in Proposition 9.2.

9.4 A Distributed Algorithm

Here, we present a distributed algorithm for determining the robustness of a connected network. Instead of simply using the centralized algorithm, *DetermineRobustness*, after obtaining the adjacency matrix, as was done in Algorithm 9.3, in this case the computation required to determine robustness is reduced by only checking the edge reachability properties in subsets in which the node is *not* a member. The BFS tree construction is again used to elect a leader and provide an efficient means to broadcast information. In this algorithm, however, a second convergecast/broadcast sequence must be performed after the estimates of r and s are determined in order for the nodes to obtain the true values of r and s. *DistributedDetermineRobustness* is given in Algorithm 9.4.

Before describing *DeterminePartialRobustness*, we explain the difference in the second convergecast/broadcast sequence. In the second sequence, the nodes must determine the true values of r and s from their estimates. Therefore, the *downstream* and *upstream* messages of *InitiateConvergecast2* and *ParticipateConvergecast2* contain the minimum value of r seen along the downstream path, along with the minimum value of s seen for the given minimum value of r. For example, if a node's estimate is $(\hat{r} = 4, \hat{s} = 1)$ and it receives a *downstream* message containing $(\hat{r} = 3, \hat{s} = 3)$, then the pair sent in the next *downstream* message is $(\hat{r} = 3, \hat{s} = 3)$. Similarly, once the downstream paths reach leaf nodes, the *upstream* messages are determined similarly. Once the leader node receives the *upstream* messages from its children, it can determine the true values of r and s. These values are used in the second broadcast.

DeterminePartialRobustness is shown in Algorithm 9.5. Because the network is assumed to be connected, the network is at least 1-robust. Therefore, all subsets *not* including node i are always checked in *DeterminePartialRobustness* when called from *DistributedDetermineRobustness* under the assumption of a connected network. This implies that all nodes require finish within $\mathcal{O}(n^2)$ steps of each other.

For the performance improvement of *DeterminePartialRobustness*, observe that by eliminating the sets in which i is an element, *DeterminePartialRobustness* effectively reduces the problem from size n to $n-1$. That is, there are $R(n-1)$ pairs of subsets to check in *DeterminePartialRobustness*, instead of $R(n)$ pairs of subsets, as in *DetermineRobustness*. By using Pascal's Rule

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1},$$

we can show that the number of pairs of subsets that are avoided in *DeterminePartialRobustness* is

$$R(n) - R(n-1) = \sum_{k=1}^{n-1} \binom{n-1}{k} (2^k - 1).$$

Notice that the number of pairs above is also the number of subsets in which i is a member. The difference between this number and $R(n-1)$ is

$$n - 1 + \sum_{k=2}^{n-1} \binom{n-1}{k} (2^{k-1} + 1) > R(n-1)$$

Therefore, it is more than twice as efficient to check the subsets in which i is *not* member rather than checking only subsets in which it *is* a member. However, the worst-case complexity of the algorithm is not improved. Also, the round and message complexity for rounds in which communication is needed coincide with the round and message complexity of the decentralized algorithm.

Algorithm 9.4: DISTRIBUTEDDETERMINEROBUSTNESS$(node\mathrm{ID}_i, \mathcal{N}_i)$

```
$(Leader, Parent, Children) \leftarrow$ LEADERELECTBFSTREE$(node\mathrm{ID}_i, \mathcal{N}_i)$
if $(Leader == nodeID_i)$
  then
    $A(\mathcal{G}) \leftarrow$ INITIATECONVERGECAST$(node\mathrm{ID}_i, Children)$
    BROADCAST$(A(\mathcal{G}), Children)$
  else
    PARTICIPATECONVERGECAST$(node\mathrm{ID}_i, \mathcal{N}_i, Parent, Children)$
    $A(\mathcal{G}) \leftarrow$ PARTICIPATEBROADCAST$(Parent, Children)$
$(\hat{r}, \hat{s}) \leftarrow$ DETERMINEPARTIALROBUSTNESS$(A(\mathcal{G}), node\mathrm{ID}_i)$
if $(Leader == nodeID_i)$
  then
    $(r, s) \leftarrow$ INITIATECONVERGECAST2$(\hat{r}, \hat{s}, Children)$
    BROADCAST2$(r, s, Children)$
  else
    PARTICIPATECONVERGECAST2$(\hat{r}, \hat{s}, Parent, Children)$
    $(r, s) \leftarrow$ PARTICIPATEBROADCAST2$(Parent, Children)$
return $(r, s)$
```

Algorithm 9.5: DETERMINEPARTIALROBUSTNESS$(A(\mathcal{D}), i)$

```
$r \leftarrow \min\{\delta^{\text{in}}(\mathcal{D}), \lceil n/2 \rceil\}$
$s \leftarrow n$
for $k \leftarrow 2$ to $n-1$
do
    for each $K'_i \in \mathcal{K}'_k$ $(i = 1, 2, \ldots, \binom{n-1}{k})$
    comment: $\mathcal{K}'_k$ is the set of $\binom{n-1}{k}$ unique $k$-subsets of $\mathcal{V} \setminus \{i\}$
    do
        for each $P'_j \in \mathcal{P}'_{K_i}$ $(j = 1, 2, \ldots, 2^{k-1} - 1)$
        comment: $\mathcal{P}'_{K_i}$ is the set of partitions of $K'_i$ with exactly two nonempty parts
        do
            comment: $P'_j = \{\mathcal{S}_1, \mathcal{S}_2\}$
            $isRSRobust \leftarrow$ ROBUSTNESSHOLDS$(A(\mathcal{D}), \mathcal{S}_1, \mathcal{S}_2, r, s)$
            if $(isRSRobust ==$ false $)$ and $(s > 0)$
              then $s \leftarrow s - 1$
            while $(isRSRobust ==$ false $)$ and $(r > 0)$
            do
                while $(isRSRobust ==$ false $)$ and $(s > 0)$
                do
                    $isRSRobust \leftarrow$ ROBUSTNESSHOLDS$(A(\mathcal{D}), \mathcal{S}_1, \mathcal{S}_2, r, s)$
                    if not $isRSRobust$
                      then $s \leftarrow s - 1$
                if $(isRSRobust ==$ false $)$
                then
                    $r \leftarrow r - 1$
                    $s \leftarrow n$
            if $r == 0$
              then return $(r, s)$
return $(r, s)$
```

9.5 Construction of Robust Digraphs

Robustness requires checking every possible nonempty disjoint pair of subsets of nodes in the digraph for edge reachability conditions. Proposition 9.2 shows that the direct manner to check for robustness is $\mathcal{O}(n^2 3^n)$. Therefore, constructive methods of building robust networks are highly desirable. In [210] it is shown that the common *preferential-attachment* model for complex networks (e.g., [2]) produces r-robust graphs, provided that a sufficient number of links are added to new nodes as they are attached. Here we show that preferential attachment also leads to (r, s)-robust graphs.

Theorem 9.3 (Robust Network Growth)**.** *Let $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ be a nonempty, nontrivial (r, s)-robust digraph. Then the digraph $\mathcal{D}' = (\mathcal{V} \cup \{v_{new}\}, \mathcal{E} \cup \mathcal{E}_{new})$, where v_{new} is a new vertex added to $\mathcal{D}$ and $\mathcal{E}_{new}$ is the directed edge set related to v_{new}, is (r, s)-robust if $d^{in}_{v_{new}} \geq r + s - 1$.*

Proof. For a pair of nonempty, disjoint sets $\mathcal{S}_1$ and $\mathcal{S}_2$, there are three cases to check: $v_{\text{new}} \notin \mathcal{S}_i$, $\{v_{\text{new}}\} = \mathcal{S}_i$ and $v_{\text{new}} \in \mathcal{S}_i$, $i \in \{1, 2\}$. In the first case, since $\mathcal{D}$ is (r, s)-robust, the conditions in Definition 5.10 must hold. In the second case, $\mathcal{X}^r_{\mathcal{S}_i} = \mathcal{S}_i$, and we are done. In the third case, suppose, without loss of generality, $\mathcal{S}_2 = \mathcal{S}'_2 \cup \{v_{\text{new}}\}$. Since $\mathcal{D}$ is (r, s)-robust, at least one of the following conditions hold: $|\mathcal{X}^r_{\mathcal{S}_1}| + |\mathcal{X}^r_{\mathcal{S}'_2}| \geq s$, $|\mathcal{X}^r_{\mathcal{S}_1}| = |\mathcal{S}_1|$, or $|\mathcal{X}^r_{\mathcal{S}'_2}| = |\mathcal{S}'_2|$. If either of the first two hold, then the corresponding conditions hold for the pair $\mathcal{S}_1, \mathcal{S}_2$ in $\mathcal{D}'$. So assume only $|\mathcal{X}^r_{\mathcal{S}'_2}| = |\mathcal{S}'_2|$ holds. Then, the negation of the first condition $|\mathcal{X}^r_{\mathcal{S}_1}| + |\mathcal{X}^r_{\mathcal{S}'_2}| \geq s$ implies $|\mathcal{X}^r_{\mathcal{S}'_2}| = |\mathcal{S}'_2| < s$. Hence, $|\mathcal{N}^{\text{in}}_{v_{\text{new}}} \setminus \mathcal{S}_2| \geq r$, and $|\mathcal{X}^r_{\mathcal{S}_2}| = |\mathcal{S}_2|$, completing the proof. ☐

The above result indicates that to construct an (r, s)-robust digraph with n nodes (where $n > r$), we can start with an (r, s)-robust digraph with relatively smaller order (such as a complete graph), and continually add new nodes with incoming edges from at least $r + s - 1$ nodes in the existing digraph. Note that this method does not specify *which* existing nodes should be chosen. The preferential-attachment model corresponds to the case when the nodes are selected with a probability proportional to the number of edges that they already have. This leads to the formation of so-called *scale-free* networks [2], and is cited as a plausible mechanism for the formation of many real-world complex networks. Theorem 9.3 indicates that a large class of scale-free networks are resilient to the threat models studied in this dissertation (provided the number of edges added in each round is sufficiently large when the network is forming).

For example, Figure 49 illustrates a (3, 2)-robust graph constructed using preferential attachment by starting with the complete graph on 5 nodes K_5 – which is also (3,3)-robust and is the only (3,2)-robust digraph on 5 nodes (recall Property 5.19 on page 171) – and by adding 4 new edges to each

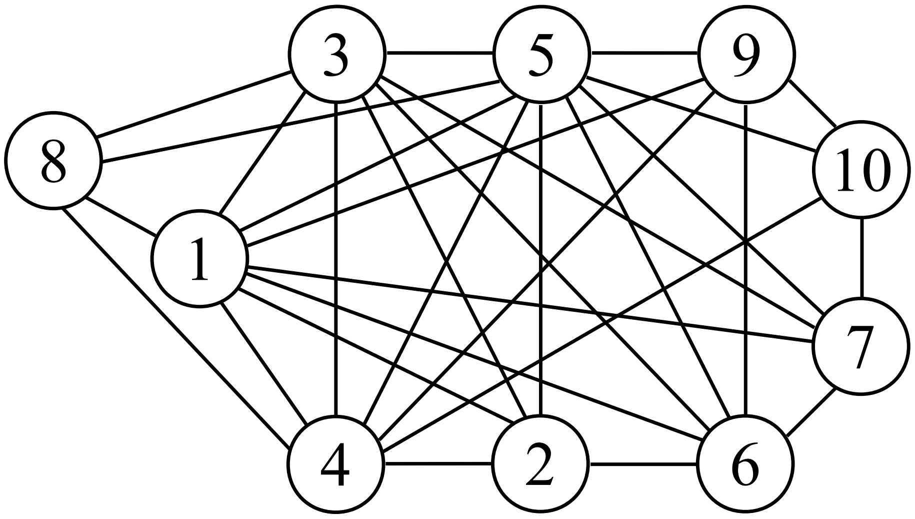


Figure 49: A $(3, 2)$-robust graph constructed from K_5 using preferential attachment.

new node in each step. Note that this graph is also 4-robust, which could *not* be predicted from Theorem 9.3 since K_5 is not 4-robust. Therefore, it is possible (but not guaranteed) to end up with a *more* robust digraph than the initial one using the preferential-attachment growth model.

9.6 Summary

In this chapter we have presented several algorithms for checking and determining robustness of a network. We presented two centralized algorithms, a decentralized algorithm, and a distributed one. All algorithms are inefficient; they are $\mathcal{O}(m3^{\sqrt{m}})$ in the size of the input. The structure of the problem of determining robustness suggests that it may be NP-hard. Proving this, however, is beyond the scope of this work.

We also looked at a growth model for robust networks that entails the preferential-attachment growth model of scale-free networks. Constructive methods such as the one given in Theorem 9.3 are advantageous, in particular given the inefficiency of determining the robustness of an existing network. Another recent result of Zhang and Sundaram shows that random networks exhibit a thresholding behavior with respect to robustness [209]. In particular, robustness and connectivity coincide in random networks [209].

CHAPTER X

CONCLUSIONS

The last decade has seen a surge of research in the cooperative control of networked multi-agent systems. Over this time, many researchers have studied robustness and resilience to certain implementation effects. However, the issue of security has just recently begun to be explored. The work presented here takes an important step by defining group objectives for multi-agent networks that provide explicit requirements on resilience in the presence of adversaries. We have introduced and studied algorithms that can be employed to ensure resilience against both adversaries and implementation effects. We foresee that these resilient consensus and synchronization control protocols will be applicable to power networks, mobile robotics, and sensor networks.

10.1 Summary of Contributions

The contributions of the dissertation may be broken into four categories: resilient consensus, resilient synchronization, network robustness, and compositionality of passivity,.

Resilient Consensus. We provide several contributions in resilient consensus under different timing models, including continuous time, synchronous discrete time, and asynchronous discrete time. We introduce a novel Continuous-Time Resilient Asymptotic Consensus (CTRAC) problem, along with new adversary models in continuous time. We formulate the Adversarial Robust Consensus Protocol (ARC-P) and its variant, ARC-P2. We demonstrate necessary and sufficient conditions under which ARC-P2 achieves CTRAC for the various adversary models studied. Similarly, we study the W-MSR algorithm in discrete-time synchronous networks in the presence of adversaries. We study an asynchronous version of W-MSR, applicable in asynchronous networks under the local broadcast model. Finally, we provide a quantization result that enables the normal nodes to achieve approximate agreement, where the error is at most one quantization step of the quantizer used in the algorithm.

Resilient Synchronization. We demonstrate how resilient consensus can be used in resilient synchronization of linear time-invariant (LTI) systems. In particular, we provide a dynamic resilient synchronization control law that uses ARC-P2 as one component of the control law. We analyze resilient asymptotic synchronization of the identical normal nodes under the control law, where the states of the normal nodes synchronize to a common open-loop solution of their local dynamics. Both full state feedback and output feedback is considered under the F-total malicious model.

Network Robustness. We introduce a definition of network robustness that has finer granularity than the one introduced in [210]. We prove several properties of robust networks, including the minimum in-degree and connectivity of robust networks. We present several algorithms for checking and determining robustness of a network. We give two centralized algorithms, a decentralized algorithm, and a distributed one. We also extend a method for constructing robust networks (introduced in [210]) to the finer granularity definition of robustness. The growth model entails the preferential-attachment growth model of scale-free networks [2].

Compositionality of passivity. We prove a novel compositional result for the interconnection of discrete-time passive systems in arbitrary bidirectional networks. It is shown that under certain message passing rules and interface components, the passivity of the networked system is maintained even in the presence of time-varying delays and data loss.

10.2 Directions for Future Work

There are several interesting directions for future work. The techniques for resilience studied in the dissertation may be extended to more complex group objectives such as flocking, formation control and coordinated path tracking. In these objectives the agents are always mobile, which leads to an important *codependence* between the *locality of the agents* and the *topology of the network*. Examining ways to *maintain network robustness* under spatial constraints is an interesting and important research problem.

Another interesting direction is resilient clock synchronization in multi-agent networks. Resilient clock synchronization has been studied [129, 109, 120]. However, these techniques achieve agreement resiliently on logical clock values, instead of agreement on the oscillator dynamics. This type of clock synchronization is a physically dependent consensus problem, but is *not* synchronization. A more interesting direction would be to study resilient synchronization of the *physical clocks* using phase-locked loops.

REFERENCES

[1] N. Agmon and D. Peleg. Fault-tolerant gathering algorithms for autonomous mobile robots. *SIAM Journal on Computing*, 36(1):56–82, July 2006.

[2] R. Albert and A. L. Barabási. Statistical mechanics of complex networks. *Rev. Mod. Phys.*, 74(1):47–97, Jan. 2002.

[3] J. Almeida, C. Silvestre, A. M. Pascoal, and P. J. Antsaklis. Continuous-time consensus with discrete-time communication. In *European Control Conference (ECC)*, pages 749–754, Budapest, Hungary, Aug. 2009.

[4] S. Amin, A. A. Cárdenas, and S. S. Sastry. Safe and secure networked control systems under denial-of-service attacks. In *Proceedings of the 12th International Conference on Hybrid Systems: Computation and Control*, HSCC '09, pages 31–45, San Francisco, CA, 2009.

[5] H. Ando, Y. Oasa, I. Suzuki, and M. Yamashita. Distributed memoryless point convergence algorithm for mobile robots with limited visibility. *IEEE Transactions on Robotics and Automation*, 15(5):818–828, Oct. 1999.

[6] P. J. Antsaklis and A. N. Michel. *Linear systems*. McGraw-Hill,, New York, 1997.

[7] M. Arcak. Passivity as a design tool for group coordination. *IEEE Transactions on Automatic Control,*, 52(8):1380–1390, Aug. 2007.

[8] A. H. Azadmanesh and H. Bajwa. Global convergence in partially fully connected networks (pfcn) with limited relays. *The 27th Annual Conference of the IEEE Industrial Electronics Society*, 3:2022–2025, 2001.

[9] M. H. Azadmanesh and R. M. Kieckhafer. Asynchronous approximate agreement in partially connected networks. *International Journal of Parallel and Distributed Systems and Networks*, 5(1):26–34, 2002.

[10] H. Bai, M. Arcak, and J. T. Wen. Rigid body attitude coordination without inertial frame information. *Automatica*, 44(12):3170–3175, 2008.

[11] H. Bai, M. Arcak, and J. T. Wen. *Cooperative Control Design: A Systematic, Passivity-Based Approach*. Communication and Control Engineering. Springer, New York, NY, 2011.

[12] T. Balch and R. C. Arkin. Behavior-based formation control for multirobot teams. *IEEE Transactions on Robotics and Automation*, 14(6):926–939, Dec. 1998.

[13] J. Bang-Jensen and G. Gutin. *Digraphs: Theory, Algorithms, and Applications*. Springer-Verlag, New York, NY, 2007.

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

[15] D. Bauso, L. Giarré, and R. Pesenti. Nonlinear protocols for optimal distributed consensus in networks of dynamic agents. *Systems & Control Letters*, 55(11):918–928, Nov.

[16] M. Ben-Or. Another advantage of free choice: Completely asynchronous agreement protocols. In *Proceedings of the 2nd annual ACM symposium on Principles of distributed computing*, (PODC), pages 27–30, Montreal, Quebec, Canada, 1983.

[17] M. Ben-Or, D. Dolev, and E. N. Hoch. Simple gradecast based algorithms. *CoRR*, abs/1007.1049, 2010.

[18] P. Berestesky, N. Chopra, and M. W. Spong. Discrete time passivity in bilateral teleoperation over the internet. In *Int. Conference on Robotics and Automation (ICRA)*, volume 5, pages 4557–4564, New Orleans, LA, 2004.

[19] D. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. Prentice-Hall, Inc., Englewood Cliffs, NJ, 1989.

[20] S. Bhattacharya and T. Başar. Spatial approaches to broadband jamming in heterogeneous mobile networks: a game-theoretic approach. *Autonomous Robots*, 31(4):367–381, 2011.

[21] F. Blanchini and S. Miani. *Set-Theoretic Methods in Control*. Birkhauser, Boston, Massachusetts, 2008.

[22] V. D. Blondel, J. M. Hendrickx, A. Olshevsky, and J. N. Tsitsiklis. Convergence in multiagent coordination, consensus, and flocking. In *IEEE Conference on Decision and Control and the European Control Conference*, pages 2996–3000, Seville, Spain, Dec. 2005.

[23] J. A. Bondy and U. S. R. Murty. *Graph Theory with Applications*. Elsevier Science Publishing Co., Inc., New York, NY, 1976.

[24] Z. Bouzid, M. G. Potop-Butucaru, and S. Tixeuil. Byzantine convergence in robot networks: The price of asynchrony. In *International Conference on Principles of Distributed Systems*, pages 54–70, Nimes, France, Dec. 2009.

[25] Z. Bouzid, M. G. Potop-Butucaru, and S. Tixeuil. Byzantine-resilient convergence in oblivious robot networks. In V. Garg, R. Wattenhofer, and K. Kothapalli, editors, *Distributed Computing and Networking*, volume 5408 of *Lecture Notes in Computer Science*, pages 275–280. Springer Berlin / Heidelberg, Jan. 2009.

[26] Z. Bouzid, M. G. Potop-Butucaru, and S. Tixeuil. Optimal Byzantine resilient convergence in asynchronous robots networks. *Stabilization, Safety, and Security of Distributed Systems*, pages 165–179, 2009.

[27] Z. Bouzid, M. G. Potop-Butucaru, and S. Tixeuil. Optimal Byzantine-resilient convergence in uni-dimensional robot networks. *Theoretical Computer Science*, 411(34-36):3154–3168, July 2010.

[28] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. *IEEE Transactions on Information Theory*, 52(6):2508–2530, June 2006.

[29] G. Bracha and S. Toueg. Asynchronous consensus and broadcast protocols. *Journal of the ACM*, 32:824–840, Oct. 1985.

[30] A. A. Cárdenas, S. Amin, and S. S. Sastry. Research challenges for the security of control systems. In *Proceedings of the 3rd conference on Hot topics in security*, pages 1–6, San Jose, CA, July 2008.

[31] D. W. Casbeer, D. B. Kingston, R. W. Beard, T. W. Mclain, S. M. Li, and R. Mehra. Cooperative forest fire surveillance using a team of small unmanned air vehicles. *International Journal of Systems Sciences*, 37(6):351–360, May 2006.

[32] G. Chartrand and F. Harary. Graphs with prescribed connectivities. In *Theory of Graphs*, pages 61–63. Akadémiai Kiadó, Budapest, 1968.

[33] N. Chopra. Passivity results for interconnected systems with time delay. In *IEEE Conference on Decision and Control*, Dec. 2008.

[34] N. Chopra, P. Berestesky, and M. W. Spong. Bilateral teleoperation over unreliable communication networks. *IEEE Transactions on Control Systems Technology.*, 16(2):304–313, 2008.

[35] N. Chopra and M. W. Spong. Passivity-based control of multi-agent systems. In *Advances in Robot Control, From Everyday Physics to Human-Like Movements*, S. Kawamura and M. Svinin (Eds.), pages 107–134. Springer Verlag, Berlin, 2006.

[36] F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, Providence, RI, 1997.

[37] S. J. Chung and J. J. E. Slotine. Cooperative robot control and concurrent synchronization of lagrangian systems. *IEEE Transactions on Robotics*, 25(3):686–700, 2009.

[38] M. Cieliebak, P. Flocchini, G. Prencipe, and N. Santoro. Solving the robots gathering problem. In *Proc. 30th International Colloquium on Automata, Languages and Programming (ICALP '03)*, pages 1181–1196, 2003.

[39] F. H. Clarke. *Optimization and Nonsmooth Analysis*. Canadian Mathematical Society Series of Monographs and Advanced Texts. John Wiley and Sons, Inc., New York, NY, 1983.

[40] J. Cortés. Finite-time convergent gradient flows with applications to network consensus. *Automatica*, 42(11):1993–2000, 2006.

[41] J. Cortés. Distributed algorithms for reaching consensus on general functions. *Automatica*, 44(3):726–737, 2008.

[42] J. Cortés, S. Martínez, and F. Bullo. Robust rendezvous for mobile autonomous agents via proximity graphs in arbitrary dimensions. *IEEE Transactions on Automatic Control*, 51(8):1289–1298, Aug. 2006.

[43] J. Cortés, S. Martínez, T. Karatas, and F. Bullo. Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation*, 20(2):243–255, April 2004.

[44] G. Dahlquist. Stability and error bounds in the numerical integration of ordinary differential equations. In *Trans. Royal Institute of Technology*, number 130, Stockholm, Sweden, 1959.

[45] S. Dasgupta, C. Papadimitriou, and U. Vazirani. *Algorithms*. McGraw-Hill Higher Education, 2006.

[46] X. Défago, M. Gradinariu, S. Messika, and P. Raipin-Parvédy. Fault-tolerant and self-stabilizing mobile robots gathering. In S. Dolev, editor, *Distributed Computing*, volume 4167 of *Lecture Notes in Computer Science*, pages 46–60. Springer Berlin, Heidelberg, 2006.

[47] M. H. DeGroot. Reaching a consensus. *Journal of the American Statistical Association*, 69(345):118–121, 1974.

[48] A. J. Van der Schaft. *L2-Gain and Passivity in Nonlinear Control*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1999.

[49] D. V. Dimarogonas and K. J. Kyriakopoulos. On the rendezvous problem for multiple nonholonomic agents. *IEEE Transactions on Automatic Control*, 52(5):916–922, May 2007.

[50] D. Dolev. The Byzantine generals strike again. *Journal of Algorithms*, 3(1):14–30, 1982.

[51] D. Dolev, N. A. Lynch, S. S. Pinter, E. W. Stark, and W. E. Weihl. Reaching approximate agreement in the presence of faults. *Journal of the ACM*, 33(3):499–516, 1986.

[52] D. Easley and J. Kleinberg. *Networks, Crowds and Markets: Reasoning about a Highly Connected World*. Cambridge University Press, 2010.

[53] M. Egerstedt, X. Hu, and A. Stotsky. Control of mobile platforms using a virtual vehicle approach. *IEEE Transactions on Automatic Control*, 46(11):1777–1782, Nov. 2001.

[54] M. Epstein, K. Lynch, K. H. Johansson, and R. M. Murray. Using hierarchical decomposition to speed up average consensus. In *Proceedings of the 17th IFAC World Congress*, 2008.

[55] A. Fagiolini, F. Babboni, and A. Bicchi. Dynamic distributed intrusion detection for secure multi-robot systems. In *Int. Conference on Robotics and Automation*, pages 2723–2728, Kobe, Japan, May 2009.

[56] A. Fagiolini, A. Bicchi, G. Dini, and I. M. Savino. Tolerating malicious monitors in detecting misbehaving robots. In *IEEE Int. Workshop on Safety, Security, and Rescue Robotics*, pages 108–114, Sendai, Japan, Oct. 2008.

[57] A. Fagiolini, M. Pellinacci, M. Valenti, G. Dini, and A. Bicchi. Consensus-based distributed intrusion detection for multi-robot systems. In *Int. Conference on Robotics and Automation*, pages 120–127, Pasadena, California, May 2008.

[58] F. Fagnani, K. H. Johansson, A. Speranzon, and S. Zampieri. On multi-vehicle rendezvous under quantized communication. In *Proc. Mathematical Theory of Networks and Systems*, Leuven, Belgium, July 2004. Electronic Proceedings.

[59] L. Fang and P. J. Antsaklis. Information consensus of asynchronous discrete-time multi-agent systems. In *Proceedings of the American Control Conference*, volume 3, pages 1883–1888, Portland, OR, June 2005.

[60] J. A. Fax and R. M. Murray. Information flow and cooperative control of vehicle formations. *IEEE Transactions on Automatic Control*, 49(9):1465–1476, 2004.

[61] A. D. Fekete. Asymptotically optimal algorithms for approximate agreement. *Distributed Computing*, 4:9–29, Mar. 1990.

[62] A. Fettweis. Wave digital filters: theory and practice. *Proceedings of the IEEE*, 74(2):270–327, 1986.

[63] A. F. Filippov. *Differential equations with discontinuous right-hand side*. Mathematics and its applications (Vol. 18). Kluwer Academic Publishers, Dordrecht, The Netherlands, 1988.

[64] M. J. Fischer, N. A. Lynch, and M. S. Paterson. Impossibility of distributed consensus with one faulty process. *Journal of the ACM*, 32:374–382, April 1985.

[65] P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer. Gathering of asynchronous robots with limited visibility. *Theoretical Computer Science*, 337:147–168, June 2005.

[66] P. A. Forero, A. Cano, and G. B. Giannakis. Consensus-based distributed support vector machines. *The Journal of Machine Learning Research*, 11:1663–1707, August 2010.

[67] M. Franceschelli, M. Egerstedt, and A. Giua. Motion probes for fault detection and recovery in networked control systems. In *Proceedings of the American Control Conference*, pages 4358–4363, Seattle, WA, June 2008.

[68] V. Gazi and K. M. Passino. Stability analysis of social foraging swarms. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 34(1):539–557, Feb. 2004.

[69] D. Geller and F. Harary. Connectivity in digraphs. In *Recent Trends in Graph Theory*, volume 186 of *Lecture Notes in Mathematics*, pages 105–115. Springer Berlin / Heidelberg, 1971.

[70] R. Ghabcheloo, A. Pascoal, C. Silvestre, and I. Kaminer. Non-linear co-ordinated path following control of multiple wheeled robots with bidirectional communication constraints. *International Journal of Adaptive Control and Signal Processing*, 21(2-3):133–157, 2007.

[71] S. Gilbert, N. Lynch, S. Mitra, and T. Nolte. Self-stabilizing robot formations over unreliable networks. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 4(3):1–29, 2009.

[72] A. Giridhar and P. R. Kumar. Toward a theory of in-network computation in wireless sensor networks. *IEEE Communications Magazine*, 44(4):98–107, April 2006.

[73] C. Godsil and G. Royle. *Algebraic Graph Theory*. Springer-Verlag New York, Inc., 2001.

[74] J. Gray. Notes on data base operating systems. In *Operating Systems: An Advanced Course*, volume 60 of *Lecture Notes in Computer Science*, pages 393–481. Springer-Verlag, New York, 1978.

[75] V. Gupta, C. Langbort, and R. M. Murray. On the robustness of distributed algorithms. In *IEEE Conference on Decision and Control*, pages 3473–3478, San Diego, California, Dec. 2006.

[76] B. Hannaford and J. H. Ryu. Time-domain passivity control of haptic interfaces. *IEEE Transactions on Robotics and Automation*, 18(1):1–10, Feb. 2002.

[77] G. H. Hardy, J. E. Littlewood, and G. Pólya. *Inequalities*. Cambridge University Press, 1952. Second Edition.

[78] P. Hartman. *Ordinary Differential Equations*. Birkhauser, Boston, Massachusetts, 1982. Second Edition.

[79] S. Hirche, T. Matiakis, and M. Buss. A distributed controller approach for delay-independent stability of networked control systems. *Automatica*, 45(8):1828–1836, 2009.

[80] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 1985.

[81] P. Hovareshti, J. S. Baras, and V. Gupta. Average consensus over small world networks: A probabilistic framework. In *IEEE Conference on Decision and Control*, pages 375–380, Cancun, Mexico, Dec. 2008.

[82] J. Hromkovic, R. Klasing, A. Pelc, P. Ruzicka, and W. Unger. *Dissemination of Information in Communication Networks*. Springer-Verlag, 2005.

[83] M. Huang and J. H. Manton. Stochastic Lyapunov analysis for consensus algorithms with noisy measurements. In *Proceedings of the American Control Conference*, pages 1419–1424, New York City, NY, July 2007.

[84] Q. Hui. Finite-time rendezvous algorithms for mobile autonomous agents. *IEEE Transactions on Automatic Control*, 56(1):207–211, Jan. 2011.

[85] Q. Hui, W. M. Haddad, and S. P. Bhat. On robust control algorithms for nonlinear network consensus protocols. *Int. J. Robust Nonlinear Control*, 20(3):269–284, 2010.

[86] A. Ichimura and M. Shigeno. A new parameter for a broadcast algorithm with locally bounded Byzantine faults. *Information Processing Letters*, 110:514–517, 2010.

[87] Y. Igarashi, T. Hatanaka, M. Fujita, and M. W. Spong. Passivity-based output synchronization in SE(3). In *Proceedings of the American Control Conference*, pages 723–728, Seattle, WA, June 2008.

[88] Y. Igarashi, T. Hatanaka, M. Fujita, and M. W. Spong. Passivity-based attitude synchronization in SE(3). *IEEE Transactions on Control Systems Technology*, 17(5):1119–1134, Sept. 2009.

[89] I. A. F. Ihle, M. Arcak, and T. I. Fossen. Passivity-based designs for synchronized path-following. *Automatica*, 43(9):1508–1518, 2007.

[90] A. Jadbabaie, J. Lin, and A. S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001, June 2003.

[91] A. Jadbabaie, N. Motee, and M. Barahona. On the stability of the Kuramoto model of coupled nonlinear oscillators. In *Proceedings of the American Control Conference*, volume 5, pages 4296–4301, Boston, MA, July 2004.

[92] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, and M. Medard. Resilient network coding in the presence of Byzantine adversaries. In *26th IEEE International Conference on Computer Communications, INFOCOM*, Anchorage, AL, May 2007.

[93] T. T. Johnson and S. Mitra. Safe flocking in spite of actuator faults using directional failure detectors. *Journal of Nonlinear Systems and Applications*, 2(1-2):73–95, April 2011.

[94] A. Kashyap, T. Başar, and R. Srikant. Quantized consensus. *Automatica*, 43(7):1192–1203, 2007.

[95] D. Kempe, A. Dobra, and J. Gehrke. Gossip-based computation of aggregate information. In *44th Annual IEEE Symposium on Foundations of Computer Science*, pages 482–491, Cambridge, MA, Oct. 2003.

[96] H. Khalil. *Nonlinear Systems*. Prentice-Hall, Inc., Upper Saddle River, NJ, 2002.

[97] R. M. Kieckhafer and M. H. Azadmanesh. Low cost approximate agreement in partially connected networks. *Journal of Computing and Information*, 3(1):53–85, 1993.

[98] R. M. Kieckhafer and M. H. Azadmanesh. Reaching approximate agreement with mixed mode faults. *IEEE Transactions on Parallel and Distributed Systems*, 5(1):53–63, 1994.

[99] D. Kingston, R. W. Beard, and R. S. Holt. Decentralized perimeter surveillance using a team of UAVs. *IEEE Transactions on Robotics*, 24(6):1394–1404, Dec. 2008.

[100] N. Kottenstette and P. J. Antsaklis. Stable digital control networks for continuous passive plants subject to delays and data dropouts. In *IEEE Conference on Decision and Control*, pages 4433–4440, New Orleans, LA, Dec. 2007.

[101] N. Kottenstette and P. J. Antsaklis. Control of multiple networked passive plants with delays and data dropouts. In *Proceedings of the American Control Conference*, pages 3126–3132, Seattle, Washington, June 2008.

[102] N. Kottenstette and N. Chopra. Lm2-stable digital-control networks for multiple continuous passive plants. In *1st IFAC Workshop on Estimation and Control of Networked Systems (NecSys'09)*, pages 120–125, Venice, Italy, Sept. 2009. International Federation of Automatic Control, International Federation of Automatic Control.

[103] N. Kottenstette, J. Hall, X. Koutsoukos, P. J. Antsaklis, and J. Sztipanovits. Digital control of multiple discrete passive plants over networks. *International Journal of Systems, Control and Communications*, 3(2):194–228, April 2011.

[104] N. Kottenstette, J. Hall, X. Koutsoukos, J. Sztipanovits, and P. J. Antsaklis. Design of networked control systems using passivity. *IEEE Transactions on Control Systems Technology*. Accepted for publication.

[105] N. Kottenstette, X. Koutsoukos, J. Hall, J. Sztipanovits, and P. Antsaklis. Passivity-based design of wireless networked control systems for robustness to time-varying delays. In *Real-Time Systems Symposium*, pages 15–24, Washington D.C., USA, Dec. 2008.

[106] N. Kottenstette, H. J. LeBlanc, E. Eyisi, and X. Koutsoukos. Multi-rate networked control of conic (dissipative) systems. In *Proceedings of the American Control Conference*, San Francisco, CA, July 2011.

[107] N. Kottenstette and J. Porter. Digital passive attitude and altitude control schemes for quadrotor aircraft. In *7th International Conference on Control and Automation (ICCA)*, pages 1761–1768, Christchurch, New Zealand, Dec. 2009.

[108] X. Koutsoukos, N. Kottenstette, J. Hall, E. Eyisi, H. J. LeBlanc, J. Porter, and J. Sztipanovits. A passivity approach for model-based compositional design of networked control systems. *ACM Transactions on Embedded Computing Systems, Special Issue on the Synthesis of Cyber-Physical Systems*. Accepted for publication.

[109] L. Lamport and P. M. Melliar-Smith. Synchronizing clocks in the presence of faults. *Journal of the ACM*, 32(1):52–78, Jan. 1985.

[110] L. Lamport, R. Shostak, and M. Pease. The Byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(2):382–401, 1982.

[111] H. J. LeBlanc, E. Eyisi, N. Kottenstette, X. Koutsoukos, and J. Sztipanovits. A passivity-based approach to deployment in multi-agent networks. In *7th International Conference on Informatics in Control, Automation and Robotics*, volume 1, pages 53–62, Funchal, Madeira - Portugal, 2010.

[112] H. J. LeBlanc, E. Eyisi, N. Kottenstette, X. Koutsoukos, and J. Sztipanovits. A passivity-based approach to group coordination in multi-agent networks. In *Informatics in Control, Automation and Robotics*, volume 89 of *Lecture Notes in Electrical Engineering*, pages 135–149. Springer Berlin Heidelberg, 2011.

[113] H. J. LeBlanc and X. D. Koutsoukos. Consensus in networked multi-agent systems with adversaries. In *Proceedings of the 14th international conference on Hybrid systems: computation and control*, (HSCC '11), pages 281–290, Chicago, IL, 2011.

[114] H. J. LeBlanc and X. D. Koutsoukos. Low complexity resilient consensus in networked multi-agent systems with adversaries. In *Proceedings of the 15th international conference on Hybrid systems: computation and control*, (HSCC '12), pages 5–14, Beijing, China, 2012.

[115] H. J. LeBlanc, H. Zhang, X. D. Koutsoukos, and S. Sundaram. Resilient asymptotic consensus in robust networks. *IEEE Journal on Selected Areas in Communications*, 2012. Submitted and under review.

[116] H. J. LeBlanc, H. Zhang, S. Sundaram, and D. Koutsoukos, X. Consensus of multi-agent networks in the presence of adversaries using only local information. In *Proceedings of the 1st International Conference on High Confidence Networked Systems (HiCoNS)*, pages 1–10, Beijing, China, 2012.

[117] A. Ledeczi, M. Maroti, A. Bakay, G. Karsai, J. Garrett, C. Thomason IV, G. Nordstrom, J. Sprinkle, and P. Volgyesi. The generic modeling environment. *Workshop on Intelligent Signal Processing*, May 2001.

[118] D. Lee and M. W. Spong. Agreement with non-uniform information delays. In *Proceedings of the American Control Conference*, pages 756–761, Minneapolis, MN, June 2006.

[119] J. Li, E. Elhamifar, I. J. Wang, and R. Vidal. Consensus with robustness to outliers via distributed optimization. In *IEEE Conference on Decision and Control*, pages 2111–2117, Atlanta, GA, Dec. 2010.

[120] Q. Li and D. Rus. Global clock synchronization in sensor networks. *IEEE Transactions on Computers*, 55(2):214–226, Feb. 2006.

[121] D. Liberzon. *Switching in Systems and Control*. Birkhauser, Boston, MA, USA, 2003.

[122] J. Lin, A. S. Morse, and B. D. O. Anderson. The multi-agent rendezvous problem. In *IEEE Conference on Decision and Control*, volume 2, pages 1508–1513, Maui, Hawaii, Dec. 2003.

[123] J. Lin, A. S. Morse, and B. D. O. Anderson. The multi-agent rendezvous problem. Part 1: The synchronous case. *SIAM Journal on Control and Optimization*, 46(6):2096–2119, 2007.

[124] J. Lin, A. S. Morse, and B. D. O. Anderson. The multi-agent rendezvous problem. Part 2: The asynchronous case. *SIAM Journal on Control and Optimization*, 46:2120–2147, 2007.

[125] Z. Lin, M. Broucke, and B. Francis. Local control strategies for groups of mobile autonomous agents. *IEEE Transactions on Automatic Control*, 49(4):622–629, April 2004.

[126] W. Lohmiller and J. J. E. Slotine. On contraction analysis for nonlinear systems. *Automatica*, 34(6):683–696, June 1998.

[127] J. Lorenz and D. A. Lorenz. On conditions for convergence to consensus. *IEEE Transactions on Automatic Control*, 55(7):1651–1656, July 2010.

[128] D. Luenberger. An introduction to observers. *IEEE Transactions on Automatic Control*, 16(6):596–602, Dec. 1971.

[129] J. Lundelius and N. A. Lynch. A new fault-tolerant algorithm for clock synchronization. In *Proceedings of the 3rd ACM symposium on Principles of distributed computing*, (PODC), pages 75–88, Vancouver, British Columbia, Canada, 1984.

[130] N. A. Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers Inc., San Francisco, California, 1997.

[131] S. Martínez. Practical multiagent rendezvous through modified circumcenter algorithms. *Automatica*, 45(9):2010–2017, 2009.

[132] S. Martínez, F. Bullo, J. Cortés, and E. Frazzoli. On synchronous robotic networks - Part II: Time complexity of rendezvous and deployment algorithms. *IEEE Transactions on Automatic Control*, 52(12):2214–2226, Dec. 2007.

[133] S. Martínez, F. Bullo, J. Cortés, and E. Frazzoli. On synchronous robotic networks Part I: Models, tasks, and complexity. *IEEE Transactions on Automatic Control*, 52(12):2199–2213, Dec. 2007.

[134] T. Matiakis, S. Hirche, and M. Buss. The scattering transformation for networked control systems. In *Proceedings of the IEEE Conference on Control Applications (CCA)*, pages 705–710, Aug. 2005.

[135] M. Mehyar, D. Spanos, J. Pongsajapan, S. H. Low, and R. M. Murray. Asynchronous distributed averaging on communication networks. *IEEE/ACM Transactions on Networking*, 15(3):512–520, June 2007.

[136] M. Mesbahi and M. Egerstedt. *Graph Theoretic Methods in Multiagent Networks*. Princeton University Press, Princeton, New Jersey, 2010.

[137] R. K. Miller and A. N. Michel. *Ordinary Differential Equations*. Dover Publications, Inc., Mineola, New York, 1982.

[138] Y. Mo, E. Garone, A. Casavola, and B. Sinopoli. False data injection attacks against state estimation in wireless sensor networks. In *IEEE Conference on Decision and Control*, pages 5967–5972, Dec. 2010.

[139] Y. Mo and B. Sinopoli. Secure control against replay attacks. In *47th Annual Allerton Conference on Communication, Control, and Computing*, pages 911–918, Sept. 2009.

[140] C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review*, 45(1):3–49, 2003.

[141] G. Montemayor and J. T. Wen. Decentralized collaborative load transport by multiple robots. In *Int. Conference on Robotics and Automation (ICRA)*, pages 372–377, Barcelona, Spain, April 2005.

[142] E. Montijano, S. Martínez, and S. Sagués. De-RANSAC: robust distributed consensus in sensor networks. *European Journal of Control*. Submitted 2012.

[143] L. Moreau. Stability of continuous-time distributed consensus algorithms. In *IEEE Conference on Decision and Control*, volume 4, pages 3998–4003, Dec. 2004.

[144] L. Moreau. Stability of multiagent systems with time-dependent communication links. *IEEE Transactions on Automatic Control*, 50(2):169–182, Feb. 2005.

[145] U. Münz, A. Papachristodoulou, and F. Allgöwer. Delay robustness in consensus problems. *Automatica*, 46(8):1252–1265, 2010.

[146] G. Niemeyer and J. J. E. Slotine. Stable adaptive teleoperation. *IEEE Journal of Oceanic Engineering*, 16(1):152–162, Jan. 1991.

[147] G. Niemeyer and J. J. E. Slotine. Telemanipulation with time delays. *International Journal of Robotics Research*, 23(9):873–890, 2004.

[148] P. Ogren, E. Fiorelli, and N. E. Leonard. Cooperative control of mobile sensor networks: Adaptive gradient climbing in a distributed environment. *IEEE Transactions on Automatic Control*, 49(8):1292–1302, Aug. 2004.

[149] M. Ohlin, D. Henriksson, and A. Cervin. *TrueTime 1.5 Reference Manual*. Dept. of Automatic Control, Lund University, Sweden, Jan. 2007. http://www.control.lth.se/truetime/.

[150] R. Olfati-Saber. Ultrafast consensus in small-world networks. In *Proceedings of the American Control Conference*, pages 2371–2378, Portland, OR, June 2005.

[151] R. Olfati-Saber. Flocking for multi-agent dynamic systems: algorithms and theory. *IEEE Transactions on Automatic Control*, 51(3):401–420, March 2006.

[152] R. Olfati-Saber, J. A. Fax, and R. M. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

[153] R. Olfati-Saber, E. Franco, E. Frazzoli, and J. Shamma. Belief consensus and distributed hypothesis testing in sensor networks. In *Networked Embedded Sensing and Control*, volume 331 of *Lecture Notes in Control and Information Sciences*, pages 169–182. Springer Berlin / Heidelberg, 2006.

[154] R. Olfati-Saber and R. M. Murray. Distributed cooperative control of multiple vehicle formations using structural potential functions. In *IFAC World Congress*, 2002.

[155] R. Olfati-Saber and R. M. Murray. Consensus protocols for networks of dynamic agents. In *Proceedings of the American Control Conference*, pages 951–956, Denver, CO, 2003.

[156] R. Olfati-Saber and R. M. Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on Automatic Control*, 49(9):1520–1533, Sep. 2004.

[157] F. Pasqualetti, A. Bicchi, and F. Bullo. Distributed intrusion detection for secure consensus computations. In *IEEE Conference on Decision and Control*, pages 5594–5599, New Orleans, LA, Dec. 2007.

[158] F. Pasqualetti, A. Bicchi, and F. Bullo. On the security of linear consensus networks. In *IEEE Conference on Decision and Control*, pages 4894–4901, Shangai, China, Dec. 2009.

[159] F. Pasqualetti, A. Bicchi, and F. Bullo. Consensus computation in unreliable networks: A system theoretic approach. *IEEE Transactions on Automatic Control*, 57(1):90–104, Jan. 2012.

[160] F. Pasqualetti, R. Carli, A. Bicchi, and F. Bullo. Identifying cyber attacks under local model information. In *IEEE Conference on Decision and Control*, pages 5961–5966, Atlanta, GA, Dec. 2010.

[161] M. Pease, R. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *Journal of the ACM*, 27(2):228–234, 1980.

[162] A. Pelc and D. Peleg. Broadcasting with locally bounded Byzantine faults. In *Information Processing Letters*, pages 109–115, 2005.

[163] G. Prencipe. CORDA: Distributed coordination of a set of autonomous mobile robots. In *Proc. 4th European Research Seminar on Advances in Distributed Systems (ERSADS)*, pages 185–190, May 2001.

[164] G. Prencipe. On the feasibility of gathering by autonomous mobile robots. In A. Pelc and M. Raynal, editors, *Structural Information and Communication Complexity*, volume 3499 of *Lecture Notes in Computer Science*, pages 246–261. Springer, Mont Saint-Michel, France, 2005.

[165] M. O. Rabin. Randomized Byzantine generals. In *24th Annual Symposium on Foundations of Computer Science*, pages 403–409, Nov. 1983.

[166] W. Ren and R. W. Beard. Consensus seeking in multiagent systems under dynamically changing interaction topologies. *IEEE Transactions on Automatic Control*, 50(5):655–661, May 2005.

[167] W. Ren, R. W. Beard, and E. M. Atkins. A survey of consensus problems in multi-agent coordination. In *Proceedings of the American Control Conference*, volume 3, pages 1859–1864, Portland, OR, June 2005.

[168] W. Ren, R. W. Beard, and E. M. Atkins. Information consensus in multivehicle cooperative control. *IEEE Control Systems Magazine*, 27(2):71–82, April 2007.

[169] C. W. Reynolds. Flocks, herds and schools: A distributed behavioral model. *SIGGRAPH Comput. Graph.*, 21:25–34, August 1987.

[170] N. Rouche, P. Habets, and M. Laloy. *Stability Theory by Liapunov's Direct Method*, volume 22 of *Applied Mathematical Sciences*. Springer-Verlag, New York, NY, 1977.

[171] L. Scardovi and R. Sepulchre. Synchronization in networks of identical linear systems. *Automatica*, 45(11):2557–2562, 2009.

[172] I. D. Schizas, G. Mateos, and G. B. Giannakis. Distributed LMS for consensus-based in-network adaptive processing. *IEEE Transactions on Signal Processing*, 57(6):2365–2382, June 2009.

[173] D. P. Spanos, R. Olfati-Saber, and R. M. Murray. Approximate distributed Kalman filtering in sensor networks with quantifiable performance. In *4th Int. Symposium on Information Processing in Sensor Networks*, pages 133–139, Los Angeles, CA, April 2005.

[174] D. P. Spanos, R. Olfati-Saber, and R. M. Murray. Dynamic consensus on mobile networks. In *Proc. IFAC*, 2005.

[175] S. Stramigioli, C. Secchi, A. J. van der Schaft, and C. Fantuzzi. Sampled data systems passivity and discrete port-Hamiltonian systems. *IEEE Transactions on Robotics*, 21(4):574–587, 2005.

[176] S. H. Strogatz. From Kuramoto to Crawford: exploring the onset of synchronization in populations of coupled oscillators. *Physica D: Nonlinear Phenomena*, 143(1):1–20, 2000.

[177] T. Ström. On logarithmic norms. *SIAM Journal on Numerical Analysis*, 12(5):741–753, 1975.

[178] S. Sundaram and C. N. Hadjicostis. Distributed function calculation and consensus using linear iterative strategies. *IEEE Journal on Selected Areas in Communications*, 26(4):650–660, May 2008.

[179] S. Sundaram and C. N. Hadjicostis. Distributed function calculation via linear iterations in the presence of malicious agents; Part I: Attacking the network. In *Proceedings of the American Control Conference*, pages 1350–1355, Seattle, WA, June 2008.

[180] S. Sundaram and C. N. Hadjicostis. Distributed function calculation via linear iterations in the presence of malicious agents; Part II: Overcoming malicious behavior. In *Proceedings of the American Control Conference*, pages 1356–1361, Seattle, WA, June 2008.

[181] S. Sundaram and C. N. Hadjicostis. Distributed function calculation via linear iterative strategies in the presence of malicious agents. *IEEE Transactions on Automatic Control*, 56(7):1495–1508, July 2011.

[182] Ichiro Suzuki and Masafumi Yamashita. Distributed anonymous mobile robots: Formation of geometric patterns. *SIAM Journal on Computing*, 28:1347–1363, 1999.

[183] J. Sztipanovits, X. Koutsoukos, G. Karsai, N. Kottenstette, P. J. Antsaklis, V. Gupta, B. Goodwine, J. Baras, and S. Wang. Toward a science of cyber-physical system integration. *Proceedings of the IEEE*, 100(1):29–44, Jan. 2012.

[184] H. G. Tanner, A. Jadbabaie, and G. J. Pappas. Flocking in fixed and switching networks. *IEEE Transactions on Automatic Control*, 52(5):863–868, 2007.

[185] H. G. Tanner, G. J. Pappas, and V. Kumar. Leader-to-formation stability. *IEEE Transactions on Robotics and Automation*, 20(3):443–455, 2004.

[186] A. Teixeira, H. Sandberg, and K. H. Johansson. Networked control systems under cyber attacks with applications to power networks. In *Proceedings of the American Control Conference*, pages 3690–3696, Baltimore, MD, July 2010.

[187] D. Tennenhouse. Proactive computing. *Communications of the ACM*, 43:43–50, May 2000.

[188] The MathWorks, Inc. Simulink. http://www.mathworks.com.

[189] B. Touri and A. Nedić. On ergodicity, infinite flow, and consensus in random models. *IEEE Transactions on Automatic Control*, 56(7):1593–1605, July 2011.

[190] B. I. Triplett, D. J. Klein, and K. A. Morgansen. Discrete time Kuramoto models with delay. In *Network Embedded Sensing and Control (Proceedings of NESC'05 Workshop)*, volume 331 of *Lecture Notes in Control and Information Sciences*, pages 9–24. Springer, New York, NY, 2006.

[191] J. N. Tsitsiklis. *Problems in Decentralized Decision Making and Computation*. PhD thesis, Department of EECS, MIT, 1984.

[192] J. N. Tsitsiklis, D. Bertsekas, and M. Athans. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE Transactions on Automatic Control*, 31(9):803–812, 1986.

[193] N. H. Vaidya. Matrix representation of iterative approximate Byzantine consensus in directed graphs. *CoRR*, abs/1201.1888, 2012.

[194] N. H. Vaidya, L. Tseng, and G. Liang. Iterative approximate Byzantine consensus in arbitrary directed graphs. *CoRR*, abs/1201.4183, 2012.

[195] N. H. Vaidya, L. Tseng, and G. Liang. Iterative approximate Byzantine consensus in arbitrary directed graphs - Part II: Synchronous and asynchronous systems. *CoRR*, abs/1202.6094, 2012.

[196] C. Van Loan. The sensitivity of the matrix exponential. *SIAM Journal on Numerical Analysis*, 14(6):971–981, 1977.

[197] S. Vanka, V. Gupta, and M. Haenggi. Power-delay analysis of consensus algorithms on wireless networks with interference. *Int. Journal of Systems, Control and Communications*, 2(1/2/3):256–274, 2010.

[198] G. Varghese and N. A. Lynch. A tradeoff between safety and liveness for randomized coordinated attack. *Information and Computation*, 128:57–71, 1996.

[199] T. Vicsek, A. Czirók, E. Ben-Jacob, I. Cohen, and O. Shochet. Novel type of phase transition in a system of self-driven particles. *Phys. Rev. Lett.*, 75(6):1226–1229, Aug. 1995.

[200] J. Wang and N. Elia. Dynamic average consensus over random networks with additive noise. In *IEEE Conference on Decision and Control*, pages 4789–4794, Atlanta, GA, Dec. 2010.

[201] W. Wang and J. J. E. Slotine. Contraction analysis of time-delayed communications and group cooperation. *IEEE Transactions on Automatic Control*, 51(4):712–717, April 2006.

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

[203] J. T. Wen and M. Arcak. A unifying passivity framework for network flow control. *IEEE Transactions on Automatic Control*, 49(2):162–174, Feb. 2004.

[204] J. Wolfowitz. Products of indecomposable, aperiodic, stochastic matrices. *Proceedings of the American Mathematical Society*, 14(5):733–737, 1963.

[205] L. Xiao and S. Boyd. Fast linear iterations for distributed averaging. *Systems & Control Letters*, 53:65–78, 2004.

[206] L. Xiao, S. Boyd, and S. Lall. A scheme for robust distributed sensor fusion based on average consensus. In *4th Int. Symposium on Information Processing in Sensor Networks*, pages 63–70, Los Angeles, CA, April 2005.

[207] G. Zames. On the input-output stability of time-varying nonlinear feedback systems Part one: Conditions derived using concepts of loop gain, conicity, and positivity. *IEEE Transactions on Automatic Control*, 11(2):228–238, April 1966.

[208] M. M. Zavlanos, H. G. Tanner, A. Jadbabaie, and G. J. Pappas. Hybrid control for connectivity preserving flocking. *IEEE Transactions on Automatic Control*, 54(12):2869–2875, Dec. 2009.

[209] H. Zhang and S. Sundaram. Robustness of complex networks: Reaching consensus despite adversaries. *CoRR*, abs/1203.6119. submitted to the 2012 Conference on Decision and Control.

[210] H. Zhang and S. Sundaram. Robustness of information diffusion algorithms to locally bounded adversaries. In *Proceedings of the American Control Conference*, June 2012. to appear.

[211] J. Zhao, D. J. Hill, and T. Liu. Synchronization of dynamical networks with nonidentical nodes: Criteria and control. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 58(3):584–594, March 2011.

[212] M. Zhu and S. Martinez. Attack-resilient distributed formation control via online adaptation. In *IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pages 6624–6629, Orlando, FL, Dec. 2011.