Submodularity in Dynamics and Control of Networked Systems

Andrew Clark

A dissertation submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2014

Reading Committee:

Radha Poovendran, Chair

Linda Bushnell, Chair

James Ritcey

Program Authorized to Offer Degree:
Electrical Engineering

University of Washington

Abstract

Submodularity in Dynamics and Control of Networked Systems

Andrew Clark

Co-Chairs of the Supervisory Committee:
Radha Poovendran
Electrical Engineering

Linda Bushnell
Electrical Engineering

Controlling a networked dynamical system to reach a desired state is a fundamental challenge in applications including transportation, energy, social, and biological systems. One scalable approach to controlling such systems is to directly control the states of a subset of leader nodes, while relying on local interactions to steer the remaining nodes towards their desired states. The choice of leader nodes is known to affect system metrics including robustness to noise, rate of convergence to a desired state, and controllability of the system. Selecting an optimal subset of leader nodes, however, is inherently a combinatorial problem, making optimal leader node selection intractable in general.

This thesis presents a submodular optimization framework to selecting leader nodes for control of networked systems. We investigate the problem of selecting a subset of leader nodes in order to minimize node state errors due to noise in the communication links between nodes. We prove that the error due to link noise is a supermodular function of the set of leader nodes, leading to the first polynomial-time algorithms for minimizing error due to link noise with provable optimality bounds. We develop our approach for networks with static topologies, as well as dynamic topologies due to random link failures, switching between predefined topologies, and arbitrary mobility.

We study selecting leader nodes in order to minimize convergence error, defined as the error in the intermediate node states prior to reaching their desired values. We derive upper

bounds for a class of convergence error metrics based on the hitting time of random walk on the network, which we prove to be a supermodular function of the set of input nodes. We present polynomial-time algorithms for minimizing convergence error with provable optimality bounds, for static as well as dynamic networks.

Efficient algorithms have recently been proposed for selecting leader nodes to satisfy controllability, defined as the ability to drive the non-input nodes from any initial state to any desired state in finite time. These algorithms, however, do not incorporate performance criteria including robustness to noise and convergence rate. We study the problem of leader selection for joint performance and controllability, and prove that controllability can be formulated as a matroid constraint on the set of leader nodes. We propose efficient algorithms with provable optimality gap for selecting leader nodes for joint performance and controllability, and characterize the submodular structure of the largest controllable subgraph of a network.

We also investigate selecting input nodes for guaranteeing synchronization in networked systems. Using the widely-studied Kuramoto model of nonlinear phase-coupled oscillators, we develop novel threshold-based conditions for a set of input nodes to ensure synchronization of the remaining nodes from almost any initial state (global practical synchronization). We formulate selection of input nodes to satisfy these conditions as a submodular optimization problem, leading efficient algorithms for selecting the minimum-size input nodes for synchronization.

Finally, we study algorithms for distributed online submodular maximization, with leader selection in distributed networks as one motivating application. We present algorithms that achieve provable optimality bounds while minimizing computation, communication, and storage overhead at the nodes. Our approach is developed for unconstrained optimization of non-monotone submodular functions, as well as cardinality-constrained monotone submodular maximization.

TABLE OF CONTENTS

Page

LIST OF FIGURES

ACKNOWLEDGMENTS

I would like first and foremost to thank my thesis advisors, Radha Poovendran and Linda Bushnell, for their invaluable mentoring and guidance over the years. They have taught me not only how to conduct research, disseminate research results, and teach undergraduate and graduate students, but also the skills and mindset that are required of a professional engineer and educator. They have exemplified to me the values of curiosity, hard work, and collaboration that form the foundation of an academic institution. I also thank the members of my PhD supervisory committee, Jim Ritcey, Emo Todorov, Mehran Mesbahi, and Arvind Krishnamurthy, for their time, service, and helpful discussions and comments.

I extend my deep thanks to the current and former graduate students and post-docs of the Network Security Lab (NSL). The current generation, consisting of Phillip Lee, Chou-Chang (Jack) Yang, Xuhang (Shaun) Ying, Elisabeth Senmarti Robla, Zhipeng (Leo) Liu, Kalikinkar Mandal, Hossein Hosseini, Sean Rice, and Leila Abudahi, have created a warm and intellectually stimulating work environment. I also thank the former NSL students, including Patrick Tague, Sidharth Nabar, David Slater, Jeff Vander Stoep, Basel Alomair, Deepak Vijaywargi, He (Tony) Wu, Ferney Maldonado, and Truc Pham, for their help and advice throughout my PhD.

During my studies, I have been fortunate to have many excellent collaborators. I thank Dr. Jorge Cuellar of Siemens Corporation for our collaboration early in my PhD on RFID security and privacy, and location privacy in sensor networks. I thank Rommie Hardy and his colleagues at the Army Research Lab for their efforts in our joint work on performance and vulnerability metrics in wireless routing, and for being generous hosts during my summer there in 2010. I thank Professor Quanyan Zhu of the NYU Polytechnic School of Engineering and Professor Tamer Başar of UIUC for our collaboration on deception-based defenses in wireless and social networks. I would like to thank Professor Kun Sun of George

Mason University for our collaborative work on performance-aware decoy-based moving target defense. Finally, I thank Dr. Fu Lin of Argonne National Laboratories and Professor Mihailo Jovanovic of University of Minnesota for many interesting discussions on leader selection and related topics. I would also like to acknowledge the generous funding support I have received from the following sources: ARL CTA, DAAD19-01-2-0011; ARO MURI, W911NF-07-1-0287; ARO PECASE, W911NF-05-1-0491; ARO grant W911NF-12-1-0448; ONR grant N00014-14-1-0029; NSF CNS-1446866; and a grant from the King Abdulaziz City for Science and Technology (KACST).

I acknowledge and thank faculty mentors who have helped me throughout my studies. Professors Loukas Lazos of University of Arizona, Guevara Noubir of Northeastern University, and Payman Arabshahi of University of Washington gave me valuable advice as I was approaching the end of my PhD and entering the job market.

Finally, I extend my deepest gratitude to my family. To my family in Seattle, you have made this city a home for me during the past several years. To my mother, father, and sister, this thesis is written with my pen and your support.

DEDICATION

To my parents.

Chapter 1

INTRODUCTION

Networked systems play ever-increasing roles in sectors such as transportation, energy, and medicine. In order to provide needed guarantees on stability, performance, and availability, these networked systems must be controlled via external inputs. Their immense scale and heterogeneity, however, renders individual control of each network node impractical. A promising alternative approach is to leverage the interactions between nodes by controlling only a relatively small subset of nodes (leaders), which act as external inputs and influence the remaining nodes (followers). The leader-follower architecture has been implemented in initial studies of unmanned vehicles [41], in which the leader nodes are controlled by remote operators and the remaining nodes follow the trajectory defined by the leaders. Anchor nodes in localization and time synchronization has also been studied within the leader-follower framework [11]. In biological systems, leader nodes represent regulated genes [62] or pacemaker cells [71].

The elevated role of leader nodes in the system implies that they must satisfy a variety of performance criteria. Inputs from the leader nodes should reach the followers in spite of noise in the communication links and node states, as well as changes in the network topology. Information should be disseminated from the leader nodes in a timely fashion, in order to enable fast convergence of the followers to their desired states. The system should also satisfy controllability, defined as the ability to drive the followers to any desired state using the leader inputs. Finally, the leader nodes should perform any specialized tasks required by the system, such as synchronization of the network to a desired state.

The need to satisfy these requirements motivates an analytical approach to selecting leader nodes in networked systems. While traditional design problems in control systems have been solved using continuous optimization, the problem of selecting a subset of nodes to act as leaders is inherently combinatorial in nature. In general, problems of this type

do not admit efficient solution algorithms. Identifying problem structure that leads to efficient solution algorithms is a key step towards achieving controllability and performance of networked systems.

1.1 Contributions of this Thesis

The contribution of this thesis is a submodular optimization framework for selecting leader nodes in networked systems (Figure 1.1). Submodularity is a diminishing returns property of set functions, analogous to concavity of continuous functions, that leads to efficient and scalable algorithms for solving discrete optimization problems. Submodular structure of the leader selection metrics implies that, as the set of leader nodes increases, the incremental increase in the metric from adding a leader node diminishes. For each metric, we investigate the problems of (i) Selecting a set of up to k leader nodes in order to optimize the performance or controllability metric, and (ii) Selecting the minimum-size set of leader nodes in order to achieve a desired bound on the metric. The specific contributions are described as follows.

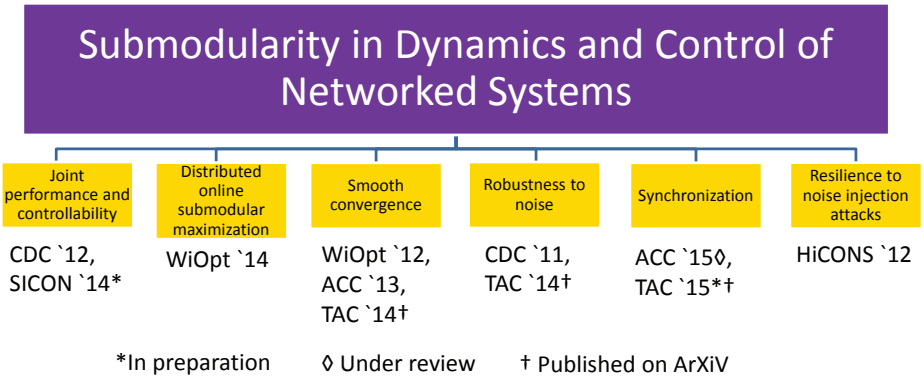


Figure 1.1: Organization of this thesis.

1.1.1 Minimizing Error Due to Link Noise

Nodes cooperate to achieve a shared goal via distributed protocols. In such protocols, nodes periodically broadcast their current state information and receive state updates from their neighbors. Each node computes and updates its state based on this information before the next iteration. Protocols of this type are predominant in consensus and formation maintenance applications, and provide a mechanism for coordination even when the network topology is dynamic.

Data exchanged via distributed protocols may be corrupted by noise in the communication links between nodes. Sources of noise include interference from neighboring nodes, malicious attacks, and environmental noise. Noise in the communication link between nodes will corrupt the inputs received by the nodes comprising the link, which in turn causes errors in the nodes' state updates. Since the state update of each node is broadcast to its neighbors at the next protocol iteration, these errors will propagate through the network and cause the entire network to deviate from its desired state. The total deviation will depend on the chosen leader nodes; if the leaders are centrally located and communicate with their neighbors via relatively noise-free links, then errors in the overall system are reduced.

We investigate and develop a submodular optimization approach to selecting leader nodes in order to minimize errors due to link noise. The design metric under consideration is the total mean-square error due to link noise in the steady-states of the nodes. Our first contribution is to prove that the error due to link noise is proportional to the commute time of a random walk on the network graph. We then prove that the commute time is a supermodular function of the set of leader nodes, implying that the error due to link noise is supermodular as well. Supermodularity implies that the problem of selecting up to k leader nodes in order to minimize error, as well as selecting the minimum-size leader set to achieve a bound on the error, can be approximated up to a constant factor via a simple greedy algorithm. We develop this approach for static networks, as well as networks with dynamics due to random link failures, switching between predefined topologies, and random node mobility. Numerical results suggest that our approach outperforms random and degree-based heuristics, and provides comparable performance to convex relaxation

techniques.

1.1.2 Robustness to Link Noise Injection Attacks

Link noise may arise either due to environmental effects, unintentional interference from neighboring nodes, or intelligent adversaries. In the latter case, an adversary broadcasts an interfering (jamming) signal in the vicinity of a receiving node, causing received packets to be decoded incorrectly. The characteristics of the noise injected by an adversary (e.g., interfering signal power, receiver that is targeted) will depend on the network topology and choice of leader nodes, since an intelligent adversary will select links for noise injection in order to maximize the impact of the attack.

We study leader selection under link noise injection attacks within a game-theoretic framework, and focus on two leader selection problems. In the first problem, the set of leader nodes is constant, and the network owner must select the leaders in order to minimize the worst-case error due to link noise arising from any possible noise injection strategy. We formulate this problem as a Stackelberg game between the network owner and adversary, in which the network owner moves first by selecting the leader set, and then the adversary moves by selecting a noise injection strategy. We show that the adversary's optimal noise injection can be computed in polynomial time, while the leader set can be approximated up to a provable bound, due to the submodularity of the error due to link noise proved in the previous chapter.

The second problem that we study is leader selection under link noise injection when the leader set varies over time. Under this approach, the system owner dynamically changes the set of leader nodes in response to changes in noise characteristics. We formulate the problem as a simultaneous-move (Nash) game, and develop an algorithm for computing an approximate Nash equilibrium. Numerical results confirm the intuition that a time-varying leader set provides improved robustness to link noise injection compared to a constant leader set.

1.1.3 Minimizing Convergence Error

Information from the leader nodes propagates through the network via distributed protocols such as gossip and consensus. The information should be disseminated from the leaders to the followers in a timely fashion. Timely information propagation is equivalent to convergence of the distributed protocols to a desired state.

One notion of convergence that has been studied in formation control [66] is containment, which occurs when the follower node states are in the convex hull of the leader node states. Containment is a desirable property when the states of the follower nodes must be bounded within a certain region, which can be specified by fixing the states of the leader nodes. Consensus is a special case of containment in which the leader nodes all have the same state, and the goal is for the followers to converge to that state.

We investigate leader selection in order to minimize convergence error in networked systems. We formulate a family of metrics, defined as the l^p-norm of the deviation of the follower node states from the convex hull of the leader node states (i.e., from containment). We denote these metrics as *convergence error* of the system. We introduce an upper bound on the convergence error as a function of the hitting time of a random walk, defined as the time required from an input from a follower node to reach any leader node. We prove that the hitting time is a supermodular function of the leader set, leading to efficient algorithms for selecting a set of up to k leaders in order to minimize convergence error, as well as selecting the minimum-size leader set to achieve a convergence error bound. We develop this approach for static networks, networks with random link failures, switching between predefined topologies, and mobile networks. In the latter case, we derive lower bounds on the best achievable performance from any leader selection algorithm.

1.1.4 Joint Performance and Controllability

Controllability is defined as the ability to drive the network nodes from any initial state to any desired state in finite time by introducing inputs to the leader nodes. Controllability is a necessary condition for control synthesis techniques and has been studied for complex networks such as biological systems, the power grid, vehicular networks, and social networks.

While the controllability property is applicable when the system operates in a benign, noise-free environment and any control inputs are feasible, typical networked systems experience errors in the communication links and node states or have a limited set of feasible control laws. For such systems, controllability must be jointly considered alongside other performance metrics such as error due to link noise and convergence error. Current algorithms for selecting leader nodes, however, either focus on one of these performance criteria [82, 51, 34], or on guaranteeing controllability [85], leaving joint optimization of performance and controllability as an open problem.

In this dissertation, we investigate the problem of leader selection for joint performance and controllability of networked systems. We find that controllability properties of the system have an inherent submodular structure that, combined with our related results on submodularity of performance metrics, enable joint consideration of performance and controllability. We formulate two specific problems. First, we investigate the problem of selecting a set of up to k leader nodes that optimize a performance metric, subject to a constraint that the network must be controllable from the leader set. We prove that the problem is equivalent to a matroid optimization problem, implying that the optimal leader set can be approximated up to a factor of $\frac{1}{2}$ using a greedy algorithm and up to a factor of $\left(1-\frac{1}{e}\right)$ using a convex optimization approach. Second, we investigate selection of leader nodes based on a trade-off between performance and controllability. We define a *Graph Controllability Index (GCI)*, defined as the size of the largest controllable subgraph of the network. We show that the GCI is a submodular function of the leader set by deriving a connection to graph matching, in which nodes that can be controlled by a given leader set are matched under a maximal matching of the graph.

1.1.5 Synchronization of Phase-Coupled Oscillators

Synchronization underlies man-made and natural phenomena, including flocking behavior of animals, Circadian rhythms, memory and cognition in the brain, time synchronization in communication systems, and stability of power systems. Phase-coupled oscillators provide a common theoretical framework for studying synchronization. Phase-coupled oscillator

dynamics consist of diffusive coupling between nodes (typically assumed to be sinusoidal), along with an intrinsic frequency that models the behavior of the node in the absence of any coupling.

Heterogeneity in the intrinsic frequencies of the nodes implies that synchronization from an arbitrary initial phase is not guaranteed in general. One approach to achieving synchronization is to pin a subset of the oscillators to a desired phase and frequency by introducing external inputs. The external inputs drive the remaining nodes to the desired phase and frequency via the local diffusive couplings between the nodes. While conditions for a given set of external inputs to drive the remaining nodes under specialized network structure (e.g., one-to-all or all-to-all coupling) have been studied in the literature, at present selecting a set of nodes to act as external inputs in networks with arbitrary topology and heterogeneous frequencies is an open problem.

We investigate selection of external input nodes in order to guarantee practical synchronization, defined as all nodes achieving the same frequency and all node phases residing within a desired region, from any initial state. We first develop sufficient conditions for guaranteeing practical synchronization from a given input set. Our sufficient condition is based on a graph threshold, in which each node reaches the synchronized state if a threshold number of neighbors (which depends on the coupling coefficient and the intrinsic frequency of the node) are synchronized. We then formulate the problem of selecting a set of input nodes that satisfies this condition as an optimization problem, in which the goal is to maximize the number of nodes that achieve synchronization. We prove that the problem is inherently submodular and propose efficient algorithms for selecting an input set for synchronization. Numerical results investigate the inputs required for synchronization, and the relationship between the input set and the network topology, in neuronal, power system, and wireless communication networks.

1.1.6 Distributed Online Submodular Maximization

The submodular optimization framework for leader selection developed in this thesis maps leader selection problems to submodular maximization. Current submodular maximization

algorithms assume the existence of a centralized entity that can compute the objective function for any value of the set, as in the greedy algorithm.

In practice, tasks such as leader selection must be performed by distributed nodes that coordinate with only local information in the absence of any centralized authority. Such nodes face severe restrictions on computation, communication, and storage. Algorithmic assumptions such as global time synchronization, which are required by existing techniques, are often infeasible. The leaders must also change over time in response to changes in network topology or operating environment. These characteristics are shared by a variety of network design problems that are solved using submodular optimization, including data caching, sensor placement, and wireless scheduling.

We investigate and develop efficient, distributed, and online algorithms for submodular maximization. We introduce algorithms for selecting a set of k nodes in order to maximize a submodular objective function, as well as unconstrained optimization of a non-monotone objective function. Our approach is based on an exchange-based algorithm, in which each node decides whether to join or leave the set at each tick of an internal clock based on previous observations of the objective function. In order to ensure that the decisions of each node approach the optimum, each node uses a multi-armed bandit algorithm. We prove that our approach achieves an asymptotic error bound of 1/3 in the unconstrained case and 1/2 in the constrained case, with lower communication, computation, and storage complexity than the current state of the art. Through simulation study on a sensor placement dataset, we find that our approach outperforms existing distributed algorithms and asymptotically approaches the performance of the centralized greedy algorithm.

1.2 Related Work

In this section, we give an overview of the related work in each of the areas considered in this thesis.

1.2.1 Minimizing Errors Due to Link Noise

Leader selection in the presence of noise in networks with static topology was considered in [108]. The authors of [108] introduce the network coherence metric, which measures how

close the follower agents states are to their desired consensus values, and equals the H_2-norm of the leader-follower system. It is shown that the network coherence is a monotone nonincreasing function of the leader set, and a greedy algorithm for maximizing the network coherence is presented. While the network coherence is equivalent to the metric we derive for static networks, provable bounds on the optimality of the selected leader sets cannot be derived from monotonicity alone.

In [47] and [82], the authors propose a semidefinite programming relaxation of the problem of selecting a set of up to k leaders to minimize the H_2-norm defined in [108]. These algorithms, however, do not provide any guarantees on the optimality of the chosen set of leaders. Furthermore, while current approaches consider selecting a set of up to k leaders in static networks in order to minimize the error in the agent states, the problem of selecting the minimum-size set of leader agents to meet a given bound on the error, as well as leader selection in dynamic networks, is not studied in [47, 82, 108].

When the leader set is given, the effect of noise on leader-follower MAS protocols, such as consensus, has been studied using a variety of approaches. For a leader-follower system with additive link noise, the steady-state error due to noise was shown to be proportional to the graph effective resistance between the leader and follower agents in [12]. In [83], decentralized control for vehicular networks with static topology, single- and double-integrator dynamics, and noise in the agent states was considered, and it was proved that at least one leader node must be present in the network to achieve stability. In [10], existing schemes for consensus and vehicle formation control were studied in the H_2-norm framework. While these methods can be used to design and evaluate a leader-follower system with given leaders, they do not address the question of selecting a leader set in the presence of noise.

Properties of the commute time of a random walk on a graph, defined as the expected time for a walk originating at a node u to reach a node v, has been studied [80]. In [27], it was shown that the graph effective resistance between two nodes u and v is proportional to the commute time of a random walk between u and v. To the best of our knowledge, however, our result that the commute time between a node u and a set S is a supermodular function of S does not appear in the existing literature.

Current approaches to mitigating link noise injection attacks on leader-follower systems

focus on securing the communication protocol used by the agents, or designing the agent state dynamics to be robust to noise. Protocol-based methods, such as frequency hopping, attempt to hide the communication channel used for inter-agent communication and thereby prevent the adversary from injecting noise into the channel [111].

From a control-theoretic standpoint, the state dynamics of the agents can be designed to be robust to noise. In [139], a convex optimization approach to deriving the agent state dynamics in order to minimize the mean-square error due to link noise was proposed. While both protocol-based and control-theoretic methods can be used to improve the resilience of a MAS with given leaders to link noise injection, neither of these methods specifies which leaders should be chosen.

1.2.2 Minimizing Convergence Error

The asymptotic convergence of distributed algorithms has been extensively studied. Surveys of recent results can be found in [24, 88]. The convergence rate and errors in the intermediate follower node states have been analyzed for static consensus networks in [103]. Subsequent works analyzed the convergence rate in networks experiencing link failures [109] and quantized consensus networks [20]. The connection between the convergence rate and the spectrum of the network graph has been observed in leader-follower systems [112] and networks without leaders [17]. The problem of containment, defined as guaranteeing that the follower node states converge to the convex hull of the leader node states, was studied for static networks in [66] and dynamic networks in [98].

Current approaches to synthesizing networked systems, either with a given leader set or without leader nodes, with minimum convergence error are mainly focused on optimizing the node dynamics. In [17], the authors minimize the convergence error by optimizing the weights that each node assigns to each of the inputs from its neighbors in a distributed fashion. In [141], an approach for dynamically modifying the weights assigned to each node in order to increase the rate of consensus was introduced. In addition to optimizing the behavior of the follower nodes, the inputs from the leader nodes can be designed using optimal control theory in order to efficiently steer the follower nodes to the desired state [67].

These existing methods focus on minimizing convergence errors when the leader set is given, rather than determining which nodes should act as leaders, and are therefore complementary to the approach presented in this chapter.

Submodular optimization has also been applied in the related context of influence maximization in social networks in [68] and [119]. The main interest of these works, however, is in maximizing the number of nodes that are influenced by the leaders in steady-state, rather than the rate of convergence. Surveys of submodular functions and submodular optimization can be found in [53, 75, 137].

The connection between multi-node system protocols and Markov chains was first explored in [30]. More recently, a Markov chain interpretation was used to prove asymptotic convergence of consensus in networks with dynamic topologies in [23] and [128]. To the best of our knowledge, however, the connection between the bounds we derive on the convergence error and the Markov process consisting of a random walk on the graph does not appear in the existing literature on MAS and Markov chains.

1.2.3 Controllability

Structural controllability of linear systems with given inputs was first studied by Lin in [81], in the case where all nonzero matrix entries are independent free parameters. Controllability of systems with additional relationships between the matrix entries was considered subsequently [39, 94, 113]. In [113], graph-based conditions for structural controllability of descriptor systems were introduced. The work of [94] provided a matroid-based framework for structural controllability of mixed-matrix descriptor systems, containing both fixed and free entries, as well as polynomial-time algorithms for verifying controllability of such systems. For a detailed survey of controllability results in linear descriptor systems, see [42]. In these works, conditions and algorithms for verifying structural controllability with a given input set are provided, but the problem of selecting the input nodes is not considered.

Selecting input nodes to satisfy controllability has been extensively studied in recent years. Necessary and sufficient conditions for a set of input nodes to guarantee controllability in leader-follower consensus dynamics were presented in [129]. Graph-based necessary

conditions were derived in [112]. These works considered controllability from a given set of input nodes, but did not introduce efficient algorithms for selecting the input nodes. In [85], a polynomial-time graph matching algorithm was introduced for selecting a minimum-size set of input nodes to satisfy structural controllability. The problem of selecting input nodes for controllability was further considered in [35, 107, 118] for the case where all matrix entries are either zero or are free, independent parameters. For the case where all entries of the system matrices are fully known, input selection algorithms and optimality bounds were derived in [102]. Here we consider a broader class of system matrices that contain both free parameters and fixed entries, taking the existing works as special cases. We also study joint consideration of performance and controllability, which has not been considered in the existing literature.

1.2.4 Synchronization in Complex Networks

The phase-coupled oscillator framework for modeling synchronization phenomena was introduced in the seminal work of Winfree [135]. The sinusoidally-coupled Kuramoto model was later introduced in [76]. Extensive studies have been performed on the mean-field behavior of the Kuramoto model with all-to-all coupling (i.e., each node is coupled to each other node) in the limit as the network size grows large [3].

Synchronization of the Kuramoto model with a finite number of oscillators and arbitrary connection topology was studied in [64]. The authors proved that convergence to the synchronized state is guaranteed when all nodes have identical intrinsic frequencies, and analyzed the feasibility and stability of synchronization under non-identical intrinsic frequencies. Stable equilibria of the Kuramoto model in finite networks, including both synchronized and non-synchronized states, were analyzed in [92]. More recently, the existence, uniqueness, and stability of partially synchronized states was studied in [44], with application to power networks [43]. These prior works considered synchronization in the absence of external inputs.

Synchronization in the presence of external inputs has achieved relatively less study. Numerical studies have estimated the region of attraction, defined as the set of initial states

that converge to the desired state, of the Kuramoto model with inputs for the case of all-to-all coupling [134]. Sufficient conditions for synchronization when there is a single input node that is connected to all other nodes were presented in [71, 110, 133]. These works do not, however, consider synchronization with external inputs in networks with arbitrary topology, and do not propose methods for selecting a subset of input nodes.

Phase-coupled oscillators in general, and Kuramoto oscillators in particular, have found extensive applications. Coupled oscillators were initially used to model natural phenomena such as bird flocking [79] and fish schooling [105]. At the level of individual cells, phase-coupled oscillators also provide a framework for heart pacemakers [90] and neuronal networks [70]. The prevalence of phase-coupled oscillators in nature has inspired engineering techniques for formation control [105] and time synchronization [9]. The phases of buses and generators in the power grid have also been modeled using phase-coupled oscillators, in order to understand whether the grid maintains a required level of synchronization for stability [50].

1.2.5 Distributed Online Submodular Maximization

In the seminal work on submodular maximization with cardinality constraints, the authors derived worst-case bounds of $(1 - 1/e)$ for the greedy algorithm and $1/2$ for the exchange algorithm for centralized submodular maximization [96]. The bound of $(1 - 1/e)$ was later shown to be the best achievable unless $P = NP$ [48]. Subsequently, centralized submodular maximization algorithms with provable performance guarantees, based on convex optimization of a relaxed problem, have been proposed [22]. Unconstrained submodular maximization has also been considered, with the best known randomized algorithm achieving an expected optimality gap of $1/2$ [19].

Recently, algorithms have been proposed for submodular maximization in an online setting [54, 126]. In this case, instead of selecting a fixed set in order to maximize a given submodular objective function, a set S_t is chosen at time t without any information on the objective function f_t. The set is instead chosen based on the past objective functions $f_1, f_2, \ldots$. It has been shown that, using experts algorithms, the average optimality gap

between current online submodular maximization algorithms and the utility of the best fixed set converges to $(1 - 1/e)$ for t sufficiently large [126].

Our approach differs from the current online submodular maximization algorithms, in that we do not assume a centralized entity with knowledge of the past objective functions. Moreover, the centralized algorithms of [126] are based on online implementation of the greedy heuristic of [96], while our algorithms are based on local exchanges, which are a fundamentally different class of algorithms.

In a submodular game, each player chooses its actions from a feasible set in a noncooperative manner in order to maximize its submodular utility function [4]. Network design problems such as multi-sensor allocation in sensor networks have been approached within the submodular game framework [138]. Our work differs from these related works in two ways. First, while the actions of a player in a submodular game are only constrained by that player's feasible set, we consider networks where the constraints on the actions of different nodes are coupled. Second, the analysis of [4, 138] assumes that each node can compute the impact of each of its actions on the objective function at each time instant. Since the computation of many common objective functions requires global knowledge of the network topology and current value of the set [35, 57, 73], we do not make this assumption.

To the best of our knowledge, the only existing work on distributed submodular maximization in either the online or offline case is in the context of distributed online sensor selection [56]. This approach is based on a stronger computational model, which assumes time synchronization and requires each node to compute the incremental benefit of being included in the set S_t, which we do not assume. Moreover, unconstrained submodular maximization is not considered in [56].

1.3 Organization of this Thesis

This thesis is organized as follows. Chapter 2 contains background on submodular functions, matroid theory, and submodular optimization. Chapter 3 discusses leader selection to minimize errors due to link noise in networked systems. Chapter 4 presents leader selection techniques for robustness to link noise injection attacks. Chapter 5 considers the problem of selecting leader nodes for smooth convergence to the desired state. Leader selection for

joint performance and controllability is discussed in Chapter 6. Chapter 7 discusses input selection to ensure global practical synchronization in complex networks. Distributed online submodular maximization algorithms are presented in Chapter 8.

Chapter 2

BACKGROUND ON SUBMODULAR FUNCTIONS AND OPTIMIZATION

This chapter gives needed background on submodular functions and submodular optimization. An introduction to the theory of matroids is also presented.

2.1 Submodularity

We define V to be a finite set. We are concerned with functions $f : 2^V \to \mathbb{R}$, i.e., functions that take a subset of V as input and give a real number as output. The submodularity property is defined as follows.

Definition 2.1. *A function $f : 2^V \to \mathbb{R}$ is* submodular *if, for any S and T, with $S \subseteq T \subseteq V$, and any $v \notin T$,*

$$f(S \cup \{v\}) - f(S) \geq f(T \cup \{v\}) - f(T). \tag{2.1}$$

Eq. (2.1) is a diminishing returns property of set functions. Intuitively, adding an element v to a set S creates a larger incremental increase in the value of f than adding v to a larger set T. The following lemma gives an equivalent definition of submodularity.

Lemma 2.1. *A function $f : 2^V \to \mathbb{R}$ is submodular if and only if, for any sets S and T,*

$$f(S) + f(T) \geq f(S \cap T) + f(S \cup T). \tag{2.2}$$

One can also show that, for any set of submodular functions $f_1, \ldots, f_m$ and any nonnegative real numbers $a_1, \ldots, a_m$, the function $f(S) = \sum_{i=1}^{m} a_i f_i(S)$ is submodular. A function is *supermodular* if $-f$ is submodular. A monotone set function is defined as follows.

Definition 2.2. *A set function $f : 2^V \to \mathbb{R}$ is* monotone *if, for any sets S and T with $S \subseteq T$, $f(S) \leq f(T)$.*

The value of a monotone set function increases as more elements are added to the set.

Lemma 2.2. *There exist monotone functions that are not submodular, and submodular functions that are not monotone.*

Proof. First, we give an example of a monotone set function that is not submodular. Consider the function $f : 2^V \to \mathbb{R}$ with $V = \{1, 2\}$, defined by $f(\emptyset) = 0$, $f(\{1\}) = 1$, $f(\{2\}) = 2$, and $f(\{1, 2\}) = 5$. The function f is clearly monotone, but violates the submodularity property since $f(\{1\}) - f(\emptyset) < f(\{1, 2\}) - f(\{2\})$.

Now, we show that submodular functions are not necessarily monotone. Suppose that $V = \{1, 2\}$ and define $f : 2^V \to \mathbb{R}$ by $f(\emptyset) = 0$, $f(\{1\}) = 4$, $f(\{2\}) = 3$, and $f(\{1, 2\}) = 2$. This function is submodular, since

$$f(\{1\}) - f(\emptyset) = 4 > -1 = f(\{1, 2\}) - f(\{2\})$$

and

$$f(\{2\}) - f(\emptyset) = 3 > -2 = f(\{1, 2\}) - f(\{1\}).$$

On the other hand, since $f(\{1\}) > f(\{1, 2\})$ and $f(\{2\}) > f(\{1, 2\})$, the function is not monotone. □

2.1.1 Composition Rules for Submodular Functions

Submodular functions can be constructed via composition rules, analogous to composition rules for convex functions. Some of these composition rules are described as follows. We first describe composition of a convex and a submodular function.

Lemma 2.3. *Let $f : 2^V \to \mathbb{R}_{\geq 0}$ be an increasing submodular function, and suppose that g is a nondecreasing, convex, and differentiable function. Then the composition $h = g \circ f$ is submodular.*

The following lemmas give additional techniques for constructing submodular functions.

Lemma 2.4. *If $f_1(S), \ldots, f_r(S)$ is a set of submodular functions of S and $\alpha_1, \ldots, \alpha_r$ are nonnegative constants, then $f(S) = \sum_{i=1}^{r} \alpha_i f_i(S)$ is a submodular function.*

Lemma 2.5. *If $f(S)$ is a submodular function and $r \geq 0$ is constant, then the function $\hat{f}(S) = \min\{f(S), r\}$ is submodular.*

Lemma 2.6. *If $f(S)$ is submodular as a function of S, then $g(S) \triangleq f(V \setminus S)$ is submodular as a function of S.*

Submodular functions can also arise as limits of sequences of submodular functions. The following theorem gives a general limit-based condition for submodularity.

Theorem 2.1. *Suppose $\{f_k(S)\}_{k=1}^{\infty}$ is a collection of supermodular functions in S, and suppose that there exists a function $f : 2^V \to \mathbb{R}$ such that, for every $S \subseteq V$ and every $\epsilon > 0$, there exists K such that $k > K$ implies*

$$|f_k(S) - f(S)| < \epsilon. \tag{2.3}$$

Then $f(S)$ is supermodular.

2.2 Examples of Submodular Functions

Submodular functions arise naturally in many applications. Some examples of submodular functions are given below.

2.2.1 Colors Represented in a Set of Balls

Let V represent a set of balls, where each ball $v \in V$ has a color (red, green, blue). Let $f(S)$ denote the number of colors represented in a set of balls $S \subseteq V$. We have that $f(S \cup \{v\}) - f(S) = 1$ if the color of ball v is not represented in the set S. If $S \subseteq T$, then any color not represented in T will automatically be unrepresented in S. Hence

$$f(S \cup \{v\}) - f(S) \geq f(T \cup \{v\}) - f(T).$$

2.2.2 Matrix Rank

Let V denote a set of column vectors, and let $f(S)$ denote the rank of the matrix with columns given in S. The increment $f(S \cup \{v\}) - f(S) = 1$ if v is linearly independent from S and 0 if v is in the span of S. If v is in the span of S, then v is automatically in the span of T for any $T \supseteq S$. Hence (2.1) holds. In fact, the matrix rank is a special case of a matroid rank function, which will be introduced in the following section.

2.2.3 Set Cover

Consider a collection of subsets $U_1, \ldots, U_m \subseteq B$. Let $V = \{1, \ldots, m\}$, and let

$$f(S) = \left| \bigcup_{i \in S} U_i \right|,$$

so that $f(S)$ denotes the number of elements of B that are contained in the union of the sets indexed in S. We have that

$$\begin{aligned} f(S \cup \{v\}) - f(S) &= \left| \left(\bigcup_{i \in S} U_i \right) \cup U_v \right| - \left| \bigcup_{i \in S} U_i \right| \\ &= \left| \bigcap_{i \in S} (U_v \setminus U_i) \right| \\ &\geq \left| \bigcap_{i \in T} (U_v \setminus U_i) \right| \\ &= f(T \cup \{v\}) - f(T), \end{aligned}$$

establishing the submodularity of the set cover.

2.2.4 Sensor Placement

The information gathered from a set of sensors deployed over a monitoring area can be viewed as a submodular function. Here, V denotes the set of possible sensor locations, and S denotes the set of locations where sensors are placed. Suppose that at each location $v \in V$, there is an observable random variable X_v. Letting $X_S = \{X_v : v \in S\}$, one metric for the amount of information gathered by the set of sensors is the mutual information

$$f(S) = I(X_S; X_V),$$

i.e., the total information gathered from the sensors. In [74], it was shown that the mutual information is a submodular function of the set of sensors, leading to efficient algorithms for selecting near-optimal sensor placements.

2.2.5 Influence in Social Networks

Ideas, including political opinions and product preferences, propagate through social interactions between individuals. A widely-studied problem in advertising is to select a subset

of individuals to directly target for advertising, in order to ensure that the largest possible subset of users adopts a certain idea or product.

This problem has been formalized by letting V denote a set of users, who are connected via a social network with edge set E. The adoption of an idea is determined by dynamics on the social network graph; for example, in the threshold model, each user adopts an idea if a threshold number of neighbors adopts the idea.

Let S denote a subset of users that initially hold the idea, and let $f(S)$ denote the expected number of users who eventually adopt the idea. In [68], it was shown that $f(S)$ is submodular as a function of S for a broad class of interaction dynamics. Submodularity has since become one of the main tools for analyzing social networks and identifying influential individuals.

2.3 Matroids: Definitions and Properties

In this section, we first define matroids, as well as the concepts of basis sets and matroid rank functions. Several example matroids are defined. We then define basic properties of matroids and methods for constructing matroids.

Definition 2.3. *Let V be a finite set, and let $\mathcal{I}$ be a collection of subsets of V. Suppose that $\mathcal{I}$ satisfies the following properties:*

1. $\emptyset \in \mathcal{I}$
2. *If $A \subseteq B$ and $B \in \mathcal{I}$, then $A \in \mathcal{I}$*
3. *If A, $B \in \mathcal{I}$ with $|A| < |B|$, then there exists $v \in B \setminus A$ such that $A \cup \{v\} \in \mathcal{I}$.*

Then $\mathcal{M} = (V, \mathcal{I})$ defines a matroid.

A *basis* is a maximal independent set in a matroid. The matroid rank function is defined as follows.

Definition 2.4. *The rank function $\rho : 2^V \to \mathbb{Z}$ of a matroid $\mathcal{M} = (V, \mathcal{I})$ is defined by*

$$\rho(A) = \max\{|B| : B \subseteq A, B \in \mathcal{I}\}.$$

The following are properties of matroid rank functions.

Lemma 2.7. *Suppose that ρ is a matroid rank function. Then ρ satisfies the following:*

(R1) $\rho(\emptyset) = 0$

(R2) *For any S and $v \in V$, $\rho(S \cup \{v\}) - \rho(S) \in \{0, 1\}$.*

(R3) *$\rho(S)$ is submodular as a function of S.*

In fact, the converse of Lemma 2.7 holds as well, enabling us to construct matroids based on rank functions satisfying Lemma 2.7.

Lemma 2.8. *Suppose $\rho : 2^V \to \mathbb{Z}_{\geq 0}$ satisfies (R1)–(R3). Then $\mathcal{M} = (V, \mathcal{I})$ with $\mathcal{I} = \{A : \rho(A) = |A|\}$ is a matroid.*

An alternative characterization of matroid rank is given by the following lemma.

Lemma 2.9. *Let $\rho : 2^V \to \mathbb{Z}_{\geq 0}$. Function ρ is a matroid rank function if and only if (i) $\rho(\emptyset) = 0$, (ii) For any $X \subseteq V$ and $v \in V \setminus X$, $\rho(X) \leq \rho(X \cup \{v\}) \leq \rho(X) + 1$, and (iii) For any $X \subseteq V$ and $v, w \notin X$, if $\rho(X) = \rho(X \cup \{v\})$ and $\rho(X) = \rho(X \cup \{w\})$, then $\rho(X) = \rho(X \cup \{x, w\})$.*

2.4 Examples of Matroids

2.4.1 Partition Matroid

Consider the example of colors represented by balls in a bin in Section 2.2.1. The number of balls represented $f(S)$ satisfies $f(\emptyset) = 0$, $f(S \cup \{v\}) - f(S) \in \{0, 1\}$ (since adding one ball to the set can increase the number of colors by at most one), and submodularity. Hence the number of balls is a matroid rank function. In this case, a set is independent if no two balls in the set have the same color. This matroid is referred to as a *partition matroid*, since the colors create a partition on the set of balls and a set is independent if it contains at most one ball of each color.

2.4.2 Matrix Rank

Let V be a set of vectors, and let $\mathcal{I}$ be the set of linearly independent vectors. We say that the empty set is independent by convention, and clearly $A \subseteq B$ is independent if B is independent. Furthermore, if A and B are independent and $|B| > |A|$, then we can find a

vector $v \in B \setminus A$ such that $A \cup \{v\}$ is independent. The rank function of this matroid is equal to the rank of the matrix with columns in S.

2.4.3 Matchings on a Bipartite Graph

Let V denote a set of edges on the graph, and suppose that a set of edges $A \in \mathcal{I}$ if no two edges share a vertex in common. Then $\mathcal{I}$ is a matching matroid. The rank of $\mathcal{I}$ is equal to the size of a maximal matching on the graph.

2.4.4 Spanning Forests of Graphs

For a graph $G = (V, E)$, a matroid can be defined by denoting a set as independent if the set does not contain a cycle. In this matroid, the set of bases is exactly the set of spanning trees.

2.5 Matroid Constructions

In this section, we describe two types of matroid constructions. The first is the matroid union, described as follows.

Lemma 2.10. *Let $(V_1, \mathcal{I}_1)$ and $(V_2, \mathcal{I}_2)$ be matroids. Define $\mathcal{I}$ by*

$$\mathcal{I} = \{A_1 \cup A_2 : A_1 \in \mathcal{I}_1, A_2 \in \mathcal{I}_2\}.$$

Then $(V_1 \cup V_2, \mathcal{I})$ is a matroid.

The matroid union gives a method for constructing a matroid by combining two other matroids. The rank function of the matroid union is given by

$$\rho(S) = \min\{\rho_1(Y) + \rho_2(Y) + |X \setminus Y| : Y \subseteq X\}.$$

The second construction technique is the *matroid dual*.

Definition 2.5. *Let $\mathcal{M} = (V, \mathcal{I})$ be a matroid. The dual matroid $\mathcal{M}^* = (V^*, \mathcal{I}^*)$ is constructed by setting $V^* = V$, and choosing the basis sets of $\mathcal{I}^*$ to be the complements of the bases in $\mathcal{I}$.*

The concept of matroid dual is motivated by duality of vector spaces. Indeed, if matroid $\mathcal{M}$ is defined as the set of linearly independent subsets of a given collection of vectors, then the dual $\mathcal{M}^*$ is defined by the independence set of the transpose of those vectors. It can be shown that if $\rho(S)$ is the rank function of matroid $\mathcal{M}$, then the dual $\mathcal{M}^*$ has rank function $\rho^*(S)$ given by

$$\rho^*(S) = \rho(V - S) + |S| - \rho(V).$$

2.6 Submodular Optimization

Submodularity enables efficient approximation of a variety of discrete optimization problems, which otherwise may not admit any approximation guarantees. In this thesis, most of the problems will have the form of submodular maximization. The basic submodularity maximization problem is to maximize a submodular objective function $f(S)$ subject to a cardinality constraint, i.e.,

$$\begin{aligned} &\text{maximize} && f(S) \\ &\text{s.t.} && |S| \leq k \end{aligned} \tag{2.4}$$

More general constraints are possible, such as the matroid constraints considered below, as well as unconstrained formulations. In general, when $f(S)$ is not submodular, problems of the form (2.4) have exponential complexity in the cardinality bound k. Submodularity, however, enables efficient approximation using a greedy algorithm, described as follows.

In the greedy algorithm, the set S is initialized to $S_0 = \emptyset$. At each iteration i, the element v_i^* that maximizes $f(S_{i-1} \cup \{v\}) - f(S_{i-1})$ is found by searching over all elements $v \notin S_{i-1}$. If $f(S_{i-1} \cup \{v^*\}) - f(S_{i-1}) \leq 0$, then the algorithm terminates and returns S_{i-1}. Otherwise, the set S_i is then updated as $S_i \leftarrow S_{i-1} \cup \{v_i^*\}$. The algorithm terminates after k iterations, and the constraint $|S_i| = k$ is satisfied with equality. A pseudocode description is given as **GreedyMaximization** below.

The simple greedy algorithm achieves a provable $O(1)$ optimality gap, as described in the following theorem.

Algorithm 1 Algorithm for approximate submodular maximization.

procedure GREEDYMAXIMIZATION(V, f, k)
  Input: Ground set V
    Monotone submodular function $f : 2^V \to \mathbb{R}$
    Cardinality constraint k
  Output: Set S
  $S \leftarrow \emptyset$, $i \leftarrow 1$

  while $i \leq k$ **do**
    $v_i^* \leftarrow \arg\max \{f(S_{i-1} \cup \{v\}) - f(S_{i-1}) : v \in V \setminus S_{i-1}\}$
    if $f(S_{i-1} \cup \{v\}^*) - f(S_{i-1}) \leq 0$ **then**
      $S \leftarrow S_{i-1}$
      return S
    end if
    $S_i \leftarrow S_{i-1} \cup \{v^*\}$
  end while
  $S \leftarrow S_k$
  return S
end procedure

Theorem 2.2 ([96]). *Let S^* denote an optimal solution to (2.4), and let $\hat{S}$ denote the set returned by* ***GreedyMaximization****. If $f(S)$ is a monotone submodular function, then*

$$f(\hat{S}) \geq \left(1 - \frac{1}{e}\right) f(S^*).$$

Proof. Write $S^* = \{v_1^*, \ldots, v_k^*\}$. Then

$$
\begin{aligned}
f(S^*) &\leq f(S^* \cup S_i) && (2.5)\\
&= f(S_i) + \sum_{j=1}^{k} \left[f(S_i \cup \{v_1^*, \ldots, v_j^*\}) - f(S_i \cup \{v_1^*, \ldots, v_{j-1}^*\}) \right] && (2.6)\\
&\leq f(S_i) + \sum_{j=1}^{k} \left[f(S_i \cup \{v_j^*\}) - f(S_i) \right] && (2.7)\\
&\leq f(S_i) + \sum_{j=1}^{k} \left[f(S_{i+1}) - f(S_i) \right] && (2.8)\\
&= f(S_i) + k(f(S_{i+1}) - f(S)) && (2.9)
\end{aligned}
$$

Here, (2.5) follows from monotonicity, (2.6) is a telescoping sum, and (2.7) follows from submodularity. Eq. (2.8) follows from the fact that the element that maximizes $f(S_{i+1}) - f(S_i)$ is chosen at each iteration of the greedy algorithm.

Let $\delta_i \triangleq f(S^*) - f(S_i)$, so that (2.9) is equivalent to $\delta_i \leq k(\delta_i - \delta_{i+1})$. Rearranging terms, we have that $\delta_{i+1} \leq \left(1 - \frac{1}{k}\right) \delta_i$. Hence $\delta_k \leq \left(1 - \frac{1}{k}\right)^k$. Using the bound $\left(1 - \frac{1}{k}\right)^k \leq \frac{1}{e}$ and rearranging terms gives the desired result. □

If the submodular function has the additional structure of a matroid rank function, then the greedy algorithm is optimal, as described by the following theorem.

Theorem 2.3. *Suppose that $r : 2^V \to \mathbb{R}$ is a matroid rank function. Then the greedy algorithm returns the optimum of*

$$
\begin{aligned}
&\textit{maximize} && r(A)\\
&\textit{s.t.} && |A| \leq k
\end{aligned}
$$

Chapter 3

MINIMIZING ERRORS DUE TO LINK NOISE

3.1 Introduction

Networked control systems operate in environments that experience noise and interference on the communication links between nodes. When state updates that are broadcast by a node are corrupted by noise at the receiver, the receiving node will update its state based on incorrect information, causing state errors that propagate through the network. The choice of leader nodes determines the level of error in the follower node states due to the propagation of leader inputs through noisy communication links [83, 108].

Leader selection algorithms based on convex optimization have been proposed for static networks in order to minimize the error in the follower node states due to noise in [47] and [82]. These convex optimization-based algorithms, however, do not provide provable guarantees on the optimality of the resulting leader set. Currently there is no analytical framework for leader selection for minimizing error due to noise that provides such guarantees.

3.1.1 Our Contributions

In this chapter, we present a supermodular optimization approach for selecting the leader set that minimizes the overall system error, defined as the mean-square error of the follower node states from their desired steady-state value, for a specific choice of link weights based on the second-order statistics of the noise. Our approach leads to efficient algorithms that provide provable bounds on the gap between the mean-square error resulting from the leader set under our approach and the minimum possible error in both static and dynamic networks. We make the following specific contributions:

- We develop a supermodular optimization approach for choosing leaders in a linear network in order to minimize the sum of the mean-square errors of the follower node states.

- In order to prove that the mean-square error due to link noise is a supermodular function of the set of leader nodes, we demonstrate that the error of each follower node's state is proportional to the commute time of a random walk between the follower node and the leader set. We then show that the commute time is a supermodular function of the leader set.
- We analyze two classes of the leader selection problem: the problem of choosing a fixed number of leaders in order to minimize the error due to noise, and the problem of finding the minimum number of leaders needed, as well as the identities of the leaders, in order to meet a given error bound.
- We extend our approach to a broad class of network topologies, including systems with: (1) static network topology, (2) random link and node failures, (3) switching between predefined topologies, and (4) network topologies that vary arbitrarily in time.
- We compare our results with other leader selection methods, including random heuristics and choosing high- and average-degree nodes, through a numerical study and show that our supermodular approach outperforms both schemes. We also show that the supermodular optimization approach provides comparable performance to state-of-the-art methods based on convex optimization.

This chapter is organized as follows. Section 3.2 states our basic definitions and assumptions. In Section 3.3, we formulate the leader selection problem for the case of a static network topology and derive a supermodular optimization solution. In Section 3.4, we analyze leader selection in networks with time-varying, dynamic topologies. Section 3.5 evaluates our approach and compares with other widely-used leader selection algorithms through a numerical study. Section 3.6 presents our conclusions and future work.

3.2 Model and Preliminaries

In this section, we describe the system model and define the mean square error due to link noise metric considered in this chapter.

3.2.1 System Model

We consider a network consisting of n nodes, indexed by the set $V = \{1, \ldots, n\}$. An edge (i, j) exists between nodes i and j iff nodes i and j can receive inputs from each other. Letting E denote the set of edges, the graph structure of the network is given by $G = (V, E)$. For an node indexed i, the neighbor set of i, denoted $N(i)$, is defined by $N(i) \triangleq \{j : (j, i) \in E\}$. The degree of i is defined to be the number of its neighbors $|N(i)|$. It is assumed that the edges are undirected and the graph G is connected.

Each node i has a time-varying state, denoted x_i(t), which may, in different contexts, represent i's position, velocity, or sensed measurement. Let $\mathbf{x} \in \mathbf{R}^n$ represent the vector of node states. The set of leader nodes, denoted S, consists of nodes who receive their state values directly from the network owner. By broadcasting these state values to their one-hop neighbors, the leaders influence the dynamics of the follower nodes. Without loss of generality, we choose the indices such that $\mathbf{x}(t) = [\mathbf{x}_f(t)^T \quad \mathbf{x}_l(t)^T]^T$, where $\mathbf{x}_f(t)$ and $\mathbf{x}_l(t)$ denote the vectors of follower and leader states, respectively.

The goal of the network is for the differences between the states of neighboring nodes i and j, $x_i - x_j$, to reach desired values, denoted r_{ij}, for all $(i, j) \in E$, so that $x_i - x_j = r_{ij}$; as a special case, when $r_{ij} = 0$, the goal of the nodes is to reach consensus. The desired state x_i^* is defined as the state satisfying $x_i^* - x_j^* = r_{ij}$ for all $j \in N(i)$. We assume that r_{ij} is known to nodes i and j and the network owner, and that there exists at least one value of $\mathbf{x}^*$ such that $x_i^* - x_j^* = r_{ij}$ for all $(i, j) \in E$.

In the noiseless case, in order to reach the desired state $\mathbf{x}^*$, the follower node $i \in V \setminus S$ updates its state according to the linear model

$$\dot{x}_i(t) = -\sum_{j \in N(i)} W_{ij}(x_i(t) - x_j(t) - r_{ij}),$$

where W is a real-valued weight matrix with nonnegative entries. Furthermore, it is assumed that each link (i, j) is affected by an additive, zero-mean white noise process, denoted $\epsilon_{ij}(t)$, with autocorrelation function $\mathbf{E}(\epsilon_{ij}(t)\epsilon_{ij}(t + \tau)) = \nu_{ij}\delta(\tau)$, where $\delta(\cdot)$ denotes the unit impulse function. The noise values on each link are assumed to be independent; in particular ϵ_{ij} and ϵ_{ji} are independent and identically distributed. This leads to the overall

linear dynamics of node $i \in V \setminus S$

$$\dot{x}_i(t) = -\sum_{j \in N(i)} W_{ij}(x_i(t) - x_j(t) - r_{ij} + \epsilon_{ij}(t)). \tag{3.1}$$

The link weights are chosen as $W_{ij} = D_i^{-1}\nu_{ij}$, where $D_i = \sum_{j \in N(i)} \nu_{ij}^{-1}$. This choice of link weights yields a best linear unbiased estimator of the leader node states if $x_j(t) \equiv x_j^*$ for all $j \in N(i)$. This approximation enables node i to estimate its desired state using local information from its one-hop neighbors, although it does not necessarily yield the mean square error-optimal linear distributed controller.

Define the elements of the weighted Laplacian matrix L by

$$L_{ij} = \begin{cases} -\nu_{ij}^{-1}, & (i,j) \in E \\ D_i, & i = j \\ 0, & \text{else} \end{cases} \tag{3.2}$$

which can be further decomposed as

$$L = \left(\begin{array}{c|c} L_{ff} & L_{fl} \\ \hline L_{lf} & L_{ll} \end{array}\right),$$

where L_{ff} and L_{fl} characterize the impact of the follower and leader node states, respectively, on the follower update dynamics. The dynamics of the follower node (3.1) for $i \in V \setminus S$ can be written in terms of L as

$$\begin{aligned} \dot{x}_i(t) &= -D_i^{-1} \sum_{j \in N(i)} \nu_{ij}^{-1}(x_i(t) - x_j(t) - r_{ij} + \epsilon_{ij}(t)) \\ &= -D_i^{-1} \left(\sum_{j \in N(i)} \nu_{ij}^{-1} x_i(t) - \sum_{j \in N(i)} \nu_{ij}^{-1} x_j(t) \sum_{j \in N(i)} \nu_{ij}^{-1} r_{ij} \right) + w_i(t) \\ &= -D_i^{-1} \left(L_{ii} x_i(t) + \sum_{j \in N(i)} L_{ij} x_j(t) + \sum_{j \in N(i)} L_{ij} r_{ij} \right) + w_i(t), \end{aligned}$$

where $w_i(t)$ is a zero-mean white noise process. Define the matrix B to be an $(n - |S|) \times |E|$ matrix, with $B_{il} = L_{ij}$ if edge l is given by (i,j) for some $j \in N(i)$ and $B_{il} = 0$ otherwise. The dynamics of the follower nodes are given in vector form by

$$\dot{\mathbf{x}}_f(t) = -D_f^{-1}(L_{ff}\mathbf{x}_f(t) + L_{fl}\mathbf{x}_l(t) + B\mathbf{r}) + \mathbf{w}(t),$$

where D_f is a $(n-|S|)\times(n-|S|)$ diagonal matrix with D_i as the i-th diagonal entry.

We assume that the leaders maintain a constant state, $\mathbf{x}_l(t) \equiv \mathbf{x}_l^*$. Based on this assumption, the desired state of the followers is defined to be $\mathbf{x}_f^* = -L_{ff}^{-1}(L_{fl}\mathbf{x}_l^* + B\mathbf{r})$, where we use the fact that L_{ff}^{-1} exists when G is connected [88, Lemma 10.36].

3.2.2 Derivation of Link Weights

In this section, a derivation of the link weights W_{ij} is given. All notations are as in Section 3.2. From the perspective of node i with neighbor set $N(i) = \{j_1, \ldots, j_d\}$, where d is the degree of node i the goal is to estimate $x_i - x_i^*$ using a set of noisy relative measurements

$$y_1 = x_i - x_{j_1} - r_{ij_1} + \epsilon_{ij_1}, \ldots, y_d = x_i - x_{j_d} - r_{ij_d} + \epsilon_{ij_d},$$

where $\epsilon_{ij_l} \sim N(0, \nu_{ij_l})$ for $l = 1, \ldots, d$. The linear estimation rule $x_i - x_i^* \approx \hat{x}_i = \sum_{j\in N(i)} W_{ij} y_j$ must satisfy two requirements. First, the expected value of $\hat{x}_i$ should be equal to the actual value of $x_i - x_i^*$, so that

$$\begin{aligned}\mathbf{E}\left(\sum_{j\in N(i)} W_{ij} y_j\right) = x_i - x_i^* &\Rightarrow \sum_{j\in N(i)} W_{ij}\mathbf{E}(y_j) = x_i - x_i^* \\ &\Rightarrow (x_i - x_i^*)\sum_{j\in N(i)} W_{ij} = x_i - x_i^*,\end{aligned}$$

which implies that $\sum_{j\in N(i)} W_{ij} = 1$. Second, the mean-square error $\mathbf{E}((\hat{x}_i - (x_i - x_i^*))^2)$ should be minimized. By making the additional approximation that $x_j(t) \equiv x_j^*$, these two problems make the optimal weight selection equivalent to finding the Best Linear Unbiased Estimator (BLUE) for node i, based on the relative measurements with errors due to link noise as well as errors in the positions of the other nodes.

The BLUE for the set of inputs given above is defined by the weights $W_{ij} = \left(\sum_{l\in N(i)} \nu_{il}^{-1}\right)^{-1} \nu_{ij}^{-1}$. This motivates the choice of dynamics

$$\dot{x}_i(t) = -D_i^{-1} \sum_{j\in N(i)} \nu_{ij}^{-1}(x_i(t) - x_j(t) - r_{ij} + \epsilon_{ij}(t)), \tag{3.3}$$

where $D_i = \sum_{j\in N(i)} \nu_{ij}^{-1}$.

3.2.3 Quantifying System Error

The mean-square error in the follower node states due to link noise in steady-state is defined as follows.

Definition 3.1. *Let $\mathbf{x}_f^* \in \mathbf{R}^n$ denote a desired state for the follower nodes. The total error of the follower nodes at time t is defined by $\mathbf{E}||\mathbf{x}_f(t) - \mathbf{x}_f^*||_2^2$. The system error, denoted $R(S)$, is defined as*

$$R(S) \triangleq \lim_{t\to\infty} \mathbf{E}||\mathbf{x}_f(t) - \mathbf{x}_f^*||_2^2.$$

The following theorem gives an explicit formula for $R(S)$ in terms of the matrix L.

Theorem 3.1. *$R(S)$ is equal to $\frac{1}{2}Tr(L_{ff}^{-1}) = \frac{1}{2}\sum_{u\in V\setminus S}(L_{ff}^{-1})_{uu}$, where $(L_{ff}^{-1})_{uu}$ denotes the (u,u)-entry of L_{ff}^{-1}.*

Proof. In the absence of noise, the follower node dynamics are given by

$$\dot{\mathbf{x}}_f(t) = -D_f^{-1}(L_{ff}\mathbf{x}_f(t) + L_{fl}\mathbf{x}_l^* + B\mathbf{r}). \tag{3.4}$$

Since L_{ff} is positive definite when G is connected [88, Lemma 10.36], $\mathbf{x}_f^* = -L_{ff}^{-1}(L_{fl}\mathbf{x}_l^* + B\mathbf{r})$ is a global asymptotic equilibrium of (3.4). In the presence of noise, $\mathbf{x}_f(t)$ is given by

$$\mathbf{x}_f(t) = e^{-D_f^{-1}L_{ff}t}\mathbf{x}_f(0) - \int_0^t e^{-D_f^{-1}L_{ff}(t-\tau)}D_f^{-1}(L_{fl}\mathbf{x}_l^* + B\mathbf{r})\,d\tau + \int_0^t e^{-D_f^{-1}L_{ff}(t-\tau)}\mathbf{w}(\tau)\,d\tau. \tag{3.5}$$

The first two terms converge to $\mathbf{x}_f^*$, since $\mathbf{x}_f^*$ is a global asymptotic equilibrium of (3.4). Since $\mathbf{w}$ is a zero-mean white process, the expected value of the third term of (3.5) is zero by linearity of expectation. Thus $\lim_{t\to\infty}\mathbf{E}(\mathbf{x}_f(t) - \mathbf{x}_f^*) = 0$, leading to

$$R(S) = \liminf_{t\to\infty} \mathbf{E}||\mathbf{x}_f(t) - \mathbf{x}_f^*||_2^2 = \sum_{u\in V\setminus S}\mathbf{E}((x_u(t) - x_u^*)^2) = \sum_{u\in V\setminus S} var(x_u(t)), \tag{3.6}$$

which follows from the fact that $x_u(t) - x_u^*$ is zero-mean, where $x_u(t)$ denotes the state of node u at time t. In [12, Section IV-B], it was shown that $\lim_{t\to\infty} var(x_u(t)) = \frac{1}{2}(L_{ff}^{-1})_{uu}$, implying that $R(S) = \frac{1}{2}\sum_{u\in V\setminus S}(L_{ff}^{-1})_{uu}$, as desired. □

For $u \in V \setminus S$, let $R(S,u) = \frac{1}{2}(L_{ff}^{-1})_{uu}$, so that $R(S) = \sum_{u\in V\setminus S} R(S,u)$. Note that L_{ff}^{-1} can be computed in worst-case $O(n^3)$ time for a given leader set S.

3.3 Leader Selection in Static Networks

In this section, we consider the problem of selecting a leader set in order to minimize the system error in static networks, in which the set of links E and the error variances ν_{ij} do not change over time. We address this problem for two cases. In the first case, no more than k nodes can act as leaders, and the goal is to choose a set of leaders that minimizes the system error. In the second case, the system error cannot exceed an upper bound α, and the goal is to find the minimum number of leaders, as well as the identities of the leader, such that the system error is less than or equal to α. In both cases, we construct algorithms for leader selection.

3.3.1 Case I – Choosing up to k Leaders to Minimize System Error

The problem of choosing a set S of k leaders in order to minimize the system error $R(S)$ is given by

$$\begin{aligned} \text{minimize} \quad & R(S) \\ \text{s.t.} \quad & |S| \leq k \end{aligned} \tag{3.7}$$

Note that problem (3.7) always has at least one feasible solution when $k \geq 1$, for example any set consisting of a single node. In what follows, we prove that the total system error $R(S)$ is a supermodular function of the leader set S, leading to efficient algorithms for approximating the optimal solution to (3.7) up to a provable bound. We first prove that for each node u, $R(S,u)$ is proportional to the commute time of a random walk on the graph G from node u to the leader set S, and then show that the commute time is supermodular. The supermodularity of $R(S)$ follows as a corollary.

Theorem 3.2. *Define a random walk on the graph G starting at node $u \in V \setminus S$, in which the probability of a transition from node i to node $j \in N(i)$, denoted $P(i,j)$, is given by*

$$P(i,j) = \frac{\nu_{ij}^{-1}}{D_i}. \tag{3.8}$$

The commute time $\kappa(S,u)$, defined as the expected number of steps for the random walk to reach the leader set S and return to u, is proportional to $R(S,u) = (L_{ff}^{-1})_{uu}$.

Before proving Theorem 3.2, the following intermediate lemmas are needed.

Lemma 3.1. *Consider the equation $Lv = J$. Define v^* to be the unique vector satisfying $Lv^* = J$. When $v^*_u = 1$, $J_i = 0$ for $i \in V \setminus (S + u)$, and $v^*_s = 0$ for $s \in S$, $(L_f^{-1})_{uu}$ is equal to J_u^{-1}.*

Proof. Write $v^* = [v_f^{*T} \quad 0]^T$ and $J = [J_f^T \quad J_l^T]^T$. The equation $Lv^* = J$ then reduces to $L_f v_f^* = J_f$, which is equivalent to $v_f^* = L_f^{-1} J_f$. Multiplying both sides of the equation by e_u^T, where e_u has a 1 in the u-th entry and 0s elsewhere, yields $v_u^* = e_u^T L_f^{-1} J_f$. The right hand side can be expanded to

$$(L_f^{-1})_{u1} J_1 + \cdots + (L_f^{-1})_{u(n-|S|)} J_{n-|S|} = 1. \tag{3.9}$$

By definition of J, all terms of the left-hand side of (3.9) are zero except $(L_f^{-1})_{uu} J_u$. Dividing both sides by J_u yields the desired result. □

Lemma 3.2. *Let v^* be defined as in Lemma 3.1. Then v^* satisfies*

$$v_i^* = \sum_{j \in N(i)} P(i,j) v_j^* \qquad \forall i \in V \setminus (S + u), \tag{3.10}$$

where $P(i,j)$ is defined as in (3.8).

Proof. By definition of L and the fact that $J_i = 0$,

$$\left(\sum_{j \in N(i)} L(i,j) \right) v_i^* = \sum_{j \in N(i)} L(i,j) v_j^*. \tag{3.11}$$

Dividing both sides by $\sum_{j \in N(i)} L(i,j)$ gives (3.10). □

Lemma 3.3. *Define $\tilde{v}_i(S,u)$ to be the probability that a random walk starting at i with transition probabilities given by (3.8) reaches node $u \in V \setminus S$ before any node in the set S. Then $\tilde{v}_i = v_i^*$ for all i.*

Proof. By definition, $\tilde{v}_i = v_i^*$ for all $i \in S$ and for $i = u$. It remains to show the result for $i \in V \setminus (S + u)$. By the stationarity of the random walk,

$$\tilde{v}_i = \sum_{j \in N(i)} P(i,j) \tilde{v}_j. \tag{3.12}$$

Hence, by (3.12) and Lemma 3.2, both v^* and $\tilde{v}$ are harmonic functions of the index i. By the Maximum Principle [86], $v^* = \tilde{v}$. □

Proof. By Lemma 3.1,

$$(L_f^{-1})_{uu} = \left(\sum_{t \in N(u)} \nu_{ut}^{-1} (1 - v_t^*) \right)^{-1}. \tag{3.13}$$

Hence, it suffices to show that the commute time $\kappa(S, u)$ is proportional to the right-hand side of (3.13). To show this, first observe that the probability $Q(u, S)$ that a random walk originating at u will reach S before returning to u is given by $1 - \sum_{t \in N(u)} \tilde{v}_t P(u, t)$, where $\tilde{v}$ is as in Lemma 3.3. Proposition 2.3 of [86] gives $Q(u, S)$ as a function of the commute time,

$$Q(u, S) = \frac{2 \sum_{(s,t) \in E} \nu_{st}^{-1}}{\kappa(S, u) D_u}.$$

This yields

$$1 - \sum_{t \in N(u)} \tilde{v}_t P(u, t) = \frac{2 \sum_{(s,t) \in E} \nu_{st}^{-1}}{\kappa(S, u) D_u}.$$

Rearranging terms and applying Lemma 3.3 gives

$$\begin{aligned} \kappa(S, u) &= \left(2 \sum_{(s,t) \in E} \nu_{st}^{-1} \right) D_u^{-1} \left(1 - \sum_{t \in N(u)} v_t^* P(u, t) \right)^{-1} \\ &= \left(2 \sum_{(s,t) \in E} \nu_{st}^{-1} \right) \left[\sum_{t \in N(u)} \nu_{uv}^{-1} (1 - v_t^*) \right]^{-1} \\ &= \left(2 \sum_{(s,t) \in E} \nu_{st}^{-1} \right) (L_{ff}^{-1})_{uu}, \end{aligned}$$

which proves the theorem. □

Theorem 3.3. *The commute time $\kappa(S, u)$ is a nonincreasing supermodular function of S.*

Proof. The nonincreasing property follows from the fact that, if $S \subseteq T$, then any walk that reaches S and returns to u has also reached T and returned to u. Thus $\kappa(T, u) \leq \kappa(S, u)$.

By Definition 2.1, $\kappa(S, u)$ is a supermodular function of S if and only if, for any sets A and B with $A \subseteq B$ and for any $j \notin B$,

$$\kappa(A, u) - \kappa(A \cup \{j\}, u) \geq \kappa(B, u) - \kappa(B \cup \{j\}, u). \tag{3.14}$$

Consider the quantity $\kappa(A, u) - \kappa(A \cup \{j\}, u)$. Define $A' = A \cup \{j\}$, and define T_{Au} and $T_{A'u}$ to be the (random) times for a random walk to reach A (respectively A') and return to u. Then

by definition, $\kappa(A,u) = \mathbf{E}(T_{Au})$ and $\kappa(A',u) = \mathbf{E}(T_{A'u})$. This implies $\kappa(A,u) - \kappa(A',u) = \mathbf{E}(T_{Au} - T_{A'u})$.

Let $\tau_j(A)$ denote the event where the random walk reaches node j before any of the nodes in A. Further, let h_{jAu} be the time for the random walk to travel from j to A and then to u, while h_{ju} is the time to travel directly from j to u. We have

$$\begin{aligned}\kappa(A,u) - \kappa(A',u) &= \mathbf{E}(T_{Au} - T_{A'u}|\tau_j(A))Pr(\tau_j(A)) + \mathbf{E}(T_{Au} - T_{A'u}|\tau_j(A)^c)Pr(\tau_j(A)^c) \\ &= \mathbf{E}(h_{jAu} - h_{ju})Pr(\tau_j(A))\end{aligned}$$

noting that if the walk reaches A first, then T_{Au} and $T_{A'u}$ are equal. Hence (3.14) becomes

$$\mathbf{E}(h_{jAu} - h_{ju})Pr(\tau_j(A)) \geq \mathbf{E}(h_{jBu} - h_{ju})Pr(\tau_j(B)). \tag{3.15}$$

In order to prove (3.15) holds, it suffices to prove that $h_{jAu} \geq h_{jBu}$ and $Pr(\tau_j(A)) \geq Pr(\tau_j(B))$. Let W^k_{jau} denote the event that, after k steps, a random walk starting at j has either not reached node $a \in A$, or has reached a but has not yet reached u. Define $W^k_{jAu} = \cap_{a\in A} W^k_{jau}$. Let $I(W^k_{jAu})$ denote the indicator function of the set W_{jAu}, and note that

$$I(W^k_{jAu}) = \prod_{a\in A} I(W^k_{jau}) \tag{3.16}$$

by the observations above and the definition of indicator functions. h_{jAu} can be rewritten as

$$\begin{aligned}h_{jAu} &= \min\{k : I(W^k_{jAu}) = 0\} =: k^* \\ &\overset{(a)}{=} \sum_{k=0}^{k^*-1} I(W^k_{jAu}) \overset{(b)}{=} \sum_{k=0}^{k^*-1} I(W^k_{jAu}) + \sum_{k=k^*}^{\infty} I(W^k_{jAu}) \\ &= \sum_{k=0}^{\infty} \prod_{a\in A} I(W^k_{jau}) \\ &\overset{(c)}{\geq} \sum_{k=0}^{\infty} \left(\prod_{a\in A} I(W^k_{jau}) \prod_{b\in B\setminus A} I(W^k_{jbu}) \right) \\ &= \sum_{k=0}^{\infty} \prod_{b\in B} I(W^k_{jbu}) = \sum_{k=0}^{\infty} I(W^k_{jBu}) = h_{jBu}\end{aligned}$$

where (a) follows from the definition of $I(W^k_{jAu})$, (b) follows from the fact that $I(W^k_{jAu}) = 0$ for $k \geq k^*$, and (c) follows from the fact that $\left(\bigcap_{a\in A} W^k_{jau}\right) \cap \left(\bigcap_{b\in B\setminus A} W^k_{jbu}\right) \subseteq \bigcap_{a\in A} W^k_{jau}$.

In order to show that $Pr(\tau_j(A)) \geq Pr(\tau_j(B))$, first let $\tau_j(v)$ denote the event that a random walk reaches node $j \neq v$ before v. Hence,

$$\tau_j(B) = \bigcap_{v \in B} \tau_j(v) = \left(\bigcap_{v \in A} \tau_j(v) \right) \cap \left(\bigcap_{v \in B \setminus A} \tau_j(v) \right) \subseteq \bigcap_{v \in A} \tau_j(v) = \tau_j(A).$$

$\tau_j(B) \subseteq \tau_j(A)$ then implies that $Pr(\tau_j(B)) \leq Pr(\tau_j(A))$, as desired. Hence (3.15) holds, thus proving that the commute time is supermodular as a function of the leader set S. □

Corollary 3.1. *$R(S)$ is a nonincreasing supermodular function of the leader set, S.*

Proof. By Theorem 3.3, $\kappa(S, u)$ is supermodular as a function of S. Since $R(S, u) = (L_f^{-1})_{uu}$ is proportional to $\kappa(S, u)$, $(L_f^{-1})_{uu}$ is supermodular as a function of S as well. By Theorem 3.1, $R(S) = \sum_{u \in V \setminus S} (L_f^{-1})_{uu}$, hence $R(S)$ is a sum of supermodular functions. $R(S)$ is therefore supermodular. □

Corollary 3.1 implies that the problem of selecting up to k leader nodes in order to minimize the system error (3.7) is a supermodular optimization problem. Although supermodular optimization problems of this form are NP-hard in general, a greedy algorithm will return a set S^* such that $R(S^*)$ is within a factor of $(1 - 1/e)$ of the optimum value, denoted R^* [137].

We now present a greedy algorithm for selecting a set of k leaders for a static network topology, which is an approximate solution to (3.7). Let S_i^* denote the set of leader nodes at the i-th iteration of the algorithm. S_0^* is initialized to $\emptyset$. At the i-th iteration of the algorithm, the element $s_i^* \in V$ is found such that $\{R(S_{i-1}^*) - R(S_{i-1}^* \cup \{s_i^*\})\}$ is maximized. S_i^* is then updated to $(S_{i-1}^* \cup \{s_i^*\})$. The algorithm terminates when either $R(S_i^*) = R(S_i^* \cup \{j\})$ for all j, or when $i = k$ (i.e., when the number of leaders is equal to k), whichever condition is reached first. A pseudocode description of the algorithm is given as algorithm STATIC-k.

The following theorem gives a bound on the performance of STATIC-k, making use of the fact that (3.7) is a supermodular optimization problem.

Algorithm 2 Algorithm for choosing up to k leaders in a static network topology.

procedure STATIC-k($G = (V, E)$, $\{\nu_{ij} : (i, j) \in E\}$, k)
  Input: $G = (V, E)$, link error variances ν_{ij}
    Maximum number of leader nodes k
  Output: Set of leader nodes S^*
  Initialization: $S^* \leftarrow \emptyset$, $i \leftarrow 0$
  while $i \leq k$ **do**
    $s_i^* \leftarrow \arg\max_{j \in V \setminus S} \{R(S^*) - R(S^* \cup \{j\})\}$
    if $R(S^*) - R(S^*) \cup \{s_i^*\} \leq 0$ **then**
      return S^*
    else
      $S^* \leftarrow S^* \cup \{s_i\}$
      $i \leftarrow i + 1$
    end if
  end while
  return S^*
end procedure

Theorem 3.4. *Define $R_{max} \triangleq \max_i R(\{i\})$, which is the worst-case error when a single node is chosen as a leader, and let R^* be the optimal value of (3.7). Then the algorithm STATIC-k terminates in polynomial time and returns a set S^* satisfying*

$$R(S^*) \leq \left(1 - \left(\frac{k-1}{k}\right)^k\right) R^* + \frac{1}{e} R_{max} \approx \left(1 - \frac{1}{e}\right) R^* + \frac{1}{e} R_{max}.$$

Proof. By Theorem 9.3 of [137, Ch III.3.9], for any nonnegative monotone nondecreasing submodular function $f(S)$, the greedy algorithm returns a set S^* satisfying $f(S^*) \geq (1 - 1/e) f^*$, where f^* is defined as $f^* \triangleq \max\{f(S) : |S| \leq k\}$. Since $R(S)$ is nonincreasing and supermodular by Corollary 3.1, the function $f(S) = R_{max} - R(S)$ is nonnegative, nondecreasing, and submodular. Maximizing $f(S)$ is thus equivalent to minimizing $R(S)$. Hence, the set S^* returned by the greedy algorithm satisfies $f(S^*) \geq \left(1 - \frac{1}{e}\right) f^*$. Substituting the definition of $f(S)$ yields the desired result.

Algorithm STATIC-k requires k iterations. At each iteration, the function $R(S)$ is evaluated $O(n)$ times. Since each evaluation of $R(S)$ involves inverting the Laplacian matrix, which requires $O(n^3)$ operations (and can be reduced to $O(n^2)$ operations, with some loss in computation accuracy, if the Laplacian matrix is sparse [108]), the total runtime is $O(kn^4)$. □

In [95], it was shown that there is no polynomial-time algorithm that improves on the approximation bound $\left(1 - \frac{1}{e}\right)$ unless $P = NP$. There may, however, be additional structure to $R(S)$ other than supermodularity that may improve the guarantees of Theorem 3.4. This remains an open problem.

3.3.2 Case II – Choosing the Minimum-Size Leader Set to Achieve an Error Bound

When the system is required to operate below a given error bound, denoted α, the problem of choosing a minimal set of leaders that achieves this bound can be stated as

$$\begin{aligned} \text{minimize} \quad & |S| \\ \text{s.t.} \quad & R(S) \leq \alpha \end{aligned} \tag{3.17}$$

Note that, for any $\alpha \geq 0$, there exists at least one S meeting the condition $R(S) \leq \alpha$, namely the set $S = V$.

The supermodularity of $R(S)$ enables an efficient approximate solution of (3.17). The set of leaders is initialized to $S_0^* = \emptyset$. As with the leader selection algorithm STATIC-k, the node s_i^* that maximizes $\{R(S_{i-1}^*) - R(S_{i-1}^* \cup \{s_i^*\})\}$ is added at the i-th iteration, so that $S_i^* = S_{i-1}^* \cup \{s_i^*\}$. The algorithm terminates when $R(S_i^*) \leq \alpha$ and returns the set $S^* = S_i^*$. A pseudocode description of the algorithm is given as algorithm STATIC-α.

Algorithm 3 Algorithm for selecting minimum-size set of leaders to achieve error bound α in a static network topology.

procedure STATIC-$\alpha(G = (V, E)$, $\{\nu_{ij} : (i, j) \in E\}$, $\alpha)$
  Input: $G = (V, E)$, link error variances ν_{ij}
    Maximum tolerable error α
  Output: Set of leader nodes S^*
  Initialization: $S^* \leftarrow \emptyset$, $error \rightarrow 0$
  while $error > \alpha$ **do**
    $s_i^* \leftarrow \arg\max_{j \in V \setminus S} \{R(S^*) - R(S^* \cup \{j\})\}$
    if $R(S^*) - R(S^*) \cup \{s_i^*\} \leq 0$ **then**
      return S^*
    else
      $S^* \leftarrow S^* \cup \{s_i\}$
      $error \leftarrow R(S^*)$
    end if
  end while
  return S^*
end procedure

The following theorem gives bounds on the optimality of the set S^* returned by SWITCHING-k.

Theorem 3.5. *Let k^* be the smallest integer such that a set S exists with $|S| = k$ and $R(S) \leq \alpha$ (i.e., k^* is the optimal value of (3.17)). The algorithm STATIC-α terminates in*

polynomial time in n. If the algorithm terminates after step k, so that $|S^*| = k$, then

$$\frac{k}{k^*} \leq 1 + \log\left\{\frac{R_{max}}{R(S_{k-1})}\right\}$$

holds, where R_{max} is as defined in Theorem 3.4.

Proof. Theorem 9.4 of [137, Ch III.3.9] states that the greedy algorithm for solving problems of the form $\min\{|S| : f(S) \geq \lambda\}$, where $f(S)$ is a nondecreasing submodular function, returns a set S^* satisfying

$$\frac{|S^*|}{|S'|} \leq 1 + \log\left\{\frac{f(V) - f(\emptyset)}{f(V) - f(S_{k-1})}\right\},$$

where S' is the optimal solution and S_{k-1} is the set obtained at the $(k-1)$-th iteration of the greedy algorithm. Letting $f(S) = R_{max} - R(S)$, we have that $f(S)$ is a nondecreasing submodular function of S. Since the greedy algorithm for optimizing $f(S)$ is equivalent to optimizing $R(S)$, we have

$$\frac{|S^*|}{|S'|} \leq 1 + \log\left\{\frac{f(V) - f(\emptyset)}{f(V) - f(S_{k-1})}\right\} = 1 + \log\left\{\frac{-f(\emptyset)}{-f(S_{k-1})}\right\} = 1 + \log\left\{\frac{R_{max}}{f(S_{k-1})}\right\}.$$

In the worst case, the algorithm will not terminate until $S = V$, i.e., after n iterations. This requires $O(n^2)$ evaluations of R, each of which requires $O(n^3)$ computations, for a runtime of $O(n^5)$. □

3.4 Leader Selection in Dynamic Networks

Multi-node systems may undergo changes in topology or link noise characteristics for three reasons. First, the nodes may experience link or device failures [59] . Second, the network may switch between prespecified topologies [101] . Third, the topology may vary arbitrarily over time due to node mobility [100] . Under each case, when the topology changes vary sufficiently slowly with respect to the system dynamics, metrics based on the asymptotic mean-square error due to link noise, $R(S)$, can be used to quantify the performance of the time-varying network[2]. In this section, we study leader selection for each of these dynamic networks.

[2]If the topology does not vary sufficiently slowly, then the asymptotic analysis of Theorem 3.1 no longer holds.

3.4.1 Leader Selection Under Random Link Failures

Random topology changes may occur due to link failures, which are reflected in the weighted Laplacian matrix L of (3.2). Since the set of links may not be known in advance under these circumstances, leaders can be selected to minimize the expected system error based on the distribution of possible weighted Laplacians.

Let $\mathcal{L}$ denote the set of possible weighted Laplacians. Define π to be a probability distribution on $\mathcal{L}$, so that $\pi(L)$ is the probability that the Laplacian is $L \in \mathcal{L}$. An example distribution is the *random link failure* model, in which each link in an underlying link set E fails independently with equal probability p. The expected system error is defined by $\mathbf{E}_\pi(R(S)) = \sum_{L\in\mathcal{L}} R(S|L)\pi(L)$, where $R(S|L)$ denotes the system error when the leader set is S and the Laplacian is L.

The problems of (a) choosing a set of k leaders to minimize the expected error $\mathbf{E}_\pi(R(S))$ and (b) choosing the smallest possible leader set S such that the error is within an upper bound α are formulated as

Choosing up to k leaders		**Minimizing number of leaders**	
minimize	$\mathbf{E}_\pi(R(S))$	minimize	$\lvert S\rvert$
s.t.	$\lvert S\rvert \le k$	s.t.	$\mathbf{E}_\pi(R(S)) \le \alpha$
	(a)		(b)

(3.18)

In order to solve (3.18a) and (3.18b), it is necessary to compute $\mathbf{E}_\pi(R(S))$ for a given distribution π and leader set S. The summation, however, can have up to $2^{|E|}$ possible topologies under the random link failure model. We present two approaches to approximating $\mathbf{E}_\pi(R(S))$: first, a Monte Carlo approximation that is valid for any distribution π, and second, a gradient-based approximation that is valid when the probability of link failures is small.

Monte Carlo Approximation: Under the Monte Carlo approach, a set consisting of M Laplacian matrices $\{L_1, \ldots, L_M\}$, each chosen independently according to distribution π, is generated. $\mathbf{E}_\pi(R(S))$ is then approximated by $R_{mc}(S) = \frac{1}{M}\sum_{i=1}^{M} R(S|L_i)$, allowing an arbitrarily close approximation of $\mathbf{E}_\pi(R(S))$ by the weak law of large numbers [78].

Gradient Approximation: Consider the random link failure model, and let $\tilde{G} = (V, \tilde{E})$, where

$\tilde{E}$ denotes the set of links that have failed. Define matrix Δ by

$$\Delta_{ij} = \begin{cases} -\nu_{ij}^{-1}, & (i,j) \in \tilde{E} \\ \sum_{(i,j)\in\tilde{E}} \nu_{ij}^{-1}, & i = j \\ 0, & \text{else} \end{cases}$$

Let $\tilde{L}_{ff} \triangleq L_f - \Delta$, so that $R(S|\tilde{L}) = \sum_{u\in V\setminus S} (\tilde{L}_{ff}^{-1})_{uu}$.

Lemma 3.4. *Suppose that links fail independently and randomly with probability p. Let* $X = \max_{(i,j)\in E} \nu_{ij}^{-1}$*, and let d denote the maximum node degree of the graph G. Define* $||\Delta||_2$ *to be the maximum singular value of* Δ*. Then* $||\Delta||_2 \leq 2pdX$*.*

Proof. Since Δ is symmetric and positive semidefinite, $||\Delta||_2$ is equal to the maximum eigenvalue of Δ. Let E_i denote the set of edges incident on node i which fail. Then

$$\sum_{j\neq i} |\Delta_{ij}| = \sum_{j\in E_i} \nu_{ij}^{-1} \leq X \sum_{j\in E_i} 1 = X|E_i| \leq pdX, \tag{3.19}$$

for each $i \in V$. Furthermore, $\Delta_{ii} = \sum_{j\neq i} |\Delta_{ij}| \approx pdX$. Hence by the Gershgorin Disc Theorem [130], the eigenvalues of Δ must lie in the interval $[0, 2pdX]$. □

Lemma 3.4 leads to the following gradient approximation for $\mathbf{E}_\pi(R(S))$.

Theorem 3.6. *Let p, d, and X be as defined above, and let* $\delta = 2pdX$*. Then*

$$R(S|\tilde{G}) = Tr(\tilde{L}_{ff}^{-1}) \leq Tr(L_{ff}^{-1}) + \frac{(n-|S|)\delta}{\lambda_{min}(L_{ff})^2}, \tag{3.20}$$

where $\lambda_{min}(L_{ff})$ *is the smallest eigenvalue of* L_f*.*

Proof. $R(S)$ can be written as

$$\begin{aligned} Tr((L_{ff}-\Delta)^{-1}) &\approx Tr(L_{ff}^{-1}) + Tr(L_{ff}^{-1}\Delta L_{ff}^{-1}) \\ &\leq Tr(L_{ff}^{-1}) + Tr(U\Lambda^{-1}U^T\Delta U\Lambda^{-1}U^T) \\ &\leq Tr(L_{ff}^{-1}) + \sup_{\Delta} Tr(U\Lambda^{-1}U^T\Delta U\Lambda^{-1}U^T), \end{aligned}$$

where $L_f = U\Lambda U^T$ is the eigen-decomposition of L_f. The upper bound occurs when $\Delta = U\Omega U^T$ for some positive semidefinite diagonal matrix Ω. Together with Lemma 3.4, this

implies that

$$Tr((L_f - \Delta)^{-1}) \leq Tr(L_{ff}^{-1}) + Tr(U\Lambda^{-1}\Omega\Lambda^{-1}U^T) = Tr(L_{ff}^{-1})Tr(U^TU\Lambda^{-1}\Omega\Lambda^{-1})$$
$$= Tr(L_{ff}^{-1}) + Tr(\Lambda^{-1}\Omega\Lambda^{-1}) \leq Tr(L_{ff}^{-1}) + \frac{(n-|S|)\delta}{\lambda_{min}^2}.$$

□

Note that the upper bound on $R(S)$ can be used as a worst-case value for $R(S)$ in the presence of link failures. Once an appropriate method for computing or estimating $\mathbf{E}_\pi(R(S))$ has been chosen, the following lemma can be used to derive efficient approximation algorithms for both problems (3.18a) and (3.18b).

Lemma 3.5. *For any distribution π on $\mathcal{L}$, the function $\mathbf{E}_\pi(R(S))$ is supermodular.*

Proof. By definition, $\mathbf{E}_\pi(R(S)) = \sum_{L\in\mathcal{L}} R(S|L)\pi(L)$. Since the set $\mathcal{L}$ of possible weighted Laplacians is finite, this is a nonnegative weighted sum of supermodular functions, and hence is supermodular. □

As a corollary to Lemma 3.5, problems (3.18a) or (3.18b) can be solved using algorithms analogous to STATIC-k and SWITCHING-k respectively. The modified algorithms take the distribution π as an additional input parameter, and replace line seven in both algorithms with

$s_i^* \leftarrow \arg\max_{j\in V\setminus S} \{\mathbf{E}_\pi(R(S^*)) - \mathbf{E}_\pi(R(S^*\cup\{j\}))\}$.

3.4.2 Leader Selection Under Switching Between Predefined Topologies

A network may switch between a set of predefined topologies $\mathcal{T} = \{G_1, \ldots, G_M\}$, each with different link error variances represented by the corresponding weighted Laplacians $L_1, \ldots, L_M$ (e.g., a set of possible formations) in response to a switching signal from the network owner or environmental changes [101]. Leader selection under switching topologies can be divided into two cases. In the first case, the set of leaders is updated after each change in topology [89]. For this case, a different set of leaders S_i can be selected for each topology G_i using either STATIC-k (if a fixed number k of leaders is chosen to minimize

error) or SWITCHING-k (if the minimum number of leaders is chosen to achieve an error bound α).

In the second case, the same leader set S is chosen and used for all topologies [84]. Under this strategy, we consider two possible leader selection metrics, namely the average-case error, given as $R_{avg}(S) = \frac{1}{M}\sum_{i=1}^{M} R(S|L_i)$, and the worst-case error, given as $R_{worst}(S) = \max\{R(S|L_i) : i = 1, \ldots, M\}$. $R_{avg}(S)$ is a nonnegative weighted sum of supermodular functions, and hence is supermodular as a function of S. The problems of selecting up to k leaders in order to minimize $R_{avg}(S)$ and selecting the minimum number of leaders to achieve an error bound α can therefore be solved by modified versions of STATIC-k and SWITCHING-k, respectively. The modified versions of both leader selection algorithms take the topologies $\{G_1, \ldots G_M\}$ as input, and replace line 7 in both algorithms with $s_i^* \leftarrow \arg\max_{j \in V \setminus S} \{R_{avg}(S^*) \ - R_{avg}(S^* \cup \{j\})\}$.

If $R_{worst}(S)$ is used as a metric, however, note that the maximum of supermodular functions is, in general, not supermodular [72]. Alternate approaches for leader selection problems are given as follows.

Choosing k leaders to minimize worst-case error

The problem of choosing k leaders in order to minimize R_{worst} is stated as

$$\begin{aligned} \text{minimize} \quad & R_{worst}(S) \triangleq \max_{i=1,\ldots,M} R(S|L_i) \\ \text{s.t.} \quad & |S| \leq k \end{aligned} \tag{3.21}$$

Let S^* be the solution to (3.21), and let $R^*_{worst} = R_{worst}(S^*)$. R^*_{worst} is bounded below by 0 and bounded above by $R_{max} = \max_{i=1,\ldots,M} \max_{u \in V} \{R(\{u\}|L_i)\}$. As a preliminary, define

$$F_c(S) \triangleq \frac{1}{M}\sum_{i=1}^{M} \max\{R(S|L_i), c\}. \tag{3.22}$$

Note that $F_c(S)$ is a supermodular function of S.

An algorithm for approximating S^* is as follows. First, select parameters $\beta \geq 1$ and $\delta > 0$. The algorithm finds a set S satisfying $R(S) \leq R^*_{worst}$ and $|S| \leq \beta k$. Parameter δ determines the convergence speed of the algorithm. Define $\alpha^0_{min} = 0$ and $\alpha^0_{max} = R_{max}$. At the j-th iteration, let $\alpha^j = \frac{\alpha^{j-1}_{max} + \alpha^{j-1}_{min}}{2}$. The goal of the j-th iteration is to determine

if there is a set S^j such that $|S^j| \leq \beta k$ and $R(S^j|L_i) \leq \alpha^j$ for all $i = 1, \ldots, M$. This is accomplished by solving the optimization problem

$$\begin{aligned} \text{minimize} \quad & |S| \\ \text{s.t.} \quad & F_{\alpha^j}(S) \leq \alpha^j \end{aligned} \tag{3.23}$$

Since $F_{\alpha^j}(S)$ is supermodular, the solution to (3.23) can be approximated by an algorithm analogous to STATIC-α. If the approximate solution to (3.23), denoted S^j, satisfies $|S^j| \leq \beta k$, then set $\alpha^j_{max} = \alpha^{j-1}$ and $\alpha^j_{min} = \alpha^{j-1}_{min}$. Otherwise, set $\alpha^j_{max} = \alpha^{j-1}_{max}$ and $\alpha^j_{min} = \alpha^j$. The algorithm terminates when $|\alpha^j_{max} - \alpha^j_{min}| < \delta$ and returns the current set S^j. A pseudocode description of this algorithm is given as algorithm SWITCHING-k.

Since $\alpha^j_{max} - \alpha^j_{min}$ is strictly decreasing as j increases, SWITCHING-k converges. The optimality of SWITCHING-k is given by the following theorem.

Theorem 3.7. *When $\delta = \frac{1}{M}$ and β satisfies*

$$\beta \geq 1 + \log\left(\max_{v \in V}\left\{\sum_i R(\{v\}|L_i)\right\}\right),$$

SWITCHING-k returns a set S^* such that $\max_i \{R(S^*|L_i)\} \leq R^*_{worst}$ and $|S^*| \leq \beta k$.

Proof. The proof follows from the fact that $R(S|L_i)$ is a supermodular function for all i and Theorem 3 of [72]. □

Choosing leaders to achieve an error bound

In order to choose a minimum-size set of leaders to achieve an error bound α, the following optimization problem must be solved

$$\begin{aligned} \text{minimize} \quad & |S| \\ \text{s.t.} \quad & R(S|L_i) \leq \alpha \quad \forall i = 1, \ldots, M \end{aligned} \tag{3.24}$$

The following lemma leads to efficient algorithms for solving (3.24).

Algorithm 4 Algorithm for selecting up to k leaders to minimize worst-case error under switching topologies.

procedure STATIC-*alpha*($G_1, \ldots, G_M$, $\{\nu_{ij}^{(1)}, \ldots, \nu_{ij}^{(M)} : (i,j) \in E\}$, βk, δ)
  Input: Topologies $G_1, \ldots, G_M$
    Link error variances $\nu_{ij}^{(1)}, \ldots, \nu_{ij}^{(M)}$
    Maximum number of leaders βk, threshold δ
  Output: Set of leader nodes S^*
  Initialization: $S^* \leftarrow \emptyset$, $j \leftarrow 0$, $\alpha_{min}^j \leftarrow 0$, $\alpha_{max}^j \leftarrow R_{max}$
  while $\alpha_{max}^j - \alpha_{min}^j \geq \delta$ **do**
    $\alpha^j \leftarrow \frac{\alpha_{max}^j + \alpha_{min}^j}{2}$
    $r \leftarrow 0$, $S^j \leftarrow \emptyset$
    while $F_{\alpha^j}(S) \leq \alpha^j$ **do**
      $s_i^* \leftarrow \arg\max_{v \in V \setminus S} \{F_{\alpha^j}(S) - F_{\alpha^j}(S \cup \{v\})\}$
      $S^j \leftarrow S^j \cup \{s_i^*\}$, $r \leftarrow r + 1$
    end while
    if $r > \beta k$ **then**
      $\alpha_{max}^j \leftarrow \alpha^j$, $\alpha_{min}^j \leftarrow \alpha_{min}^{j-1}$
    else
      $\alpha_{min}^j \leftarrow \alpha^j$, $\alpha_{max}^j \leftarrow \alpha_{max}^{j-1}$
    end if
    $j \leftarrow j + 1$
  end while
  $S^* \leftarrow S^j$, **return** S^*
end procedure

Lemma 3.6. *Problem (3.24) is equivalent to*

$$\begin{aligned} \textit{minimize} \quad & |S| \\ \textit{s.t.} \quad & F_\alpha(S) \leq \alpha \end{aligned} \tag{3.25}$$

where $F_\alpha(S)$ is defined as in (3.22).

Proof. The proof follows from the facts that the objective functions of (3.24) and (3.25) are the same, as well as the fact that $F_\alpha(S) \leq \alpha$ if and only if $R(S|L_i) \leq \alpha$ for all i. □

Since $F_\alpha(S)$ is a supermodular function of S, this is a supermodular optimization problem similar to (3.17), and hence can be solved by an algorithm analogous to SWITCHING-k.

Leader selection for switching topologies under random node and link failures

As in the static network case, network with switching topologies may experience random link or node failures. In this case, the expected values of the average and worst-case system error are of interest when selecting the leaders.

Under random failures, the i-th topology can be represented as a random variable $\mathbf{G}_i$. Let π denote the joint distribution of $\mathbf{G}_1, \ldots, \mathbf{G}_M$, so that $\pi(G_1, \ldots, G_M) = Pr(\mathbf{G}_1 = G_1, \ldots, \mathbf{G}_M = G_M)$, and let $\pi_i(G) = Pr(\mathbf{G}_i = G)$. Note that the $\mathbf{G}_i$'s may not be independent; for example, under the random failure model of Section 3.4.1, the failure of an node in any topology implies failure in all topologies.

The average-case expected error $\mathbf{E}_\pi(R_{avg}(S))$ can be further simplified by

$$\mathbf{E}_\pi(R_{avg}) = \mathbf{E}_\pi\left(\frac{1}{M}\sum_{i=1}^{M} R(S|\mathbf{L}_i)\right) = \frac{1}{M}\sum_{i=1}^{M} \mathbf{E}_{\pi_i}(R(S|L_i)),$$

which follows from linearity of expectation and the fact that, by definition of π_i, $\mathbf{E}_\pi(R(S|L_i)) = \mathbf{E}_{\pi_i}(R(S|L_i))$. By Lemma 3.5, $\mathbf{E}_{\pi_i}(R(S|\mathbf{L}_i))$ is supermodular as a function of S for all i. Hence, $\mathbf{E}_\pi(R_{avg}(S))$ is a nonnegative weighted sum of supermodular functions, and is therefore a supermodular function of S.

The problem of minimizing $\mathbf{E}_\pi(R_{avg}(S))$ when the number of leaders cannot exceed k can be solved using an algorithm analogous to STATIC-k. Similarly, the problem of finding the smallest leader set S that is within an upper bound α on $\mathbf{E}_\pi(R_{avg}(S))$ can be solved

using an algorithm analogous to SWITCHING-k. Both modified algorithms take $\pi_1, \ldots, \pi_M$ as additional inputs and have line 7 changed to

$$s_i^* \leftarrow \arg\max_{v \in V \setminus S} \left\{ \frac{1}{M} \sum_{i=1}^{M} \mathbf{E}_{\pi_i} (R(S|\mathbf{L}_i) - R(S \cup \{v\}|\mathbf{L}_i)) \right\}.$$

Considering the expected worst-case error, $\mathbf{E}_\pi(R_{worst}(S))$, by Jensen's inequality [78],

$$\mathbf{E}_\pi \left(\max_{i=1,\ldots,M} \{R(S|\mathbf{L}_i)\} \right) \geq \max_{i=1,\ldots,M} \{\mathbf{E}_\pi(R(S|\mathbf{L}_i))\} = \max_{i=1,\ldots,M} \{\mathbf{E}_{\pi_i}(R(S|\mathbf{L}_i))\}. \quad (3.26)$$

The lower bound (3.26) is the worst-case expected error experienced when the leader set is S. Since $\mathbf{E}_{\pi_i}(R(S|\mathbf{L}_i))$ is supermodular as a function of S, the function

$$\tilde{F}_c(S) = \max \{\mathbf{E}_{\pi_i}(R(S|\mathbf{L}_i)), c\} \quad (3.27)$$

is supermodular in S. Therefore, the problem of selecting up to k leaders in order to minimize $\mathbf{E}_\pi(R_{worst}(S))$ can be approximately solved by an algorithm similar to SWITCHING-k. The modified algorithm takes $\pi_1, \ldots, \pi_M$ as additional input, and replaces the function $F_{\alpha^j}(S)$ at line 5 with the function $\tilde{F}_{\alpha^j}(S)$ defined in (3.27).

In order to choose the minimum-size set of leaders such that $\max_{i=1,\ldots,M} \{\mathbf{E}_\pi(R(S|\mathbf{L}_i))\}$ is below an error bound α, a supermodular optimization problem analogous to (3.24) can be used. The constraint of the modified problem is given by $\tilde{F}_\alpha(S) \leq \alpha$ with $\tilde{F}_\alpha(S)$ defined as in (3.27).

3.4.3 Leader Selection Under Arbitrarily Time-Varying Topologies

In this section, network with topologies that vary arbitrarily in time are considered. We assume that time is divided into steps, and that at step t, the network owner has knowledge of the topologies $G_1, \ldots, G_{t-1}$ for steps $1, 2, \ldots, t-1$, respectively, as well as their corresponding weighted Laplacians $L_1, \ldots, L_{t-1}$, but does not know the topology G_t for step t.

Under this model, a fixed set of leaders chosen at step $t = 1$, may give poor performance for the subsequent topologies $G_2, \ldots, G_T$. Instead, it is assumed that, at step t, a new set of leaders S_t is selected based on the observed topologies $G_1, \ldots, G_{t-1}$. The error for each

topology is given by $R(S_t|L_t)$. The leader selection problem for time-varying topologies is stated as

$$\begin{array}{ll} \text{minimize}_{S_1,\ldots,S_T} & \sum_{t=1}^{T} R(S_t|L_t) \\ \text{s.t.} & |S_t| \leq k \end{array} \tag{3.28}$$

Problem (3.28) is an online supermodular optimization problem [125]. The method for choosing a set of leader nodes S_t^* for the t-th time step is as follows. Consider the STATIC-k algorithm. Since G_t is unknown and random, the node j that maximizes $\{R(S_{t,i}^*) - R(S_{t,i}^* \cup \{j\})\}$ at the i-th iteration of the algorithm is also random. Let

$$\pi_{t,i}(l) = Pr(\arg\max_j \{R(S_{t,i}^*) - R(S_{t,i}^* \cup \{j\})\} = l). \tag{3.29}$$

Then for step t, instead of selecting $s_{t,i}^*$ deterministically as in Line 7 of STATIC-k, $s_{t,i}^*$ is selected probabilistically with distribution $\pi_{t,i}$ in (3.29).

In general, the exact values of $\pi_{t,i}$ will not be known during leader selection. To address this, an online learning technique is used to estimate $\pi_{t,i}$ based on observations from the previous $t-1$ time steps. Under this approach, a set of weights $\mathbf{w}_{t,1}, \ldots, \mathbf{w}_{t,k}$ is maintained, where $\mathbf{w}_{t,i}$ is a vector in $\mathbf{R}^n$ with $\mathbf{w}_{t,i}(j)$ representing the weight assigned to choosing node j as the i-th leader during step t. Define

$$s_{t,i}^{opt} \triangleq \arg\max_j \{R(S_{t,i-1}^*|L_t) - R(S_{t,i-1}^* \cup \{j\}|L_t)\}. \tag{3.30}$$

In other words $s_{t,i}^{opt}$ in (3.30) is the best possible choice of $s_{t,i}^*$ for the topology G_t. Then define the loss $l_{t,i,j}$ associated with choosing $s_{t,i}^* = j$ to be

$$l_{t,i,j} \triangleq 1 - \frac{R(S_t^*|L_t) - R(S_t^* \cup \{j\}|L_t)}{R(S_t^*|L_t) - R(S_t^* \cup \{s_{t,i}^{opt}\}|L_t)}. \tag{3.31}$$

At the end of step t, the value of $\mathbf{w}_{t,i}(j)$ is updated to $\mathbf{w}_{t+1,i}(j) = \beta^{l_{t,i,j}} \mathbf{w}_{t,i}(j)$, where $\beta \in (0, 1]$ is a system parameter that can be tuned to adjust the performance of the learning algorithm. This is interpreted as penalizing node j for suboptimal performance during interval t. By decreasing the value of β, nodes experiencing higher losses $l_{t,i,j}$ will be much less likely to be selected during the $(t+1)$-th time step. $\mathbf{w}_{t+1,i}$ is then normalized to obtain the estimated distribution $\pi_{t+1,i}$. This process is described in detail in algorithm ONLINE-k.

In analyzing this approach, the total error $\sum_t R(S_t|G_t)$ can be compared to the error achievable when all T topologies are known in advance. The following theorem gives a bound on the difference between these two errors.

Algorithm 5 Algorithm for selecting up to k leaders for an arbitrarily time-varying topology.

procedure ONLINE-k($\mathbf{w}_{t,1}, \ldots, \mathbf{w}_{t,k}$, $G_t = (V, E_t)$, ν_{ij}^t, β, k, S_t^*)
  Input: Current weights $\mathbf{w}_{t,1}, \ldots, \mathbf{w}_{t,k}$
    $G_t = (V, E_t)$, link error variances ν_{ij}^t
    Parameter $\beta \in (0, 1)$
    Maximum number of leader nodes k
    Current set of leader nodes S_t^*
  Output: Updated weights $\mathbf{w}_{t+1,1}, \ldots, \mathbf{w}_{t+1,k}$
    Set of leader nodes S_{t+1}^*
  Initialization: $S_{t+1}^* \leftarrow \emptyset$
  for $i = 1, \ldots, k$ **do**
    $s_{t,i}^{opt} \leftarrow \arg\max_j \{R(S_{t,i-1}^*) - R(S_{t,i-1}^* \cup \{j\})\}$
    for $j = 1, \ldots, n$ **do**
      $l_{t,i,j} \leftarrow 1 - \frac{R(S_{t,i-1}^*) - R(S_{t,i-1}^* \cup \{j\})}{R(S_{t,i-1}^*) - R(S_{t,i-1}^* \cup \{s_{t,i}^{opt}\})}$
      $\mathbf{w}_{t+1,i}(j) \leftarrow (\mathbf{w}_{t,i}(j))\beta^{l_{t,i,j}}$
    end for
    $\pi_{t+1,i} \leftarrow \mathbf{w}_{t+1,i} / \mathbf{1}^T \mathbf{w}_{t+1,i}$
    Choose $s_{t+1,i}^*$ randomly with distribution $\pi_{t+1,i}$
    $S_{t+1}^* \leftarrow S_{t+1}^* \cup \{s_{t+1,i}^*\}$
  end for
  return S
end procedure

Theorem 3.8. *Suppose that the algorithm ONLINE-k is executed for T steps, and let $G_1, \ldots, G_T$ be the topologies during those steps. Let R_{max} be defined as in Theorem 3.4.*

Define the error K to be

$$K \triangleq (1-1/e)\sum_{t=1}^{T} R(S_t|L_t) - \left(\max_{|S|=k}\left\{\sum_{t=1}^{T} R(S|L_t)\right\}\right). \tag{3.32}$$

Then $K \leq \mathcal{O}(\sqrt{R_{max}kT\log n})$.

Proof. $R(S|L_1),\ldots,R(S|L_T)$ is a sequence of supermodular functions bounded above by R_{max}. Then, by Lemma 4 of [125],

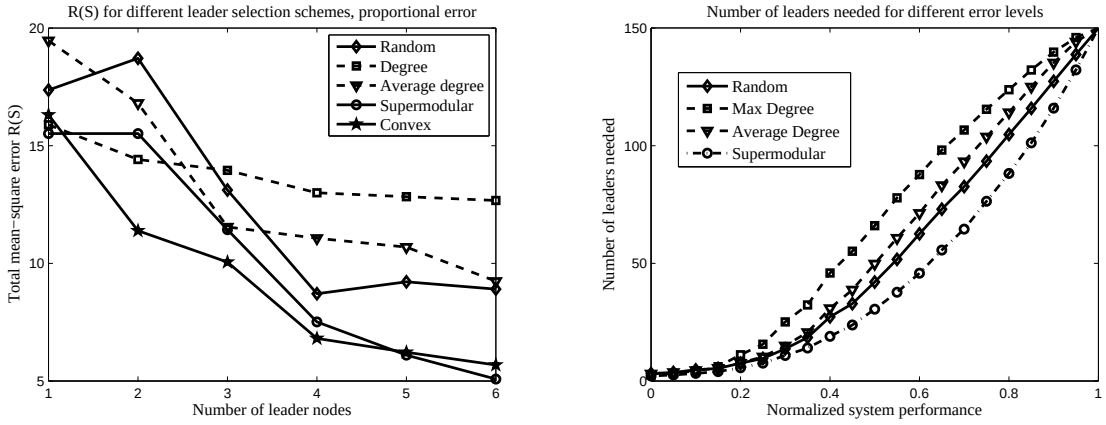
$$(1-1/e)\sum_{t=1}^{T} R((S_t^*|L_t) - \left(\max_{|S|=k}\left\{\sum_{t=1}^{T} R(S|L_t)\right\}\right) \leq \mathcal{O}(\sqrt{R_{max}kT\log n}) \tag{3.33}$$

as desired. □

3.5 Numerical Study

In this section, we evaluate the performance of our leader selection algorithms. A numerical study is carried out using Matlab. A network of 100 nodes is simulated, with nodes placed at random positions within a 1000m x 1000m rectangular area. Two nodes are assumed to share a link if they are deployed within 300m of each other. The error variance ν_{ij} of link (i,j) is assumed to be proportional to the distance between nodes i and j. Each data point in the following figures represents an ensemble average of 50 trials, unless otherwise indicated.

For comparison, the following leader selection algorithms are simulated. In the first algorithm, a random subset of nodes is chosen to act as leaders. In the second algorithm, the k nodes with highest degree (i.e., largest number of neighbors) are chosen to act as leaders. In the third algorithm, the k nodes with degree closest to the average degree are selected. The fourth algorithm simulated, for the case of selecting up to k leaders in a static topology, is the convex optimization approach of [47]. The fifth algorithm is our supermodular optimization. Note that, while the random, degree-based, and supermodular schemes are simulated for both static and dynamic networks, the convex relaxation approach of [47] is only applicable for selecting up to k leaders in a static network, and hence is only simulated for that case.



(a) (b)

Figure 3.1: (a) A comparison of our supermodular optimization approach to leader selection with the convex optimization approach of [47], as well as random and degree-based leader selection. Either the supermodular or convex optimization approach provides minimal error, depending on the number of leader nodes. (b) The supermodular optimization approach SWITCHING-k requires fewer leaders to satisfy the error constraint than either the degree-based or random heuristics. Error values are normalized to between 0 and 1.

Case 1: network with static network topology – Figure 3.1(a) compares the performance of the five algorithms considered for the problem of choosing up to k leaders in order to minimize the total system error. For this comparison, in order to reduce the runtime of the convex optimization approach, a smaller network of 25 nodes is used. Figure 3.1(a) shows the error achieved by the different leader selection algorithms for a fixed network topology and varying leader set size, in which either the convex or supermodular approaches provide optimal performance depending on the number of leader nodes. When $k = 1, 2, 5, 6$, our supermodular optimization approach results in lower mean-square error, while the convex optimization approach in [47] selects leaders with lower error when $k = 3, 4$.

For the problem of choosing the minimum number of leaders to achieve an error bound, the supermodular optimization approach requires only 40 leaders to achieve normalized error of 0.7 (for example), compared to 50 leaders for the random heuristic and over 60 leaders for the maximum degree method (Figure 3.1(b)). Figure 3.1(b) also suggests that the random heuristic consistently outperforms both degree-based algorithms. Selecting the

nodes with average degree also performs better than selecting the maximum-degree nodes to act as leaders.

The total system error experienced by the network as a function of network size is explored in Figure 3.2(a). The number of leaders is equal to $0.1n$, where n is the number of nodes. Since the deployment area remains constant, adding nodes to the network increases the number of links, resulting in smaller overall error. Hence, while the supermodular approach still outperforms the other methods, the difference in overall error decreases as the node density grows large.

Case 2: network experiencing random link failures – Figure 3.2(b) shows the error experienced for each method when links fail independently and at random, with probability ranging from 0 to 0.2. The number of leaders is equal to 10, while the network size is 100. The supermodular optimization algorithm uses the Monte Carlo approach described in Section 3.4.1. While each scheme sees a degradation in performance as the probability of failure increases, this degradation is minimized by the supermodular optimization method.

Case 3: network that switch between predefined topologies – Figure 3.2(c) shows the number of leaders needed to achieve an error level of 6 for each algorithm for network under switching topologies. For this evaluation, a set of M topologies, where M varied from 1 to 10, is generated at random based on the deployment area and node communication range described in the first paragraph of Section 3.5. A fixed leader set S is then selected using each heuristic. Each scheme requires a larger leader set as the number of prespecified topologies increased; however, for the supermodular optimization approach, a fixed set of 10 leaders provides an error of less than 6 for 10 different topologies. Overall, fewer leaders are needed for the supermodular optimization approach. Random leader selection requires fewer leaders than the average degree heuristic, which in turn outperforms selection of the highest-degree nodes as leaders.

The case of network with switching topologies and independent, random link failures is shown in Figure 3.2(d). Link failures increase the number of leaders required to achieve the error bound for each value of the number of topologies, M. While the supermodular

optimization approach continued to perform better than the other heuristics in most cases, the performance improvement was less significant.

Case 4: network with arbitrarily time-varying topology – Figure 3.3 shows the performance of leader selection schemes when the topology varies over time due to node mobility. Nodes are assumed to move according to a group mobility model, in which nodes attempt to maintain their positions with respect to a reference point [60]. The reference point varies according to a random walk with speed 30 m/s. Each node's position is equal to its specified position relative to the reference plus a uniformly distributed error. A new set of 10 leaders is selected every 10 seconds. As in the other cases, the supermodular optimization approach consistently provides the lowest error, followed by the random, average degree, and maximum degree heuristics. Moreover, the ONLINE-k algorithm improves its performance over time by observing which nodes provided the best performance when chosen as leaders and assigning those nodes a higher weight.

3.6 Conclusions and Future Work

In this chapter, the problem of selecting leaders in networked systems in order to minimize error due to communication link noise was studied. We analyzed the total mean-square error in the follower node states, and formulated the problem of selecting up to k leaders in order to minimize the error, as well as the problem of selecting the minimum-size set of leaders to achieve a given upper bound on the error. We examined both problems for different cases of network, including network with (a) static network topology, (b) topologies that experience random link failures, (c) switching between predefined topologies, and (d) topologies that vary arbitrarily over time. We showed that all of these cases can be solved within a supermodular optimization framework. We introduced efficient algorithms for selecting a set of leaders that approximates the optimum set up to a provable bound for each of the four cases. Our proposed approach was evaluated and compared with other methods, including random leader selection, selecting high-degree nodes as leaders, selecting average-degree nodes as leaders, and a convex optimization-based approach through a simulation study. Our study showed that supermodular leader selection significantly outperformed the random and degree-based leader selection algorithms in static as well as dynamic network

while providing provable bounds on the network performance, and provides performance comparable to the convex optimization approach.

3.6.1 Future Work

Generalized node dynamics and noise models: Current analysis and algorithms for leader selection under link noise make several assumptions on the network dynamics and noise model. The interactions between nodes are assumed to be linear with either uniform [47, 82, 108] or BLUE [12]. Additive Gaussian noise is also assumed [12, 47, 82, 108]. Generalization of these results to arbitrary linear as well as nonlinear node dynamics, as well as different types of communication channels between nodes (e.g., fading or interference) is one direction of future work, and would enable leader selection under a variety of realistic scenarios.

Leader selection based on graph invariants: The mean-square error due to link noise is proportional to the graph effective resistance, which is an example of a *graph invariant* [28]. Other invariants include spectral quantities such as the algebraic connectivity of the graph. Generalizing the supermodular optimization approach to arbitrary convex graph invariants would enable leader selection based on a broader class of metrics.

Phase transition in number of leader nodes: Figure 3.1(b) shows the number of leader nodes required to achieve a desired bound on the mean square error due to noise. When the error bound is large, very few leaders are required to achieve the bound, while almost all nodes must act as leaders when the desired error is small. In between these extremes, a phase transition occurs where an intermediate number of leaders is needed. Quantifying the properties of this phase transition, including the error bound at which the transition occurs and the sharpness of the transition, would provide insight into the fundamental limits on performance of networked systems in the presence of noise. The network topology and link noise characteristics are two factors that we expect to impact the phase transition.

Effectiveness of random and degree-based heuristics: The numerical results suggest that selecting a random set of nodes to act as leaders provides lower error, and hence better performance, then selecting high-degree nodes to act as leaders. A rigorous explanation of this phenomenon remains an open problem. One possible reason is that high-degree nodes

form cliques that are all connected to each other, while random nodes are evenly distributed throughout the network. Characterizing the class of networks for which random outperforms max-degree selection, including study of Erdos-Renyi and scale-free graphs, are additional directions of future work.

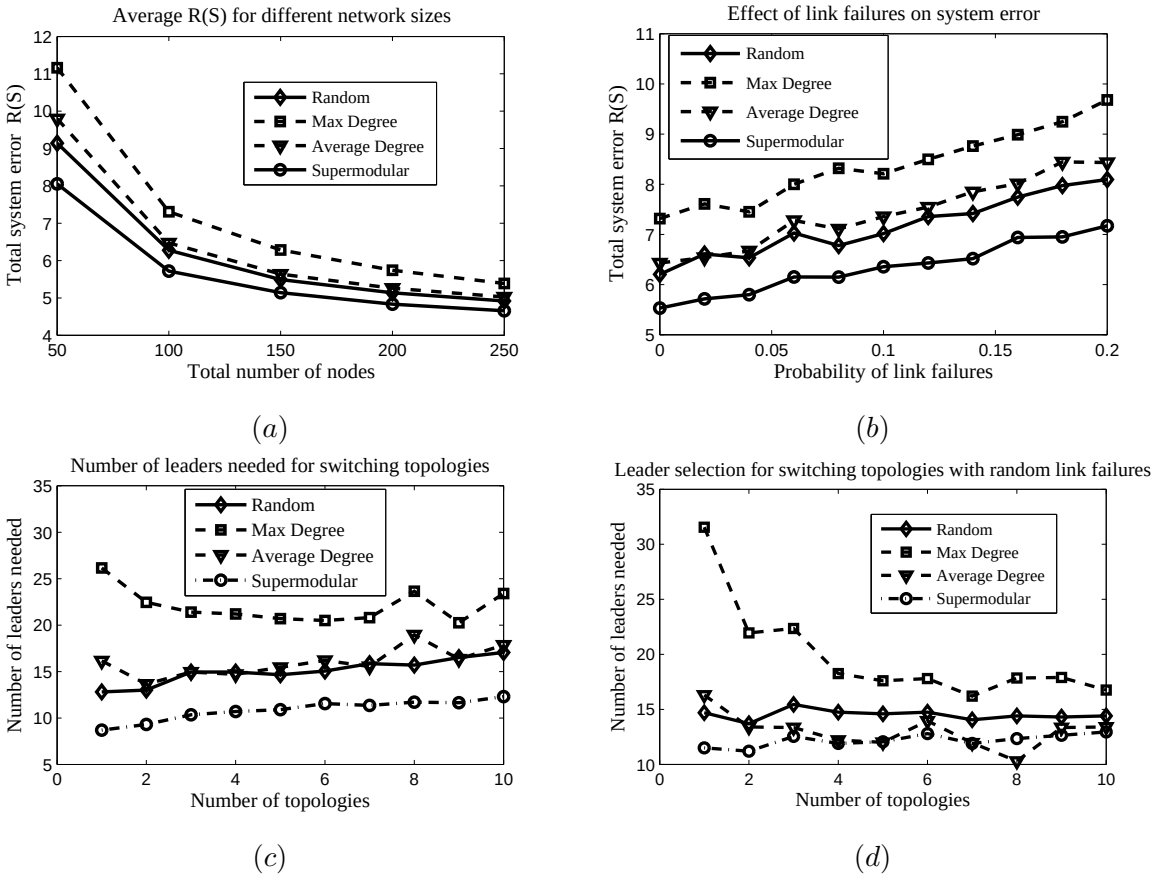


Figure 3.2: (a) Error experienced by network as the number of nodes increases. For each network size, the supermodular optimization approach provides the lowest overall error, followed by random selection, average degree-based selection, and maximum degree-based selection. Overall error decreases in all cases due to the increase in node density. (b) Effect of random failures on network error when number of nodes is 100 and number of leaders is 10. Although all selection schemes experience an increase in error due to link failures, the increase is smallest for the supermodular approach. (c) Number of leaders required to achieve system error of 6 when the network switches between M randomly generated topologies, where M varies from 1 to 10. The number of leaders needed increases with the number of topologies; the supermodular selection method requires the fewest number of leaders. (d) Number of leaders required when the network switches between M randomly generated topologies and links fail independently with probability $p = 0.05$. The number of leaders required is greater due to link failures.

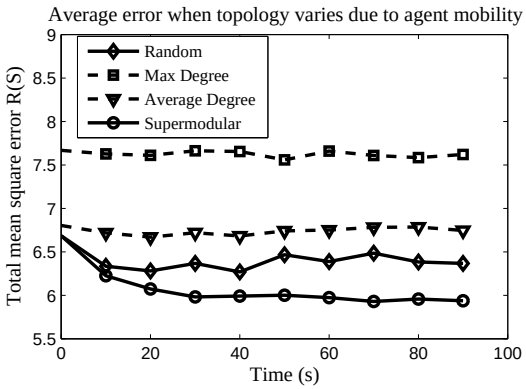


Figure 3.3: Leader selection under arbitrary time-varying topologies. Nodes move according to a group mobility model [60] with speed 30 m/s. A new set of leaders is selected every 10 seconds. While the performance of the four selection algorithms is comparable, the online supermodular approach ONLINE-k performs better over time by incorporating observed network topology information.

Chapter 4

ROBUSTNESS TO LINK NOISE INJECTION ATTACKS

In the previous chapter, we considered the problem of selecting leader nodes in order to ensure robustness to environmental link noise with known distribution. In hostile environments, networked systems may be disrupted by malicious adversaries in addition to environmental noise. By broadcasting an interfering signal in the vicinity of follower nodes, an adversary can corrupt the inputs broadcast by each node to its neighbors, leading to incorrect state updates and performance degradation.

Leaders that are selected in order to minimize error due to benign, environmental noise may leave the network vulnerable to an attack because an intelligent adversary can observe the set of leaders and inject link noise accordingly. Leaders can be selected in order to improve robustness to these noise injection attacks in two ways. In the first case, a fixed set of leaders is used for the lifetime of the system, and hence must be chosen to minimize the worst-case error [84]. In the second case, the nodes may be equipped with sensing hardware that allows them to monitor their environment, observe increased noise levels, and update the set of leaders accordingly [89]. Currently, however, there is no analytical approach for selecting leaders in either of these cases.

In this chapter, we study the problem of leader selection to mitigate the effects of noise injection attacks. We develop our approach within a two-player game framework, in which the networked system selects a set of leaders in order to minimize the mean-square error in the node states, while the adversary injects noise on a set of communication links in order to maximize this error. We make the following specific contributions:

- We study the problem of leader selection in the presence of an adversary mounting a link noise injection attack. We study two classes of the leader selection problem: (a) the problem of selecting a fixed set of leaders, and (b) the problem of adaptively choosing leaders in response to an attack.

- We model the selection of a fixed set of leaders as a supermodular Stackelberg game, leading to efficient algorithms for approximating the optimal leader set up to a provable bound. As an intermediate step, we prove that the limit of a sequence of supermodular functions and the integral of a collection of supermodular functions are supermodular.
- We formulate a repeated, simultaneous-move game modeling the interaction between the adversary and the networked system for the case where the leader set may change over time. We develop efficient algorithms for approximating a mixed-strategy Nash equilibrium for the game, and provide bounds on its optimality for each player.
- We evaluate the performance of our approach under both models via simulation study. We compare our leader selection methods to other approaches, including random and degree-based leader selection, and show that our scheme leads to lower overall mean-square error in the node states under link noise injection attacks. We further observe that allowing the leader set to vary over time improves the resilience to noise injection.

This chapter is organized as follows. In Section 4.1, the adversary model is presented, along with background on game theory. In Section 4.2, we introduce a game-theoretic model for selection of a fixed set of leaders in the presence of adversaries. In Section 4.3, we formulate a game for the case where the set of leader nodes changes in response to adversarial actions. Section 4.4 presents our simulation results. Section 4.5 concludes the chapter.

4.1 Model and Preliminaries

The system dynamics considered in this chapter are as in Chapter 3. In what follows, we describe the adversary model and give needed background on game theory.

4.1.1 Adversary Model

The system is assumed to be deployed in the presence of an adversary who is capable of injecting noise on the links between nodes by broadcasting an interfering signal in the vicinity of the nodes. This noise injection leads to an overall error variance $\nu_{ij} = \nu_{ij}^0 + \nu_{ij}^A$, where ν_{ij}^0 and ν_{ij}^A are the variances of the error on link (i, j) due to ambient noise and adversarial noise, respectively. We let $R(S, \nu^0 + \nu^A)$ denote the total mean-square error in the node states when the leader set is S and the noise distribution is $\nu = \nu^0 + \nu^A$.

The variance ν_{ij}^A is equal to the received strength of the interfering signal broadcast by the adversary. The received strength depends on the position of the receiver, denoted $y_j \in \mathbb{R}^3$, the position of the adversary, denoted $z \in \mathbb{R}^3$, the transmit power of the adversary for link (i, j), denoted P_{ij}, and the path-loss constant of the propagation medium, denoted α. We assume that, in order to avoid detection, the adversary does not choose its position to coincide with any nodes, so that $z \neq y_j$ for all j. The resulting error variance is given by $\nu_{ij}^A = P_{ij}||y_j - z||_2^{-\alpha}$. It is assumed that the adversary has a constraint P_A on the total power available, so that

$$\sum_{(i,j)\in E} P_{ij} = \sum_{(i,j)\in E} \nu_{ij}^A ||z - y_j||_2^\alpha \leq P_A \tag{4.1}$$

The adversary is assumed to know the network topology and the environmental noise characteristics ν_{ij}^0 for all $(i, j) \in E$. Furthermore, since the leader nodes do not follow the dynamics (3.1), the adversary can determine the leader set S by eavesdropping on the nodes' state values and observing which nodes do not update their states according to (3.1).

4.1.2 Background – Game Theory

A game is defined by a set of players $\{\mathcal{P}_1, \ldots, \mathcal{P}_m\}$. Each player $\mathcal{P}_i$ has a set of strategies $\mathcal{S}_i$ and a utility function $U_i : \mathcal{S}_1 \times \cdots \times \mathcal{S}_m \to \mathbf{R}$. The utility function U_i represents $\mathcal{P}_i$'s benefit from its action $s_i \in \mathcal{S}_i$ and the actions of the remaining players. The goal of each player $\mathcal{P}_i$ is to maximize its utility U_i.

For simplicity, we restrict ourselves to two-player games ($m = 2$). It is assumed that each player knows the strategy space and utility function of the other player. In addition, in

a *Stackelberg* game, player $\mathcal{P}_2$ observes the strategy $s_1 \in \mathcal{S}_1$ chosen by $\mathcal{P}_1$ before choosing a strategy $s_2 \in \mathcal{S}_2$. In order to maximize its utility, $\mathcal{P}_2$ will therefore choose strategy $s_2^* \in \mathcal{S}_2$ that satisfies[1]

$$s_2^* \in \arg\max_{s_2 \in \mathcal{S}_2} U_2(s_1, s_2) \tag{4.2}$$

Let $s_2^*(s_1)$ denote $\mathcal{P}_2$'s optimal strategy when $\mathcal{P}_1$ chooses strategy s_1. $\mathcal{P}_1$ will therefore choose strategy s_1^* that maximizes its utility given $\mathcal{P}_2$'s response:

$$s_1^* \in \arg\max_{s_1 \in \mathcal{S}_1} U_1(s_1, s_2^*(s_1)) \tag{4.3}$$

By contrast, in a *simultaneous-move game*, the two players choose their strategies at the same time, so that neither player observes the strategy of the other player. In this case, when the players are rational, they will choose their strategies according to a *Nash equilibrium*, defined as follows [8].

Definition 4.1. *A pair of strategies* $(s_1^*, s_2^*) \in \mathcal{S}_1 \times \mathcal{S}_2$ *is a* Nash equilibrium *if and only if*

$$s_1^* = \arg\max_{s_1 \in \mathcal{S}_1} U_1(s_1, s_2^*) \tag{4.4}$$

$$s_2^* = \arg\max_{s_2 \in \mathcal{S}_2} U_2(s_1^*, s_2) \tag{4.5}$$

In words, if (s_1^*, s_2^*) is a Nash equilibrium, then $\mathcal{P}_i$ cannot change its strategy s_i^* without decreasing his utility. Any strategy s_1^* (resp. s_2^*) satisfying (4.4) (resp. (4.5)) is a *best response* for player $\mathcal{P}_1$ (resp. $\mathcal{P}_2$).

Each player may attempt to improve its performance by randomizing over a set of strategies. This concept is defined as follows.

Definition 4.2. *A mixed strategy for player* $\mathcal{P}_i$ *consists of a set of ordered pairs* $\{(s_i^{(1)}, p_1), \ldots, (s_i^{(r)}, p_r)\}$, *where* $s_i^{(j)} \in \mathcal{S}_i$ *for all* j *and the* p_j*'s are nonnegative real numbers that sum to* 1. *Under this strategy, player* $\mathcal{P}_i$ *chooses strategy* $s_i^{(j)}$ *with probability* p_j. *A* mixed strategy equilibrium *is a Nash equilibrium in which one or more players uses a mixed strategy.*

[1] When more than one strategy s_2^* satisfies (4.2), we assume that $\mathcal{P}_2$ chooses the strategy satisfying (4.2) that minimizes U_1.

4.2 Problem Formulation – Fixed Leader Set

In this section, we consider the problem of choosing a fixed set of leaders in the presence of an adversary injecting noise.

4.2.1 Game Definition

In this setting, the networked system first chooses a set of up to k leaders. The adversary then observes the set of leaders and chooses a set of error variances ν_{ij}^A satisfying the adversary's power constraint, given by (4.1). The goal of the adversary is to choose the vector ν^A such that the total system error $R(S, \nu^0 + \nu^A)$ is maximized, while the networked system's goal is to choose a set of leaders S such that the total error in the worst case is minimized.

We formulate this problem as a Stackelberg game, in which the first player, $\mathcal{P}_1$, is the networked system and the second player, $\mathcal{P}_2$, is the adversary. The strategy space $\mathcal{S}_1$ of the networked system is given by the set of possible leader sets, $\mathcal{S}_1 = \{S \subseteq V : |S| \leq k\}$. The adversary's strategy space, $\mathcal{S}_2$, consists of the set of feasible error variances (4.1).

The MAS utility is given by $U_N(S, \nu^A) = -R(S, \nu^0 + \nu^A)$, while the adversary's utility is given by $U_A(S, \nu^A) = R(S, \nu^0 + \nu^A)$, so that the networked system's goal is to minimize the total error variance, while the adversary's goal is to maximize it. In what follows, we explore the optimal pure strategies for each player in detail, leaving the analysis of possible mixed strategies as future work.

4.2.2 Solution Algorithms for Adversary

For a given leader set, the adversary's goal is to choose the error variances ν^A such that the total system error $R(S, \nu)$ is maximized. Hence the adversary's optimal strategy is given by the solution to the optimization problem

$$\begin{aligned} \text{maximize} \quad & R(S, \nu^0 + \nu^A) \\ \nu_{ij}^A & \\ \text{s.t.} \quad & \nu_{ij}^A \geq 0 \quad \forall (i,j) \in E \\ & \textstyle\sum_{(i,j)\in E} \nu_{ij}^A ||z - y_j||_2^\alpha \leq P_A \end{aligned} \tag{4.6}$$

The following theorem leads to efficient solution algorithms for (4.6).

Theorem 4.1. *The function $R(S,\nu)$ is a concave function of $\{\nu_{ij} : (i,j) \in E\}$.*

As a first step towards proving Theorem 4.1, the following intermediate lemmas are needed.

Lemma 4.1. *Let v and J denote vectors in $\mathbf{R}^n$ such that $Lv = J$, $v_i = 0$ for all $i \in S$, $J_u = 0$, and $J_i = 0$ for all $i \in V \setminus (S + u)$.*

Proof. The equation $Lv = J$ can be written in long form as

$$\begin{pmatrix} L_{ff} & L_{fl} \\ L_{lf} & L_{ll} \end{pmatrix} \begin{pmatrix} v_f \\ v_l \end{pmatrix} = \begin{pmatrix} J_f \\ J_l \end{pmatrix} \tag{4.7}$$

Substituting $v_l = 0$ yields $L_{ff}v_f = J_f$, which in turn implies that $v_f = L_{ff}^{-1}J_f$. Let e_u denote the vector with a 1 in the u-th position and 0s elsewhere. Then multiplying both sides by e_u^T yields

$$v_u = (L_{ff}^{-1})_{u1}J_1 + \cdots + (L_{ff}^{-1})_{u(n-|S|)}J_{n-|S|} \tag{4.8}$$

Since $J_i = 0$ for $i \neq u$, (4.8) reduces to $(L_{ff}^{-1})_{uu}J_u = (L_{ff}^{-1})_{uu} = v_u$, as desired. □

In order to prove the next lemma, the following definition is required.

Definition 4.3. *A function $\mu : E \to \mathbf{R}$ is a* flow *if the following conditions hold for any $(i,j) \in E$. First, $\mu_{ij} = -\mu_{ji}$. Second, $\sum_{j \in N(i)} \mu_{ij} = 0$ for $i \in V \setminus (S + u)$. μ is defined to be a* unit flow *if the additional condition $\sum_{j \in N(u)} \mu_{uj} = 1$ holds.*

The following lemma gives a property of unit flows.

Lemma 4.2. *Let $w : V \to \mathbf{R}$ be any function on V and let μ be a flow. Then for any $a, b \in V$,*

$$(w_a - w_b) \sum_{j \in N(a)} \mu_{aj} = \frac{1}{2} \sum_{(i,j) \in E} (w_i - w_j)\mu_{ij} \tag{4.9}$$

The proof of this lemma can be found in [45, Ch 1]. One final lemma is needed before the proof of Theorem 4.1.

Lemma 4.3. *$(L_{ff}^{-1})_{uu}$ is equivalent to*

$$(L_{ff}^{-1})_{uu} = \min \left\{ \sum_{(i,j)\in E} \nu_{ij}\mu_{ij}^2 : \ \mu_{ij} \text{ is a unit flow} \right\}$$

Proof. Let v be as in the statement of Lemma 4.1, and define $\lambda_{ij} : E \to \mathbf{R}$ by $\lambda_{ij} = \nu_{ij}^{-1}(v_i - v_j)$. First, we show that λ_{ij} is a unit flow. $\lambda_{ij} = -\lambda_{ji}$ follows from the definition. Further, $\sum_{j\in N(i)} \nu_{ij}^{-1}(v_i - v_j) = J_i$, since v is defined by $Lv = J$. Thus $\sum_{j\in N(i)} \lambda_{ij} = 0$ if $i \in V \setminus (S + u)$ and $\sum_{j\in N(i)} \lambda_{ij} = 1$ if $i = u$, satisfying the second and third requirements for a unit flow.

We next show that

$$\sum_{(i,j)\in E} \nu_{ij}\lambda_{ij}^2 = (L_{ff}^{-1})_{uu} \tag{4.10}$$

From Lemma 4.2,

$$(v_u - v_s) \sum_{j\in N(u)} \lambda_{uj} = \frac{1}{2} \sum_{(i,j)\in E} (v_i - v_j)\lambda_{ij} \tag{4.11}$$

for any $s \in S$. By definition of λ, $(v_i - v_j) = \nu_{ij}\lambda_{ij}$, and so the right hand side of (4.11) is equivalent to $\frac{1}{2}\sum_{(i,j)\in E} \nu_{ij}\lambda_{ij}^2$. Meanwhile, since λ is a unit flow, the left-hand side of (4.11) is equal to $v_u - v_s$. By definition, $v_s = 0$, and by Lemma 4.1, $v_u = (L_{ff}^{-1})_{uu}$.

Now, let μ_{ij} be another unit flow, and note that $\mu_{ij} = \lambda_{ij} + \phi_{ij}$, where ϕ_{ij} is a flow with $\sum_{j\in N(u)} \phi_{uj} = 0$. Then

$$\begin{aligned}
\sum_{(i,j)\in E} \nu_{ij}\mu_{ij}^2 &= \sum_{(i,j)\in E} \nu_{ij}(\lambda_{ij} + \phi_{ij})^2 \\
&= \sum_{(i,j)} \nu_{ij}\lambda_{ij}^2 + 2\sum_{(i,j)} \phi_{ij}(v_i - v_j) + \sum_{(i,j)\in E} \nu_{ij}\phi_{ij}^2 && (4.12)\\
&= \sum_{i,j} \nu_{ij}\lambda_{ij}^2 + 4v_u \sum_{j\in N(u)} \phi_{uj} + \sum_{i,j} \nu_{ij}\phi_{ij}^2 && (4.13)\\
&= \sum_{i,j} \nu_{ij}\lambda_{ij}^2 + \sum_{i,j} \nu_{ij}\phi_{ij}^2 \geq \sum_{i,j} \nu_{ij}\lambda_{ij}^2 && (4.14)
\end{aligned}$$

where (4.12) follows by expanding the previous equation and by definition of λ, (4.13) follows from Lemma 4.2 and (4.14) follows from the fact that $\sum_{j\in N(u)} \phi_{uj} = 0$. □

Proof of Theorem 4.1. First, note that $\sum_{(i,j)\in E} \nu_{ij}\mu_{ij}^2$ is linear as a function of ν_{ij}. By Lemma 4.3, $(L_f^{-1})_{uu}$ is a pointwise minimum of linear functions and is therefore concave. □

Corollary 4.1. *Problem (4.6) is a concave optimization problem.*

Proof. The proof follows from Theorem 4.1 and the fact that the constraints of (4.6) are convex in ν_{ij}^A. □

As a result, the optimal strategy for the adversary can be computed in polynomial time using interior point algorithms [18].

4.2.3 Solution Algorithms for Leader Selection

Since the adversary will choose the noise injection strategy that maximizes the error due to link noise, the goal of the networked system is to select leaders such that this worst-case error is minimized. The optimal strategy is therefore given as the solution to the optimization problem

$$\begin{array}{ll} \text{minimize} & \max_{\nu^A} R(S, \nu^0 + \nu^A) \\ S & \\ \text{s.t.} & |S| \leq k \end{array} \tag{4.15}$$

Define $R(S) \triangleq \max_{\nu^A} R(S, \nu)$. Problem (4.15) involves optimizing over $\binom{n}{k}$ possible leader sets, which is infeasible when n or k is large. Moreover, functions of the form $g(S) = \max_i f_i(S)$, where $f_i(S)$ is supermodular, are not supermodular in general. We instead introduce an equivalent, supermodular formulation and derive a solution algorithm for (4.15) as a result.

As a preliminary, define $F_\zeta(S)$ by

$$F_\zeta(S) \triangleq \frac{1}{|\mathcal{S}_2|} \int_{\mathcal{S}_2} \max\left\{R(S, \nu^0 + \nu^A), \zeta\right\} \, d\nu^A \tag{4.16}$$

where $|\cdot|$ denotes the Lebesgue measure of a set. The function $F_\zeta(S)$ is supermodular as a function of S (Lemma 4.4). An algorithm for solving (4.15), based on the supermodularity of $F_\zeta(S)$, is as follows.

First, select parameters $\beta \geq 1$ and $\delta > 0$. The algorithm finds a set S satisfying $R(S) \leq R^*$, where R^* is the optimal value of (4.15), and $|S| \leq \beta k$.

Define $\zeta^0_{min} = 0$ and $\zeta^0_{max} = R(\{1\})$, the error R corresponding to leader set $S = \{1\}$. At the j-th iteration, let $\zeta^j = \frac{\zeta^{j-1}_{max}+\zeta^{j-1}_{min}}{2}$. The goal of the j-th iteration is to determine if there is a set S^j satisfying $R(S^j) \leq \zeta^j$ with $|S| \leq \beta k$. This is accomplished by solving the optimization problem

$$\begin{aligned} \text{minimize} \quad & |S| \\ \text{s.t.} \quad & F_\zeta(S) \leq \zeta^j \end{aligned} \tag{4.17}$$

noting that $R(S) \leq \zeta^j$ if and only if $F_\zeta(S) \leq \zeta^j$.

Problem (4.17) is solved as follows. Initialize $S^j = \emptyset$. At each iteration, choose v^* as

$$v^* = \arg\max \{F_\zeta(S^j) - F_\zeta(S^j \cup \{v\}) : v \in V \setminus S\} \tag{4.18}$$

Set $S^j = S^j \cup \{v\}$. The process continues until $F_\zeta(S^j) \leq \zeta^j$.

If S^j satisfies $|S^j| \leq \beta k$, then set $\zeta^j_{max} = \zeta^j$ and $\zeta^j_{min} = \zeta^{j-1}_{min}$. Otherwise, set $\zeta^j_{min} = \zeta^j$ and $\zeta^j_{max} = \zeta^{j-1}_{max}$. The algorithm terminates when $\zeta^j_{max} - \zeta^j_{min} < \delta$ and returns the set S^j. A pseudocode description of this approach is given as algorithm **Fixed-k**.

Theorem 4.2. *If β satisfies*

$$\beta \geq 1 + \log \left\{ \frac{\max_{v \in V} F_\zeta(\{v\})}{\zeta^*} \right\} \tag{4.19}$$

*then **Fixed**-k returns a set S satisfying $R(S) \leq R(S^*)$ and $|S| \leq \beta k$.*

The following lemma is needed to prove Theorem 4.2.

Lemma 4.4. *The function $F_\zeta(S)$ defined in (4.16) is supermodular as a function of S.*

Proof. By the theory of Riemann integration, $F_\zeta(S)$ can be approximated by a sequence of functions

$$F^k_\zeta(S) = \sum_{l=0}^{L_k} \max \{R(S, R(P^k_l)), \zeta\} |C^k_l| \tag{4.20}$$

where $|\cdot|$ denotes the Lebesgue measure of a set, the sets C^k_l satisfy $|C^k_l| = \delta_k$, with $\delta_k \to 0$ as $k \to \infty$, $P_l \in C^k_l$, and $\mathcal{S}_2 \subseteq \bigcup_{l=0}^{L_k} C^k_l$. Now, since $R(S, R)$ is supermodular for fixed R (Theorem 3.1), $\max \{R(S, R), \zeta\}$ is supermodular as a function of S (Lemma 2.5). $F^k_\zeta(S)$ is therefore a nonnegative weighted sum of supermodular functions, and thus is supermodular by Lemma 2.4. This implies that $F_\zeta(S)$ is the limit of a sequence of supermodular functions, and so $F_\zeta(S)$ is supermodular by Theorem 2.1. □

Algorithm 6 Algorithm for choosing up to k fixed leaders under noise injection.

procedure FIXED-$k(G = (V, E)$, $\{\nu_{ij}^0 : (i, j) \in E\}$, k, β, δ, $\{y_j : j \in V\}$, $p)$
  Input: $G = (V, E)$, link error variances ν_{ij}^0
    Maximum number of leader nodes k
    Parameters β and δ
    Node positions y_j, $j \in V$, adversary position p
  Output: Set of leader nodes S^*
  while $\zeta_{max} - \zeta_{min} \geq \delta$ **do**
    $\zeta_{min} \leftarrow 0$, $\zeta_{max} \leftarrow \max_{\nu^A} R(\{1\}, \nu^0 + \nu^A)$
    $j \leftarrow 0$
    while $F_\zeta(S^j) \geq \zeta$ **do**
      $v^* \leftarrow \arg\max_{v \in V \setminus S^j} \left\{F_\zeta(S^j) - F_\zeta(S^j \cup \{v\})\right\}$
      $S^j \leftarrow S^j \cup \{v^*\}$
    end while
    if $|S^j| > \beta k$ **then**
      $\zeta_{min} \leftarrow \zeta$
    else
      $\zeta_{max} \leftarrow \zeta$
    end if
  end while
  $S \leftarrow S^j$, **return** S
end procedure

Lemma 4.5. *For fixed ζ, for the optimization problem*

$$\begin{aligned} \textit{minimize} \quad & |S| \\ \textit{s.t.} \quad & F_\zeta(S) \leq \zeta \end{aligned} \tag{4.21}$$

the greedy algorithm of lines 6-8 in ***Fixed****-k returns a set S with*

$$\frac{|S|}{|S^*|} \leq 1 + \log_e \left\{ \frac{\max_v F_\zeta(\{v\})}{\zeta} \right\}, \tag{4.22}$$

where S^* is the optimum solution to (4.21).

Proof. The proof follows from Theorem 1 of [136] and the supermodularity of $F_\zeta(S)$. □

Proof of Theorem 4.2. From Lemma 4.5, solving a problem of the form (4.21) with $\zeta = \zeta^*$ will return a set S with $|S| \geq \beta k$ and β is as in the statement of Theorem 4.2. Furthermore, the algorithm is guaranteed to reach $\zeta = \zeta^*$, because ζ^j is strictly decreasing as long as $|S^j| \leq \beta k$, which will hold for all $\zeta^j > \zeta^*$. □

Corollary 4.1 implies that the adversary can compute the best-response to the chosen leader set S, while Theorem 4.2 proves that the leader selection strategy is within a provable bound of the optimum. These strategies, when taken together, therefore form an approximate Stackelberg equilibrium.

4.3 Problem Formulation – Dynamic Leader Set

We now consider the case where the set of leaders can change over time.

4.3.1 Game Definition

Under this model, the networked system periodically updates the leader set S in order to minimize the overall system error $R(S, \nu)$, based on the observed noise characteristics ν. The adversary, upon observing a change in S, chooses a new noise injection strategy in order to maximize the error experienced by the networked system. This leads to a repeated game model for the interaction between the networked system and the adversary.

Formally, the networked system is the first player, $\mathcal{P}_1$, while the adversary is the second player, $\mathcal{P}_2$. At the t-th iteration of the game, the networked system selects a leader set

S_t satisfying $|S_t| \leq k$. The adversary is unaware of the leader set for a time T, and hence chooses a vector of link error variances $\nu_t^A \in \mathcal{S}_2$ without knowledge of S_t, where $\mathcal{S}_2$ is defined as in Section 4.2. After time T elapses, the adversary discovers S_t and chooses a new vector of link error variances, $\tilde{\nu}_t^A \in \mathcal{S}_2$. An additional time T' elapses until the next iteration.

The penalty of the network at the t-th iteration is given by the average system error experienced, so that

$$U_N(S_t, \nu_t^A, \tilde{\nu}_t^A) = -\left(\frac{T}{T+T'}R(S, \nu^0 + \nu_t^A) + \frac{T'}{T+T'}R(S, \nu^0 + \tilde{\nu}_t^A)\right)$$

Similarly, the utility of the adversary is equal to the average system error:

$$U_A(S_t, \nu_t^A, \nu_t^{A\prime}) = \frac{T}{T+T'}R(S, \nu^0 + \nu_t^A) + \frac{T'}{T+T'}R(S, \nu^0 + \tilde{\nu}_t^A)$$

In what follows, it is assumed that it takes more time for the adversary to determine the leader set than for the MAS to detect the increase in error due to the noise injection attack, so that $T \gg T'$. Since the adversary and MAS are not aware of each other's strategies during this interval, this is a repeated simultaneous game with $U_N(S_t, \nu_t^A) \approx -R(S, \nu^0 + \nu_t^A)$ and $U_A(S_t, \nu_t^A) \approx R(S, \nu^0 + \nu_t^A)$. We first study the best-response behavior of each player. We then analyze the equilibria of the game based on the best-response behavior.

4.3.2 Best-Response Strategies

We first analyze the best-response strategy for the adversary at each iteration t. For a given choice of S_t, the adversary's best response to S_t is given by

$$\begin{aligned} \text{maximize} \quad & R(S_t, \nu^0 + \nu_t^A) \\ \text{s.t.} \quad & \nu_t^A \in \mathcal{S}_2 \end{aligned} \tag{4.23}$$

The function $R(S_t, \nu^0 + \nu_t^A)$ is a convex function of ν_t^A by Theorem 4.1. Hence the adversary's optimal noise allocation ν_t^A at each iteration can be obtained by solving (4.23) via convex optimization.

The MAS's problem of choosing the optimal leader set to minimize noise in response to ν_t^A is formulated as

$$\begin{aligned} \text{minimize} \quad & R(S_t, \nu^0 + \nu_t^A) \\ \text{s.t.} \quad & |S_t| \leq k \end{aligned} \tag{4.24}$$

Problem (4.24) can be approximated by supermodular optimization as discussed in Chapter 3.

While minimizing a supermodular function is NP-hard in general, a greedy algorithm can be used to approximate the optimal leader set S_t [96]. In the algorithm, the leader set $S_t = \emptyset$ initially. At each iteration, the node v^* satisfying

$$v^* = \arg\max \left\{ v \in V \setminus S_t : R(S_t, \nu^0 + \nu_t^A) - R(S_t \cup \{v\}, \nu^0 + \nu_t^A) \right\}$$

is added to the leader set. The algorithm terminates after k iterations. A pseudocode description as algorithm **BestResponse**-k.

Algorithm 7 Algorithm for selecting up to k leaders to minimize worst-case error under noise injection attack.

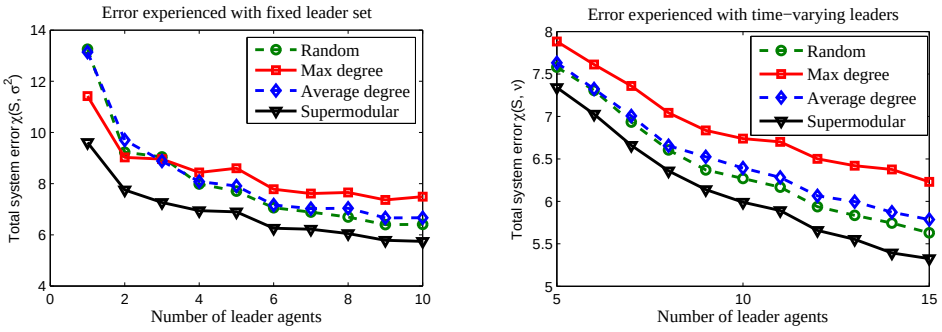
procedure BESTRESPONSE-$k(G = (V, E)$, $\{\nu_{ij}^t : (i, j) \in E\}$, $k)$
  Input: $G = (V, E)$, link error variances ν_{ij}^t
    Maximum number of leader nodes k
  Output: Leader set S_t
  $S_t \leftarrow \emptyset$, $l \leftarrow 0$
  while $l < k$ **do**
    $v^* \leftarrow \arg\max \{v \in V \setminus S_t : R(S_t, \nu_t) - R(S_t \cup \{v\}, \nu_t)\}$
    $S_t \leftarrow S_t \cup \{v^*\}$, $l \leftarrow l + 1$
  end whilereturn S_t
end procedure

Theorem 4.3. *Let S_t^* be the optimal solution to (4.24). Then the set S returned by* ***BestResponse****-k satisfies*

$$R(S_t, \nu_t^0 + \nu_t^A) \leq \left(1 - \frac{1}{e}\right) R(S_t^*, \nu_t^0 + \nu_t^A) + \frac{1}{e} R_{max} \tag{4.25}$$

where $R_{max} \triangleq \max_i \sum_{j \in V} R(i, j)$.

Proof. The proof follows from Proposition 4.1 of [96] and the supermodularity of the error R. □



(a) (b)

Figure 4.1: (a) Simulation of selection of a fixed set of leaders in the presence of adversarial noise using random, degree-based, and supermodular algorithms. While all schemes experience a decrease in performance, the supermodular selection approach provides the most robustness to noise. (b) Simulation of the approximate Nash equilibrium that arises from updating the leader set over time. All schemes outperform the case of fixed leader selection, while the supermodular optimization approach outperforms both random and degree-based approaches.

4.3.3 Equilibrium Analysis

As discussed in Section 4.1.2, the networked system and the adversary will maximize their utilities by playing a Nash equilibrium strategy at each iteration t. In general, determining Nash equilibria of two-player games is PPAD-complete [123, Ch. 4.1]. Moreover, in this case, the MAS's best-response strategy is NP-hard to compute. Instead, the following algorithm, first proposed for general two-player games [40], can be used to efficiently compute an approximate mixed-strategy equilibrium. We note that there may be other equilibria, which we will characterize in future work.

The MAS first chooses a set of leaders S. The adversary computes $\hat{\nu}^A$ by solving the best-response problem of (4.23) based on the leader set S. The MAS approximates the best-response S' to $\hat{\nu}^A$ using **BestResponse**-k.

The MAS's strategy is to choose leader set S with probability $1/2$ and to choose S' with probability $1/2$, corresponding to a mixed strategy $\{(S, 1/2), (S', 1/2)\}$. The adversary's strategy is to choose link error variances $\hat{\nu}^A$ with probability 1. A pseudocode description of the algorithm for approximating a Nash equilibrium is given as algorithm **Approx-NE**.

Algorithm 8 Algorithm for computing an approximate Nash equilibrium.

procedure APPROX-NE($G = (V, E)$, $\{\nu_{ij}^0 : (i,j) \in E\}$, k, y, P)
  Input: $G = (V, E)$, link error variances ν_{ij}^0
    Maximum number of leader nodes k
    Adversary's position y and power constraint P
  Output: Set of noise error variances $\nu_{ij}^A : (i,j) \in E$ for adversary
    Mixed strategy $\{(S^{(1)}, p_1), (S^{(2)}, p_2)\}$ for leader selection
  $S \leftarrow$ **BestResponse**-$k(\nu^0, G, k)$
  $\hat{\nu}^A \leftarrow$ **NoiseInjection**(ν^0, G, y, P)
  $S' \leftarrow$ **BestResponse**-$k(\nu^0 + \hat{\nu}^A, G, k)$ **return** $\{(S, 1/2), (S', 1/2)\}$, $\hat{\nu}^A$
end procedure

The following theorem gives a bound on the approximation error of **Approx-NE**.

Theorem 4.4. *Let $\hat{U}_{MAS}$ be the utility of the MAS under the strategy defined above, and let $\hat{U}_A$ be the utility of the adversary. Let U_N^* be the best-response utility of the MAS to the adversary's strategy, and let U_A^* be the best-response utility of the adversary. Then*

$$\hat{U}_{MAS} \geq \frac{1}{2}\left[\left(1 - \frac{1}{e}\right) U_N^* - \frac{1}{e} R_{max}\right] - \frac{1}{2} R_{max} \tag{4.26}$$

$$\hat{U}_A \geq \frac{1}{2} U_A^* \tag{4.27}$$

Proof. Let S^* be the MAS's best response to the adversary's strategy $\hat{\nu}^A$, and let $U_N(S^*)$ be the resulting utility of the MAS. Let $\mathbf{S}$ be a random variable corresponding to the leader set under the mixed strategy returned by **Approx-NE**. Then under this mixed strategy,

$$\begin{aligned} \mathbf{E}(U_N(\mathbf{S}, \hat{\nu}^A)) &= U_N(S, \hat{\nu}^A) Pr(\mathbf{S} = S) + U_N(S', \hat{\nu}^A) Pr(\mathbf{S} = S') \\ &\geq -\frac{1}{2} R_{max} + \frac{1}{2}\left[\left(1 - \frac{1}{e}\right) U_N(S^*) - \frac{1}{e} R_{max}\right] \end{aligned} \tag{4.28}$$

where (4.28) follows from Theorem 4.3 and the fact that R_{max} is an upper bound on the error experienced by the MAS. This proves (4.26).

Suppose that the adversary's best response to $\{(S, \frac{1}{2}), (S', \frac{1}{2})\}$ is given by ν^{A*}. Then the adversary's payoff from the strategy $\hat{\nu}^A$ is given by

$$\begin{aligned}\mathbf{E}(U_A(\mathbf{S}, \hat{\nu}^A)) &= U_A(S, \hat{\nu}^A)Pr(\mathbf{S} = S) + U_A(S', \hat{\nu}^A)Pr(\mathbf{S} = S') \\ &= \frac{1}{2}U_A(S, \hat{\nu}^A) + \frac{1}{2}U_A(S', \hat{\nu}^A) \\ &\geq \frac{1}{2}U_A(S, \hat{\nu}^{A*}) \end{aligned} \tag{4.29}$$

The last inequality follows from the fact that $U_A \geq 0$ and $U_A(S, \hat{\nu}^A) \geq U_A(S, \hat{\nu}^*)$, since $\hat{\nu}^A$ is by definition the best response to leader set S. □

4.4 Numerical Study

The performance of leader-follower systems in the presence of adversaries, including the case where the leader set is fixed as well as the case where the leader set varies over time, was analyzed using Matlab$^{\text{TM}}$. Simulations were conducted assuming a set of 100 nodes, deployed uniformly at random over a 1000m x1000m square area, with each node's radio range set to 300m. It was assumed that the environmental noise had variance proportional to the distance between nodes. An adversary positioned at random within the deployment area, and with power budget equal to 10^6 was simulated. The path-loss parameter was set to $\alpha = 2$. Each plotted data point represents an average of 30 trials.

For both cases, the proposed leader selection algorithms were compared with three alternative heuristics: random leader selection, selection of the highest-degree nodes as leaders, and selection of the nodes with average degree as leaders.

Case 1 – Fixed leader set: The performance of leader-follower systems under noise injection attack when the leader set is fixed is illustrated in Figure 4.1(a). The algorithm **Fixed**-k was used to select leaders, with $\beta = 1$ and $\delta = 1$. The supermodular optimization approach **Fixed**-k outperforms the degree-based and random selection heuristics, achieving an error of 5, compared to 7 for random selection and 8 for maximum degree-based selection. Furthermore, we observe that the random selection approach achieves comparable performance to the average degree-based selection, and outperforms selection of high-degree nodes as leaders.

Case 2 – Time-varying leader set: Figure 4.1(b) shows the total error variance when the leader set is allowed to vary over time, based on the approximate Nash equilibrium computed using **Approx-NE**. By allowing the leader set to vary in response to attack, all of the schemes considered achieve better performance than the case of fixed leaders. The supermodular optimization approach outperforms the other three heuristics, achieving a total error variance of 5.25 when 15 leaders are chosen, compared to an error variance of roughly 6 for the random and average degree heuristics, and an error variance of 6.5 for the maximum degree heuristic. Furthermore, random selection of leaders outperforms selecting high-degree nodes to act as leaders.

4.5 Conclusions and Future Work

In this chapter, improving the resilience of leader-follower multi-node systems to noise injection attacks through leader selection was studied. Two leader selection problems were considered: first, the problem of choosing a fixed set of leaders to maximize robustness to a noise injection attack, and second, the problem of choosing a set of leaders that varies over time in response to attacks. Both cases were analyzed within a supermodular game framework. The first case was studied as a Stackelberg game between a networked system and an adversary. It was shown that the adversary's optimum strategy can be computed for a given leader set, while the best choice of leader set can be approximated up to a provable bound. In the second case, a simultaneous game framework was developed and an algorithm for efficiently approximating a mixed-strategy equilibrium was presented. Both cases were analyzed through simulation study, which demonstrated that choosing a varying set of leaders provides better robustness to noise injection than a fixed set, and that for both cases the supermodular optimization approach outperformed other leader selection algorithms.

In future work, we will study leader selection algorithms that improve on the bounds given in Sections 4.2 and 4.3. Moreover, we note that an adversary may employ additional techniques in order to disrupt system performance, including removing links from the network altogether through denial-of-service attack. We will further study leader selection methods for mitigating these different classes of attack.

4.5.1 Future Work

Improved bounds on leader selection strategy: The algorithm approx-NE only considers two iterations of interaction between the system and adversary. Learning-based equilibrium-seeking algorithms in both discrete [97] and continuous [122] settings provide a possible approach for tightening the error bounds and modeling the interaction over an extended time period. Investigating whether the supermodularity of the objective function for the system can be exploited to improve the bounds on these algorithms is an open problem.

Generalized adversary models: We note that an adversary may employ additional techniques in order to disrupt system performance, including removing links from the network altogether through denial-of-service attack. We will further study leader selection methods for mitigating these different classes of attack. In addition, distributed attacks at multiple locations by coordinating adversaries provide a further direction of study. In this case, the system must also learn the true position of the adversary, and change the topology accordingly, in order to effectively reduce the impact of the attack.

Chapter 5

SMOOTH CONVERGENCE

Linear weighted averaging algorithms are widely used in domains such as parallel computing [58], distributed control [115], and networking [121] due to their computational efficiency, distributed operation, and robustness to network topology changes. In such algorithms, the leader and follower nodes periodically broadcast their state information and each follower node computes its new state as a weighted average of the state values of its neighbors. The dynamics of the follower nodes are influenced by the leader nodes through the weighted averaging rule. The distributed nature of these algorithms, however, can lead to errors in either the asymptotic states of the follower nodes, when the followers converge to an incorrect value, or in the intermediate states of the follower nodes, prior to convergence of the algorithm. Both types of errors in the follower node states impact the system performance, for example, by causing formation errors in unmanned vehicle networks [141] and inaccurate estimates in sensor networks [17].

In [63], it was shown that linear weighted averaging algorithms achieve asymptotic consensus to the leader node state in both static networks and networks with dynamic topologies, provided the network is connected. Moreover, in [84, 129], the authors showed that the follower node states are controllable from a set of leader nodes if the eigenvalues of the graph Laplacian are distinct. While these existing results imply that the follower node states can be driven to any value asymptotically, the convergence requires an arbitrary length of time, and the follower nodes' intermediate states prior to convergence may deviate significantly from their desired steady-state values.

The intermediate behavior of nodes under linear weighted averaging algorithms was studied in [103], in which upper and lower bounds on the errors in the follower nodes' intermediate states were derived for static networks without leaders. These errors were analyzed for leader-follower systems with given leader sets based on the eigenvalues of the

graph Laplacian in [106, 112], where it was observed that the errors depend on the given leader set. Hence, an efficient, analytical approach to selecting leaders in order to minimize errors in the follower nodes' intermediate states would enable the leader nodes to steer the followers to the desired state while improving performance prior to convergence.

In this chapter, we study leader selection in order to minimize the convergence error in the intermediate states of the follower nodes, defined as the l^p-norm of the distance between the follower states and the convex hull of the leader node states. In the special case where the leader node states are equal, this error reduces to the l^p-norm of the difference between the follower node states and their desired steady-state values. We formulate two leader selection problems, namely (a) selecting a set of up to k leaders in order to minimize the convergence error, and (b) selecting the minimum-size set of leaders to achieve a given bound on the convergence error. We make the following specific contributions towards addressing these problems:

- We derive upper bounds on the convergence error at a given time that depend on the network topology and leader set but not on the initial node states. We establish the equivalence between the derived upper bounds and the time for a random walk on the network graph to travel from a given follower node to any leader node.
- Using the connection to random walks, we prove that the upper bound on the convergence error is a supermodular function of the leader set. We then introduce polynomial-time leader selection algorithms for problems (a) and (b) in static networks and use supermodularity to prove that the algorithms approximate the optimal convergence errors for each problem up to a provable bound.
- We extend our approach to dynamic networks, including networks with topologies that vary in time according to a known probability distribution, and dynamic topologies that vary with unknown and unpredictable distributions. For dynamic topologies with unknown distributions, we prove a lower bound on the best possible convergence error that can be achieved by any leader selection algorithm, as well as an upper bound on the error achieved by our proposed algorithm.
- Our results are illustrated through a numerical study, which compares our supermod-

ular optimization approach with random and degree-based leader selection in static and dynamic networks.

The chapter is organized as follows. The system model, as well as background on submodular functions and experts algorithms, are described in Section 5.1. The connection between convergence error and random walks, along with our proposed leader selection algorithms for networks with static topology, are presented in Section 5.2. Leader selection algorithms for dynamic networks are described and analyzed in Section 5.3. Section 5.4 presents our numerical study. Section 5.5 concludes the chapter.

5.1 System Model and Preliminaries

In this section, we define the system model, which is a generalization of the previous sections to the case of arbitrary link weights. We then define the convergence error metric used in this chapter.

5.1.1 System Model

We consider a network of n nodes, indexed in the set $V = \{1, \ldots, n\}$. An edge (i, j) exists from node i to node j if node j is within communication range of node i. Let E denote the set of edges. The set of neighbors of node i is defined by $N(i) \triangleq \{j : (i, j) \in E\}$. The number of edges outgoing from i is denoted the outdegree of i, while the number of edges incoming to i is the indegree of i. We assume that the graph G is strongly connected, so that for each pair of nodes i and j, there exists a directed path from i to j.

Each node i is assumed to have a time-varying internal state, $x_i(t) \in \mathbb{R}$. Each node j in the leader set, denoted S, maintains a constant state value equal to $x_j^* \in \mathbb{R}$. For every leader node $j \in S$, the state value is initialized to $x_j(0) = x_j^*$. The leader states may be distinct, so that $x_j^* \neq x_{j'}^*$ for $j, j' \in S$.

The follower nodes compute their states according to the distributed rule

$$\dot{x}_i(t) = \sum_{j \in N(i)} W_{ij}(x_j(t) - x_i(t)), \tag{5.1}$$

where W_{ij} is a nonnegative constant for a given node pair (i, j). The Laplacian matrix L,

which will be used to derive the convergence properties of the dynamics (5.1), is defined by

$$L_{ij} = \begin{cases} -W_{ij}, & j \in N(i),\ i \notin S \\ \sum_{j \in N(i)} W_{ij}, & i = j,\ i \notin S \\ 0, & i \in S \\ 0, & \text{else} \end{cases}$$

By [30], for any $t > 0$, e^{-Lt} is a stochastic matrix with nonzero entries. Letting $\mathbf{x}(t) \in \mathbb{R}^n$ denote the vector of node states at time t, equation (5.1) is written in vector form as $\dot{\mathbf{x}}(t) = -L\mathbf{x}(t)$. The node states $\mathbf{x}(t)$ depend on the leader set, S. When defining the convergence error, we represent this dependence explicitly by writing $\mathbf{x}(t, S)$ to denote the node states. For simplicity of notation, we write $\mathbf{x}(t)$ when describing the node dynamics. The following lemma describes the asymptotic behavior of the dynamics in (5.1).

Lemma 5.1 ([66])**.** *The vector of follower node states converges to a fixed point of* $\dot{\mathbf{x}}(t) = -L\mathbf{x}(t)$*, denoted* $\mathbf{x}^*$*. Furthermore, for each* $j \in V$*,* $x_j^* \in co\,(\{x_i^* : i \in S\})$*, where* $co(\cdot)$ *denotes the convex hull.*

Let $A = \{x_i^* : i \in S\}$ and $\overline{A} = \text{co}(A)$. If the state of each follower node converges to $\overline{A}$, then the system is said to achieve containment [66].

5.1.2 Definition of Convergence Error

Although Lemma 5.1 implies that the nodes will asymptotically converge to the convex hull of the leader node states, the node states will deviate from their desired values at each finite time. A family of metrics for quantifying these *convergence errors* in the intermediate states is defined as follows. We let $||\cdot||_p$ (where $1 \leq p < \infty$) denote the l^p-norm of a vector.

Definition 5.1. *Suppose* $t > 0$ *and* $1 \leq p < \infty$*. The convergence error* $f_t(S)$ *for leader set* S *is defined by*

$$f_t(S) \triangleq \left(\sum_{i \in V} \left(d(x_i(t, S), \overline{A})^p \right) \right)^{1/p} = \left(\sum_{i \in V} \min_{y \in \overline{A}} \{ |x_i(t, S) - y|^p \} \right)^{1/p}. \tag{5.2}$$

When the leaders all have the same state, $A = \overline{A} = \{x^*\}$ *and* $f_t(S) = ||\mathbf{x}(t, S) - x^*\mathbf{1}||_p$*.*

The metric $f_t(S)$ measures the deviation of $\mathbf{x}(t,S)$ from containment at time t as a function of the leader set S. In general, $f_t(S)$ is not a monotonic function of t. When the graph G is undirected and the matrix W satisfies $W_{ij} = W_{ji}$ for all i, j, however, the system response, and hence the convergence error is monotonic, as described in the following lemma.

Lemma 5.2. *If the weight matrix W is symmetric and $t_0 > 0$, then $f_t(S) \leq f_{t_0}(S)$ for all $S \subseteq V$ and $t \geq t_0$.*

Proof. We have that $\mathbf{x}(t) = e^{-L(t-t_0)}\mathbf{x}(t_0)$. Since $e^{-L(t-t_0)}$ is a stochastic matrix, we have that, for all $i \in V \setminus S$, $x_i(t) = \sum_{j\in V} \alpha_{ij} x_j(t_0)$ for some $\alpha_{ij} \geq 0$ satisfying $\sum_{j\in V} \alpha_{ij} = 1$ (also, by definition of S, we have that $\alpha_{ij} \equiv 0$ for all $i \in S$, $j \neq i$). Let $y_j^0 \in \arg\min\{|x_j(t_0) - y|^p : y \in \overline{A}\}$, and define $\overline{y}_i = \sum_{j\in V} \alpha_{ij} y_j^0$, noting that $\overline{y}_i \in \overline{A}$. We have

$$\begin{aligned} |x_i(t) - \overline{y}_i|^p &= \left|\sum_{j\in V} \alpha_{ij} x_j(t_0) - \sum_{j\in V} \alpha_{ij} y_j^0\right|^p = \left|\sum_{j\in V} \alpha_{ij}(x_j(t_0) - y_j^0)\right|^p \\ &\leq \sum_{j\in V} \alpha_{ij} |x_j(t_0) - y_j^0|^p \leq \sum_{j\in V\setminus S} \alpha_{ij} |x_j(t_0) - y_j^0|^p, \end{aligned}$$

where the first inequality follows from convexity of $|\cdot|^p$. We then have

$$\begin{aligned} f_t(S) &= \left(\sum_{i\in V} \min_{y\in\overline{A}} \{|x_i(t) - y|^p\}\right)^{1/p} \leq \left(\sum_{i\in V} |x_i(t) - \overline{y}_i|^p\right)^{1/p} \\ &\leq \left(\sum_{i\in V\setminus S} \sum_{j\in V\setminus S} \alpha_{ij} |x_j(t_0) - y_j^0|^p\right)^{1/p} = \left(\sum_{i\in V\setminus S} \sum_{j\in V\setminus S} \alpha_{ji} |x_j(t_0) - y_j^0|^p\right)^{1/p} \\ &= \left(\sum_{j\in V\setminus S} \left(|x_j(t_0) - y_j^0|^p \sum_{i\in V\setminus S} \alpha_{ji}\right)\right)^{1/p} \\ &\leq \left(\sum_{j\in V\setminus S} \left(|x_j(t_0) - y_j^0|^p\right)\right)^{1/p} = f_{t_0}(S), \end{aligned}$$

as desired. □

The value of $f_t(S)$ depends on the value of t that is chosen. One approach to choosing t is to select a set S at random and then select the smallest t such that $f_t(S) \leq \beta$, where

$\beta > 0$ is a desired bound on the convergence error. This choice of t guarantees that the convergence error arising from our proposed approach will be no more than $f_t(S)$, and hence no more than the constraint β. Alternatively, the *total convergence error*, defined as

$$w(S) \triangleq \int_0^\infty f_t(S)\ dt,$$

does not depend on any fixed value of t. In Section 5.2, we show that $w(S)$ can be substituted for $f_t(S)$ in our algorithms while preserving the optimality guarantees that we derive.

5.2 Leader Selection in Static Networks

In this section, we discuss leader selection in order to minimize the convergence error in the intermediate states of networks with static topologies. We first derive an upper bound on the convergence error that is independent of the initial leader and follower node states, $\mathbf{x}(0)$. We then introduce a connection between the derived upper bound and a random walk on the network graph, which provides the critical step towards formulating the leader selection problem as supermodular optimization. Using the upper bound on the convergence error as a cost function, we formulate two leader selection problems, namely: (i) the problem of selecting a fixed set of up to k leaders to minimize the convergence error, and (ii) the problem of selecting the minimum-size leader set in order to satisfy a given bound on the convergence error. In order to efficiently approximate the solutions to (i) and (ii), we prove that the upper bound on the convergence error is a supermodular function of S. Supermodularity leads to polynomial-time algorithms for approximating (i) and (ii) up to a provable constant factor.

5.2.1 Random Walk-based Upper Bound on Convergence Error

As the first step in analyzing the convergence error and establishing a connection between the convergence error and a random walk on the graph G, an upper bound on the convergence error $f_t(S)$ is given by the following theorem.

Theorem 5.1. *Let q satisfy $\frac{1}{p} + \frac{1}{q} = 1$, and suppose $||\mathbf{x}(0)||_q \leq K$, where K is a positive constant. Further, define $P_t = e^{-Lt}$ and let e_i denote the canonical vector with a 1 in the*

i-th entry and 0*'s elsewhere. Then for any leader set* S*, the convergence error satisfies*

$$f_t(S) \leq K \left(\sum_{i \in V \setminus S} \left[\sum_{j \in V \setminus S} (e_i^T P_t)_j^p + \left(1 - \sum_{j \in S} (e_i^T P_t)_j \right)^p \right] \right)^{1/p}. \tag{5.3}$$

Proof. Define $\Pi(S) \triangleq \{\pi \in \mathbf{R}^n_{\geq 0} : \mathbf{1}\pi = 1, \pi_i = 0 \ \forall i \notin S\}$, so that each π defines a convex combination. The convergence error is defined by

$$f_t(S) = \left(\sum_{i \in V} \left[\min_{y \in \overline{A}} \{|x_i(t) - y|^p\} \right] \right)^{1/p} = \left(\sum_{i \in V} \left[\min_{\pi \in \Pi(S)} \left\{ |e_i^T P_t \mathbf{x}(0) - \pi^T \mathbf{x}(0)|^p \right\} \right] \right)^{1/p}.$$

Bounding the above equation using Hölder's inequality yields

$$f_t(S) \leq \left(\sum_{i \in V} \left[\min_{\pi \in \Pi(S)} \{ ||\mathbf{x}(0)||_q^p ||e_i^T P_t - \pi^T||_p^p \} \right] \right)^{1/p} \leq K \left(\sum_{i \in V} \left[\min_{\pi \in \Pi(S)} \left\{ ||e_i^T P_t - \pi^T||_p^p \right\} \right] \right)^{1/p}.$$

Now, suppose that a distribution $\pi_i^* \in \Pi(S)$ is chosen such that $\pi_i^*(j) \geq (e_i^T P_t)_j$ for all $j \in S$. It is always possible to construct such a distribution since $\mathbf{1}^T \pi = 1$ for all $\pi \in \Pi(S)$ and $\sum_{j \in S} (e_i^T P_t)_j \leq 1$. Define $\hat{\pi}_i(j) = \pi_i^*(j) - (e_i^T P_t)_j$. Then we have

$$\begin{aligned} f_t(S) &\leq K \left(\sum_{i \in V \setminus S} \left[\sum_{j \in V \setminus S} |(e_i^T P_t)_j|^p + \sum_{j \in S} \hat{\pi}_i(j)^p \right] \right)^{1/p} \\ &\leq K \left(\sum_{i \in V \setminus S} \left[\sum_{j \in V \setminus S} (e_i^T P_t)_j^p + \left(\sum_{j \in S} \hat{\pi}_i(j) \right)^p \right] \right)^{1/p}, \end{aligned}$$

where the bound on the first term follows from the fact that $\hat{\pi}_i(j) \geq 0$ for all $i \in V \setminus S$ and $j \in S$. Using the facts that $\hat{\pi}_i(j) = \pi_i^*(j) - (e_i^T P_t)_j$ and $\sum_{j \in S} \pi_i^*(j) = 1$ yields the desired result. □

We observe that, while the bound (5.3) is loose in general, the left and right hand sides of (5.3) become arbitrarily close as t grows large.

We define the notation $\hat{f}_t(S)$ for the upper bound on the convergence error $f_t(S)$ as given in Theorem 5.1, as

$$\hat{f}_t(S) \triangleq \sum_{i \in V \setminus S} \left[\sum_{j \in V \setminus S} (e_i^T P_t)_j^p + \left(1 - \sum_{j \in S} (e_i^T P_t)_j \right)^p \right]. \tag{5.4}$$

We now establish a mathematical relationship between the terms of the inner summation of (5.4) and a random walk on G that will be used to prove supermodularity of $\hat{f}_t(S)$ later. Intuitively, the inputs from the leader nodes can be viewed as diffusing from the leader nodes to the follower nodes via a random walk. The connection is described formally as follows.

We define a random walk on G as follows. Choose $\delta > 0$ such that $t = \tau\delta$ for some positive integer τ. Define $X(\tau)$ to be a random walk with transition matrix $P_\delta \triangleq e^{-L\delta}$ (as in Theorem 5.1). The following is a standard result, which we prove for completeness.

Theorem 5.2. *Choose $\delta > 0$ such that $t = \tau\delta$ for some integer τ. Let $X(\tau)$ be a random walk on G with transition matrix P_δ. Then*

$$(e_i^T P_\delta^\tau)_j = Pr(X(\tau) = j | X(0) = i), \tag{5.5}$$

$$1 - \sum_{j \in S} (e_i^T P_\delta^\tau)_j = Pr(X(\tau) \notin S | X(0) = i), \tag{5.6}$$

where P_δ^τ is equal to the matrix P_δ raised to the τ-th power.

Proof. The vector e_i defines a probability distribution on the set of nodes V, corresponding to the case where $X(0) = i$. Hence, $e_i^T P_\delta^\tau$ is the probability distribution of $X(\tau)$, conditioned on $X(0) = i$, so that $(e_i^T P_\delta^\tau)_j = Pr(X(\tau) = j | X(0) = i)$. Eq. (5.5) follows immediately, while

$$1 - \sum_{j \in S} (e_i^T P_\delta)_j = 1 - \sum_{j \in S} Pr(X(\tau) = j | X(0) = i) = 1 - Pr(X(\tau) \in S) = Pr(X(\tau) \notin S)$$

yields (5.6). □

5.2.2 Problem Formulation – Selecting up to k Leaders

In this section, we first formulate the problem of selecting a set of up to k leaders, denoted S, in order to minimize the convergence error bound $\hat{f}_t(S)$. We then prove that $\hat{f}_t(S)$ is supermodular as a function of S, leading to an efficient algorithm for approximating the optimal leader set.

Selecting up to k leaders in order to minimize the convergence error bound $\hat{f}_t(S)$ is given by the optimization problem

$$\begin{aligned} \text{minimize} \quad & \hat{f}_t(S) \\ S & \\ \text{s.t.} \quad & |S| \leq k \end{aligned} \tag{5.7}$$

Since an exhaustive search over the feasible values of S is computationally prohibitive, we investigate the structure of the convergence error bound $\hat{f}_t(S)$ in order to develop an efficient algorithm for approximating the solution to (5.7). By showing that $\hat{f}_t(S)$ satisfies supermodularity as a function of S, we derive polynomial-time algorithms with provable $O(1)$ optimality gap.

As a first step to proving that $\hat{f}_t(S)$ is supermodular, we prove the following intermediate result that the probability that a random walk on G, originating at follower node $i \in V \setminus S$, does not reach any node in the leader set is a supermodular function of S.

Lemma 5.3. *Define $g_{ij}^{\tau}(S) \triangleq Pr(X(\tau) = j | X(0) = i)$ and $h_i^{\tau}(S) \triangleq Pr(X(\tau) \notin S | X(0) = i)$. Then for all $i \in V \setminus S$, j, and τ, $g_{ij}^{\tau}(S)$ and $h_i^{\tau}(S)$ are both supermodular functions of S.*

Proof. Let $S \subseteq T$ and let $u \in V \setminus T$ (and hence $u \in V \setminus S$). Let $A_{ij}^{\tau}(S)$ denote the event that $X(\tau) = j$ and $X(r) \notin S$ for all $1 \leq r \leq \tau$, and define $\chi(\cdot)$ to be the indicator function of an event. Since each node in S is an absorbing state of the walk, we have

$$g_{ij}^{\tau}(S) = Pr(A_{ij}^{\tau}(S) | X_0 = i) = \mathbf{E}(\chi(A_{ij}^{\tau}(S)) | X_0 = i).$$

Furthermore, let $B_{ij}^{\tau}(S, u)$ denote the event where $X(0) = i$, $X(\tau) = j$, $X(r) \notin S$ for $0 \leq r \leq \tau$, and $X(m) = u$ for some $0 \leq m \leq \tau$. We then have $A_{ij}^{\tau}(S) = A_{ij}^{\tau}(S \cup \{u\}) \cup B_{ij}^{\tau}(S, u)$, $A_{ij}^{\tau}(S \cup \{u\}) \cap B_{ij}^{\tau}(S, u) = \emptyset$, and

$$\chi(A_{ij}^{\tau}(S)) = \chi(A_{ij}^{\tau}(S \cup \{u\})) + \chi(B_{ij}^{\tau}(S, u)).$$

Since $S \subseteq T$, $X(r) \notin T$ for all $0 \leq r \leq \tau$ implies $X(r) \notin S$ for all $0 \leq r \leq \tau$, i.e., $B_{ij}^{\tau}(T, u) \subseteq B_{ij}^{\tau}(S, u)$. We have

$$\begin{aligned} \chi(A_{ij}^{\tau}(S)) - \chi(A_{ij}^{\tau}(S \cup \{u\}) &= \chi(B_{ij}^{\tau}(S, u)) \geq \chi(B_{ij}^{\tau}(T, u) \\ &= \chi(A_{ij}^{\tau}(T)) - \chi(A_{ij}^{\tau}(T \cup \{u\})). \end{aligned}$$

Taking expectations of both sides yields

$$\begin{aligned} g_{ij}^{\tau}(S) - g_{ij}^{\tau}(S \cup \{u\}) &= \mathbf{E}(\chi(A_{ij}^{\tau}(S))) - \mathbf{E}(\chi(A_{ij}^{\tau}(S \cup \{u\}))) \\ &\geq \mathbf{E}(\chi(A_{ij}^{\tau}(T))) - \mathbf{E}(\chi(A_{ij}^{\tau}(T \cup \{u\}))) \\ &= g_{ij}^{\tau}(T) - g_{ij}^{\tau}(T \cup \{u\}), \end{aligned}$$

implying that $g_{ij}^{\tau}(S) = \mathbf{E}(\chi(A_{ij}^{\tau}(S)))$ is supermodular as a function of S. A similar argument shows that $h_i^{\tau}(S)$ is supermodular as a function of S. □

We can now prove that $\hat{f}_t(S)$ is supermodular as a function of S by using Lemma 5.3 and the composition result of Lemma 2.3.

Theorem 5.3. *The convergence error bound $\hat{f}_t(S)$ is supermodular as a function of S.*

Proof. By Theorem 5.2, the convergence error bound $\hat{f}_t(S)$ can be written as

$$\begin{aligned} \hat{f}_t(S) &= \sum_{i \in V \setminus S} \left[\sum_{j \in V \setminus S} (e_i^T P_t)_j^p + \left(1 - \sum_{j \in S} (e_i^T P_t)_j \right)^p \right] \\ &= \sum_{i \in V} \sum_{j \in V \setminus S} g_{ij}^{\tau}(S)^p + h_i^{\tau}(S)^p. \end{aligned}$$

Since x^p is a nondecreasing, convex, and differentiable function of x, $g_{ij}^{\tau}(S)^p$ and $h_i^{\tau}(S)^p$ are supermodular functions of S for all i, j, τ, and $p \in [1, \infty)$ by Lemmas 2.3 and 5.3. Hence $\hat{f}_t(S)$ is a sum of supermodular functions, and is therefore supermodular. □

As a corollary to Theorem 5.3, the total convergence error $w(S)$ defined in Section 5.1.2 is the integral of a family of supermodular functions, and is therefore supermodular. As a result, the algorithms *Select-k-leaders* and *Select-minimal-leaders* can be modified by replacing $\hat{f}_t(S)$ with $w(S)$.

An algorithm for approximating the optimal solution to (5.7) is as follows. Initialize the set $S = \emptyset$. At the j-th iteration, choose the node $v_j \in V \setminus S$ such that $\hat{f}_t(S) - \hat{f}_t(S \cup \{v_j\})$ is maximized. The algorithm terminates after k iterations. A pseudocode description is given as algorithm *Select-k-leaders*.

Algorithm 9 Algorithm for choosing up to k fixed leaders to minimize convergence error.

procedure SELECT-k-LEADERS($G = (V, E)$, k, W)
  Input: $G = (V, E)$
    Maximum number of leader nodes k
    Weight matrix W
  Output: Set of leader nodes S^*
  Initialization: $S \leftarrow \emptyset$, $i \leftarrow 0$
  while $i < k$ **do**
    $v_i \leftarrow \arg\max \{\hat{f}_t(S) - \hat{f}_t(S \cup \{v\}) : v \in V \setminus S\}$
    $S \leftarrow S \cup \{v_j\}$, $i \leftarrow i + 1$
  end while
  $S^* \leftarrow S$, **return** S^*
end procedure

The following theorem gives a worst-case bound on the optimality gap between the best possible solution to (5.7) and the convergence error of the set S returned by *Select-k-leaders*. The bound guarantees that the convergence error of *Select-k-leaders* is within a constant factor of the lowest possible convergence error.

Theorem 5.4. *Let S^* denote the set of leader nodes that is the solution of (5.7). Then algorithm* Select-k-leaders *returns a set S' satisfying*

$$\hat{f}_t(S') \leq \left(1 - \frac{1}{e}\right) \hat{f}_t(S^*) + \frac{1}{e} f_{max}, \tag{5.8}$$

where $f_{max} = \max_{v \in V} \hat{f}_t(\{v\})$ and e is the base of natural logarithms.

Proof. By Theorem 2.2, for a nonnegative nondecreasing submodular function $f(S)$, a maximization algorithm that chooses

$$v_t = \arg\max \{f(S \cup \{v\}) - f(S) : v \in V \setminus S\}$$

returns a set S' satisfying $f(S') \geq \left(1 - \frac{1}{e}\right) f(S)$ for all $S \subseteq V$. Algorithm *Select-k-leaders* is equivalent to greedy maximization of the nonnegative, nondecreasing submodular function

$f_{max} - \hat{f}_t(S)$. Hence $f_{max} - \hat{f}_t(S') \geq \left(1 - \frac{1}{e}\right)(f_{max} - \hat{f}_t(S^*))$. Rearranging terms gives (5.8). □

5.2.3 Problem Formulation – Selecting the Minimum-size Leader Set that Achieves an Error Bound

In this section, we consider the problem of selecting the minimum-size leader set S in order to achieve a given constraint, $\alpha \geq 0$, on the convergence error bound $\hat{f}_t(S)$. We first give the problem formulation, followed by an efficient algorithm for approximating the optimal leader set. The supermodularity of $\hat{f}_t(S)$ leads to a provable $O(1)$ optimality gap between the size of the selected leader set and the size of the smallest possible leader set that achieves the convergence error α.

Selecting the minimum-size leader set that achieves a bound on the convergence error is given by the optimization problem

$$\min \{|S| : \hat{f}_t(S) \leq \alpha\} \tag{5.9}$$

Since $\hat{f}_t(S)$ is a supermodular function of S by Theorem 5.3, equation (5.9) is a supermodular optimization problem, which can be efficiently approximated by a greedy algorithm analogous to *Select-k-leaders*. The algorithm begins by initializing $S = \emptyset$. At the j-th iteration, the algorithm selects the node v_j that maximizes $(\hat{f}_t(S) - \hat{f}_t(S \cup \{v_j\}))$ and sets $S = S \cup \{v_j\}$. A pseudocode description of the algorithm is given as *Select-minimal-leaders* below.

Bounds on the optimality gap of the solutions returned by *Select-minimal-leaders* are given by the following theorem.

Theorem 5.5. *Let S^* be the optimum set of leaders for problem (5.9), and let S' be the set of leaders returned by* Select-minimal-leaders. *Then* $\frac{|S'|}{|S^*|} \leq 1 + \ln\left(\frac{f_{max}}{\alpha}\right)$.

Proof. Theorem 9.4 of [137, Ch III.3.9] implies that, for any nonnegative, nondecreasing, submodular function $f(S)$, the set S' returned by the greedy maximization algorithm and

Algorithm 10 Algorithm for selecting the minimum-size leader set to achieve a given convergence error bound.

procedure SELECT-MINIMAL-LEADERS($G = (V, E)$, α, W)
  Input: $G = (V, E)$
    Error bound α
    Weight matrix W
  Output: Set of leader nodes S^*
  Initialization: $S \leftarrow \emptyset$
  while $\hat{f}_t(S) > \alpha$ **do**
    $v^* \leftarrow \arg\max \{\hat{f}_t(S) - \hat{f}_t(S \cup \{v\}) : v \in V \setminus S\}$
    $S \leftarrow S \cup \{v^*\}$
  end while
  $S^* \leftarrow S$, **return** S^*
end procedure

the optimal set S^* satisfy

$$\frac{|S'|}{|S^*|} \leq 1 + \ln\left\{\frac{f(V) - f(\emptyset)}{f(V) - f(S_{T-1})}\right\},$$

where S_{T-1} denotes the leader set at the second-to-last iteration of *Select-minimal-leaders*. Applying this result to the submodular function $f(S) = f_{max} - \hat{f}_t(S)$, and using the fact that $\hat{f}_t(S') \leq \alpha$ yields

$$\frac{|S'|}{|S^*|} \leq 1 + \ln\left\{\frac{-f_{max}}{-\hat{f}_t(S_{k-1})}\right\} = 1 + \ln\left\{\frac{f_{max}}{\hat{f}_t(S_{k-1})}\right\} \leq 1 + \ln\left\{\frac{f_{max}}{\alpha}\right\},$$

as desired. □

We have that, for any $l \in V$,

$$\hat{f}_t(\{l\}) = \sum_{i \in V \setminus \{l\}} ||e_i^T P_t - e_0^T||_p^p \leq \sum_{i \in V \setminus \{l\}} ||e_i^T P_t - e_0^T||_1^p \leq n - 1,$$

and hence $f_{max} = \max\{\hat{f}_t(\{l\}) : l \in V\} \leq n - 1$. Thus for fixed α, the bound of Theorem 5.5 is of $O(\ln n)$ in the worse case.

5.3 Leader Selection in Time-Varying Networks

In this section, we consider leader selection for two cases of networks with time-varying topologies. In the first case, we assume that the network topology evolves according to a stochastic process with known probability distribution. Examples of this type of topology include networks that experience random link failures with known spatial correlation [65] and networks that change topology according to a known switching signal. In the second case, the network topology evolves according to a stochastic process with distribution that is unknown to the system designer at the time when the leaders are selected, for example, if the nodes move to avoid unforeseen obstacles [100]. For each case, we first define the graph model, and then present leader selection algorithms along with bounds on the optimality gap between the convergence error guaranteed by our algorithm and the optimal leader set. In the case of topology dynamics with unknown distribution, we also give the best-case performance of any possible algorithm without probability distribution information and prove that our approach achieves a constant factor of this performance.

5.3.1 Dynamic Topology Model

We assume that the set of network nodes, denoted V, is constant. The set of edges is represented by a random process $E(t)$, resulting in a time-varying network topology $G(t) = (V, E(t))$. We assume that there exists a sequence of random variables $t_1, t_2, \ldots$, such that $E(t) = E(t')$ for all $t, t' \in [t_m, t_{m+1}]$, and that $\inf_m |t_{m+1} - t_m| = \gamma > 0$. We assume that $G(t)$ is strongly connected for all $t \geq 0$.

The weight matrix at time t is denoted $W(t)$, while the corresponding Laplacian matrix is given by $L(t)$. The dynamics of the follower nodes are governed by

$$\dot{\mathbf{x}}(t) = -L(t)\mathbf{x}(t), \tag{5.10}$$

while each leader node $i \in S$ maintains a constant state x_i^* and $\overline{A} = \text{co}\{x_i^* : i \in S\}$ is the convex hull of the leader node states. Let r denote the number of network topology changes up to time t, i.e., $r = \max\{m : t_m \leq t\}$, and let $\delta_m = t_m - t_{m-1}$. The node states at time t

are given by

$$\mathbf{x}(t) = \left(\prod_{m=1}^{r} e^{-L(t_{m-1})\delta_m} \right) \mathbf{x}(0).$$

The following lemma describes the convergence properties of (5.10).

Lemma 5.4 ([98]). *Under the assumption that $G(t)$ is strongly connected $\forall t$, $\lim_{t\to\infty} d(x_i, \overline{A}) = 0$ for every node $i \in V$.*

As a corollary, when the leaders have the same initial state x^*, the follower nodes will asymptotically converge to x^*. While Lemma 5.4 implies that any leader set guarantees asymptotic convergence provided that the graph $G(t)$ is strongly connected for all $t \geq 0$, asymptotic convergence alone does not imply that the errors in the intermediate states of the follower nodes are minimized. Selecting the leader set in order to minimize the intermediate state convergence errors is the focus of the next section.

One example of a type of topology dynamics that can be analyzed within this framework is the *random waypoint model* [16]. Under this model, each node chooses a destination uniformly at random from within a certain deployment area and moves toward that destination with a randomly chosen velocity, choosing a new destination upon arrival. If the mobility model has reached its stationary distribution, then the expected convergence error $\hat{f}_t(S)$ can be bounded by

$$\mathbf{E}\left\{ \sum_{i\in V\setminus S} \sum_{j\notin S} \left(e_i^T \prod_{m=1}^{r} e^{-L(t_{m-1})\delta_m} \right)_j^p + \left(1 - \sum_{j\in S} \left(e_i^T \prod_{m=1}^{r} e^{-L(t_{m-1})\delta_m} \right)_j \right)^p \right\}. \quad (5.11)$$

The value of (5.11) can be estimated using Monte Carlo methods, yielding an approximation of the convergence error when the topology varies according to a known random waypoint mobility model.

5.3.2 Leader Selection Under Known Topology Distribution

We first treat the case where, at each time t, the distribution of the random variable $G(t)$ is known to the system designer. In this case, if the leader set varies over time, then selecting a leader set for each topology in order to minimize the convergence error is equivalent to solving a series of problems of the form (5.7).

The case of minimizing the convergence error when the leader set is fixed is discussed as follows. As in Section 5.1.1, we measure the convergence error by

$$f_t(S) \triangleq \left(\sum_{i \in V} \left(d(x_i(t,S), \overline{A})^p \right) \right)^{1/p}.$$

A straightforward extension of Theorem 5.1 implies that $f_t(S)$ is bounded above by $\hat{f}_t(S)$, which is defined for time-varying networks by

$$\hat{f}_t(S) \triangleq \mathbf{E} \left\{ \sum_{i \in V \setminus S} \left[\sum_{j \in V \setminus S} \left(e_i^T \prod_{m=1}^{r} e^{-L(t_{m-1})\delta_m} \right)_j^p + \left(1 - \sum_{j \in S} \left(e_i^T \prod_{m=1}^{r} e^{-L(t_{m-1})\delta_m} \right)_j \right)^p \right] \right\}, \quad (5.12)$$

where the expectation is over the possible realizations of $\{G(\tau) : 0 \leq \tau \leq t\}$. The following theorem leads to efficient algorithms for minimizing the convergence error for topology dynamics with known distribution.

Theorem 5.6. *The convergence error bound $\hat{f}_t(S)$ is supermodular as a function of S.*

Proof. Suppose that the topology changes occur at given times $t_1 < \ldots < t_r$, with $\delta_m = t_m - t_{m-1}$. Choose δ sufficiently small, and let $\tau_0 = \lceil \frac{t_1 - t_0}{\delta} \rceil$, ..., $\tau_r = \lceil \frac{t - t_r}{\delta} \rceil$. Note that the difference between $\tau_m \delta$ and $(t_{m+1} - t_m)$ can be made arbitrarily small by decreasing δ. Since P_δ^m (where $P_\delta = e^{-L\delta}$) is a stochastic matrix for each m, the product $\prod_{m=1}^{r} P_\delta^m$ is also stochastic. Let $X(l)$ represent a random walk on V such that the (τ_{m-1})-th to (τ_m)-th steps are taken using transition matrix P_δ^m.

We first show that the term of the inner summation of (5.12) with indices i and j is equivalent to the probability that $X(t_r)$ is equal to j when the random walk starts at i. Formally, in the limit as $\delta \to 0$, this relation is given by

$$\left(e_i^T \prod_{m=1}^{r} e^{-L(t_{m-1})\delta_m} \right)_j = Pr(X(\tau_0 + \tau_1 + \cdots + \tau_r) = j | X(0) = i). \quad (5.13)$$

The proof of (5.13) is given by induction on r. When $r = 1$, the proof follows from Theorem 5.2. Now, assume that (5.13) holds for $r < r_0 \in \mathbb{Z}_+$, and define the probability distribution

π by $\pi^T = e_i^T \prod_{m=1}^{r_0-1} e^{-L(t_{m-1})\delta_m}$. This definition yields

$$\begin{aligned}\left(e_i^T \prod_{m=1}^{r_0} e^{-L(t_{m-1})\delta_m}\right)_j &= \sum_{j'=0}^{n} \pi_{j'} (e^{-L(t_{r_0-1})(t_{r_0}-t_{r_0-1})})_{j'j} \\ &= \sum_{j'=0}^{n} \left[Pr(X(\tau_{r_0-1}+\cdots+\tau_0)=j'|X(0)=i) Pr(X(\tau_{r_0})=j|X(0)=j')\right] \quad (5.14) \\ &= \sum_{j'=0}^{n} \Big[Pr(X(\tau_{r_0-1}+\cdots+\tau_0)=j'|X(0)=i) \\ &\quad \times Pr(X(\tau_0+\cdots+\tau_{r_0})=j|X(\tau_0+\cdots+\tau_{r_0-1})=j')\Big], \quad (5.15)\end{aligned}$$

where (5.14) follows by using the inductive assumption and Theorem 5.2. Equation (5.15) follows from the stationarity of the random walk. Equation (5.15) and the law of total probability then imply (5.13). Hence $||e_i^T \prod_{m=1}^{r} e^{-L(t_{m-1})\delta_m}||_p^p$ is supermodular by (5.13) and Lemma 5.3. The function $\hat{f}_t(S)$ is therefore a nonnegative weighted sum of supermodular functions, and hence is supermodular. □

Theorem 5.6 implies that the algorithms *Select-k-leaders* and *Select-minimal-leaders* from Section 5.2 can be modified to select leaders in the known topology distribution case by using the generalized version of $\hat{f}_t(S)$ defined above. The optimality gap provided by Theorems 5.4 and 5.5 are maintained in the known topology dynamics case.

5.3.3 Leader Selection Under Unknown Topology Distribution

We next consider the case where the distribution of the random process $G(t)$ is not known. The distribution of the topology dynamics may be unknown, for example, due to node mobility, which creates and removes links at arbitrary and random points in time. For this case, we first define an upper bound on the convergence error for topologies with unknown distribution, and then formulate the problem of selecting up to k leaders in order to minimize this upper bound on the convergence error. We then define the notion of regret, which will be used to evaluate our proposed leader selection approach, and give a lower bound on the regret that can be achieved by any algorithm. We give background on expert algorithms that are called as subroutines of our approach, and finally we present our leader selection approach and derive a bound on the regret achieved by our approach.

Problem Formulation

If a fixed set of leaders is maintained for the lifetime of the network, then high convergence errors will result since the future topologies are not known in advance. Hence, we assume that a new leader set S_m is selected for each time interval $[t_m, t_{m+1}]$. We note that the node dynamics defined in Section 5.1.1 will converge to $\mathbf{x}^*$ if $\mathbf{x}^* = x^*\mathbf{1}$, i.e., if all leader nodes maintain the same state x^*. For this case, letting $\delta_m = t_m - t_{m-1}$, and recalling that $r = \max\{m : t_m \leq t\}$ represents the number of topology changes before time t, we have the following upper bound on the convergence error for dynamic networks.

Proposition 5.1. *For any topology dynamics $G(t)$,*

$$\sum_{i=1}^{n}\left[\sum_{j\in V\setminus S}\left(e_i^T\prod_{m=1}^{r}e^{-L(t_{m-1})\delta_m}\right)_j^p+\left(1-\sum_{j\in S}\left(e_i^T\prod_{m=1}^{r}e^{-L(t_{m-1})\delta_m}\right)_j\right)^p\right]$$
$$\leq\sum_{m=1}^{r}\sum_{i\in V\setminus S_m}\left[\sum_{j\in V\setminus S}\left(e_i^Te^{-L(t_{m-1})\delta_m}\right)_j^p+\left(1-\sum_{j\in S}\left(e_i^Te^{-L(t_{m-1})\delta_m}\right)_j\right)^p\right]. \quad (5.16)$$

Proof. Since e_0^T is an absorbing state of the Markov chain with transition matrix $e^{-L_m(S_m)\delta_m}$ for all $m \in \{1,\dots,r\}$, $e_0^T e^{-L_m(S_m)\delta_m} = e_0^T$ for all $m \in \{1,\dots,r\}$. For any $i \in V$, this implies

$$||e_i^T\prod_{m=1}^{r}e^{-L_m(S_m)\delta_m}-e_0^T||_p^p \quad (5.17)$$
$$=||(e_ie^{-L_r(S_r)\delta_r}-e_0^T)e^{-L_{r-1}(S_{r-1})\delta_{r-1}}\cdots e^{-L_1(S_1)\delta_1}||_p^p \quad (5.18)$$
$$\leq||e_i^Te^{-L_r(S_r)\delta_r}-e_0^T||_p^p||e^{-L_{r-1}(S_{r-1})\delta_{r-1}}||_p^p\cdots||e^{-L_1(S_1)\delta_1}||_p^p \quad (5.19)$$
$$\leq||e_i^Te^{-L_r(S_r)\delta_r}-e_0^T||_p^p, \quad (5.20)$$

where (5.19) follows from the definition of the matrix 2-norm, while (5.20) follows from the fact that $e^{-L_m(S_m)\delta_m}$ is a stochastic matrix for all $m \in \{1,\dots,r-1\}$. Eq. (5.20), together with the fact that $||e_i^Te^{-L_m(S_m)\delta_m}-e_0^T||_p^p \geq 0$ for all $m \in \{1,\dots,r-1\}$, yields the desired result. □

Based on Proposition 5.1, we define an error metric for the unknown topology case by

$$\hat{f}_t(S_1,\ldots,S_r) \triangleq \frac{1}{r}\sum_{m=1}^{r}\sum_{i\in V\setminus S_m}\left[\sum_{j\in V\setminus S}\left(e_i^T e^{-L(t_{m-1})\delta_m}\right)_j^p + \left(1-\sum_{j\in S}\left(e_i^T e^{-L(t_{m-1})\delta_m}\right)_j\right)^p\right]. \tag{5.21}$$

Using the metric (5.21), minimizing the convergence error is achieved by selecting leaders according to the optimization problem

$$\begin{aligned} &\text{minimize} && \hat{f}_t(S_1,\ldots,S_r) \\ &S_1,\ldots,S_r && \\ &\text{s.t.} && |S_m| \le k,\ m=1,\ldots,r \end{aligned} \tag{5.22}$$

At time t_m, the system designer has access to the sequence of Laplacian matrices $L(t_1)$, ..., $L(t_{m-1})$, and hence can compute the convergence error bound $\hat{f}_t(S_1,\ldots,S_{m-1})$ arising from any sequence of leader sets $S_1,\ldots,S_{m-1}$.

In order to analyze the optimality of our proposed algorithm and prove lower bounds on the achievable optimality gap, we introduce the concept of regret, denoted $R(S_1,\ldots,S_r)$, which is defined as the difference between the convergence error from sets $S_1,\ldots,S_r$ and the minimum convergence error from any fixed leader set. The regret is defined as

$$R(S_1,\ldots,S_r) = \hat{f}_t(S_1,\ldots,S_r) - \min_S\left\{\frac{1}{r}\sum_{m=1}^{r}\hat{f}_t(S|L(t_m))\right\}. \tag{5.23}$$

The lower bounds and optimality gap derived below are based on the regret (5.23).

Lower bounds on regret

In what follows, we give a lower bound on the minimum possible regret for any algorithm without knowledge of the topology distribution. This bound provides a comparison for evaluating possible leader selection algorithms, including our proposed approach. In order to prove the bound, we construct a lower bound on the regret for a specific topology distribution in which, at each time epoch, each node is independently connected to all other nodes with probability 1/2, and only one node can act as leader. Intuitively, the leader selected by

any algorithm without topology information will provide very high convergence error with probability 1/2, leading to a large regret. On the other hand, an algorithm with knowledge of the network topology will choose a leader node that is connected to all other nodes, with low resulting convergence error. Formally, the theorem is stated as follows.

Theorem 5.7. *For any leader selection algorithm for solving (5.22) with $k = 1$ and for n and r sufficiently large, there exists a sequence of topologies $G(t_1), \ldots, G(t_r)$ such that the regret is bounded by*

$$R \geq \frac{1}{r}\sqrt{r/2 \ln n}. \tag{5.24}$$

Proof. We prove the theorem by constructing a sequence of random topologies $(G(t_1), \ldots, G(t_r))$ such that, when $k = 1$, the expected regret of any leader selection algorithm that does not take the distribution of the network topology as input satisfies (5.24). An outline of the proof is as follows. We begin by defining the distribution of $G(t_1), \ldots, G(t_r)$. The next step is to derive an expression for R for this choice of topologies. We then prove the bound (5.24) by analyzing the asymptotic behavior of our expression for R. Details of the proof are given below.

Consider a directed graph $G(t)$ in which, at time t_m, the edge set $E(t_m)$ is chosen such that, for each node i, the neighbor set $N(i)$ satisfies

$$N(i) = \begin{cases} V \setminus \{i\}, & \text{w.p. } 1/2 \\ \emptyset, & \text{w.p. } 1/2 \end{cases}$$

In other words, each node has outdegree $(n-1)$ with probability 1/2 and outdegree 0 with probability $\frac{1}{2}$. Define $\sigma(n)$ to be the normalized expected convergence error when a node with outdegree $(n-1)$ acts as leader. The value of $\sigma(n)$ is normalized so that the convergence error is 1 when the outdegree of the leader is 0. This yields

$$\hat{f}_t(\{i\}|G(t_m)) = \begin{cases} \sigma(n), & \text{w.p. } 1/2 \\ 1, & \text{w.p. } 1/2. \end{cases} \tag{5.25}$$

It is assumed that each node's edge set is chosen independently at random at each time step.

The first term of R in (5.23) is given as follows. Any algorithm that does not have foreknowledge of the network topology will have expected error that satisfies

$$\begin{aligned}\mathbf{E}(\hat{f}_t(S_1,\ldots,S_r)) &= \sum_{m=1}^{r}\sum_{i=1}^{n}[\mathbf{E}(\hat{f}_t(\{i\})|S_m=\{i\}) \\ &\qquad \cdot Pr(S_m=\{i\})] \\ &= \sum_{m=1}^{r}\left[\frac{1}{2}(1+\sigma(n))\sum_{i=1}^{n}Pr(S_m=\{i\})\right] \\ &= \frac{r}{2}(1+\sigma(n)).\end{aligned}$$

We now consider the second term of R in (5.23). Let $A_{r,i} \triangleq \sum_{m=1}^{r}\hat{f}_t(\{i\}|G(t_m))$ to be the convergence error for r topologies when the leader node is i, so that the second term of (5.23) is equal to $\min_i A_{r,i}$. The mean of $A_{r,i}$ is $\frac{r}{2}(1+\sigma(n))$ from (5.25), and the variance is $\frac{r}{4}(1-\sigma(n))^2$. When $\sigma(n)=0$, the value of $A_{r,i}$ is minimized, and hence the second term of (5.23) is minimized. In what follows, we therefore assume that $\sigma(n)=0$. Under this assumption, $A_{r,i}$ is a binomial random variable. We now have that the expected value of the regret, when $\sigma(n)=0$, is equivalent to $R=\frac{1}{r}(\frac{r}{2}-\min_i A_{r,i})$, so that (5.24) holds iff

$$\frac{\frac{1}{r}\left(\frac{r}{2}-\min_i A_{r,i}\right)}{\frac{1}{r}\sqrt{r/2\ln n}} = \frac{\frac{r}{2}-\min_i A_{r,i}}{\sqrt{r/2\ln n}} \geq 1 \tag{5.26}$$

for r and n sufficiently large.

In order to prove (5.26), define $B_{r,n}$ by multiplying the left-hand side of (5.26) by -1, so that

$$B_{r,n} \triangleq \frac{\min_i A_{r,i}-\frac{r}{2}}{\sqrt{r/2\ln n}}.$$

Proving Equation (5.26) is then equivalent to showing that, for r and n sufficiently large and any $\kappa>0$, $\mathbf{E}(B_{r,n})\leq -1+\kappa$. For any random variable X, we have that

$$\begin{aligned}\mathbf{E}(X) &= \mathbf{E}(X|X\leq -1+\frac{\kappa}{3})Pr\left(X\leq -1+\frac{\kappa}{3}\right)+\mathbf{E}(X|X\in[-1+\frac{\kappa}{3},0])Pr(X\in[-1+\kappa,0]) \\ &\quad +\mathbf{E}(X|X\geq 0)Pr(X\geq 0) \\ &\leq \left(-1+\frac{\kappa}{3}\right)Pr\left(X\leq -1+\frac{\kappa}{3}\right)+0\cdot Pr\left(X\in[-1+\frac{\kappa}{3},0]\right)+\mathbf{E}(X|X\geq 0)Pr(X\geq 0).\end{aligned}$$

Hence for $B_{r,n}$ in particular we have

$$\mathbf{E}(B_{r,n}) \leq \left(-1+\frac{\kappa}{3}\right)Pr\left(B_{r,n}\leq -1+\frac{\kappa}{3}\right)+\int_0^\infty Pr(B_{r,n}\geq c)\,dc. \tag{5.27}$$

First, note that

$$\begin{aligned} Pr(B_{r,n} \geq c) &= Pr(A_{r,1} - r/2 \geq c\sqrt{r(\ln n)/2}, \ldots, \\ &\quad A_{r,n} - r/2 \geq c\sqrt{r(\ln n)/2}) \qquad (5.28) \\ &= \prod_{i=1}^{n} Pr(A_{r,i} \geq r/2 + c\sqrt{r(\ln n)/2}) \qquad (5.29) \\ &= \left(Pr\left(A_{r,1} \geq \frac{r}{2} + r\left(\frac{c\sqrt{(\ln n)/2}}{\sqrt{n}} \right) \right) \right)^n \\ &\leq \left(\exp\left(-2\left(c\sqrt{\frac{2\ln n}{r}} \right)^2 r \right) \right)^n \qquad (5.30) \\ &= \exp(-4c^2 n \ln n), \qquad (5.31) \end{aligned}$$

where (5.29) follows from the fact that $(A_{r,1}, \ldots, A_{r,n})$ are i.i.d. random variables and (5.30) follows from Hofferding's inequality. Hence

$$\int_0^\infty Pr(B_{r,n} \geq c)\ dc \leq \int_0^\infty \exp\left(-4c^2 n \ln n\right)\ dc = \frac{1}{8}\sqrt{\frac{\pi}{n \ln n}}$$

which is less than $\epsilon/3$ for n sufficiently large.

Examining the first term of (5.27), we first define $Y_{r,i,n} = \frac{A_{r,i} - \frac{r}{2}}{\sqrt{r/2}}$. By the Central Limit Theorem and the fact that $A_{r,i}$ is binomial, $Y_{r,i,n}$ converges to a $N(0,1)$ random variable as $r \to \infty$. Hence

$$\begin{aligned} Pr\left(B_{r,n} \leq -1 + \frac{\kappa}{3}\right) &= Pr\left(\min_{1 \leq i \leq n} Y_{r,i,n} \leq -1 + \frac{\kappa}{3}\right) = 1 - Pr\left(\min_{1 \leq i \leq n} Y_{r,i,n} \geq -1 + \frac{\kappa}{3}\right) \\ &= 1 - Pr\left(Y_{r,1,n} \geq -1 + \frac{\kappa}{3}\right)^n \geq 1 - \frac{\kappa}{3} \end{aligned}$$

for n sufficiently large. We therefore have

$$(-1 + \kappa/3) Pr(B_{r,n} \leq -1 + \kappa/3) \leq \left(-1 + \frac{\kappa}{3}\right)\left(1 - \frac{\kappa}{3}\right) < -1 + \frac{2\kappa}{3},$$

and hence, (5.27) reduces to $\mathbf{E}(B_{r,n}) \leq -1 + \frac{2\kappa}{3} + \frac{\kappa}{3} < -1 + \kappa$, which yields (5.26), thus proving the theorem. □

Experts Algorithms

We now give background on experts algorithms, which will be used as subroutines in our algorithms. All results can be found in [26]. As a preliminary, we define a set of n actions

$\mathcal{A} = \{a_1, \ldots, a_n\}$ (we use the notation n because, in the next subsection, each expert will correspond to selecting a different node as leader). We also define a sequence of loss functions $\ell_1, \ldots, \ell_T : \mathcal{A} \to \mathbb{R}_+$ for time epochs $1, \ldots, T$. The loss for action a_j in epoch i is defined to be $l_i(a_j)$.

An *experts algorithm* outputs an action $a \in \mathcal{A}$ at each epoch i based on observations of past losses $\{l_{i'}(a) : a \in \mathcal{A}, i' = 1, \ldots, i-1\}$, in order to minimize the total losses. The effectiveness of an experts algorithm can be quantified using the regret, defined as follows.

Definition 5.2. *The regret of an experts algorithm that chooses action $a^{(j)}$ at time epoch j is defined as*

$$R \triangleq \sum_{i=1}^{T} \ell_i(a^{(j)}) - \min_{a \in \mathcal{A}} \left\{ \sum_{i=1}^{T} \ell_i(a) \right\}.$$

One such experts algorithm is given below as Algorithm *Randomized-experts*.

Algorithm 11 Experts algorithm with sublinear regret.

procedure RANDOMIZED-EXPERTS($\mathcal{A}$)
  Input: Set of actions $\mathcal{A} = \{a_1, \ldots, a_n\}$
  Output: Actions $a^{(1)}, \ldots, a^{(T)}$ for epochs $1, \ldots, T$
  Choose $\eta \in [0, 1]$, $\mathbf{w}_j \leftarrow 1$, $j = 1, \ldots, n$
  $p_j \leftarrow \frac{1}{n}$, $j = 1, \ldots, n$
  for Epochs $i = 1, \ldots, T$ **do**
    Select $a^{(i)}$ from probability distribution $\mathbf{p}$
    Receive losses $\ell_i(a_1), \ldots, \ell_I(a_n)$
    for $j = 1, \ldots, n$ **do**
      $\mathbf{w}_j \leftarrow \mathbf{w}_j \exp\left(-\eta \ell_i(a_j)\right)$
    end for
    $\mathbf{p} \leftarrow \left(\mathbf{1}^T \mathbf{w}\right)^{-1} \mathbf{w}$
  end for
end procedure

Intuitively, the *Randomized-experts* algorithm maintains a probability distribution $\mathbf{p} \in \mathbf{R}^n$, where $\mathbf{p}_j$ represents the probability of selecting action j. The algorithm assigns higher weight, and hence higher selection probability, to actions that have generated low losses in the past. **Randomized-experts** updates the weights according to an exponential rule. While alternative update algorithms such as polynomial weighting have been studied, exponential weighting schemes have been found to provide lower error in empirical and analytical studies [26]. The following proposition characterizes the regret of *Randomized-experts*.

Proposition 5.2. *The regret of the set* $a^{(1)}, \ldots, a^{(T)}$ *returned by the algorithm* Randomized-experts *satisfies* $R(a^{(1)}, \ldots, a^{(T)}) \leq O\left(\sqrt{\frac{T \ln n}{2}}\right)$.

Leader selection algorithms for unknown topology distribution

Our approach to solving the optimization problem (5.22) is based on the greedy algorithm *Select-k-leaders*. In the unknown topology setting, however, the exact value of $\hat{f}_t(S)$ cannot be computed. Instead, we use the experts algorithm *Randomized-experts* to estimate which leader v_j to add to the leader set at the j-th iteration of the algorithm.

The algorithm maintains a set of weights w_{ij}^m for $i = 1, \ldots, n$, $j = 1, \ldots, k$, and each time step $m = 1, \ldots, r$. The weight w_{ij}^m represents the current estimate of node i's utility when selected as the j-th leader node at time t_m. Initially, $w_{ij}^0 = 1$ for all i and j, since no information is available regarding the effectiveness of each node as a leader. At step m, a set of probability distributions $\pi_1^m, \ldots, \pi_k^m$ is obtained by setting $\pi_j^m(i) = w_{ij}^m / \sum_{i=1}^n w_{ij}^m$. The leader set S_m for time interval $[t_m, t_{m+1}]$ is obtained by first selecting a leader according to distribution π_1^m. In general, the j-th leader is selected by choosing a node from $V \setminus S_{j-1}$ according to distribution π_j^m, which can be viewed as selecting the leader according to an experts algorithm where the weights are computed based on the convergence error during previous time steps.

After stage m, the weights are updated in order to reflect the convergence error that each node would have provided if it had acted as leader during $[t_m, t_{m+1}]$. Define $z_{m,i,j}$ to be $\hat{f}_t(S_m^{j-1}) - \hat{f}_t(S_m^{j-1} \cup \{i\})$. The weight w_{ij}^m is updated to $w_{ij}^{m+1} = \beta^{z_{m,i,j}} w_{ij}^m$, as in algorithm **Randomized-experts**, where $\beta \in [0, 1]$ is chosen in order to vary the rate at

which the algorithm adapts to changes in topology. For low β values, a large convergence error will result in a large penalty (i.e., a much lower weight) for a node. The algorithm is summarized as *Select-dynamic-leaders* below.

Algorithm 12 Algorithm for selecting up to k leaders for time interval $[t_r, t_{r+1}]$ under unknown topology dynamics.

procedure SELECT-DYNAMIC-LEADERS(V, k, w_{ij}^{r-1}, $G(t_r)$)
  Input: Set of nodes V, number of leaders k
    Node weights w_{ij}^{r-1}, $i \in V$, $j = 1, \ldots, k$, topology $G(t_r)$
  Output: Leader set S_r for state r, updated weights w_{ij}^{r}, $i \in V$, $j = 1, \ldots, k$
  $z_{ij}^{r} \leftarrow 0$, $i \in V$, $j = 1, \ldots, k$, $S_r \leftarrow \emptyset$
  for $j = 1, \ldots, k$ **do**
    for $i = 1, \ldots, n$ **do**
      $z_{ij}^{r} \leftarrow \hat{f}_t(S_{r-1}^{j-1}) - \hat{f}_t(S_{r-1}^{j-1} \cup \{i\})$
      $w_{ij}^{r} \leftarrow w_{ij}^{r-1} \beta^{z_{ij}^{r}}$
    end for
    for $i = 1, \ldots, n$ **do**
      $\pi_{ij}^{r} \leftarrow w_{ij}^{r} / \left(\sum_{v \in V} w_{vj}^{r} \right)$
    end for
    $v_j^r \leftarrow$ choose $i \in V \setminus S_r$ according to π_j^r
    $S_r \leftarrow S_r \cup \{v_j^r\}$
  end forreturn S_r
end procedure

The following theorem describes bounds achieved on the regret of the algorithm *Select-dynamic-leaders*.

Theorem 5.8. *The algorithm* Select-dynamic-leaders *returns a sequence of sets* $S_1, \ldots, S_r$ *that satisfies*

$$\hat{f}_t(S_1, \ldots, S_r) \leq \left(1 - \frac{1}{e}\right) \hat{f}_t(S^*) + \frac{1}{e} f_{max} + \sum_{j=1}^{k} R_j,$$

where $S^* = \arg\min \left\{ \sum_{m=1}^{r} \hat{f}_t(S|L(t_m)) : |S| \leq k \right\}$ *and the regret* R_j *in choosing the* j*-th leader is given by* $R_j \leq \frac{1}{r}\left(\sqrt{2 f_{max} r \ln n} + \ln n\right)$.

Proof. By Theorem 6 of [125], an algorithm for maximizing a submodular function $f(S)$ that introduces error R_j from the greedy approach at each stage j, i.e.,

$$\max_v f(S \cup \{v\}) - f(S \cup \{v_j\}) \leq R_j$$

satisfies

$$f(S) > \left(1 - \frac{1}{e}\right) f(S^*) - \sum_{j=1}^{k} R_j.$$

Using the fact that $\hat{f}_t(S_r)$ is supermodular as a function of S_r (Theorem 5.3), setting $f(S) = f_{max} - \hat{f}_t(S)$ and rearranging terms implies that

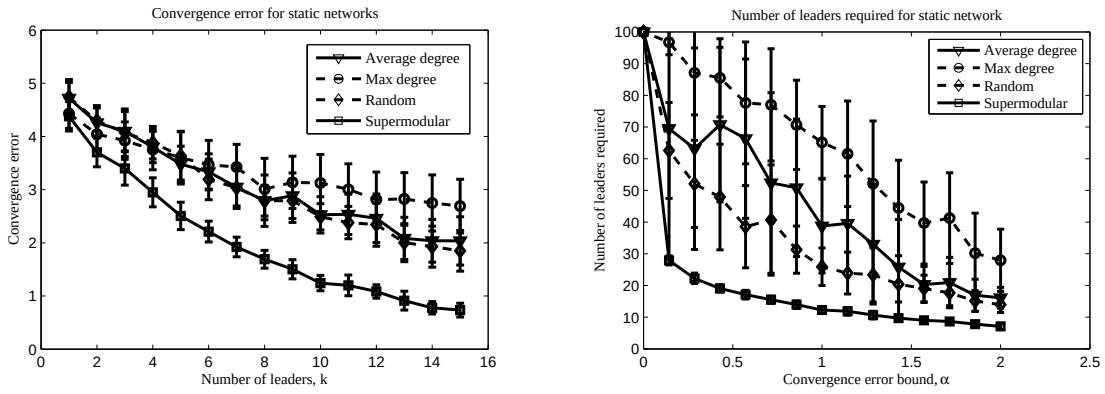
$$\hat{f}_t(S_r) \leq \left(1 - \frac{1}{e}\right) \hat{f}_t(S_r^*) + \frac{1}{e} f_{max} + \sum_{j=1}^{k} R_j.$$

It remains to bound the regret R_j. Lemma 4 of [52] yields the desired bound. □

Theorem 5.8 implies that, as the number of topologies r increases, the convergence error of algorithm *Select-dynamic-leaders* reaches a constant factor of the optimal convergence error. Indeed, *Select-dynamic-leaders* achieves the same optimality gap of $\left(1 - \frac{1}{e}\right)$ as the known topology distribution case. In other words, if the system designer observes the distribution of the network topology for a sufficiently long time, then the convergence error with the chosen leader set is equivalent to the convergence error when the topology distribution is known in advance. The algorithm incurs a cost of $O(nk)$ computations of $\hat{f}_t(S)$ for each network topology.

5.4 Numerical Study

We conduct a numerical study of our proposed approach using Matlab. We simulate a network of 100 nodes, deployed uniformly at random over a 1000m × 1000m area. A communication link exists between two nodes if they are within 300m of each other. The weight matrix W of Equation (5.1) is chosen by selecting a weight W_{ij} for each link (i, j)

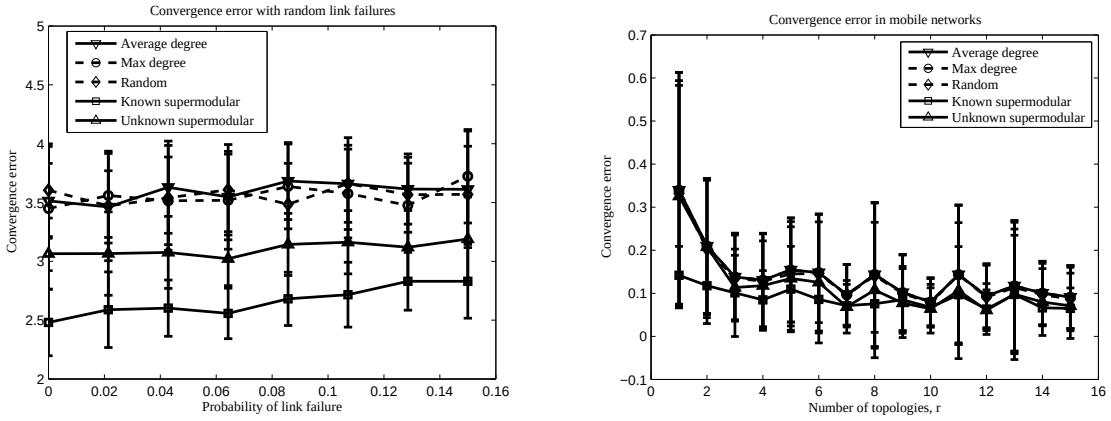


(a) (b)

Figure 5.1: Comparison of random, maximum degree, average degree, and supermodular leader selection algorithm *Select-k-leaders* for static networks. (a) The supermodular optimization approach *Select-k-leaders* consistently provides the lowest convergence error, while the average degree-based selection slightly outperforms random leader selection. (b) Evaluation of the selection of the minimum-size set of leader nodes to achieve a given bound on the convergence error, as a function of the bound, for static networks. The supermodular optimization approach *Select-minimal-leaders* requires fewer leaders in order to satisfy convergence error criteria.

uniformly from the interval $[0, 50]$. Each data point represents an average of 50 independent trials. The number of leaders varies from 1 to 15.

Three types of network topologies are considered. The first type consists of a static network topology with the parameters described above. In the second type, a topology with the parameters described above is subject to link failures. As a result, the topology during time interval $[t_m, t_{m+1}]$, with $m = 1, \ldots, 8$, is obtained by removing each link independently and with equal probability from the underlying network. The probability of link failures varies from 0 to 0.15. In the third type of network topology, the positions of the network nodes vary according to a random waypoint mobility model [60]. In this model, the nodes attempt to maintain a fixed position relative to a time-varying reference state. During each time interval, the position of each node is equal to the sum of the reference state, the desired relative position of that node, and a disturbance with magnitude chosen uniformly at random from the interval $[0, 50]$. The relative positions are chosen uniformly at random



(a) (b)

Figure 5.2: Comparison of random, degree-based, and supermodular optimization algorithms *Select-dynamic-leaders* and *Select-k-leaders* for dynamic networks. (a) Impact of random link failures on convergence errors. The convergence error for each scheme increases along with the failure probability. Furthermore, for each failure probability, knowledge of the distribution of link failures leads to lower convergence error than the unknown topology case. (b) Convergence errors when the network topology varies due to random waypoint mobility model with speed $v = 100$m/s. As the number of topologies increases, the algorithm *Select-dynamic-leaders* adaptively updates the leader set, eventually providing error comparable to the known topology case.

from the 1000m $\times$ 1000m area, while the reference state moves according to a random walk with speed 100m/s. The number of time epochs varies from 1 to 8 in order to track the performance of each simulated algorithm over time.

For each type of topology, leader selection is performed using four algorithms, namely, random selection of leader nodes, selection of maximum-degree nodes as leaders, selection of average-degree nodes, and leader selection via supermodular optimization. For the problem of selecting up to k leaders in order to minimize convergence errors, the algorithm *Select-k-leaders* is used, while algorithm *Select-minimal-leaders* is used to select the minimum-size set of leaders in order to achieve a convergence error bound. In addition, for the two types of dynamic topologies that are considered, a comparison between the algorithm *Select-k-leaders*, which incorporates prior knowledge of the topology distribution, and the algorithm *Select-dynamic-leaders*, which does not incorporate prior knowledge, is provided.

The goal of the numerical evaluation is to address the following questions: (i) How does the convergence error of the leaders selected using supermodular optimization compare with state of the art leader selection algorithms? (ii) How does the number of leaders required by each scheme vary as a function of the convergence error bound? and (iii) When the topology distribution is unknown, does the algorithm *Select-dynamic-leaders* learn the distribution and achieve convergence error comparable to the known topology case?

Question (i) is addressed for the static network case in Figure 5.1(a). The supermodular optimization approach of algorithm *Select-k-leaders* selects leaders with less than half the convergence error of random leader selection. We observe that, while the random, degree, and average degree-based algorithms achieve comparable convergence error, selection of random nodes as leaders slightly outperforms both degree-based schemes. Similarly, in Figures 5.2(a) and 5.2(b) supermodular optimization of the leader set, using the algorithms *Select-k-leaders* and *Select-dynamic-leaders*, results in lower convergence error than random and degree-based schemes for dynamic topologies.

In order to investigate the gap between the minimum achievable convergence error and the error provided by the supermodular optimization approach, we compute the minimum convergence error for a single randomly generated network by iterating over all possible leader sets. We find that the lowest achievable convergence error was 4.61, 3.82, and 3.23 for $k = 1$, $k = 2$, and $k = 3$ respectively (larger values of k are not considered due to the computation cost of iterating over all possible leader sets). By comparison, for this network the supermodular optimization approach resulted in error of 4.61, 3.82, and 3.27 for $k = 1$, $k = 2$, and $k = 3$ respectively.

Figure 5.1(b) is related to Question (ii). The supermodular optimization approach of algorithm *Select-minimal-leaders* requires less than half the number of leaders to achieve an error bound of 1 than the next highest-performing scheme, which was random leader selection. As in the case of selecting a fixed number of leaders, random leader selection outperformed degree-based leader selection on average.

Figures 5.2(a) and 5.2(b) compare the convergence error for the supermodular optimization algorithms, with and without prior topology information (Question (iii)). In the case of random link failures (Figure 5.2(a)), prior knowledge in algorithm *Select-k-leaders* provides

a consistent advantage over the case where no prior information is available, using *Select-dynamic-leaders*. This is because the link failures occur independently at each time interval, and hence the ability of the algorithm *Select-dynamic-leaders* to dynamically adapt to new topology information does not lead to lower convergence error. When the topology changes due to a mobility model (Figure 5.2(b)), the *Select-dynamic-leaders* algorithm eventually achieves the same performance as the algorithm with prior knowledge as time progresses. Given sufficient time intervals, *Select-dynamic-leaders* eventually learns which leader nodes will provide the lowest convergence error under the mobility model that is used.

5.5 Conclusions and Future Work

In this chapter, we investigated leader selection for minimizing convergence error, defined as the distance between the intermediate states of the follower nodes and the convex hull of the leader states, in linear multi-node systems. We developed efficient algorithms for leader selection through the following approach. First, we derived an upper bound on the convergence error that is independent of the initial states of the network, and proved a connection between the upper bound and a random walk on the graph. Using the connection between convergence error and the random walk, we proved that the upper bound on the convergence error is a supermodular function of the set of leader nodes. The supermodular structure of the convergence error enables formulation and approximation of leader selection as a discrete optimization problem, rather than relying on continuous extensions of the problem.

We formulated two leader selection problems for MAS with static topology. In the first problem, a fixed number of leaders is selected in order to minimize the convergence error. In the second problem, the minimum-size set of leaders is selected in order to achieve a given bound on the convergence error. We presented efficient algorithms for each problem, and proved that both algorithms achieve an $O(1)$ optimality gap with the lowest possible convergence error.

We introduced a supermodular optimization approach to leader selection in MAS with dynamic topologies, including the cases where the system designer has prior knowledge of the distribution of the topology, as well as the case where the system designer has no prior

information and must adaptively update the leader set over time. For the case where the system designer has no prior topology information, we derived a lower bound on the convergence error that can be achieved by any algorithm, as well as bounds on the convergence error achieved by our approach.

Our results were illustrated through a numerical study, in which we compared our supermodular optimization approach with random, average-degree, and maximum-degree leader selection, for static networks, networks with random link failures, and networks with mobile nodes. We found that the supermodular optimization approach significantly outperforms the other algorithms. Furthermore, the numerical evaluation showed that the leader selection approach without prior topology information eventually achieves a convergence error that is comparable to the algorithm with prior information.

5.5.1 Future Work

Joint Leader, Link Weight, and Network Topology Selection: The set of leader nodes is only one factor that influences the convergence error in distributed networked systems. The network topology and weights on the interactions between nodes also determine the convergence rate. In [37], we proposed a convex relaxation approach for joint leader and weight selection, with provable optimality under certain special cases. A more general approach that extends these optimality guarantees is a topic of future work.

Optimizing Spectral Convergence Metrics: A variety of convergence rate metrics are related to the graph spectrum, including the mixing time of random walks on the graph [80]. At present, submodular structure of these spectral convergence metrics has not been investigated in the literature to the best of our knowledge. The connection between the spectral properties and random walks on the graph may enable an approach analogous to this chapter.

Leader Selection with Node Power Constraints: In applications where the nodes communicate via wireless networks, each node may have a power budget or cost associated with acting as a leader. Hence, instead of a cardinality constraint, the goal may be to maintain a total power budget below a given budget. Knapsack constraints are one possible approach

to modeling these power budgets.

Chapter 6

JOINT PERFORMANCE AND CONTROLLABILITY

The techniques for leader selection developed over the previous chapters are designed to optimize performance criteria, including convergence error and robustness to link noise. Another important design parameter is controllability, defined as the ability to drive the network from any initial state to any desired state by controlling the leader node states. Recently, polynomial-time algorithms have been proposed for selecting the minimum-size set of leader nodes to ensure controllability [85], making use of results from [81] on the theory of structural controllability. These approaches to input selection assume either that the system matrices are fully known, or that only the structure of the system matrices is known. In the latter case, the system matrix consists of zero entries and free parameters, where the free parameters can take any arbitrary value. Many practical systems, however, lie between these two extremes, with either a mixture of known and unknown matrix entries, or structural relationships between the unknown entries (i.e., all of the entries of a row sum to zero, as in the case of linear consensus algorithms [63]). In [29], it was observed that a set of selected input nodes may not satisfy controllability if these structural properties are not taken into account.

While input selection methods that incorporate system structure have been proposed for specific applications such as consensus [55], a computationally tractable general framework that guarantees controllability of structured systems remains an open problem. Moreover, there is no design method for leader selection that incorporates both controllability and performance metrics, such as robustness to link noise and convergence error. We aim to address these problems within a unifying framework.

In this chapter, we develop a submodular optimization framework for input selection based on joint consideration of performance and controllability. The submodular structure implies that a variety of input selection problems can be solved up to a $(1 - 1/e)$ optimality

bound, using algorithms that depend on the constraints of each problem. We first show that selecting the minimum-size set of input nodes to satisfy controllability of structured linear descriptor systems can be mapped to a maximum-cardinality matroid intersection problem, leading to the first polynomial-time algorithm for ensuring controllability of such systems. We then investigate selecting a set of up to k input nodes to maximize a performance metric while satisfying a controllability constraint, and prove that this problem is equivalent to submodular maximization with two matroid basis constraints. We develop a randomized algorithm for solving a relaxed, continuous version of the problem with a $(1-1/e)$ optimality bound, which can be rounded to a feasible input set that satisfies controllability. As a third problem, we relax the requirement that the system is controllable and select input nodes based on a trade-off between performance and controllability. In this case, we prove that the problem has the structure of submodular maximization subject to a cardinality constraint, leading to $(1-1/e)$ optimality bound.

We next study input selection when the complex network is strongly connected (i.e., there exists a directed path between any two nodes). We prove that, for almost all systems with a given structure, the controllability of the networked system can be represented as a single matroid constraint. Based on this result, we derive a linear-time algorithm for selecting input nodes for controllability in structured linear descriptor systems, and prove that this algorithm is guaranteed to select the minimum-size input set. We further show that the problem of selecting a set of up to k input nodes to optimize performance subject to a controllability constraint can be solved with optimality bound of $(1-1/e)$ when the network is strongly connected.

We investigate three special cases of our framework, arising from classes of structured systems that have been studied in the existing literature, namely, linear consensus [63], second-order integrator dynamics [114], and systems in which all nonzero parameters can take arbitrary values [85]. We show that our general approach achieves at least the same optimality guarantees compared to the current state of the art for each individual problem. Our results are illustrated via numerical study in the special case of a consensus network.

This chapter is organized as follows. In Section 6.1, we present our system model, definitions and sufficient conditions for controllability, background on submodularity and

matroid theory, and examples of performance metrics that can be incorporated into our framework. In Section 6.2, we develop our submodular optimization framework for input selection in structured systems. Section 6.3 discusses input selection in strongly connected network, and shows how connectivity improves the optimality bounds of our approach. Section 6.4 presents three special cases of our framework found in the existing literature. Section 6.5 contains a numerical study. Section 6.6 concludes the chapter.

6.1 Model and Preliminaries

In this section, we describe the system model and definitions of controllability considered. We then present sufficient conditions for controllability from previous work based on matroids and an auxiliary graph construction.

6.1.1 System Model

We consider a linear, time-invariant networked system with a total of n states, where $\mathbf{x}(t) \in \mathbb{R}^n$ is the vector of states at time t. In the absence of any control inputs, the system dynamics are described by

$$F\dot{\mathbf{x}}(t) = A\mathbf{x}(t). \tag{6.1}$$

Eq. (6.1) defines a *linear descriptor system* [42]. The matrices F and A can be further decomposed as $F = Q_F + T_F$ and $A = Q_A + T_A$. The values of Q_F and Q_A are known, fixed parameters. The matrices T_F and T_A are *structure matrices*, in which any nonzero entry can take any arbitrary real value. We assume that the system defined by (6.1) satisfies solvability, defined as follows.

Definition 6.1. *An LTI system of the form (6.1) is* solvable *if for any initial state* $\mathbf{x}(0) \in \mathbb{R}^n$*, there exists a unique trajectory* $\{\mathbf{x}(t) : t > 0\}$ *that satisfies (6.1).*

We now describe the effect of control inputs on the system (6.1). A subset S of states act as control inputs, i.e., for each state $i \in S$, there exists a control input $u_i(t)$ such that $x_i(t) = u_i(t)$ for all t. The states in S correspond to the states of nodes that are controlled directly by an external entity. In the following, without loss of generality, we assume that the state indices are ordered so that states $\mathbf{x}_R = (x_1, x_2, \ldots, x_{n-|S|})$ do not act as control

inputs, and states $\mathbf{x}_S = \{x_{n-|S|+1}, \ldots, x_n\}$ act as control inputs. The system dynamics are then given by

$$\left(\begin{array}{c} \hat{F} \\ \hline 0 \end{array}\right)\left(\begin{array}{c} \dot{\mathbf{x}}_R(t) \\ \dot{\mathbf{x}}_S(t) \end{array}\right) = \left(\begin{array}{c|c} A_{RR} & A_{RS} \\ \hline 0_{(n-|S|)\times|S|} & -I_{|S|\times|S|} \end{array}\right)\left(\begin{array}{c} \mathbf{x}_R(t) \\ \mathbf{x}_S(t) \end{array}\right) + \left(\begin{array}{c} 0_{(n-|S|)\times|S|} \\ I_{|S|\times|S|} \end{array}\right)\mathbf{u}(t) \tag{6.2}$$

In (6.2), $\hat{F}$ is the $(n-|S|)\times n$ matrix consisting of the first $(n-|S|)$ rows of F. The matrices A_{RR} and A_{RS} consist of the first $(n-|S|)$ and last $|S|$ columns of the first $(n-|S|)$ rows of A, respectively. The vector $\mathbf{u}(t)$ is the control input signal. The last $|S|$ rows of the equation enforce the condition that $x_i(t) = u_i(t)$ for all $i \in S$.

As a notation, for a square matrix $X \in \mathbb{R}^{n\times n}$, $N(X)$ is a graph with n vertices, where there exists an edge (i,j) if X_{ji} is nonzero.

6.1.2 Definitions of Controllability

We now present the definition of controllability considered in this work, which can be found in more detail in [113]. For general systems of the form

$$F\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t) \tag{6.3}$$

we first have the following lemma.

Lemma 6.1 ([140])**.** *If the system (6.3) is solvable, then we can write* $\mathbf{x} = [\mathbf{x}_1\ \mathbf{x}_2]^T$ *and have matrices* F_1*,* F_2*,* B_1*, and* B_2 *such that the dynamics (6.3) are equivalent to*

$$\begin{aligned} \dot{\mathbf{x}}_1(t) &= F_1\mathbf{x}_1(t) + B_1\mathbf{u}(t) \\ F_2\dot{\mathbf{x}}_2(t) &= \mathbf{x}_2(t) + B_2\mathbf{u}(t) \end{aligned}$$

We now define the concepts of admissibility and reachability.

Definition 6.2. *An initial state* $\mathbf{x}(0)$ *is* admissible *if* $\mathbf{x}_2(0) = -\sum_{i=0}^{m-1} F_2^i u^{(i)}(0)$*, where* m *is the degree of nilpotency of* F_2 *and* $u^{(i)}$ *is the* i*-th derivative of the input* u*. A state* $\mathbf{x}^*$ *is* reachable *if there exists* $t > 0$*, an admissible initial state* $\mathbf{x}_0$*, and an input signal* $\{\mathbf{u}(t') : t' \in [0,t]\}$ *such that* $\mathbf{x}(t) = \mathbf{x}^*$ *when* $\mathbf{x}(0) = \mathbf{x}_0$*.*

We let R denote the set of reachable states. We are now ready to define the concept of controllability.

Definition 6.3. *The system (6.2) is* controllable *if, for any admissible initial state* $\mathbf{x}_0$ *and any reachable final state* $\mathbf{x}^*$ *and time* $t > 0$*, there exists a control signal* $\{\mathbf{u}(t') : t' \in [0, t]\}$ *such that* $\mathbf{x}(0) = \mathbf{x}_0$ *and* $\mathbf{x}(t) = \mathbf{x}^*$.

Finally, structural controllability is defined as follows.

Definition 6.4. *The system (6.3) is* structurally controllable *if there exist values for the free parameter matrices* T_F*,* T_A*, and* T_B *such that (6.3) is controllable.*

Structural controllability holds if the system (6.2) is controllable for almost any choice of the free parameters. Note that, if $Q_A = Q_F = 0$, then Definition 6.4 reduces to that of [85]. A matrix pencil interpretation is given by the following theorem.

Theorem 6.1 ([94])**.** *The system (6.3) is structurally controllable if and only if there exist free parameter matrices* T_F*,* T_A*, and* T_B *such that the following conditions hold:*

$$(F, A, B) \text{ is solvable} \tag{6.4}$$

$$\mathit{rank}(A|B) = n \tag{6.5}$$

$$\mathit{rank}((A - zF)|B) = n \quad \text{for all } z \in \mathbb{C} \tag{6.6}$$

We have assumed that (6.4) holds, since the system is solvable. It remains to find equivalent or sufficient conditions for (6.5) and (6.6). The following lemma gives a matroid-based sufficient condition for (6.5).

Lemma 6.2 ([94])**.** *Let* $\mathcal{M}(I|Q_A|Q_B)$ *denote the linear matroid defined by the fixed parameter matrices* Q_A *and* Q_B*, and let* $\mathcal{M}(I|T_A|T_B)$ *denote the matroid defined by the free parameter matrices* T_A *and* T_B*. Then* $\mathit{rank}(A|B) = n$ *iff* $\mathit{rank}(\mathcal{M}(I|Q_A|Q_B) \vee \mathcal{M}(I|T_A|T_B)) = 2n$.

In the following section, we give a graph construction, introduced in [94], that will be used to derive sufficient conditions for (6.6).

6.1.3 Auxiliary Graph Construction

We consider the following auxiliary graph constructed from (6.3). Define a matrix

$$\Omega = \begin{array}{c} \mathbf{w}: \\ \mathbf{x}: \\ \mathbf{u}: \end{array} \begin{pmatrix} Q_A - Q_F & Q_B \\ I & 0 \\ 0 & I \end{pmatrix} \tag{6.7}$$

Here, $\mathbf{w}$, $\mathbf{x}$, and $\mathbf{u}$ denote indices, so that the first n rows are indexed $w_1, \ldots, w_n$, the second n rows are indexed $x_1, \ldots, x_n$, and the third k rows are indexed $u_1, \ldots, u_k$.

As an intermediate step in the construction, define a bipartite graph H with vertex set

$$V_H = \{w_1^T, \ldots, w_n^T\} \cup \{x_1^Q, \ldots, x_n^Q\} \cup \{w_1^Q, \ldots, w_n^Q\}$$

and edge set

$$E_H = \{(w_i^T, w_i^Q) : i = 1, \ldots, n\} \cup \{(w_i^T, x_j^Q) : (i,j) \in N(T_A) \cup N(T_F)\}.$$

The following lemma gives properties of matchings in this bipartite graph.

Lemma 6.3 ([94])**.** *If the system (6.3) is solvable, then there exists a perfect matching m on the graph H (i.e., a matching in which all nodes in $\{w_1^T, \ldots, w_n^T\}$ are matched) such that the rows indexed in the set*

$$J = \{w_i : m(w_i^T) = w_i^Q\} \cup \{x_i : m(w_j^T) = x_i^Q \text{ for some } w_j\}$$

are linearly independent in Ω.

Let J be a set satisfying the conditions of Lemma 6.3 with matching m, and let Ω_J be a submatrix of Ω obtained from the rows in J. We have that Ω_J is an $n \times (n+k)$ matrix with full rank, and hence we can find rows J_1 such that $\Omega_{J \cup J_1}$ is a full-rank $(n+k) \times (n+k)$ matrix. Let $\tilde{\Omega} = \Omega \Omega_{J \cup J_1}^{-1}$. We index the columns of $\tilde{\Omega}$ in the set $J \cup J_1$.

We now define the auxiliary graph $\hat{G} = (\hat{V}, \hat{E})$ using the matrix $\tilde{\Omega}$. The vertex set is equal to

$$\hat{V} = \{w_i^T : i = 1, \ldots, n\} \cup \{w_i^Q : i = 1, \ldots, n\} \cup \{x_i^T : i = 1, \ldots, n\} \cup \{x_i^Q : i = 1, \ldots, n\} \\ \cup \{u_i^T : i = 1, \ldots, k\} \cup \{u_i^Q : i = 1, \ldots, k\},$$

while the edge set is given by

$$\begin{aligned}\hat{E} \;=\; & \{(w_i^T, x_j^Q) : (i,j) \in N(T_A) \cup N(T_F), m(w_i^T) \neq x_j^Q\} \\ & \cup \{(x_j^Q, w_i^T) : (i,j) \in N(T_A) \cup N(T_F), m(w_i^T) = x_j^Q\} \\ & \cup \{(w_i^T, w_i^Q) : m(w_i^T) \neq w_i^Q\} \cup \{(w_i^Q, w_i^T) : m(w_i^T) = m(w_i^Q)\} \\ & \cup \{(w_i^T, u_j^Q) : (i,j) \in N(T_B)\} \cup \{(u_j^T, u_j^Q) : j = 1, \ldots, n\} \\ & \cup \{(x^Q, y^Q) : x \in \hat{V} \setminus J, y \in J, \tilde{\Omega}_{xy} \neq 0, \tilde{\Omega}_{xz} = 0 \forall z \in J_1\}\end{aligned}$$

Based on this graph construction, the following sufficient condition for $\text{rank}(zF - A|B) = n$ can be derived.

Lemma 6.4. *Let*

$$S_- = \{v^Q : \tilde{\Omega}_{vj} \neq 0 \text{ for some } j \in J_1\}.$$

Let V'' denote the set of nodes in $\hat{V}$ that are part of a cycle. If all nodes in V'' are connected to S_- in the graph $\hat{G}$, then the condition rank$(zF - A|B) = n$ holds for almost any values of the free parameters.

The steps in the proof of Lemma 6.4 follow those in [94]. First, define a set of coefficients on the edges $\hat{E}$ of graph $\hat{G}$. The coefficient $\gamma(e)$ on edge $e \in \hat{E}$ is defined by

$$\gamma(e) = \begin{cases} r_i, & e = (w_i^T, w_i^Q), m(w_i^T) \neq w_i^Q \\ -r_i, & e = (w_i^T, w_i^Q), m(w_i^T) = w_i^Q \\ c_i, & e = (x_i^T, x_i^Q), i \notin J \\ -c_i, & e = (x_i^T, x_i^Q), i \in J \\ 1, & e = (w_i^T, x_j^T), m(w_i^T) \neq x_j^Q \\ -1, & e = (w_i^T, x_j^T), m(w_i^T) = x_j^Q \\ 0, & \text{else} \end{cases} \tag{6.8}$$

Now, let $\tilde{\Omega} = \Omega\Omega^{-1}_{J \cup J_1}$ and $V^- = \{v^Q : \tilde{\Omega}_{vj} \neq 0 \text{ for some } j \in J_1\}$. The following appears as Theorem 4.7 of [94], albeit with slightly modified notation.

Theorem 6.2. *Let $\tilde{V}$ denote the set of vertices that are not reachable to V^-, and let $\tilde{G}$ denote the subgraph of $\hat{G}$ induced by $\tilde{V}$. Then the condition rank$((zF - A)|B) = n$ holds*

for almost any choice of the free parameters if and only if the sum of the $\gamma(e)$ along any directed cycle in $\tilde{G}$ is zero.

We now prove Lemma 6.4.

Proof of Lemma 6.4. If the condition of Lemma 6.4 holds, then $\tilde{V}$ does not contain any cycle. Hence the condition of Theorem 6.2 holds automatically, and we have $\text{rank}((zF - A)|B) = n$ for almost any free parameters. □

6.2 Problem Formulation - Input Selection for Performance and Controllability

In this section, we present our submodular optimization framework for selecting input nodes based on performance and controllability. In order to provide computational tractability, we first map the sufficient conditions of Lemma 6.2 and Lemma 6.4 to matroid constraints on the input set. We present algorithms for selecting a minimum-size input set to guarantee controllability. We then formulate the problem of selecting a set of up to k input nodes to maximize a performance metric while satisfying controllability. We prove that the problem is a submodular maximization problem with two matroid basis constraints, and present efficient approximation algorithms. For the case where the number of input nodes may not be sufficient to guarantee controllability, we introduce a graph controllability index and formulate the problem of selecting input nodes based on a trade-off between performance and controllability.

6.2.1 Mapping controllability to matroid constraints

We derive matroid constraints for the set of non-input nodes that are equivalent to or sufficient for the conditions of Theorem 6.1. As a first step, we develop an equivalent representation of the dynamics (6.2), and prove that structural controllability of (6.2) is equivalent to structural controllability of the equivalent dynamics.

Lemma 6.5. *Define the dynamics*

$$F\dot{\mathbf{x}} = \left(\begin{array}{c|c} A_{RR} & A_{RS} \\ \hline A_{SR} & A_{SS} \end{array}\right) \begin{pmatrix} \mathbf{x}_R(t) \\ \mathbf{x}_S(t) \end{pmatrix} + \begin{pmatrix} 0 \\ T_B \end{pmatrix} \mathbf{u}(t) \tag{6.9}$$

where T_B is a diagonal matrix where the diagonal entries are free parameters. Then the system (6.2) is structurally controllable if and only if (6.9) is structurally controllable.

Proof. Suppose that the system (6.9) are structurally controllable with $(T_B)_{ii} = \alpha_i$ for some real α_i's. For a given initial state $\mathbf{x}_0$ and desired state $\mathbf{x}^*$, suppose that there exists a set of control inputs $u_1(t), \ldots, u_k(t)$ such that $\mathbf{x}(t) = \mathbf{x}^*$. We have

$$\alpha_i u_i(t) + \sum_j A_{ij} x_j(t) = \sum_j F_{ij} \dot{x}_j(t),$$

which is equivalent to

$$\alpha_i u_i(t) + \sum_j A_{ij} x_j(t) - \sum_j F_{ij} \dot{x}_j(t) + x_i(t) = x_i(t). \tag{6.10}$$

Rearranging terms implies that $x_i(t) = \hat{u}_i(t)$, where $\hat{u}_i(t)$ is the left-hand side of (6.10). Hence using $\hat{u}_i(t)$ as the input signal implies that structural controllability is achieved for the dynamics (6.2) as well. The proof of the converse is similar. □

Lemma 6.5 implies that it suffices to consider the conditions of Theorem 6.1 under the equivalent system (6.9). We first define a matroid constraint on the set of non-input nodes that is equivalent to (6.5). For the system (6.9), the condition $\text{rank}[\mathcal{M}(I|Q_A|Q_B) \vee \mathcal{M}([I|T_A|T_B])] = 2n$ of Lemma 6.2 is given by

$$\text{rank}[\mathcal{M}([I|Q_A|0]) \vee \mathcal{M}([I|T_A|T_B(S)])] = 2n,$$

where $T_B(S)$ is a diagonal matrix with a free parameter in the i-th diagonal entry for $i \in S$ and zeros elsewhere. In order to establish a matroid constraint for Lemma 6.2 we have the following lemma.

Lemma 6.6. *The function*

$$\rho_1(S) = \textit{rank}(\mathcal{M}([I|Q_A|0]) \vee \mathcal{M}([I|T_A|T_B(S)])) - \textit{rank}(\mathcal{M}([I|Q_A]) \vee \mathcal{M}([I|T_A]))$$

is a matroid rank function.

Proof. First, note that $\rho_1(\emptyset) = 0$. Next, consider $\rho_1(S \cup \{v\})$. Let $\hat{\rho}^S$ denote the rank function of $(\mathcal{M}(I|Q_A|0) \vee \mathcal{M}([I|T_A|T_B(S)]))$, and let $\hat{r}_1^S$ and $\hat{r}_2^S$ denote the rank functions

of $\mathcal{M}([I|Q_A|0])$ and $\mathcal{M}([I|T_A|T_B(S)])$, respectively. We have $\rho_1(S \cup \{v\}) = \hat{\rho}^{S\cup\{v\}}(V)$. Let X^* be the set that minimizes $\hat{r}_1^S(X^*) + \hat{r}_2^S(X^*) + n - |X^*|$. Observing that $\hat{r}_1^S = \hat{r}_1^{S\cup\{v\}}$, we have

$$\begin{aligned}
\hat{\rho}^{S\cup\{v\}}(V) &= \min_V \{\hat{r}_1^{S\cup\{v\}}(X) + \hat{r}_2^{S\cup\{v\}}(X) + n - |X|\} \\
&\leq \hat{r}_1^{S\cup\{v\}}(X^*) + \hat{r}_2^{S\cup\{v\}}(X^*) + n - |X^*| \\
&= \hat{r}_1^S(X^*) + \hat{r}_2^{S\cup\{v\}}(X^*) + n - |X^*| \\
&\leq \hat{r}_1^S(X^*) + \hat{r}_2^S(X^*) + 1 + n - |X^*| = \hat{\rho}^S(V) + 1 = \rho_1(S) + 1
\end{aligned}$$

Hence $\rho_1(S \cup \{v\}) \leq \rho_1(S) + 1$. On the other hand, for any set $X \subseteq V$,

$$\begin{aligned}
\hat{r}_1^{S\cup\{v\}}(X) + \hat{r}_2^{S\cup\{v\}}(X) + n - |X| &= \hat{r}_1^S(X) + \hat{r}_2^{S\cup\{v\}}(X) + n - |X| \\
&\geq \hat{r}_1^S(X) + \hat{r}_2^S(X) + n - |X|
\end{aligned}$$

and so $\rho_1(S \cup \{v\}) \geq \rho_1(S)$. Finally, suppose that $\rho_1(S \cup \{v\}) = \rho_1(S \cup \{w\}) = \rho_1(S)$. Let X be the set that minimizes $\hat{r}_1^{S\cup\{v\}}(X) + \hat{r}_2^{S\cup\{v\}}(X) + n - |X|$. We then have

$$\begin{aligned}
\hat{\rho}^{S\cup\{v,w\}}(X) &\leq \hat{r}_1^{S\cup\{v,w\}}(X) + \hat{r}_2^{S\cup\{v,w\}}(X) + n - |X| \\
&= \hat{r}_1^{S\cup\{v\}}(X) + \hat{r}_2^{S\cup\{v,w\}}(X) + n - |X| \\
&= \hat{r}_1^{S\cup\{v\}}(X) + \hat{r}_2^{S\cup\{v\}}(X) + n - |X| = \rho_1(S \cup \{v\})
\end{aligned}$$

Since $\rho_1(S)$ satisfies the three criteria of Lemma 2.9, it is a matroid rank function. □

We let $\mathcal{M}_1$ denote the matroid with rank function defined by $\rho_1(S)$ from Lemma 6.6. The following corollary allows us to express $\text{rank}([A|B]) = n$ as a matroid rank condition on the set of input nodes.

Corollary 6.1. *Let $\mathcal{M}_1$ be the matroid with rank function defined by $\rho_1(S)$ from Lemma 6.6. Then the rank condition rank$(A|B) = n$ holds for input set S if and only if $\rho_1(S) = 2n -$ rank$(\mathcal{M}([I|Q_A|0]) \vee \mathcal{M}([I|T_A|0]))$.*

Proof. By Lemma 6.2, the condition $\text{rank}([A|B]) = n$ is satisfied if and only if $\text{rank}(\mathcal{M}([I|Q_A|0]) \vee \mathcal{M}([I|T_A|T_B(S)])) = 2n$. Combining with the definition of $\rho_1(S)$ completes the proof. □

Finally, we express rank($[A|B]$) $= n$ as a matroid constraint on the set of non-input nodes.

Lemma 6.7. *The condition rank*$([A|B]) = n$ *holds if and only if the set of non-input nodes* R *satisfies* $R \in \mathcal{M}_1^*$*, the dual of the matroid induced by rank function* $\rho_1(S)$*.*

Proof. Let ρ_1^* denote the rank function of the dual matroid $\mathcal{M}_1^*$. We have $\rho_1^*(R) = \rho_1(V \setminus R) + |R| - \rho_1(V)$, which is equivalent to $\rho_1(S) = \rho_1^*(R) - |R| + \rho_1(V)$. The constraint of Corollary 6.1 is equivalent to $\rho_1(S) \geq \rho_1(V)$, which is in turn equivalent to $\rho_1^*(R) \geq |R|$. This, however, holds only when $R \in \mathcal{M}_1^*$. □

We now turn to the constraint that rank$(A - zF|B) = n$ (Eq. (6.6)). The following intermediate lemma is the first step in our approach.

Lemma 6.8. *There exists a graph* $G' = (V', E')$*, which can be constructed in polynomial time, such that the auxiliary graph* $\hat{G} = (\hat{V}, \hat{E})$ *corresponding to system (6.9) is given by*

$$\hat{V} = V' \cup \{u_1^T, \ldots, u_k^T\} \cup \{u_1^Q, \ldots, u_k^Q\} \tag{6.11}$$

$$\hat{E} = E' \cup \{(u_i^T, u_i^Q) : i = 1, \ldots, k\} \cup \{(w_i^T, u_i^T) : i \in S\} \tag{6.12}$$

In this graph, the set $J_1 = \{u_1^Q, \ldots, u_k^Q\}$ *and the condition of Lemma 6.4 is satisfied if each node of* V' *that belongs to a cycle in* G' *is reachable to a node in* $\{w_i^T : i \in S\}$ *in the graph* $\hat{G}$*.*

Proof. The matrix Ω corresponding to (6.9) is given by

$$\Omega = \begin{pmatrix} Q_A - Q_F & 0 \\ I & 0 \\ 0 & I \end{pmatrix}$$

By solvability of (6.9) and Lemma 6.3, we can select n linearly independent rows J from the first $2n$ rows of Ω. Let Ψ denote the matrix consisting of these linearly independent rows. The matrix Ω_J can be completed to a full-rank matrix by selecting $J_1 = \{u_1, \ldots, u_k\}$, giving

$$\Omega_{J \cup J_1} = \begin{pmatrix} \Psi & 0 \\ 0 & I \end{pmatrix}, \quad \Omega_{J \cup J_1}^{-1} = \begin{pmatrix} \Psi^{-1} & 0 \\ 0 & I \end{pmatrix}.$$

Note that the matrix $\Omega\Omega^{-1}_{J\cup J_1}$ does not depend on the input set S. Let G' denote the auxiliary graph when $S = \emptyset$. By construction, the auxiliary graph $\hat{G}$ with a non-empty input set S is given by (6.11) and (6.12), since adding nodes to S simply adds edges to $N(T_B)$.

To prove that reachability to the nodes $\{w_i^T : i \in S\}$ is sufficient, note that it suffices that each node is reachable to $J_1 = \{u_1^Q, \ldots, u_k^Q\}$ by Lemma 6.4. Since each node in $\{w_i^T : i \in S\}$ is reachable to J_1, it suffices for all other nodes to be reachable to $\{w_i^T : i \in S\}$. □

As a consequence of Lemmas 6.4 and 6.8, to ensure that $\text{rank}(zF - A|B) = n$, it suffices to select an input set S such that each node in G' is reachable to $\{w_i^T : i \in S\}$. In order to select such an input set, we define an equivalence relation $\sim$ on the nodes in V' as $i \sim j$ if node i is path-connected to node j in G' and vice versa. We let $[i] = \{j : i \sim j\}$, and define $\overline{V} = \{[i] : i \in \hat{V}\}$ (so that $\overline{V}$ is the quotient set of $\hat{V}$ under the relation $\sim$).

Define the graph $\overline{G} = (\overline{V}, \overline{E})$ by $(i, j) \in \overline{E}$ if there exists $i' \in [i]$ and $j' \in [j]$ such that $(i', j') \in E$. Note that $\overline{G}$ is a directed acyclic graph. We let $\overline{V}'$ denote the set of isolated nodes, i.e., nodes that have no incoming edges in $\overline{G}$.

Lemma 6.9. *All nodes in $\hat{V}$ are connected to S iff for each $[i] \in \overline{V}'$, $S \cap [i] \neq \emptyset$.*

Proof. We first show that if $S \cap [i] \neq \emptyset$ for all $[i] \in \overline{V}'$, then all nodes in $\hat{V}$ are connected to S. Let $v \in \hat{V}$. If $v \in [i]$ with $[i] \in \overline{V}'$, then there exists $j \in S \cap [i]$ such that j is path-connected to v. If $v \in [i]$ with $[i] \notin \overline{V}'$, then there are nodes $i' \in [i]$ and $j_1 \in [j_1]$ for some $[j_1] \in \overline{V}$ such that $(j_1, i') \in \hat{E}$. Now, either $[j_1] \in \overline{V}'$ or there exists $j_1' \in [j_1]$ and $j_2 \in [j_2]$ such that $(j_2, j_1') \in \hat{E}$. Since the graph $\overline{G}$ is acyclic, there exists a sequence of components $[i], [j_1], \ldots, [j_L]$, with $([j_{l+1}], [j_l]) \in \hat{E}$ and $[j_L] \in \overline{V}'$. This sequence of components defines a directed path from a node $v' \in S$ to v.

Now, suppose that all nodes in $\hat{V}$ are connected to S. For each $v \in [i]$, with $[i] \in \overline{V}'$, v must be connected to at least one input node. Since $[i] \in \overline{V}'$, only other nodes in $[i]$ are connected to v. Hence we must have $S \cap [i] \neq \emptyset$. □

Lemma 6.9 enables us to express the connectivity criterion as a matroid constraint. First, define a function $\rho_2(S) = |\{[i] \in \overline{V}' : [i] \cap S \neq \emptyset\}|$. The following lemma describes the

rank condition in terms of function $\rho_2(S)$.

Lemma 6.10. *Let $c = |\overline{V}'|$. The function $\rho_2(S)$ is a matroid rank function, and all nodes in $\hat{V}$ are connected to S iff $\rho_2(S) = c$.*

Proof. The function $\rho_2(S)$ is a matroid rank function because $\rho_2(\emptyset) = 0$ and $(\rho_2(S \cup \{v\}) - \rho_2(S)) \in \{0, 1\}$, with $\rho_2(S \cup \{v\}) - \rho_2(S) = 1$ iff there exists i such that $v \in [i]$ and $S \cap [i] = \emptyset$. Furthermore, $\rho_2(S) = c$ if and only if for every $[i] \in \overline{V}'$, $S \cap [i] \neq \emptyset$, which is exactly the condition of Lemma 6.9. □

Let $\mathcal{M}_2$ denote the matroid induced by rank function $\rho_2(S)$. We are now ready to state a sufficient matroid constraint on R for the condition (6.6).

Lemma 6.11. *Let $\mathcal{M}_2^*$ be the dual of the matroid induced by $\rho_2(S)$. If the non-input nodes R satisfy $R \in \mathcal{M}_2^*$, then the condition rank$((zF - A)|B) = n$ is satisfied.*

Proof. The rank function $\rho_2^*(R)$ of $\mathcal{M}_2^*$ can be written as $\rho_2^*(R) = \rho_2(V \setminus R) + |R| - \rho_2(V) = \rho_2(S) + |R| - c$, or equivalently, $\rho_2(S) = \rho_2^*(R) + c - |R|$. Hence $\rho_2(S) = c$ is equivalent to $\rho_2^*(R) = |R|$, which holds if and only if $R \in \mathcal{M}_2^*$. □

We combine the results of Lemmas 6.7 and 6.11 to yield the following theorem.

Theorem 6.3. *If $R \in \mathcal{M}_1^* \cap \mathcal{M}_2^*$, then the system is controllable from input set $S = V \setminus R$.*

Having defined matroid-based sufficient conditions for controllability, we will next formulate the problem of selecting the minimum-size input set to guarantee structural controllability.

6.2.2 Minimum-Size Input Set Selection for Structural Controllability

Selecting a minimum-size set S to satisfy structural controllability is equivalent to selecting a maximum-size set $R = V \setminus S$ that satisfies controllability. Based on Theorem 6.3, the problem of selecting the minimum-size set of input nodes to guarantee structural controllability can be formulated as

$$\text{maximize} \{|R| : R \in \mathcal{M}_1^*, R \in \mathcal{M}_2^*\}. \tag{6.13}$$

Lemma 6.12. *A minimum-size set of input nodes satisfying structural controllability can be obtained in polynomial-time.*

Proof. The problem of selecting a minimum-size set of input nodes to satisfy structural controllability is formulated as (6.13). Eq. (6.13) is a matroid intersection problem, which can be solved in time $O(n^{5/2}\tau)$, where τ is the time required to test if a set R is in $\mathcal{M}_1^*$ and $\mathcal{M}_2^*$ [120]. The independence of R in each set can be evaluated in polynomial time. □

Algorithm 13 gives a polynomial-time procedure for solving (6.13) using the maximum cardinality matroid intersection algorithm of [120, Ch. 41]. Lemmas 6.6 and 6.11 and the above discussion generalize the main result of [85] from free matrices to systems with a mix of free and fixed parameters.

6.2.3 Input Selection for Joint Performance and Controllability

We now consider the problem of maximizing a monotone performance metric $f(S)$ while satisfying controllability with a set of up to k input nodes. Based on Theorem 6.3, the problem formulation is given by

$$\begin{array}{ll} \text{maximize}_{S\subseteq V} & f(S) \\ \text{s.t.} & |S| \leq k \\ & (V \setminus S) \in \mathcal{M}_1^* \\ & (V \setminus S) \in \mathcal{M}_2^* \end{array} \tag{6.14}$$

Eq. (6.14) is a combinatorial optimization problem, making it NP-hard to solve in the general case. The following lemma describes an equivalent formulation to (6.14).

Lemma 6.13. *Define $r_1 = rank(\mathcal{M}_1)$ and $r_2 = rank(\mathcal{M}_2)$. Let $\hat{\mathcal{M}}_1 = \mathcal{M}_1 \vee U_{k-r_1}$ and $\hat{\mathcal{M}}_2 = \mathcal{M}_2 \vee U_{k-r_2}$, with $\hat{\mathcal{B}}_1$ and $\hat{\mathcal{B}}_2$ denoting the sets of bases of $\hat{\mathcal{M}}_1$ and $\hat{\mathcal{M}}_2$, respectively. Let S^* denote the optimal solution to the problem*

$$\begin{array}{ll} \textit{maximize}_{S\subseteq V} & f(S) \\ \textit{s.t.} & S \in \hat{\mathcal{B}}_1 \cap \hat{\mathcal{B}}_2 \end{array} \tag{6.15}$$

The set S^* is the optimal solution to (6.14).

Proof. First, we have that the optimal solution to (6.14) satisfies $|S| = k$. If not, then since $f(S)$ is monotone, we can add elements to S and increase the value of f without violating the constraints $(V \setminus S) \in \mathcal{M}_1^*$ and $(V \setminus S) \in \mathcal{M}_2^*$. Furthermore, if $R = (V \setminus S) \in \mathcal{M}_1^*$, then R can be completed to a basis $\overline{R}$ of $\mathcal{M}_1^*$, and we have $(V \setminus \overline{R}) \subseteq (V \setminus R)$, implying that $S = (V \setminus \overline{R}) \cup \overline{S}$ for some set $\overline{S}$ with $|\overline{S}| = k - r_1$. Since $\overline{R}$ is a basis of $\mathcal{M}_1^*$, $V \setminus \overline{R}$ is a basis of $\mathcal{M}_1$, and hence $S \in \hat{\mathcal{B}}_1$. A similar result holds for $\hat{\mathcal{M}}_2$.

Now, consider $S^* \in \hat{\mathcal{B}}_1 \cap \hat{\mathcal{B}}_2$. By construction $\text{rank}(\hat{\mathcal{M}}_1) = \text{rank}(\hat{\mathcal{M}}_2) = k$. We can therefore write $S^* = S_1^* \cup \hat{S}_1^*$, where S_1^* is a basis of $\mathcal{M}_1$ and $|\hat{S}_1^*| = k - r_1$. Since S_1^* is a basis of $\mathcal{M}_1$, we have $(V \setminus S_1^*) \in \mathcal{M}_1^*$, and thus the set $R = (V \setminus S^*) \subseteq (V \setminus S_1^*) \in \mathcal{M}_1^*$, implying that $R = (V \setminus S^*) \in \mathcal{M}_1^*$. A similar argument implies that $(V \setminus S^*) \in \mathcal{M}_2^*$. □

By Lemma 6.13, solving (6.14) is equivalent to solving (6.15). We present a two-stage algorithm for approximating (6.15). In the first stage, the algorithm solves a relaxed, continuous version of the problem. In the second stage, the algorithm rounds the solution to an integral value satisfying the constraints of (6.15).

The algorithm is defined as Algorithm 14 below. It contains two subroutines, namely, MAX_WEIGHTED_BASIS and SWAP_ROUND. The subroutine MAX_WEIGHTED_BASIS takes as input two matroids $\mathcal{M}'$ and $\mathcal{M}''$ with the same ground set V (with $|V| = n$), as well as a weight vector $\boldsymbol{\alpha} \in \mathbb{R}^n$, and outputs a set $I \in \mathcal{B}(\mathcal{M}') \cap \mathcal{B}(\mathcal{M}'')$ such that $\sum_{i \in I} \alpha_i$ is maximized (provided at least one common basis exists). Polynomial time algorithms for finding such sets are well-known [120, Ch. 43].

The subroutine SWAP_ROUND takes as input a vector $\mathbf{r}$ in the common base polytope of two matroids $\mathcal{M}'$ and $\mathcal{M}''$, and outputs a set $I \in \mathcal{B}(\mathcal{M}') \cap \mathcal{B}(\mathcal{M}'')$. The algorithm is randomized with the output satisfying $\mathbf{E}(f(I)) \geq F(\mathbf{r})$. The swap round algorithm was proposed in [31].

In Algorithm 14, the $\mathbf{1}(I(t))$ denotes the incidence vector of set $I(t)$, which has a 1 in the i-th entry if $i \in I$ and 0 otherwise. The following theorem describes the optimality bound of Algorithm 14.

Theorem 6.4. *Algorithm 14 runs in polynomial time with complexity $O(\tau n^5)$. Letting S^* denote the optimal solution to (6.14), the vector $\mathbf{y}(1)$ returned by the continuous relaxation satisfies $F(\mathbf{y}(1)) \geq (1 - 1/e)f(S^*)$, where F is the multilinear relaxation of $f(S)$. The rounded solution S is a feasible solution to (6.14).*

In order to prove Theorem 6.4, we first prove a sequence of lemmas, which will establish that $F(\mathbf{y}(1)) \geq (1 - 1/e)f(S^*)$. The lemmas follow Lemmas 3.1–3.3 of [22], however, the results of [22] are for a single matroid constraint, instead of two matroid basis constraints as in Theorem 6.4.

Lemma 6.14. *Let $y \in [0, 1]^n$ and let $R \subseteq V$ denote a random set such that $j \in R$ with probability y_j. Then*

$$f(S^*) \leq F(y) + \max_{I \in \hat{\mathcal{B}}_1 \cap \hat{\mathcal{B}}_2} \sum_{j \in I} \mathbf{E}(f_R(j)),$$

where $f_R(j) = f(R \cup \{j\}) - f(R)$.

Proof. By submodularity, we have that $f(S^*) \leq f(R) + \sum_{j \in S^*} f_R(j)$. Taking expectation over R yields

$$f(S^*) \leq \mathbf{E}(f(R)) + \sum_{j \in S^*} \mathbf{E}(f_R(j)) \leq F(y) + \max_{I \in \mathcal{I}} \sum_{j \in I} \mathbf{E}(f_R(j)),$$

as desired. □

Lemma 6.14 establishes an optimality result for an idealized version of Algorithm 14 where computation is performed over the actual values of $\mathbf{E}(f_R(j))$ instead of estimates computed via random sampling. The following lemma introduces bounds on the errors introduced by sampling.

Lemma 6.15. *With high probability, at each time t the algorithm finds a set $I(t)$ such that*

$$\sum_{j \in I(t)} \mathbf{E}(f_{R(t)}(j)) \geq (1 - 2k\delta)f(S^*) - F(y(t)).$$

The proof is identical to Lemma 3.2 of [22] and is omitted. Finally, we prove the optimality bound on the solution to the continuous relaxation.

Lemma 6.16. *With high probability, the fractional solution $y(1)$ found by solving the continuous problem satisfies*

$$F(y) \geq \left(1 - \frac{1}{e} - \frac{1}{3d}\right) f(S^*).$$

The proof follows that of Lemma 3.3 of [22]. We now prove Theorem 6.4.

Proof of Theorem 6.4. The fact that $F(y) \geq \left(1 - \frac{1}{3}\right) f(S^*)$ follows directly from Lemma 6.16. The feasibility of S follows from the SWAP_ROUND algorithm, which preserves membership in the bases of both matroids at each iteration and returns an integral solution, corresponding to a common basis of $\hat{\mathcal{B}}_1$ and $\hat{\mathcal{B}}_2$. The complexity of the procedure is dominated by the solution of the continuous problem, which is in turn determined by the cost of computing the maximum weight matroid intersection at each iteration. Since there are $O(n^2)$ iterations and the maximum weight matroid intersection has complexity $O(\tau n^3)$ [120], where τ is the cost of testing independence in $\hat{\mathcal{M}}_1$ and $\hat{\mathcal{M}}_2$, the overall complexity is $O(\tau n^5)$. □

The following theorem provides additional optimality guarantees when the objective function $f(S)$ is linear.

Theorem 6.5. *If the function $f(S)$ is of the form $f(S) = \sum_{i \in S} \tau_i$ for some real-valued weights $\tau_1, \ldots, \tau_n$, then the solution to (6.14) can be obtained in polynomial time.*

Proof. If the function $f(S)$ is of the form $f(S) = \sum_{i \in S} \tau_i$, then (6.14) is equivalent to maximizing a modular function subject to two matroid basis constraints. For problems of this form, the Edmonds weighted matroid intersection algorithm provides an optimal solution in polynomial time [120]. □

6.2.4 Selecting Input Nodes for Performance-Controllability Trade-Off

In this section, we study input selection based on a trade-off between performance and controllability, instead of treating controllability as a constraint that must be satisfied. This maximization may be beneficial when the number of input nodes k is insufficient to guarantee controllability.

We first introduce two graph controllability indices (GCIs) that can be traded off with a performance metric in order to maximize the level of performance and controllability. The first controllability index $c_1(S)$ is given by

$$c_1(S) = \max\{|V'| : \text{rank}(A(V')|B(V')) = |V'|\}, \tag{6.16}$$

where $A(V')$ and $B(V')$ are sub-matrices of A and B consisting of the rows and columns indexed in V'. Intuitively, $c_1(S)$ is the size of the largest subgraph of V such that the zero modes of all nodes in the subgraph are controllable. The second GCI quantifies the controllability of the nonzero modes, characterized by the constraint $\text{rank}(zF - A|B) = n$. We define $c_2(S)$ by

$$c_2(S) = |\{i \in V : i \text{ is reachable to } \mathbf{u} \text{ in } \hat{G}\}| \tag{6.17}$$

where $\hat{G}$ is defined as in Section 6.1.3. The function $c_2(S)$ quantifies the number of nodes that are reachable to the input nodes, and hence satisfy controllability of the nonzero modes. If $c_1(S) + c_2(S) = 2n$, then both the zero and nonzero modes of all nodes are controllable, and hence controllability is satisfied. Otherwise, the problem of joint maximization of performance and controllability can be formulated as

$$\begin{array}{ll} \text{maximize}_{S \subseteq V} & f(S) + \eta(c_1(S) + c_2(S)) \\ \text{s.t.} & |S| \leq k \end{array} \tag{6.18}$$

The trade-off parameter $\eta \geq 0$ is used to vary the relative weight assigned to performance or controllability criteria. When η is small, then nodes are selected for performance alone; at the other extreme, when η is large, nodes are primarily selected to maximize controllability. The following result is the first step in deriving efficient algorithms for solving (6.18).

Theorem 6.6. *The functions $c_1(S)$ and $c_2(S)$ are submodular as functions of S.*

Proof. The function $c_1(S)$ is equal to the maximum-size set of non-input nodes with controllable zero modes from the input nodes, plus the number of input nodes. This can be written as $c_1(S) = \rho_1(V \setminus S) + |S|$, where ρ_1 is defined as in Lemma 6.6. Since ρ_1 is a matroid rank function, ρ_1 is submodular and hence $\rho_1(V \setminus S)$ is submodular as well by Lemma 2.6. Since the sum of submodular functions is submodular, $\rho_1(V \setminus S) + |S|$ is submodular as a function of S.

It remains to show submodularity of $c_2(S)$. Let $S \subseteq T$, and suppose that $v \notin T$. We have that $c_2(T \cup \{v\}) - c_2(T)$ is equal to the number of nodes that are reachable to v, but not to any node in T. Since $S \subseteq T$, any node that is not reachable to T is automatically not reachable to S. Hence, any node that is reachable to v but not any node in T is also reachable to v but not any node in S, implying that $c_2(T \cup \{v\}) - c_2(T) \geq c_2(S \cup \{v\}) - c_2(S)$. □

A greedy algorithm for approximating (6.18) is as follows. The set S is initialized to be empty, and the algorithm proceeds over k iterations. At the i-th iteration, the element $v \in V$ maximizing $f(S \cup \{v\}) + \eta(c_1(S \cup \{v\}) + c_2(S \cup \{v\})$ is selected and added to S, terminating after k iterations. The following theorem gives an optimality bound for this algorithm.

Theorem 6.7. *The solution S obtained by the greedy algorithm satisfies $(f(S) + \eta(c_1(S) + c_2(S))) \geq (1 - 1/e)(f(S^*) + \eta(c_1(S^*) + c_2(S^*)))$, where S^* is the optimal solution to (6.18).*

Proof. Since $f(S)$, $c_1(S)$, and $c_2(S)$ are submodular and monotone as functions of S, the function $f(S) + \eta(c_1(S) + c_2(S))$ is monotone and submodular. Hence, Theorem 4.1 of [96] implies that the greedy algorithm returns a set S satisfying a $(1 - 1/e)$-optimality bound with the optimal set S^*, thus completing the proof. □

6.3 Input Selection in Strongly Connected Networks

The input selection algorithms of the previous section hold for any arbitrary structured linear descriptor system. In this section, we investigate the case where the graph induced by the system matrices F, A, and B is strongly connected, i.e., there exists a directed path from any node i to any node j. In this case, there is additional problem structure that reduces the complexity and improves the optimality bounds of our input selection algorithms. The following lemma gives system properties that hold with high probability for strongly connected networks.

Lemma 6.17. *If the graph induced by (F, A, B) is strongly connected and $\mathrm{rank}(A|B) = n$, then the condition $\mathrm{rank}((zF - A)|B) = n$ holds for almost any fixed parameter matrices Q_F, Q_A, and Q_B.*

Proof. If $\text{rank}(A|B) = n$, then $\det(zF - A) \neq 0$ for all $z \in \mathbb{C}$, except for some complex numbers $z_1, \ldots, z_n$. Consider z_i such that $\det(z_i F - A) = 0$. We have that

$$\det(z_i F - A) = \sum_{\sigma \in S_n} \prod_{j=1}^{n} (z_i F - A)_{j\sigma(j)}.$$

We observe that each σ corresponding to a nonzero term of the summation induces a decomposition of the graph into cycles $j_1, \ldots, j_m$, in which $j_{l+1} = \sigma(j_l)$ and $j_1 = \sigma(j_m)$. If the determinant is zero, then there exist at least two such decompositions, corresponding to distinct permutations σ and σ', with products of weights that sum to zero.

Suppose that the l-th column of the matrix $(z_i F - A)$ is linearly dependent on the other columns. Suppose that the column is replaced by one of the input columns from B. Since the graph is strongly connected, a new cycle is induced by adding the input column. For almost all values of the free parameters of B, the cycle will not cancel out with the other cycles in the graph, and hence the determinant will be nonzero. □

If $\text{rank}((zF - A)|B) = n$, then only the constraint $(V \setminus S) \in \mathcal{M}_1^*$ must hold. We now describe how this additional problem structure improves the runtime and optimality gaps of each of the input selection problems considered.

6.3.1 Minimum-Size Input Set Selection in Strongly Connected Networks

In the strongly connected case, the minimum-size input set selection problem reduces to

$$\begin{aligned} &\text{maximize} && |R| \\ &\text{s.t.} && R \in \mathcal{M}_1^* \end{aligned} \tag{6.19}$$

The following algorithm can be used to compute the solution to (6.19). Initialize the set $R = \emptyset$, and let $V = \{1, \ldots, n\}$ be the set of possible input nodes. The algorithm iterates over all nodes in V, starting with the node indexed 1. For each node i, test if $(R \cup \{i\}) \in \mathcal{M}_1^*$. If so, set $R = R \cup \{i\}$. The algorithm terminates after all n nodes have been tested.

Lemma 6.18. *When the network is strongly connected and the condition of Lemma 6.17 holds, the greedy algorithm returns the minimum-size input set to guarantee controllability within $O(n)$ computations of the matroid independence condition $R \in \mathcal{M}_1^*$.*

Proof. Let $\mathcal{B}$ be the set of bases of $\mathcal{M}_1^*$. We define a lexicographic ordering on the sets $R \in \mathcal{B}$ as follows. Let R_1, $R_2 \in \mathcal{B}$, and write $R_1 = \{a_1, \ldots, a_m\}$ and $R_2 = \{b_1, \ldots, b_m\}$, where $a_1 < a_2 < \cdots < a_m$ and $b_1 < b_2 < \cdots < b_m$. We let $R_1 \prec R_2$ if there exists i such that $a_j = b_j$ for $j < i$ and $a_i < b_i$. We have that $\prec$ induces a total ordering on $\mathcal{B}$, since for any R_1 and R_2 with $R_1 \neq R_2$, we have $R_1 \prec R_2$ or $R_2 \prec R_1$.

Let R^* denote the set in $\mathcal{B}$ that is minimal under the ordering $\prec$. We show that the algorithm described above outputs R^*. Let $R^* = \{a_1, \ldots, a_m\}$, and let $R_i^* = R^* \cap \{1, \ldots, i\}$. Finally, let R_i denote the set computed by our algorithm at iteration i. We prove by induction that $R_i^* = R_i$ for each i.

We have $R_0 = R_0^*$ trivially. Now, suppose $R_i = R_{i-1}^*$. We have two cases. First, suppose that $i \in R^*$. Since $R_{i-1} \cup \{i\} = R_{i-1}^* \cup \{i\} = R_i^*$ and $R_i^* \subseteq R^*$, we have that $(R_{i-1} \cup \{i\}) \in \mathcal{M}_1^*$. Hence the algorithm will add i to the set, and $R_i = R_i^*$.

Now, suppose that $i \notin R^*$, and suppose that $R_i^* \neq R_i$. By inductive hypothesis, we must have that $i \in R_i$, which occurs if and only if $R_i = (R_{i-1} \cup \{i\})$ is independent in $\mathcal{M}_1^*$. Since $\mathcal{M}_1^*$ is a matroid, we can complete R_i to a basis $\hat{R} \in \mathcal{B}$. By definition of the $\prec$ ordering, $\hat{R} \prec R^*$, contradicting the assumption that R^* is minimal under the ordering $\prec$. This contradiction implies that $i \notin R_i$, and so $R_i = R_i^*$.

Continuing inductively until $i = n$, we have that $R_n = R_n^* = R^*$. Since R_n is equal to $R^* \in \mathcal{B}$, R_n is a basis of $\mathcal{M}_1^*$, and hence is a solution to (6.19). The $O(n)$ runtime follows from the fact that the algorithm makes one independence check per iteration over a total of n iterations. □

The additional problem structure in the strongly connected case leads to a simplified algorithm with reduced runtime compared to Algorithm 13.

6.3.2 Joint Performance and Controllability Input Selection in Strongly Connected Networks

When the system graph is strongly connected and the conditions of Lemma 6.17 hold, the complexity and optimality bounds of input selection for joint performance and controllability

are improved. With this additional structure, the problem formulation is given by

$$\begin{aligned} \text{maximize} \quad & f(S) \\ \text{s.t.} \quad & |S| \leq k \\ & (V \setminus S) \in \mathcal{M}_1^* \end{aligned} \tag{6.20}$$

The following lemma gives an equivalent formulation to (6.20).

Lemma 6.19. *Let $r_1 = rank(\mathcal{M}_1)$ and define $\hat{\mathcal{M}}_1 = \mathcal{M}_1 \vee U_{k-r_1}$. If the objective function $f(S)$ is monotone, then the optimization problem (6.20) has the same solution as*

$$\begin{aligned} \textit{maximize} \quad & f(S) \\ \textit{s.t.} \quad & S \in \hat{\mathcal{M}}_1 \end{aligned} \tag{6.21}$$

Proof. Let S^* and $\hat{S}$ denote the optimal solutions to (6.20) and (6.21), respectively. We observe that both $|S^*| = |\hat{S}| = k$ by monotonicity of $f(S)$. We show that if $|S| = k$, then the conditions $(V \setminus S) \in \mathcal{M}_1^*$ and $S \in \hat{\mathcal{M}}_1$ are equivalent. First, suppose that $(V \setminus S) \in \mathcal{M}_1^*$. Let $R^* = (V \setminus S) \in \mathcal{M}_1^*$. The set R^* can be completed to a basis of $\mathcal{M}_1^*$, denoted $\hat{R} = R^* \cup R'$, and so we have $S = (V \setminus \hat{R}) \cup (V \setminus R')$. Now, $(V \setminus \hat{R}) \in \mathcal{M}_1$ and $|V \setminus R'| = k - r_1$, and so $S \in \hat{\mathcal{M}}_1$.

Suppose that $S \in \hat{\mathcal{M}}_1$ and $|S| = k$. Since $|S| = k$, S is a basis of $\hat{\mathcal{M}}_1$, and so S can be written as $S = S_1 \cup S_2$ where $S_1 \in \mathcal{M}_1$. Hence $(V \setminus S) \subseteq (V \setminus S_1) \in \mathcal{M}_1^*$, and so the second constraint of (6.20) is satisfied. □

Convex relaxation approaches have been proposed for solving matroid-constrained monotone submodular maximization problems [22, 31]. One such approach is to replace Line 12 in Algorithm 14 with a subroutine that computes the maximum-weighted basis of a matroid (such a basis can be computed efficiently using a greedy algorithm). It was shown in [22] that this algorithm achieves a $(1 - 1/e)$ optimality bound.

6.3.3 Performance-Controllability Trade-Off in Strongly Connected Networks

In the strongly connected network case, we define the graph controllability index (GCI)

$$c(S) = \max \{|V'| : V' \text{ controllable from } S\}.$$

The following lemma provides additional structure on $c(S)$.

Lemma 6.20. *The function $c(S) = \tilde{c}(S) + \zeta$, where ζ is a constant and $\tilde{c}(S)$ is a matroid rank function.*

Proof. The largest controllable subgraph of G corresponds to a subset of states such that, for the matrix A' with columns indexed in V', we have $\text{rank}(\mathcal{M}([I|Q_{A'}|0] \vee [I|T_{A'}|T_B(S)]) = 2|V'|$. This subset of columns, however, is exactly the maximum-size independent set in $(\mathcal{M}([I|Q_A|0]) \vee \mathcal{M}([I|T_A|T_B(S)]))$, and the value of $c(S)$ is $\text{rank}(\mathcal{M}([I|Q_A|0]) \vee \mathcal{M}([I|T_A|T_B(S)]))$. By Lemma 6.6, the rank is equal to the rank of $\mathcal{M}([I|Q_A]) \vee \mathcal{M}([I|T_A])$ plus a matroid rank function of S. □

The problem of selecting a set of up to k input nodes to maximize both a performance metric $f(S)$ and the GCI $c(S)$ is formulated as

$$\begin{aligned} &\text{maximize} && f(S) + \eta c(S) \\ &\text{s.t.} && |S| \leq k \end{aligned} \tag{6.22}$$

As in the general case, a greedy algorithm for maximizing $f(S) + \eta c(S)$ is guaranteed to return an input set S^* such that $f(S^*) + \eta c(S^*)$ is within a $(1 - 1/e)$ factor of the optimum. Moreover, when the performance metric $f(S)$ is identically zero, so that only controllability is optimized, we have the following result.

Lemma 6.21. *If $f(S) = 0$, then the greedy algorithm returns the optimal solution to (6.22).*

Proof. For the problem of maximizing a matroid rank function subject to a cardinality constraint, the greedy algorithm is known to return an optimal solution [104]. If $f(S) = 0$, then by Lemma 6.20, Eq. (6.22) is equivalent to maximizing a matroid rank function subject to a cardinality constraint, and hence the greedy algorithm returns the optimal input set S. □

6.4 Special Cases of Our Approach

In this section, we consider three special cases of our framework, namely systems with linear consensus dynamics, networked systems where each node has second integrator dynamics, and systems where all parameters are free.

6.4.1 Linear Consensus Dynamics

We first consider a network of N nodes where each node $i \in \{1, \ldots, N\}$ has a state $x_i(t) \in \mathbb{R}$. The state dynamics of the non-input nodes are given by $\dot{x}_i(t) = -\sum_{j \in N(i)} W_{ij}(x_i(t) - x_j(t))$, where W_{ij} are nonnegative weights. In [55], it was shown that, by introducing a set of states $\{x_j^e : j = 1, \ldots, M\}$, where M is the number of edges in the network, the system can be written in the form (6.3) as

$$\begin{pmatrix} I & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \dot{\mathbf{x}}(t) \\ \dot{\mathbf{x}}^e(t) \end{pmatrix} = \begin{pmatrix} 0 & K \\ K_I & W \end{pmatrix} \begin{pmatrix} \mathbf{x}(t) \\ \mathbf{x}^e(t) \end{pmatrix}. \tag{6.23}$$

In (6.23), K_I is the incidence matrix of the graph and K is the transpose of the incidence matrix. W is a diagonal matrix with e-th entry equal to the weight on edge e. We assume that the weights W are free parameters, so that Q_F, T_F, Q_A, and T_A are given by

$$Q_F = \begin{pmatrix} I & 0 \\ 0 & 0 \end{pmatrix}, \quad Q_A = \begin{pmatrix} 0 & K \\ K_I & 0 \end{pmatrix}, \quad T_A = \begin{pmatrix} 0 & 0 \\ 0 & W \end{pmatrix}. \tag{6.24}$$

As a first step towards analyzing this class of system dynamics under our framework, we consider the condition of Lemma 6.4. For this system, the matrix Ω of Section 6.1.3 is equal to

$$\Omega = \begin{array}{c} \mathbf{w} \\ \mathbf{w}^e \\ \mathbf{x} \\ \mathbf{x}^e \\ \mathbf{u} \end{array} \begin{pmatrix} -I & K & 0 \\ K_I & 0 & 0 \\ I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{pmatrix} \tag{6.25}$$

We use $\mathbf{w}^e$ and $\mathbf{x}^e$ to denote the auxiliary graph nodes corresponding to the states $\mathbf{x}^e(t)$. We observe that in the augmented graph, since the weight matrix is diagonal, there is a directed edge from $w_j^{e,T}$ to $x_j^{e,T}$ for all edges indexed $j = 1, \ldots, M$. We have the following intermediate result.

Lemma 6.22. *The matching m with $m(w_j^{e,T}) = x_j^{e,T}$ for all $j = 1, \ldots, M$ and $m(w_i^T) = w_i^Q$ for all $i = 1, \ldots, N$ satisfies the conditions of Lemma 6.3.*

Proof. The matching m is valid under the construction of Section 6.1.3. The rows of Ω from (6.25) indexed in $J = \{x_j^{e,Q} : j = 1, \ldots, M\} \cup \{w_i^Q : i = 1, \ldots, N\}$ form the matrix

$$\Omega_J = \begin{pmatrix} -I & K \\ 0 & I \end{pmatrix},$$

which has full rank. □

We can then compute $\Omega\Omega_{J\cup J_1}^{-1}$ as

$$\Omega\Omega_{J\cup J_1}^{-1} = \begin{pmatrix} I & 0 & 0 \\ -K_I & K_I K & 0 \\ -I & K & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{pmatrix} \tag{6.26}$$

The graph $G' = (V', E')$ of Lemma 6.8 defined by Eq. (6.26) is described by the following lemma.

Lemma 6.23. *For the system (6.23), the edge set E' of Lemma 6.8 is defined by*

$$\begin{aligned} E' \;=\; & \{(w_e^Q, w_i^Q) : e = (i,j) \text{ for some } j \in V\} \cup \{(x_i^Q, x_e^Q) : e = (i,j) \text{for some } j \in V\} \\ & \cup \{(w_e^Q, x_{e'}^Q) : \text{edges } e \text{ and } e' \text{ have a node in common}\} \\ & \cup \{(x_j^{e,Q}, x_j^{e,T}) : j = 1, \ldots, M\} \cup \{(w_j^{e,T}, w_j^{e,Q}) : j = 1, \ldots, M\} \\ & \cup \{(w_i^Q, w_i^T) : i = 1, \ldots, N\}. \end{aligned}$$

Proof. Considering the edge set of the auxiliary graph defined in Section 6.1.3, the set $\{(w_i^T, x_j^Q) : (i,j) \in N(T_A) \cup N(T_F), m(w_i^T) \neq x_j^Q\}$ is empty, while the set $\{(x_j^Q, w_i^T) : (i,j) \in N(T_A) \cup N(T_F), m(w_i^T) = x_j^Q\}$ is equal to $\{(x_j^{e,Q}, x_j^{e,T}) : j = 1, \ldots, M\} \cup \{(w_j^{e,T}, w_j^{e,Q}) : j = 1, \ldots, M\}$. The set $\{(w_i^T, w_i^Q) : m(w_i^T) \neq w_i^Q\} \cup \{(w_i^Q, w_i^T) : m(w_i^T) = m(w_i^Q)\}$ is equal to $\{(w_j^{e,T}, w_j^{e,Q}) : j = 1, \ldots, M\} \cup \{(w_i^Q, w_i^T) : i = 1, \ldots, N\}$.

It remains to compute the set $\{(x^Q, y^Q) : x \in \hat{V} \setminus J, y \in J, \tilde{\Omega}_{xy} \neq 0, \tilde{\Omega}_{xz} = 0 \; \forall z \in J_1\}$. This set is defined by the off-diagonal entries of $\Omega\Omega_{J\cup J_1}^{-1}$ from (6.26). The entries from $w_j^{e,Q}$ to w_i^Q correspond to the entries of the incidence matrix, and hence there is a nonzero entry

if and only if edge j is incident to node i. A similar argument holds for the $(x_i^Q, x_j^{e,Q})$ edges. Finally, an edge $(w_j^{e,Q}, w_{j'}^{e,Q})$ is formed if $(K_I K)_{jj'} \neq 0$. This matrix is the *edge Laplacian*, which has a nonzero entry if and only if either $j = j'$, or edges j and j' are incident to the same node. □

This description of the graph G' enables characterization of the input-connected nodes in $\hat{V}$.

Lemma 6.24. *The nodes x_i^T, w_i^T and w_i^Q do not belong to any cycle in G'. A node x_i^Q is input-connected in G' if and only if i is input-connected in the graph G induced by the consensus dynamics. A node $x_j^{e,T}$, $w_j^{e,T}$, $x_j^{e,Q}$, or $w_j^{e,Q}$ is input-connected in G' if and only if edge j is incident on a node that is input-connected in the consensus network G.*

Proof. By Lemma 6.23, w_i^T and w_i^Q are only connected to each other, and hence are not part of any cycle since the link is directional. Similarly, the nodes x_i^T cannot belong to any cycle, since they have no incoming edges.

Now, suppose that there is a path from node i to an input i' in G. Let (i, i_1), ..., (i_r, i') denote one such path, and let $j_0, \ldots, j_r$ denote the indices of the edges on the path. Then there is a path π from x_i^Q to $w_{i'}^T$, given by

$$\pi = (x_i^Q, x_{j_0}^{e,Q}) \cup \bigcup_{l=0}^{r-1} \left\{ (x_{j_l}^{e,Q}, x_{j_l}^{e,T}), (x_{j_l}^{e,T}, w_{j_l}^{e,T}), (w_{j_l}^{e,T}, w_{j_l}^{e,Q}), (w_{j_l}^{e,Q}, x_{j_{l+1}}^{e,Q}) \right\} \cup \{ (x_{j_r}^{e,Q}, w_{i'}^Q), (w_{i'}^Q, w_{i'}^T) \} \quad (6.27)$$

For the other direction, we have that any path from x_i^Q to $w_{i'}^T$ has the form of (6.27), and hence defines a path from i to i' in G. Finally, suppose that edge j is incident on node i and that there is a path from a node i to an input i' in G. Let $(i, i_1), \ldots, (i_r, i')$ denote one such path, and let $j_0, \ldots, j_r$ denote the indices of the edges on the path. For node $w_j^{e,Q}$, there is a path given by

$$\pi' = (w_j^{e,Q}, x_{j_0}^{e,Q}) \cup \bigcup_{l=0}^{r-1} \{ (x_{j_l}^{e,Q}, x_{j_l}^{e,T}), (x_{j_l}^{e,T}, w_{j_l}^{e,T}), (w_{j_l}^{e,T}, w_{j_l}^{e,Q}), (w_{j_l}^{e,Q}, x_{j_{l+1}}^{e,Q}) \} \cup \{ (x_{j_r}^{e,Q}, w_{i'}^Q), (w_{i'}^Q, w_{i'}^T) \}$$

A path can also be found for nodes $x_j^{e,Q}$, $x_j^{e,T}$, and $w_j^{e,T}$ by using the path $(x_j^{e,Q}, x_j^{e,T})$, $(x_j^{e,T}, w_j^{e,T})$, $(w_j^{e,T}, w_j^{e,Q})$. □

Based on Lemma 6.24, we can characterize exactly when the condition of Lemma 6.4 holds, based on the connectivity of the network graph G.

Lemma 6.25. *The condition of Lemma 6.4 holds for the system (6.23) if and only if each node is input-connected in the graph G.*

The proof follows directly from Lemma 6.24. Lemma 6.25 enables us to improve the optimality bounds for a class of metrics with a certain structure. Suppose that the connected components of the graph are equal to $G_1, \ldots, G_r$, with $G_i = (V_i, E_i)$ for $i = 0, \ldots, r$. We consider metrics of the form

$$f(S) = \sum_{i=1}^{r} f_i(S \cap V_i). \tag{6.28}$$

Eq. (6.28) has the interpretation that the performance of nodes in connected component V_i only depends on the set of input nodes for component V_i, instead of the overall input set. This structure holds for, e.g., the metrics of [108, 38, 34]. For these metrics, we have the following optimality result.

Theorem 6.8. *For the consensus system (6.23), the problem of maximizing a performance metric of the form (6.28) subject to controllability as a constraint and $|S| \leq k$, formulated as*

$$\begin{aligned} &\textit{maximize} && f(S) \\ &\textit{s.t.} && (V \setminus S) \in \mathcal{M}_1^* \cap \mathcal{M}_2^* \\ & && |S| \leq k \end{aligned} \tag{6.29}$$

can be approximated up to an optimality bound of $(1-1/e)$ in polynomial time. As a special case, if the graph G is strongly connected, then any monotone submodular performance metric can be approximated up to an optimality bound of $(1-1/e)$ in polynomial time.

Proof. The proof is by showing that the optimality bounds of Algorithm 14 are improved in this case. By Theorem 6.4, the continuous relaxation phase of Algorithm 14 returns a vector $y(1)$ such that $F(y(1)) \geq (1-1/e)f(S^*)$, where S^* is the optimal solution to (6.29). Now,

Theorem II.3 of [31] implies that the SWAP_ROUND subroutine satisfies $\mathbf{E}(f(S \cap Q)) \geq F(x_i : i \in Q)$ for any set Q of equivalent elements of one of the matroids $\hat{\mathcal{M}}_1$ or $\hat{\mathcal{M}}_2$. For the matroid $\hat{\mathcal{M}}_2$, each set of elements V_s is equivalent, and so $\mathbf{E}(f(S \cap V_s)) \geq F(x_i : i \in V_s)$. Summing over s yields the desired result.

For the special case, we have that when the graph is strongly connected, input connectivity holds provided there is at least one input. Hence we can obtain a $(1 - 1/e)$-bound on the optimal input set. □

6.4.2 Double Integrator Dynamics

We study networked systems where the second derivative of each node's state $\xi_i(t)$ is a function of its neighbor states, so that $\ddot{\xi}_i(t) = \sum_{j \in N(i)} W_{ij}\xi_j(t) + \Gamma_{ij}\dot{\xi}_j(t)$, where $N(i)$ is the neighbor set of node i. The double integrator model is applicable for larger vehicles that have inertial components in their dynamics [114]. We write this system in the form (6.3) by introducing variables $\zeta_i(t) = \dot{\xi}_i(t)$, resulting in dynamics

$$\begin{pmatrix} \dot{\xi}(t) \\ \dot{\zeta}(t) \end{pmatrix} = \begin{pmatrix} 0 & I \\ W & \Gamma \end{pmatrix} \begin{pmatrix} \xi(t) \\ \zeta(t) \end{pmatrix}. \tag{6.30}$$

where $F = I$. In analyzing this system, we observe that it is not possible to independently control the states $\xi_i(t)$ and $\zeta_i(t)$ to any arbitrary trajectories, since $\zeta_i(t) = \dot{\xi}_i(t)$. Hence we assume that the state $\zeta_i(t)$ (the velocity) is controlled in the input nodes, while the state $\xi_i(t)$ continues to follow the dynamics (6.30).

We first investigate the auxiliary graph condition of Lemma 6.4. The matrix Ω is given by

$$\Omega = \begin{array}{c} \\ \mathbf{w}^{\xi} \\ \mathbf{w}^{\zeta} \\ \mathbf{x}^{\xi} \\ \mathbf{x}^{\zeta} \\ \mathbf{u} \end{array} \begin{pmatrix} -I & I & 0 \\ 0 & -I & 0 \\ I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{pmatrix} \tag{6.31}$$

We have that the rows indexed in $\mathbf{w}$ have full rank, and hence the matching $m(w_i^{\xi,T}) = w_i^{\xi,Q}$ and $m(w_i^{\zeta,T}) = w_i^{\zeta,Q}$ for $i = 1, \ldots, N$ satisfies the conditions of Lemma 6.3. This gives

$J = \{w_i^{\zeta,Q} : i = 1, \ldots, N\} \cup \{w_i^{\xi,Q} : i = 1, \ldots, N\}$. We then have $\Omega_{J\cup J_1}$, $\Omega^{-1}_{J\cup J_1}$, and $\Omega\Omega^{-1}_{J\cup J_1}$ as

$$\Omega_{J\cup J_1} = \begin{pmatrix} -I & I & 0 \\ 0 & -I & 0 \\ 0 & 0 & I \end{pmatrix}, \quad \Omega^{-1}_{J\cup J_1} = \begin{pmatrix} -I & -I & 0 \\ 0 & -I & 0 \\ 0 & 0 & I \end{pmatrix},$$

$$\Omega\Omega^{-1}_{J\cup J_1} = \begin{array}{c} \\ \mathbf{w}_\xi \\ \mathbf{w}_\zeta \\ \mathbf{x}^\xi \\ \mathbf{x}^\zeta \\ \mathbf{u} \end{array} \begin{pmatrix} I & 0 & 0 \\ 0 & I & 0 \\ -I & -I & 0 \\ 0 & -I & 0 \\ 0 & 0 & I \end{pmatrix}. \quad (6.32)$$

These matrix values lead to the following description of the auxiliary graph.

Lemma 6.26. *The graph G' of Lemma 6.8 has edge set E' given by*

$$\begin{aligned} E' \;=\; & \{(w_i^{\xi,Q}, w_i^{\xi,T}) : i = 1, \ldots, N\} \cup \{(w_i^{\zeta,Q}, w_i^{\zeta,T}) : i = 1, \ldots, N\} && (6.33) \\ & \cup \{(x_i^{\xi,Q}, x_i^{\xi,T}) : i = 1, \ldots, N\} \cup \{(x_i^{\zeta,Q}, x_i^{\zeta,T}) : i = 1, \ldots, N\} \\ & \cup \{(w_i^{\xi,T}, x_j^{\xi,T}) : j \in N(i)\} \cup \{(w_i^{\xi,T}, x_j^{\zeta,T}) : j \in N(i)\} && (6.34) \\ & \cup \{(w_i^{\zeta,T}, w_j^{\xi,T}) : j \in N(i)\} \cup \{(w_i^{\zeta,T}, w_j^{\xi,T}) : j \in N(i)\} \\ & \cup \{(x_i^{\xi,Q}, w_i^{\xi,Q}) : i = 1, \ldots, N\} \cup \{(x_i^{\xi,Q}, w_i^{\zeta,Q}) : i = 1, \ldots, N\} && (6.35) \\ & \cup \{(x_i^{\zeta,Q}, w_i^{\zeta,Q}) : i = 1, \ldots, N\} \end{aligned}$$

Proof. The edges (6.33) correspond to the edges $\{(w_i^Q, w_i^T) : m(w_i^T) = w_i^Q\}$ in the definition of $\hat{E}$. The edges enumerated in (6.34) correspond to the edges $\{(w_i^T, x_j^T) : (i,j) \in N(T_A)$. Finally, the value of $\Omega\Omega^{-1}_{J\cup J_1}$ from (6.26) implies that the edges enumerated in (6.35) correspond to the edges $\{(x^Q, y^Q) : x \in \hat{V} \setminus J, y \in J, \tilde{\Omega}_{xy} \neq 0, \tilde{\Omega}_{xz} = 0 \forall z \in J_1\}$. □

Lemma 6.26 leads to the following result, which relates the connectivity of the auxiliary graph and the graph G induced by the node dynamics. This is analogous to Lemma 6.24.

Lemma 6.27. *For any node i, the nodes $w_i^{\xi,Q}$, $w_i^{\zeta,Q}$, $x_i^{\xi,Q}$, $x_i^{\zeta,Q}$, $w_i^{\xi,T}$, $w_i^{\zeta,T}$, $x_i^{\xi,T}$, and $x_i^{\zeta,T}$ are input-connected in the auxiliary graph G' if and only if node i is input-connected in the graph G.*

Proof. Suppose that node i is connected to an input node i' in G, with path $(i, i_1), (i_1, i_2), \ldots, (i_r, i')$. Now, consider the node $w_i^{\zeta,T}$. We can construct a path π from $w_i^{\zeta,T}$ to $w_{i'}^{\xi,T}$ as

$$\pi = (w_i^{\zeta,T}, x_{i_0}^{\xi,T}) \cup \bigcup_{l=0}^{r} \{(x_{i_l}^{\xi,T}, x_{i_l}^{\xi,Q}), (x_{i_l}^{\xi,Q}, w_{i_l}^{\xi,Q}), (w_{i_l}^{\xi,Q}, w_{i_l}^{\xi,T}), (w_{i_l}^{\xi,T}, x_{i_{l+1}}^{\xi,T})\}$$
$$\cup \{(x_{i'}^{\xi,T}, x_{i'}^{\xi,Q}), (x_{i'}^{\xi,Q}, w_{i'}^{\xi,Q}), (w_{i'}^{\xi,Q}, w_{i'}^{\xi,T})\}. \quad (6.36)$$

Paths for the other types of nodes in the auxiliary graph can be constructed in a similar fashion. Conversely, any path to an input node in the auxiliary graph will have the form (6.36), and hence can be used to construct a path to an input node in the graph G. □

Lemma 6.27 implies that, for performance metrics satisfying (6.28), Algorithm 14 returns a set S satisfying $f(S) \geq (1 - 1/e)f(S^*)$, where S^* is the optimal solution. The proof is analogous to Theorem 6.8.

6.4.3 Input Selection in Networks of Free Parameters

We now investigate systems where all of the matrix entries are free parameters, as in the models of [36, 85, 118]. We consider systems of the form $\dot{\mathbf{x}}(t) = A\mathbf{x}(t)$, where A is a free matrix and $F = I$. The matrix Ω defined in Section 6.1 is then given by

$$\Omega = \begin{matrix} \mathbf{w} \\ \mathbf{x} \\ \mathbf{u} \end{matrix} \begin{pmatrix} -I & 0 \\ I & 0 \\ 0 & I \end{pmatrix} \quad (6.37)$$

Hence the simple matching $m(w_i^T) = w_i^Q$ has full rank in (6.37), and we can compute $\Omega_{J \cup J_1}$ and $\Omega\Omega_{J \cup J_1}^{-1}$ as

$$\Omega_{J \cup J_1} = \begin{pmatrix} -I & 0 \\ 0 & I \end{pmatrix}, \quad \Omega\Omega_{J \cup J_1}^{-1} = \begin{pmatrix} I & 0 \\ -I & 0 \\ 0 & I \end{pmatrix} \quad (6.38)$$

Based on the value of $\Omega\Omega_{J \cup J_1}^{-1}$, the following lemma gives the construction of the auxiliary graph G'.

Lemma 6.28. *The auxiliary graph G' of Lemma 6.4 has edge set E' given by*

$$E' = \{(w_i^Q, w_i^T) : i = 1, \ldots, N\} \cup \{(w_i^T, x_j^T) : j \in N(i)\} \cup \{(x_i^T, x_i^Q) : i = 1, \ldots, N\} \cup \{(x_i^Q, w_i^Q) : i = 1, \ldots, N\}$$

Proof. The first term follows from the matching $m(w_i^T) = w_i^Q$. The second term arises from the matrix A, while the third term is from the definition of the auxiliary graph. The last term follows from the value of $\Omega\Omega_{J\cup J_1}^{-1}$ in (6.38). □

Hence, in the free parameter case we have a result analogous to Lemmas 6.24 and 6.27.

Lemma 6.29. *The conditions of Lemma 6.4 are met if and only if each node is path-connected to an input node in the graph G.*

Proof. Suppose that a node i is path-connected to an input node i' in G, where the path is given by $(i, i_0), \ldots, (i_{r-1}, i_r), (i_r, i')$, letting $i_{r+1} = i'$. Then the corresponding path in G' is equal to

$$\begin{aligned} \pi \quad = \quad & \{(w_i^Q, w_i^T), (w_i^T, x_{i_0}^T), (x_{i_0}^T, x_{i_0}^Q), (x_{i_0}^Q, w_{i_0}^Q)\} \\ & \cup \bigcup_{l=0}^{r} \{(w_{i_l}^Q, w_{i_l}^T), (w_{i_l}^T, x_{i_{l+1}}^T), (x_{i_{l+1}}^T, x_{i_{l+1}}^Q), (x_{i_{l+1}}^Q, w_{i_{l+1}}^Q)\} \\ & \cup (w_{i'}^Q, w_{i'}^T) \end{aligned}$$

□

We now investigate the other matroid constraint of Lemma 6.6. When all entries of A are free, the constraint reduces to

$$\operatorname{rank}(\mathcal{M}([I|0|0]) \vee \mathcal{M}([I|T_A|T_B(S)])) = 2n,$$

which is in turn equivalent to the second matroid being full rank. Hence the condition can be reduced to $\operatorname{rank}(\mathcal{M}([I|T_A|T_B(S)])) = n$.

Lemma 6.29 implies that, if a system with free parameters is strongly connected, then it suffices to find an input set such that the matroid of Lemma 6.6 is full rank. A minimum-size input set such that $\operatorname{rank}(\mathcal{M}([I|T_A|T_B(S)])) = n$ can be found efficiently using the greedy

algorithm. This enables efficient computation of an input set with the same size as in [85], but through the matroid optimization framework.

Finally, we observe that, by Lemma 6.29, Algorithm 14 returns a set S that satisfies a $(1 - 1/e)$ optimality bound, by the argument of Theorem 6.8.

6.5 Numerical Study

We numerically evaluated our framework using Matlab. We considered the consensus network case of Section 6.4.1. Network topologies were generated by placing nodes at uniform random positions within a square region, and creating a link (i, j) if node i is within the communication range of node j. The range of each node was chosen uniformly at random from the interval $[0, 600]$. We investigated the minimum-size set of input nodes for structural controllability, as well as selection of input nodes for joint performance and controllability.

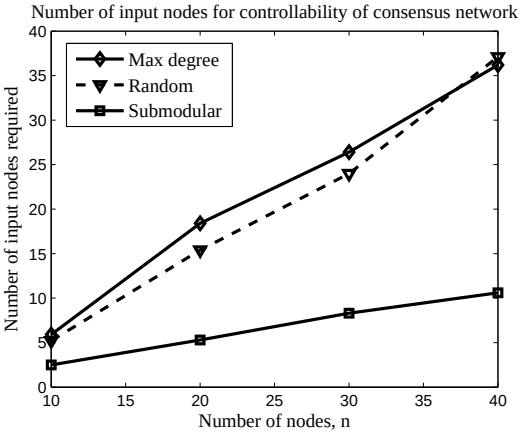


Figure 6.1: Minimum-size input set for structural controllability in a consensus network. The submodular optimization approach is compared to max degree-based and random input selection. The submodular optimization approach typically requires roughly one-quarter of the network to be controlled, while the random and degree-based heuristics select nearly all network nodes before controllability is satisfied.

In the case of selecting the minimum-size set of input nodes for structural controllability, we considered networks of size $n = \{10, 20, 30, 40\}$. The deployment area was selected to yield average node degrees $d = 3$. We compared our submodular optimization approach with

selecting high degree nodes as inputs, as well as selecting random nodes to act as inputs. The submodular optimization approach required fewer input nodes to satisfy controllability, with the other heuristics selecting nearly all network nodes before controllability is satisfied. For all schemes, the number of input nodes was increasing in the network size.

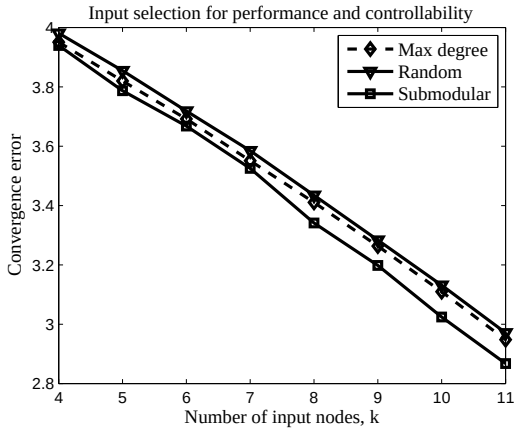


Figure 6.2: Convergence error when input nodes are selected to minimize convergence error while satisfying controllability. The number of nodes n was equal to 20. The submodular optimization approach provided lower convergence error than degree-based and random selection algorithms, especially as the number of input nodes increased.

We evaluated selection of input nodes in order to minimize the convergence error with controllability as a constraint. The convergence error was defined as $||\mathbf{x}(t) - x^*\mathbf{1}||_2$, where x^* is the state of the input nodes, $t = 1$, and the initial state and edge weights were chosen uniformly at random. The number of nodes was equal to 20, while the deployment area was chosen to achieve an average degree of 2. The submodular optimization approach provided lower convergence error than the degree-based and random heuristics, while also satisfying controllability from the input set. As the number of input nodes increased, the gap between the submodular optimization approach and the other heuristics increased.

6.6 Conclusions and Future Work

In this chapter, we studied the problem of input selection for joint performance and controllability of structured linear descriptor systems. Our main contribution was to prove

that structural controllability of linear descriptor systems can be mapped to two matroid constraints, representing controllability of the zero and nonzero modes of the system. We demonstrated that, by exploiting this matroid structure, a minimum-size set of input nodes to guarantee structural controllability of such systems can be selected in polynomial time via matroid intersection algorithms. We further showed that selection of input nodes for joint performance and controllability can be formulated as a submodular maximization problem subject to two matroid basis constraints. We presented polynomial-time algorithms for obtaining a continuous solution to the input selection problem, providing a $(1-1/e)$ optimality bound, which can then be rounded to obtain a feasible input set. We demonstrated that, when the objective function is modular, the optimal input selection for performance and controllability can be computed in polynomial time.

We investigated input selection in systems where the graph representation of the system is strongly connected, and found that for almost all systems of this type the number of matroid constraints can be reduced from two to one. This led to an $O(n)$ algorithm for selecting a minimum-size set of input nodes for structural controllability, as well as more efficient polynomial-time algorithms for approximating the optimal input set up to a factor of $(1 - 1/e)$. We studied linear consensus systems, double integrator systems, and systems consisting of free parameters within our framework, and showed that the additional structure of each system provided a provable $(1 - 1/e)$ optimality bound for input selection based on performance and controllability.

6.6.1 Future Work

While this chapter is a first step towards a unifying framework for ensuring performance and controllability, a variety of extensions are possible.

Incorporating Dynamic Network Topologies: As noted in previous chapters, networked systems may change topologies due to random failures, switching between predefined topologies, and node mobility. While some necessary and sufficient conditions for controllability of switched systems are in the existing literature [84], currently there is no efficient algorithm for selecting leaders to guarantee controllability under dynamic network topologies,

let alone joint performance and controllability. State-dependent networks, in which the network topology is determined by the node states (e.g., the node positions), which are in turn affected by the choice of control inputs, represent another class of networked systems for which guaranteeing controllability is an open problem.

Other Structure-Based Metrics: The controllability of a networked system is heavily influenced by the network topology and resulting interactions between the nodes. Structural factors also influence metrics such as robustness to disturbances and delays [131], but currently leader selection for structural robustness to disturbance and delays has not been investigated. A joint framework for selecting leaders based on performance, controllability, and robustness would enable design of safer and more reliable networked systems while still guaranteeing desired performance.

Algorithm 13 Algorithm for selecting the minimum-size input set to guarantee structural controllability.

procedure MIN_CONTROLLABLE_SET($\mathcal{M}_1^*$, $\mathcal{M}_2^*$)
  Input: Matroids $\mathcal{M}_1^*$ and $\mathcal{M}_2^*$
  Output: Set of inputs S
  $R \leftarrow \emptyset$
  while 1 **do**
    $E_{\mathcal{M}_1^*,\mathcal{M}_2^*}(R) \leftarrow \emptyset$
    for All $i \in R$, $j \notin R$ **do**
      if $(R - \{i\} \cup \{j\}) \in \mathcal{M}_1^*$ **then**
        $E_{\mathcal{M}_1^*,\mathcal{M}_2^*}(R) \leftarrow E_{\mathcal{M}_1^*,\mathcal{M}_2^*}(R) \cup \{(i,j)\}$
      end if
      if $(R - \{i\} \cup \{j\}) \in \mathcal{M}_2^*$ **then**
        $E_{\mathcal{M}_1^*,\mathcal{M}_2^*}(R) \leftarrow E_{\mathcal{M}_1^*,\mathcal{M}_2^*}(R) \cup \{(j,i)\}$
      end if
    end for
    $D_{\mathcal{M}_1^*,\mathcal{M}_2^*}(R) \leftarrow$ directed graph with vertex set V and edge set $E_{\mathcal{M}_1^*,\mathcal{M}_2^*}(R)$
    $X_1 \leftarrow \{j \in V \setminus R : (R \cup \{j\}) \in \mathcal{M}_1^*\}$
    $X_2 \leftarrow \{j \in V \setminus R : (R \cup \{j\}) \in \mathcal{M}_2^*\}$
    if path exists from a node in X_1 to a node in X_2 **then**
      $P \leftarrow$ shortest X_1-X_2 path
      $R \leftarrow R \Delta P$
    else
      break
    end if
  end while
  $S \leftarrow V \setminus R$
  return S
end procedure

Algorithm 14 Input selection algorithm for joint performance and controllability.

procedure INPUT_SELECT(f, $\hat{\mathcal{M}}_1$, $\hat{\mathcal{M}}_2$, k)
  Input: Monotone submodular objective function $f : 2^V \to \mathbb{R}$
    Matroids $\hat{\mathcal{M}}_1$, $\hat{\mathcal{M}}_2$
    Maximum number of inputs k
  Output: Set of inputs S
  $\delta \leftarrow \frac{1}{9k^2}$, $t \leftarrow 0$, $\mathbf{y}(0) \leftarrow \mathbf{0}$
  while $t < 1$ **do**
    $R(t)$ contains each $j \in V$ independently with probability $y_j(t)$
    for $j \in V$ **do**
      $\omega_j(t) \leftarrow \mathbf{E}[f(R(t) \cup \{j\}) - f(R(t))]$
    end for
    $I(t) \leftarrow$ MAX_WEIGHTED_BASIS$(\hat{\mathcal{M}}_1, \hat{\mathcal{M}}_2, \boldsymbol{\omega})$
    $\mathbf{y}(t + \delta) \leftarrow \mathbf{y}(t) + \delta \cdot \mathbf{1}(I(t))$
    $t \leftarrow (t + \delta)$
  end while
  $S \leftarrow$ SWAP_ROUND$(\mathbf{y}(1), \hat{\mathcal{M}}_1, \hat{\mathcal{M}}_2)$
  return S
end procedure

Chapter 7

SYNCHRONIZATION IN COMPLEX NETWORKS

Synchronization plays a vital role in complex networks. Stable operation of the power grid requires synchronization of buses and generators to a common frequency [43]. Synchronized oscillations of neuronal firing provide a biological mechanism for aggregating information in perception [46] and memory [70]. Coordinated motion of animals [105] occurs when a common heading is achieved. The prevalence of synchronization across different application domains motivates the study of the basic principles underlying synchronization [6].

Phase-coupled oscillators have been proposed as a widely-applicable framework for studying synchronization [116]. In phase-coupled oscillator networks, the dynamics of each node's phase are determined by a diffusive coupling with its neighbors (typically assumed to be sinusoidal [3]), together with an intrinsic frequency. The sinusoidal coupling causes the node phases to approach synchronization, while the intrinsic frequency drives each node away from synchronization.

The existence and stability of synchronized states has been studied extensively in the literature [33, 64, 127], including conditions based on the intrinsic frequencies, network topology, and degree of coupling between the nodes. An important case is synchronization in the presence of external inputs [32]. External inputs arise in applications including neuroscience, where they represent environmental stimuli [46] or deep brain stimulation [87]. From an engineering standpoint, by introducing external inputs that pin a subset of nodes to a desired phase and frequency, a network that does not synchronize in the absence of inputs can be driven to a synchronized state, thus facilitating stability and performance of the network.

Existing analytical approaches to introducing external inputs assume that the external input node is connected to all other nodes [133], or that the network has a specific topology

such as a complete graph [71]. Developing sufficient conditions for synchronization using external inputs in networks with arbitrary topology is an open problem. Efficient algorithms for selecting a subset of input nodes in order to guarantee synchronization are also currently lacking.

In this chapter, we present an optimization framework for selecting a subset of phase-coupled oscillators to act as external inputs in order to guarantee practical synchronization of the overall network. We formulate sufficient conditions for ensuring convergence of phase-coupled oscillators to a synchronized state from almost any initial state (i.e., except for a measure zero set of initial states). We then develop a submodular optimization approach for selecting a minimum-size set of input nodes to achieve a desired level of synchronization, with provable bounds on the optimality of the chosen input nodes. The main results of this chapter are:

- We investigate two synchronization problems. The first problem is to ensure that the phases of all of the oscillators converge to fixed points that are within a desired bound of a given reference phase (practical node synchronization). The second problem is to guarantee that the phase differences of neighboring nodes converge to within a desired bound of each other (practical edge synchronization). In both problems, the node frequencies must converge to the same value.
- We derive a set of sufficient conditions for a given set of input nodes to achieve practical node or edge synchronization from almost any initial state. We interpret our conditions as each node achieving a desired level of synchronization if a threshold number of neighbors reaches that level of synchronization.
- We develop a submodular optimization framework for selecting a set of input nodes to achieve these synchronization conditions. Our approach is to derive a connection between the threshold conditions and the connectivity of an augmented graph. Based on this connection, we map our sufficient conditions to constraints on submodular functions. We propose efficient algorithms for selecting input nodes to guarantee synchronization and analyze the optimality bounds of our algorithms.
- We evaluate our approach through a numerical study of synchronization in three

classes of networks. First, we consider synchronization of power grids using the IEEE 14 Bus test case [1]. Second, we study synchronization in geometric random graphs, motivated by vehicle coordination and wireless network synchronization problems. Finally, we investigate synchronization of neuronal networks based on the *C. Elegans* dataset [132].

The chapter is organized as follows. Section 7.1 contains our system model and definitions of synchronization. Sufficient conditions for synchronization are presented in Section 7.2. In Section 7.3, we describe our submodular optimization approach to selecting input nodes. Section 7.4 contains our numerical study. In Section 7.5, we chapter the paper and discuss directions for future work.

7.1 Model and Preliminaries

In this section, we describe the system model and oscillator dynamics. We then define the notions of synchronization considered in this chapter. Finally, we give a preliminary result that will be needed for the proof of Theorem 7.2 in Section 7.2.

7.1.1 System Model

A network of n oscillators, indexed in the set $V = \{1, \dots, n\}$ is considered. Each oscillator $v \in V$ has a neighbor set $N(v) \subseteq V$, consisting of the set of oscillators that are coupled to v. We assume that links are bidirectional, so that $u \in N(v)$ implies $v \in N(u)$. An edge (u, v) exists if $u \in N(v)$ and $v \in N(u)$. We let E denote the set of edges, and let $p = |E|$.

Each oscillator v has a time-varying phase $\theta_v(t)$. The vector of oscillator phases at time t is denoted $\boldsymbol{\theta}(t) \in \mathbb{R}^n$. We assume that there are two types of oscillators, denoted *input* and *non-input* oscillators. We let A denote the set of input oscillators. The phases of the non-input oscillators follow the Kuramoto dynamics

$$\dot{\theta}_v(t) = -\sum_{u \in N(v)} K_{uv} \sin\left(\theta_v(t) - \theta_u(t)\right) + \omega_v. \tag{7.1}$$

In (7.1), the first term represents the coupling between the oscillators, while ω_v is the intrinsic frequency of oscillator v and describes the phase dynamics in the absence of coupling.

The coupling coefficient $K_{uv} > 0$ determines the relative strength of the two terms. We assume throughout that the couplings between nodes are symmetric, so that $K_{uv} = K_{vu}$ for all $(u, v) \in E$. We define $\overline{K}_v \triangleq \sum_{u \in N(v)} K_{uv}$.

Each input oscillator $v \in A$ is assumed to be pinned to a desired frequency ω_0 and phase offset θ_0, so that $\dot{\theta}_v(t) = \omega_0$ and $\theta_v(t) = \omega_0 t + \theta_0$ for all $v \in A$. The overall oscillator dynamics are given by

$$\dot{\theta}_v(t) = \begin{cases} -\sum_{u \in N(v)} K_{uv} \sin\left(\theta_v(t) - \theta_u(t)\right) + \omega_v, & v \notin A \\ \omega_0, & v \in A \end{cases} \tag{7.2}$$

We define a function $f(x) : [-2\pi, 2\pi] \to [-\pi, \pi]$ as

$$f(x) = \begin{cases} x + 2\pi, & x \in [-2\pi, -\pi) \\ x, & x \in [-\pi, \pi) \\ x - 2\pi, & x \in [\pi, 2\pi] \end{cases}$$

The function $f(x)$ maps elements in $[-2\pi, 2\pi]$ into $[-\pi, \pi]$ while satisfying $\sin\left(f(x)\right) = \sin\left(x\right)$ and $\cos\left(f(x)\right) = \cos\left(x\right)$ for all x. This function will be used to define the edge cohesiveness property in the following subsection.

7.1.2 Definitions of Synchronization

We now define the notions of synchronization considered. Analogous definitions for networks without external inputs are given in [44].

The strongest form of synchronization is *phase synchronization*, defined as follows.

Definition 7.1. *The oscillators achieve* phase synchronization *if there exists* θ^* *such that* $\lim_{t \to \infty} \theta_v(t) = \theta^*$ *for all* $v \in V$.

Phase synchronization is achieved if all oscillators converge to the same phase. Since synchronization of all oscillators to the same phase is not possible in general [64], weaker notions of synchronization have been proposed. We first define *node-* and *edge-cohesiveness* as follows.

Definition 7.2 (Node Cohesiveness)**.** *A network of oscillators achieves* node cohesiveness *with parameter* $\gamma \in [0, \frac{\pi}{4}]$ *if there exists* $T > 0$ *such that, for all* $t \geq T$, $|\theta_v - (\omega_0 t + \theta_0)| < \gamma$.

Definition 7.3 (Edge Cohesiveness)**.** *A network of oscillators achieves* edge cohesiveness *with parameter* $\gamma \in [0, \frac{\pi}{2}]$ *if there exists* $T > 0$ *such that, for all* $t \geq T$ *and all* $(u, v) \in E$, $|f(\theta_v - \theta_u)| < \gamma$.

Node cohesiveness is achieved if all nodes converge to within a desired bound of the reference phase. Edge cohesiveness implies that the relative differences between any pair of neighboring nodes is bounded above. In general, node cohesiveness is desirable in applications such as coordinated motion [105] and time synchronization [9], where all nodes must agree on a common phase. Edge cohesiveness is desirable in applications including power systems [43], where the relative differences between nodes must be within a certain range to ensure stability.

We observe that a network that is node cohesive with parameter γ is edge cohesive with parameter 2γ. Indeed, node cohesiveness implies that there exists $T > 0$ such that, for all $t \geq T$,

$$\begin{aligned} |\theta_v(t) - \theta_u(t)| &= |(\theta_v(t) - (\omega_0 t + \theta_0)) - (\theta_u(t) - (\omega_0 t + \theta_0))| \\ &\leq |\theta_v(t) - (\omega_0 t + \theta_0)| + |\theta_u(t) - (\omega_0 t + \theta_0)| \\ &< \gamma + \gamma = 2\gamma. \end{aligned}$$

An additional synchronization notion is frequency synchronization, defined as follows.

Definition 7.4 (Frequency Synchronization)**.** *The oscillators achieve* frequency synchronization *if there exists* ω^* *such that* $\lim_{t\to\infty} \dot{\theta}_v(t) = \omega^*$ *for all* $v \in V$.

Note that the only possible value for ω^* in Definition 7.4 is the frequency of the input nodes, denoted ω_0, since we have $\dot{\theta}_v(t) \equiv \omega_0$ for all $v \in A$ and $t > 0$.

We now define the main types of synchronization considered in this chapter.

Definition 7.5 (Practical Node and Edge Synchronization)**.** *The oscillators achieve* practical node synchronization *with parameter* γ *if they achieve frequency synchronization and node cohesiveness with parameter* γ. *The oscillators achieve* practical edge synchronization *with parameter* γ *if they achieve frequency synchronization and edge cohesiveness with parameter* γ.

The following lemma allows us to focus on the case where $\omega_0 = 0$, so that all input oscillators have frequency 0.

Lemma 7.1. *Define* $\hat{\boldsymbol{\theta}}(t) = \boldsymbol{\theta}(t) - (\omega_0 t + \theta_0)\mathbf{1}$, *where* $\mathbf{1}$ *denotes the vector of all* 1*'s. Then* $\boldsymbol{\theta}(t)$ *achieves practical node (resp. edge) synchronization with frequency* ω_0, *phase bound* γ, *and reference frequency* θ_0 *if and only if* $\hat{\boldsymbol{\theta}}(t)$ *achieves practical node (resp. edge) synchronization with frequency* 0 *and phase bound* γ.

Proof. First, note that $\dot{\hat{\theta}}_v(t) = \dot{\theta}_v(t) - \omega_0$ for all $v \in V$. Suppose that $\boldsymbol{\theta}(t)$ achieves practical node synchronization with frequency ω_0, reference phase θ_0, and phase bound γ. Then for all $v \in V$,

$$\lim_{t\to\infty} \dot{\hat{\theta}}_v(t) = \lim_{t\to\infty} \Big(\dot{\theta}_v(t) - \omega_0\Big) = 0.$$

Furthermore, there exists $T > 0$ such that, for all $t \geq T$ and $v \in V$,

$$|\hat{\theta}_v(t)| = |\theta_v(t) - (\theta_0 + \omega_0 t)| \leq \gamma,$$

and hence practical node synchronization is achieved. The proof for practical edge synchronization is similar, as is the proof of the converse. □

7.1.3 Preliminary Result

The following preliminary result of [93] will be needed in Section 7.2. First, for any $\delta > 0$ and any matrix B, the δ-digraph is defined as the digraph where edge (i, j) exists if $B_{ij} \geq \delta$.

Theorem 7.1 ([93], Theorem 1)**.** *Consider the linear system* $\dot{x}(t) = A(t)x(t)$, *where* $A(t)$ *is a time-varying system matrix. Assume that the system matrix is a bounded piecewise continuous function of time, and that for every time* t *the system matrix is Metzler (i.e., all off-diagonal elements are nonnegative) and has zero row sums. Suppose further that there is an index* $k \in \{1, \ldots, n\}$, *a threshold value* $\delta > 0$, *and an interval length* $T > 0$ *such that for all* $t \in \mathbb{R}$ *the* δ*-digraph associated to*

$$\int_t^{t+T} A(s)\, ds$$

has the property that all nodes may be reached from the node k. *Then the set of states* $\{x^*\mathbf{1} : x^* \in \mathbb{R}\}$ *is uniformly exponentially stable. In particular, all components of any solution converge to a common value as* $t \to \infty$.

The theorem gives a sufficient condition for a linear system with time-varying weights to converge to consensus. Theorem 7.1 will be used to prove that, under certain conditions on the oscillator phases, the frequencies $\{\dot{\theta}_v(t) : v \in V\}$ converge to the frequencies of the input nodes.

7.2 Sufficient Conditions for Practical Synchronization

In this section, we provide sufficient conditions for a set of inputs to achieve practical synchronization. We first show that the oscillators achieve practical node (resp. edge) synchronization if there exists at least one stable equilibrium point satisfying node (resp. edge) cohesiveness, and no stable equilibria that do not satisfy node (resp. edge) cohesiveness. We then formulate sufficient conditions for existence of a stable and cohesive equilibrium point, as well sufficient conditions for non-existence of stable equilibria that do not satisfy cohesiveness. We provide efficient algorithms for verifying the sufficient conditions for a given set of inputs.

7.2.1 Equilibrium Analysis and Practical Synchronization

As a preliminary, we define *node cohesive*, *edge cohesive*, *node non-cohesive*, and *edge non-cohesive* equilibria as follows.

Definition 7.6 (Cohesive and Non-cohesive Equilibria)**.** *An equilibrium $\boldsymbol{\theta}^*$ of the dynamics (7.2) is* node cohesive *if $|\theta_v^*| \leq \gamma$ for all $v \in V$ and* edge cohesive *if $|\theta_v^* - \theta_u^*| \leq \gamma$ for all $(u, v) \in E$. An equilibrium is* node non-cohesive *if it is not node-cohesive, and* edge non-cohesive *if it is not edge-cohesive.*

We first prove that the dynamics (7.2) are guaranteed to converge to an equilibrium point if an equilibrium exists. An analogous result is known in the case where there are no input nodes [134].

Lemma 7.2. *Let Θ denote the set of equilibria of (7.2). If $\Theta \neq \emptyset$, then for any initial state $\boldsymbol{\theta}(0)$, there exists $\boldsymbol{\theta}^* \in \Theta$ such that $\lim_{t\to\infty} \boldsymbol{\theta}(t) = \boldsymbol{\theta}^*$.*

Proof. Define $E_1 \triangleq \{(u,v) \in E : u \notin A, v \notin A\}$ and $E_2 \triangleq \{(u,v) \in E : u \in A, v \notin A\}$. Let $U(\boldsymbol{\theta}) : \mathbb{R}^n \to \mathbb{R}$ be given by

$$U(\boldsymbol{\theta}) = \sum_{(u,v)\in E_1} K_{uv}(1 - \cos(\theta_v - \theta_u)) + \sum_{(u,v)\in E_2} K_{uv}(1 - \cos\theta_v) - \sum_{v\notin A} \omega_v \theta_v.$$

The function U is continuously differentiable as a function of $\boldsymbol{\theta}$. We have that $\dot{\boldsymbol{\theta}} = -\nabla_{\boldsymbol{\theta}} U(\boldsymbol{\theta})$, and therefore $\dot{U}(\boldsymbol{\theta}) = -||\dot{\boldsymbol{\theta}}||_2^2$. Hence $\dot{U}(\boldsymbol{\theta}) \leq 0$ and $\dot{U}(\boldsymbol{\theta}) = 0$ if and only if $\boldsymbol{\theta}$ is an equilibrium, and we have $\{\boldsymbol{\theta} : \dot{U}(\boldsymbol{\theta}) = 0\} = \Theta$.

Since Θ is the set of equilibria of (7.2), it is invariant under the dynamics of (7.2). The lemma follows from LaSalle's Theorem [69]. □

Lemma 7.2 implies that all initial states will converge to an equilibrium if at least one equilibrium exists. Furthermore, if $\boldsymbol{\theta}^*$ is an unstable equilibrium, then its stable manifold will have dimension less than n by the stable manifold theorem [69], and hence the set of initial states that converge to $\boldsymbol{\theta}^*$ will have measure zero. Based on this observation and Lemma 7.2, we have the following theorem.

Theorem 7.2. *Suppose that the dynamics (7.2) has at least one stable node cohesive (resp. edge cohesive) equilibrium and no stable non-node cohesive (resp. non-edge cohesive) equilibria. Then the oscillators achieve practical node (resp. edge) synchronization from almost any initial state.*

Proof. Let Θ' denote the set of asymptotically stable equilibria of (7.2). Lemma 7.2 and the above discussion imply that, for almost all initial states, $\boldsymbol{\theta}(t)$ converges to an equilibrium in Θ'.

Suppose first that at least one stable equilibrium exists, and that all stable equilibria are node-cohesive. We show that node cohesiveness holds. Let $\overline{\theta} = \max\{|\theta_v| : v \in V, \boldsymbol{\theta} \in \Theta'\}$. We have $\overline{\theta} < \gamma$. Since the set of limit points of $\boldsymbol{\theta}(t)$ is contained in Θ', for any solution $\boldsymbol{\theta}(t)$ to (7.2), $\lim_{t\to\infty} \boldsymbol{\theta}(t) = \boldsymbol{\theta}^*$ for some $\boldsymbol{\theta}^* \in \Theta'$. Hence there exists T such that $t > T$ implies that $|\theta_v(t) - \theta_v^*| < \gamma - \overline{\theta}$ for all $v \in V$. We then have

$$|\theta_v(t)| \leq |\theta_v^*| + |\theta_v(t) - \theta_v^*| < \overline{\theta} + \gamma - \overline{\theta} = \gamma.$$

Now, suppose that all stable equilibria are edge-cohesive. Let $\overline{\theta} = \max\{|\theta_v^* - \theta_u^*| : (u,v) \in E, \boldsymbol{\theta}^* \in \Theta'\}$. Since Θ' is the set of limit points of (7.2), every trajectory of (7.2) satisfies $|\theta_v(t) - \theta_u(t) - (\theta_v^* - \theta_u^*)| < \gamma - \overline{\theta}$ for t sufficiently large. Hence

$$|\theta_v(t) - \theta_u(t)| \leq |\theta_v^* - \theta_u^*| + |\theta_v(t) - \theta_u(t) - (\theta_v^* - \theta_u^*)| < \overline{\theta} + \gamma - \overline{\theta} = \gamma.$$

We now establish that frequency synchronization holds in both cases. Consider $\dot{\boldsymbol{\theta}}(t)$, which has dynamics

$$\ddot{\theta}_v(t) = -\sum_{u \in N(v)} \left[K_{uv} \cos\left(\theta_v(t) - \theta_u(t)\right)\left(\dot{\theta}_v(t) - \dot{\theta}_u(t)\right)\right] \tag{7.3}$$

for $v \notin A$ and $\ddot{\theta}_v(t) \equiv 0$ for $v \in A$. We now define dynamics of the form in Theorem 7.1 in order to analyze the convergence of (7.3). Let $x(t) \in \mathbb{R}^{n+1}$ denote the state variable, where x_{n+1} is a "super node" with dynamics $\dot{x}_{n+1}(t) \equiv 0$. Define the system matrix $A(t)$ by

$$A_{ij}(t) = \begin{cases} \cos K_{ij}(\theta_i(t) - \theta_j(t)), & (i,j) \in E, \\ & i \notin A \\ -\sum_{j \in N(i)} K_{ij} \cos\left(\theta_i(t) - \theta_j(t)\right), & i = j, i \notin A \\ -1, & i = j, i \in A \\ 1, & i \in A, \\ & j = (n+1) \\ 0, & i = n+1 \end{cases}$$

By the preceding analysis, for both node- and edge-cohesive equilibria, there exists $T > 0$ such that $|\theta_i(t) - \theta_j(t)| < \pi/2$ for all $t > T$. Hence $A(t)$ is a bounded, piecewise continuous Metzler matrix with rows that sum to zero, and the node $(n+1)$ is connected to all other nodes in the associated δ-digraph. By Theorem 7.1, $x(t)$ converges to a state $x^*\mathbf{1}$.

Now, if we set $x_v(0) = x_{n+1} = 0$ for all $v \in A$ and $x_v(0) = \theta_v(0)$ for all $v \notin A$, then the trajectory of $[x_1(t) \cdots x_n(t)]^T$ will be identical to the trajectory of $\boldsymbol{\theta}(t)$. Since this is a special case of the more general result in the preceding paragraph, we have that $\lim_{t\to\infty} \dot{\boldsymbol{\theta}}(t) = \omega^*\mathbf{1}$ for some $\omega^* \in \mathbb{R}$. Moreover, since $\dot{\theta}_v(t) \equiv 0$ for all $v \in A$, we must have $\omega^* = 0$, implying frequency synchronization. $\square$

Theorem 7.2 implies that, in order to prove that practical node (resp. edge) synchronization is achieved from almost any initial state, it suffices to show that there is at least one stable node-cohesive (resp. edge-cohesive) equilibrium and no stable non-cohesive equilibria. Sufficient conditions for these two properties are derived in the following two subsections.

7.2.2 Existence of Cohesive Stable Equilibria

In this section, we first demonstrate that any node or edge cohesive equilibrium is stable. We then present a sufficient condition for the existence of a cohesive equilibrium. As a preliminary, we consider the Jacobian of the non-input oscillator dynamics:

$$J_{uv} = \begin{cases} K_{uv}\cos(\theta_u - \theta_v), & (u,v) \in E, \\ & u, v \notin A \\ -\sum_{u \in N(v)\setminus A} K_{uv}\cos(\theta_u - \theta_v) & \\ -\sum_{u \in N(v)\cap A} K_{uv}\cos\theta_v, & u = v \\ 0, & \text{else} \end{cases} \tag{7.4}$$

Lemma 7.3. *All node- and edge-cohesive equilibria are asymptotically stable.*

Proof. Let $\boldsymbol{\theta}^*$ be a node cohesive equilibrium. Since $|\theta_v^*| < 2\gamma < \pi/2$ and $|\theta_u^* - \theta_v^*| < \gamma \leq \pi/2$, $\cos(\theta_v^* - \theta_u^*) \geq 0$ for all $(u,v) \in E$ and $\cos(\theta_v^*) \geq 0$ for all v. Similarly, if $\boldsymbol{\theta}^*$ is an edge cohesive equilibrium, then $|\theta_u - \theta_v| \leq \gamma$ and so $\cos(\theta_v^* - \theta_u^*) \geq 0$ for all $(u,v) \in E$. In both cases, J is negative definite, and hence the equilibrium is stable. □

Since any cohesive equilibrium is stable by Lemma 7.3, it suffices to show that at least one cohesive equilibrium exists. The following theorem gives sufficient conditions for existence of node-cohesive and edge-cohesive equilibria for a given set of input nodes.

Theorem 7.3. *Let* $\{\overline{\theta}_v : v \in V\}$ *and* $\{\overline{\theta}_{uv} : (u,v) \in E\}$ *be sets of nonnegative real numbers. Define*

$$\Lambda = \{\boldsymbol{\theta} : |\theta_v| \leq \overline{\theta}_v \ \forall v \in V\} \cap \{\boldsymbol{\theta} : |f(\theta_v - \theta_u)| \leq \overline{\theta}_{uv} \ \forall (u,v) \in E\},$$

where f *is defined in Section 7.1.1. Suppose that for all* $\boldsymbol{\theta} \in \Lambda$ *with* $\boldsymbol{\theta} = (\theta_1, \ldots, \overline{\theta}_v, \ldots, \theta_n)$,

$$\sum_{u \in N(v)} K_{uv}\sin(\overline{\theta}_v - \theta_u) > |\omega_v|$$

and that for all $\boldsymbol{\theta} \in \Lambda$ *with* $(\theta_v - \theta_u) = \overline{\theta}_{uv}$,

$$K_{uv} \sin \overline{\theta}_{uv} + \sum_{\hat{u} \in N(v) \setminus \{u\}} K_{\hat{u}v} \sin (\theta_v - \theta_{\hat{u}}) > |\omega_v|.$$

Then there exists an equilibrium in Λ.

Proof. The approach of the proof is to show that Λ is a positive invariant set and apply Brouwer's fixed point theorem [13] to show existence of an equilibrium point. Suppose that Λ is not positive invariant. For some trajectory of (7.2) with $\boldsymbol{\theta}(0) \in \Lambda$, let $t^* = \inf \{t : \boldsymbol{\theta}(t) \notin \Lambda\}$. Define $\overline{V} = \{v \in V : |\theta_v(t^*)| = \overline{\theta}_v\}$, and consider $v \in \overline{V}$. If $\theta_v(t^*) = \overline{\theta}_v$, then

$$\begin{aligned} \dot{\theta}_v(t^*) &= - \sum_{u \in N(v)} K_{uv} \sin (\theta_v(t^*) - \theta_u(t^*)) + \omega_v = - \sum_{u \in N(v)} K_{uv} \sin (\overline{\theta}_v - \theta_u(t^*)) + \omega_v \\ &< -|\omega_v| + \omega_v \leq 0 \end{aligned}$$

where the final inequality follows from the assumption of the theorem and the fact that $\boldsymbol{\theta}(t^*)$ is on the boundary of Λ by continuity of $\boldsymbol{\theta}(t)$. Since $\dot{\theta}_v(t^*) < 0$, there exists $\epsilon(v)$ such that $|\theta_v(t)| \leq \overline{\theta}_v$ for $t \in [t^*, t^* + \epsilon(v))$. Similarly, if $\theta_v(t^*) = -\overline{\theta}_v$, we have

$$\begin{aligned} \dot{\theta}_v(t^*) &= - \sum_{u \in N(v)} K_{uv} \sin (-\overline{\theta}_v - \theta_u(t^*)) + \omega_v = \sum_{u \in N(v)} K_{uv} \sin (\overline{\theta}_v - (-\theta_u(t^*))) + \omega_v \\ &> |\omega_v| + \omega_v \geq 0 \end{aligned}$$

where the inequality follows from the fact that $\boldsymbol{\theta} \in \Lambda$ implies $-\boldsymbol{\theta} \in \Lambda$.

Now, suppose that $\theta_v(t^*) - \theta_u(t^*) = \overline{\theta}_{uv}$. Then we have

$$\begin{aligned} \frac{d}{dt}(\theta_v(t^*) - \theta_u(t^*)) &= \omega_v - K_{uv} \sin (\theta_v(t^*) - \theta_u(t^*)) - \sum_{\hat{u} \in N(v) \setminus \{u\}} K_{\hat{u}v} \sin (\theta_v(t^*) - \theta_{\hat{u}}(t^*)) \\ &\quad + K_{uv} \sin (\theta_u(t^*) - \theta_v(t^*)) - \omega_u + \sum_{\hat{v} \in N(u) \setminus \{v\}} K_{u\hat{v}} \sin (\theta_u(t^*) - \theta_{\hat{v}}(t^*)) \\ &= \omega_v - K_{uv} \sin \overline{\theta}_{uv} - \sum_{\hat{u} \in N(v) \setminus \{u\}} K_{\hat{u}v} \sin (\theta_v(t) - \theta_{\hat{u}}(t)) \\ &\quad + \omega_u - K_{uv} \sin \overline{\theta}_{uv} + \sum_{\hat{v} \in N(u) \setminus \{v\}} K_{u\hat{v}} \sin (\theta_u(t^*) - \theta_{\hat{v}}(t^*)) \\ &< -|\omega_v| + \omega_v - |\omega_u| + \omega_u \leq 0 \end{aligned}$$

and hence $(\theta_v(t) - \theta_u(t))$ is decreasing in a neighborhood of t^*.

Putting the arguments of the preceding paragraphs together, we have that there exists $\epsilon > 0$ such that $t \in [t^*, t^* + \epsilon]$ implies that $\boldsymbol{\theta} \in \Lambda$, contradicting the definition of t^*. Hence if $\boldsymbol{\theta}(0) \in \Lambda$, then $\boldsymbol{\theta}(t) \in \Lambda$ for all $t \geq 0$, implying that Λ is positive invariant. Since Λ is compact, convex, and positive invariant, it contains an equilibrium [13]. □

Theorem 7.3 is a threshold-based condition for existence of cohesive equilibria. If a sufficient number of neighbors of node v lie in the cohesive region, then node v will remain in the cohesive region. The following corollary gives explicit conditions for existence of node and edge cohesive equilibria.

Corollary 7.1. *If there exist* $\{\overline{\theta}_v : v \in V\}$ *and* $\{\overline{\theta}_{uv} : (u,v) \in E\}$ *satisfying the conditions of Theorem 7.3 with* $|\overline{\theta}_v| \leq \gamma$ *for all* $v \in V$*, then there is a node cohesive equilibrium. If there exist* $\{\overline{\theta}_v : v \in V\}$ *and* $\{\overline{\theta}_{uv} : (u,v) \in E\}$ *satisfying the conditions of Theorem 7.3 with* $\overline{\theta}_{uv} \leq \gamma$ *for all* $(u,v) \in E$*, then there exists an edge cohesive equilibrium.*

Theorem 7.3 and Corollary 7.1 give sufficient conditions for existence of a stable cohesive equilibrium point. In the following section, we provide an algorithm for verifying the existence of such equilibria based on these conditions.

7.2.3 Verifying Existence of Node- and Edge-Cohesive Equilibria

In this section, we provide an algorithm for verifying the existence of stable cohesive equilibria. The goal of the algorithm is to identify sets $\{\overline{\theta}_v : v \in V \setminus A\}$ and $\{\overline{\theta}_{uv} : (u,v) \in E\}$ satisfying the conditions of Theorem 7.3. The approach consists of a contraction-based method to identify a feasible region Λ for the set of fixed points, and then prove that the conditions of Theorem 7.3 are satisfied. As a preliminary, define weights $\beta_1(m,l)$, $\beta_2(m,l)$, and $\beta_3(m,l)$ by

$$\beta_1(m,l) = \min\left\{\sin(\theta_v - \theta_u) : \theta_v \in \left[\frac{\gamma(m-1)}{M}, \frac{\gamma m}{M}\right], |\theta_u| \leq \frac{\gamma l}{M}\right\}$$
$$\beta_2(r) = \min\left\{\sin(\theta) : |\theta| \in \left[\frac{\gamma(r-1)}{M}, \frac{\gamma r}{M}\right]\right\}$$
$$\beta_3(r) = \min\left\{\sin(\theta) : |\theta| \leq \frac{\gamma r}{M}\right\}$$

In the algorithm, we select $M \in \mathbb{Z}^+$ and initialize a set $S = A$. The algorithm maintains a set of time-varying weights $\{x_v[k] : v \in V\} \cup \{x_{uv}[k] : (u,v) \in E\}$. The initialization of the weights $\{x_v[0] : v \in V\}$ is given as $x_v[0] = 0$ if $v \in A$ and $x_v[0] = M$ if $v \notin A$. For the weights $\{x_{uv}[0] : (u,v) \in E\}$, the initialization is given as $x_{uv}[0] = 0$ if $u \in A$ and $v \in A$, $x_{uv}[0] = M$ otherwise.

At each iteration k of the algorithm, loop over all nodes $v \in V \setminus A$ and check whether

$$\sum_{u \in N(v)} K_{uv}\beta_1(x_v[k-1], x_u[k-1]) > |\omega_v|. \tag{7.5}$$

If (7.5) holds, set $x_v[k] = x_v[k-1] - 1$ and $S = S \cup \{v\}$. If (7.5) does not hold for any v, then loop over all edges $(u,v) \in E$ and check whether

$$\sum_{\hat{u} \in N(v) \setminus \{u\}} K_{\hat{u}v}\beta_3(x_{\hat{u}v}[k-1]) + K_{uv}\beta_2(x_{uv}[k-1]) > |\omega_v| \tag{7.6}$$

$$\sum_{\hat{v} \in N(u) \setminus \{v\}} K_{u\hat{v}}\beta_3(x_{u\hat{v}}[k-1]) + K_{uv}\beta_2(x_{uv}[k-1]) > |\omega_u| \tag{7.7}$$

If there exists $(u,v) \in E$ satisfying (7.6) and (7.7), set $x_{uv}[k] = x_{uv}[k-1] - 1$ and $S = S \cup \{(u,v)\}$. If no such edge exists, then the algorithm terminates.

The algorithm terminates after at most $M(n+p)$ iterations, since each x_v and x_{uv} can be decremented at most M times. The following theorem relates the termination conditions of the algorithm to the sufficient condition of Theorem 3.

Theorem 7.4. *If $S = V \cup E$ when the algorithm terminates, then node and edge cohesive equilibria exist. If $\mathbf{x}^*$ denotes the vector of weights when termination occurs, then there exists an equilibrium in*

$$\Phi = \left\{ \boldsymbol{\theta} : |\theta_v| \leq \frac{x_v^* \gamma}{M} \right\} \cap \left\{ \boldsymbol{\theta} : |\theta_v - \theta_u| \leq \frac{x_{uv}^* \gamma}{M} \right\}.$$

Proof. Suppose that $V \subseteq S$ and let $v \in V \setminus A$. Define $\overline{\theta}_v = \frac{x_v^* \gamma}{M}$, and let $k_v = \min\{k : x_v^* = x_v[k]\}$.

Then for any $|\theta_u| \leq \frac{x_u[k_v-1]\gamma}{M}$, we have

$$\begin{aligned}\sum_{u\in N(v)} K_{uv}\sin\left(\overline{\theta}_v-\theta_u\right) &\geq \min\left\{\sum_{u\in N(v)} K_{uv}\sin\left(\overline{\theta}_v-\theta_u\right) : |\theta_u| \leq \frac{x_u[k_v-1]\gamma}{M}\right\}\\ &= \sum_{u\in N(v)} K_{uv}\min\left\{\sin\left(\overline{\theta}_v-\theta_u\right) : |\theta_u| \leq \frac{x_u[k_v-1]\gamma}{M}\right\}\\ &\geq \sum_{u\in N(v)} K_{uv}\beta_1(x_v[k_v-1], x_u[k_v-1]) > |\omega_v| \qquad (7.8)\end{aligned}$$

where (7.8) follows from (7.5) and the definition of β_1.

Now, suppose $E \subseteq S$ and let $(u,v) \in E$. Let $\overline{\theta}_{uv} = \frac{x^*_{uv}\gamma}{M}$, and let $k_{uv} = \min\{k : x^*_{uv} = x_{uv}[k]\}$. Then for all $(\hat{u},\hat{v}) \in E$ satisfying $|\theta_{\hat{v}} - \theta_{\hat{u}}| \leq \frac{x_{\hat{u}\hat{v}}[k_{uv}-1]\gamma}{M}$, we have

$$\begin{aligned}\sum_{\hat{u}\in N(v)} K_{\hat{u}v}\sin\left(\theta_v-\theta_{\hat{u}}\right) \geq & \sum_{\hat{u}\in N(v)\setminus\{u\}} K_{\hat{u}v}\min\left\{\sin\left(\theta_v-\theta_{\hat{u}}\right) : |\theta_v-\theta_{\hat{u}}| \leq \frac{x_{\hat{u}v}[k_{uv}-1]\gamma}{M}\right\}\\ &+K_{uv}\sin\left(\overline{\theta}_{uv}\right) \qquad (7.9)\\ \geq & \sum_{\hat{u}\in N(v)\setminus\{u\}} K_{\hat{u}v}\beta_3(x_{\hat{u}v}[k_{uv}-1]) + K_{uv}\beta_2(x_{uv}[k_{uv}-1]) \qquad (7.10)\\ > & |\omega_v| \qquad (7.11)\end{aligned}$$

where (7.10) follows from the definition of β_2 and β_3 and (7.11) follows from (7.6). Since $x_{uv}[k_{uv}-1] \geq x^*_{uv}$ for all $(u,v) \in E$, the inequalities hold for all $\boldsymbol{\theta} \in \Phi$.

Combining the arguments of the preceding paragraphs implies that the conditions of Theorem 7.3 are satisfied for set Φ, and hence cohesive equilibria exist. □

The algorithm presented in this section gives an approach for proving node and edge cohesiveness when an input set is given. In Section 7.3.1, we formulate the problem of selecting a set of input nodes A that guarantees $S = V \cup E$, and hence guarantees existence of cohesive equilibrium.

7.2.4 Conditions for Non-Existence of Stable Non-Cohesive Equilibria

In this section, we first identify a sufficient condition for an equilibrium to be unstable. We then provide two sufficient conditions for non-existence of stable equilibria within a given region.

Lemma 7.4. *Let $\boldsymbol{\theta}^*$ be an equilibrium point. Suppose that, for some $v \in V \setminus A$,*

$$\sum_{u \in N(v)} \cos\left(\theta_v^* - \theta_u^*\right) < 0. \tag{7.12}$$

Then $\boldsymbol{\theta}^*$ is unstable.

Proof. Consider the Jacobian matrix defined in (7.4). If (7.12) holds for some $v \in V \setminus A$, let e_v denote the vector with a 1 in index v and 0's elsewhere. Then $e_v^T J e_v > 0$, and hence the equilibrium $\boldsymbol{\theta}^*$ is unstable. □

In order to prove that all stable equilibria are cohesive, we must show that each subset of the non-cohesive region contains no stable equilibria. The following lemma formalizes this condition.

Lemma 7.5. *Let $v \in V \setminus A$, and let $\overline{\theta}'_v$ and $\overline{\theta}_v$ be real numbers satisfying $0 < \overline{\theta}'_v < \overline{\theta}_v < \pi$. Furthermore, let $\{\overline{\theta}_u : u \in V \setminus \{v\}\}$ be a set of real numbers in $[0, \pi]$. Define*

$$\Theta = \{\boldsymbol{\theta} : \theta_v \in [\overline{\theta}'_v, \overline{\theta}_v], |\theta_u| \leq \overline{\theta}_u\}.$$

If there exists $\lambda \geq 0$ such that

$$\sum_{u \in N(v)} K_{uv} \left[\sin\left(\theta_v - \theta_u\right) - \lambda \cos\left(\theta_v - \theta_u\right)\right] > |\omega_v|$$

for all $\boldsymbol{\theta} \in \Theta$, then there are no stable equilibria in Θ.

Proof. Observe that if, for all $\boldsymbol{\theta} \in \Theta$,

$$\sum_{u \in N(v)} K_{uv} \sin\left(\theta_v - \theta_u\right) > |\omega_v|,$$

then there are no equilibria in Θ. Combining this with the condition of Lemma 7.4, we have that there are no stable equilibria if

$$\min \left\{ \sum_{u \in N(v)} K_{uv} \sin\left(\theta_v - \theta_u\right) : \sum_{u \in N(v)} K_{uv} \cos\left(\theta_v - \theta_u\right) > 0, \boldsymbol{\theta} \in \Phi \right\} > |\omega_v|.$$

By lower-bounding the left-hand side using the Lagrange dual, we have that

$$\min_{\boldsymbol{\theta} \in \Theta} \sum_{u \in N(v)} K_{uv} \left[\sin\left(\theta_v - \theta_u\right) - \lambda \cos\left(\theta_v - \theta_u\right)\right] > |\omega_v|$$

is sufficient for non-existence of a stable equilibrium point in the region Θ. □

An analogous condition can be proved for a region Θ' defined by the relative phase differences between the nodes.

Lemma 7.6. *Let $(u,v) \in E$, and let $\overline{\theta}'_{uv}$ and $\overline{\theta}_{uv}$ be real numbers satisfying $0 < \overline{\theta}'_{uv} < \overline{\theta}_{uv} < \pi$. Furthermore, let $\{\overline{\theta}_{\hat{u}\hat{v}} : (\hat{u},\hat{v}) \in E\}$ be a set of real numbers in $[0,\pi]$. Define*

$$\Theta' = \{\boldsymbol{\theta} : |f(\theta_v - \theta_u)| \in [\overline{\theta}'_{uv}, \overline{\theta}_{uv}], |f(\theta_{\hat{v}} - \theta_{\hat{u}})| \leq \overline{\theta}_{\hat{u}\hat{v}}\}.$$

If there exists $\lambda \geq 0$ such that

$$\sum_{u \in N(v)} K_{uv} \left[\sin(\theta_v - \theta_u) - \lambda \cos(\theta_v - \theta_u)\right] > |\omega_v| \tag{7.13}$$

for all $\boldsymbol{\theta} \in \Theta'$, then there are no stable equilibria in Θ'.

The proof is similar to Lemma 7.5 and is omitted. Lemmas 7.5 and 7.6 give conditions for non-existence of non-cohesive stable equilibria. In the next section, we present an algorithm for verifying that these conditions are satisfied.

7.2.5 Verifying Non-Existence of Stable Non-Cohesive Equilibria

In this section, we present an algorithm, analogous to the algorithm of Section 7.2.3, for verifying that there are no stable non-cohesive equilibria. As in Section 7.2.3, our approach is to consider the region where stable equilibria can occur, and reduce the size of the region at each iteration. We show that non-node cohesive equilibria as well as non-edge cohesive equilibria can be checked using this approach.

As a preliminary, we define weights β_1^λ, β_2^λ, and β_3^λ, for $\lambda \in [0,\infty]$, as follows.

$$\begin{aligned}
\beta_1^\lambda(m,l) &= \min\left\{g_\lambda(\theta_v - \theta_u) : \theta_v \in \left[\frac{\pi(m-1)}{M}, \frac{\pi m}{M}\right], |\theta_u| \leq \frac{\pi l}{M}\right\} \\
\beta_2^\lambda(r) &= \min\left\{g_\lambda(\theta_v - \theta_u) : |f(\theta_v - \theta_u)| \leq \frac{\pi r}{M}\right\} \\
\beta_3^\lambda(r) &= \min\left\{g_\lambda(\theta_v - \theta_u) : |f(\theta_v - \theta_u)| \in \left[\frac{\pi(r-1)}{M}, \frac{\pi r}{M}\right]\right\}
\end{aligned}$$

with

$$g_\lambda(x) = \begin{cases} \sin(x) - \lambda\cos(x), & \lambda \in [0,\infty) \\ -\cos(x), & \lambda = \infty \end{cases}$$

The algorithm is defined as follows. Let $M \in \mathbb{Z}^+$ and define a set of time-varying indices $\{y_v[k] : v \in V\}$ and $\{y_{uv}[k] : (u,v) \in E\}$. The indices y_v are initialized as $y_v[0] = 0$ if $v \in A$ and $y_v[0] = M$ otherwise. The indices y_{uv} are initialized as $y_{uv}[0] = 0$ if $u \in A$, $v \in A$ and $y_{uv}[0] = M$ otherwise.

Choose a set $\Psi = \{\lambda_1, \ldots, \lambda_R\} \subseteq [0, \infty]$. At each iteration of the algorithm, for each node $v \in V \setminus A$, check whether

$$\sum_{u \in N(v)} K_{uv} \beta_1^\lambda (y_v[k-1], y_u[k-1]) > \tau_{\lambda,v} \tag{7.14}$$

for some $\lambda \in \Psi$, where $\tau_{\lambda,v} = |\omega_v|$ for $\lambda \in [0, \infty)$ and $\tau_{\infty,v} = 0$. If v satisfies (7.14), set $y_v[k] = y_v[k-1] - 1$ and continue to the next iteration.

If no such node v exists, iterate over all edges $(u,v) \in E$ and check whether

$$K_{uv} \beta_3^\lambda (y_{uv}[k-1]) + \sum_{\hat{u} \in N(v) \setminus \{u\}} K_{\hat{u}v} \beta_2^\lambda (y_{\hat{u}v}[k-1]) > \tau_{\lambda,v} \tag{7.15}$$

$$K_{uv} \beta_3^\lambda (y_{uv}[k-1]) + \sum_{\hat{v} \in N(u) \setminus \{v\}} K_{u\hat{v}} \beta_2^\lambda (y_{u\hat{v}}[k-1]) > \tau_{\lambda,u} \tag{7.16}$$

where $\tau_{\lambda,v}$ and $\tau_{\lambda,u}$ are defined as in the previous paragraph. If (u,v) satisfies (7.15) and (7.16), set $y_{uv}[k] = y_{uv}[k-1] - 1$ and continue. If no such edge (u,v) exists, the algorithm terminates.

Since each index y_v and y_{uv} can be decremented at most M times, the algorithm terminates within $M(n+p)$ iterations. The complexity of the algorithm depends on M, which determines the number of iterations, as well as $|\Psi|$, which determines the workload per iteration. Larger values of M and $|\Psi|$ will increase the complexity, but will also result in a tighter sufficient condition.

The following theorem establishes that the algorithm of this section verifies that no stable non-cohesive equilibria exist.

Theorem 7.5. *Let* $\mathbf{y}^*$ *denote the vector of indices upon termination. Define* Φ^* *as*

$$\Phi^* \triangleq \left\{ \boldsymbol{\theta} : |\theta_v| \leq \frac{\pi y_v^*}{M} \ \forall v \in V, |f(\theta_v - \theta_u)| \leq \frac{\pi y_{uv}^*}{M} \ \forall (u,v) \in E \right\}.$$

Then all stable equilibria lie in Φ.

Proof. Define

$$\Phi_k \triangleq \left\{ \boldsymbol{\theta} : |\theta_v| \leq \frac{\pi y_v[k]}{M} \ \forall v \in V, |f(\theta_v - \theta_u)| \leq \frac{\pi y_{uv}[k]}{M} \ \forall (u,v) \in E \right\},$$

so that $\Phi_k \subseteq \Phi_{k-1}$. It suffices to show that at each iteration k, all stable equilibria lie in Φ_k. This result is trivially true when $k = 0$.

At $k > 0$, it suffices to show that there are no stable equilibria in $\Phi_{k-1} \setminus \Phi_k$. Suppose that v satisfies $y_v[k] = y_v[k-1] - 1$. Then

$$\Phi_{k-1} \setminus \Phi_k = \left\{ \boldsymbol{\theta} : |\theta_v| \in \left[\frac{\pi(y_v[k-1]-1)}{M}, \frac{\pi y_v[k-1]}{M} \right], \right. \\ \left. |\theta_u| \leq \frac{\pi y_u[k-1]}{M}, |f(\theta_v - \theta_u)| \leq \frac{\pi y_{uv}[k-1]}{M} \right\}.$$

By (7.14), there exists $\lambda \in [0, \infty]$ such that, for all $\boldsymbol{\theta} \in \Phi_{k-1} \setminus \Phi_k$,

$$\begin{aligned} \sum_{u \in N(v)} K_{uv}[\sin(\theta_v - \theta_u) - \lambda \cos(\theta_v - \theta_u)] &\geq \min_{\boldsymbol{\theta} \in \Phi_{k-1} \setminus \Phi_k} \left\{ \sum_{u \in N(v)} K_{uv} \left[\sin(\theta_v - \theta_u) - \lambda \cos(\theta_v - \theta_u) \right] \right\} \\ &\geq \sum_{u \in N(v)} K_{uv} \beta_1^{\lambda}(y_v[k-1]) \qquad (7.17) \\ &> \tau_{\lambda, v} \qquad (7.18) \end{aligned}$$

where (7.17) follows from the definition of β_1^{λ} and (7.18) follows from (7.14). Hence the conditions of Lemma 7.5 are satisfied for $\Phi_{k-1} \setminus \Phi_k$, and so there are no stable equilibria in $\Phi_{k-1} \setminus \Phi_k$.

Now, suppose that $(u,v) \in E$ satisfies $y_{uv}[k] = y_{uv}[k-1] - 1$. We have that

$$\Phi_{k-1} \setminus \Phi_k = \left\{ \boldsymbol{\theta} : |\theta_v| \leq \frac{\pi y_v[k-1]}{M}, \ |f(\theta_{\hat{v}} - \theta_{\hat{u}})| \leq \frac{\pi y_{\hat{u}\hat{v}}[k-1]}{M}, \right. \\ \left. |f(\theta_v - \theta_u)| \in \left[\frac{\pi(y_{uv}[k-1]-1)}{M}, \frac{\pi y_{uv}[k-1]}{M} \right] \right\}.$$

By (7.15) and (7.16), there exists λ such that

$$K_{uv} g_{\lambda}(\theta_v - \theta_u) + \sum_{\hat{u} \in N(v)} K_{\hat{u}v} g_{\lambda}(\theta_v - \theta_{\hat{u}}) > \tau_{\lambda, v},$$

and by Lemma 7.6 there are no stable equilibria in $\Phi_{k-1} \setminus \Phi_k$. Hence, by induction, all stable equilibria lie in Φ^*. $\square$

The following corollary gives explicit conditions for node and edge cohesiveness.

Corollary 7.2. *If y^*_{uv} satisfies $\frac{y^*_{uv}\pi}{M} \leq \gamma$ for all $(u,v) \in E$, then all stable equilibria are edge-cohesive. If y^*_v satisfies $\frac{y^*_v\pi}{M} \leq \gamma$ for all $v \in V$, then all stable equilibria are node-cohesive.*

Sections 7.2.3 and 7.2.5 give sufficient conditions for existence of stable cohesive equilibria and non-existence of stable non-cohesive equilibria; together, these constitute sufficient conditions for global practical synchronization from a given set of inputs. In the following section, we present an approach for selecting inputs that satisfy these conditions.

7.3 Submodular Optimization Approach to Input Selection

In this section, we present our submodular optimization approach to selecting a set of input nodes to guarantee practical node or edge synchronization from almost any initial state. Our approach is based on ensuring that the conditions for existence of a cohesive equilibrium (Section 7.2.3) and non-existence of stable non-cohesive equilibria (Section 7.2.5) are satisfied.

Section 7.3.1 formulates the condition for existence of a cohesive equilibrium as a submodular constraint on the input set. We first define an augmented graph and prove that the conditions for existence shown in Section 7.2.3 are equivalent to connectivity of a class of random subgraphs of the augmented graph. We then show that this connectivity condition can be expressed as a constraint of the form $h(A) = n$, where h is a submodular function.

Section 7.3.2 formulates non-existence of stable non-cohesive equilibria as a submodular constraint on the input set. As in Section 7.3.1, we first construct an augmented graph, although the construction of this section differs from Section 7.3.2. We then prove that the conditions identified in Section 7.2.5 are equivalent to connectivity of a class of subgraphs of the augmented graph, and show that this connectivity condition is equivalent to a submodular constraint on the input set.

In Section 7.3.3, we formulate the problem of selecting a set of input nodes to achieve practical synchronization as an optimization problem that combines the constraints of Sections 7.3.1 and 7.3.2. We then present algorithms for selecting a set of input nodes and

analyze the optimality guarantees of those algorithms.

7.3.1 Submodular Constraint for Cohesive Equilibrium Existence

We first define an augmented network graph, and then describe the sufficient condition for existence of a cohesive equilibrium based on the augmented graph. We then prove that selecting a set of input nodes to satisfy this condition can be formulated as a submodular constraint.

Let $M \in \mathbb{Z}^+$ as in Section 7.2.3. The augmented graph $G' = (V', E')$ consists of a set of nodes

$$V' = \{v_m : m = 0, \dots, M,\ v \in V\} \cup \{(u, v)_r : r = 0, \dots, M,\ (u, v) \in E\}.$$

The augmented graph is directed, with $(x, y) \in E'$ if there is a directed edge from node x to node y. The set $N(v_m)$ denotes the set of nodes with outgoing edges to v_m. The neighbor set of node v_m is defined by

$$N(v_m) = \{u_l : u \in N(v), l = 0, \dots, M\} \cup \{v_l : l < m\} \cup \{(u, v)_r : u \in N(v), r = 0, \dots, M\}.$$

The set $N((u, v)_r)$ denotes the set of nodes with outgoing edges to $((u, v)_r)$, and is defined by

$$\begin{aligned} N((u, v)_r) = {} & \{u_l : l = 0, \dots, r\} \cup \{v_m : m = 0, \dots, r\} \cup \{(u, v)_{r'} : r' < r\} \\ & \cup \{(\hat{u}, v)_s : \hat{u} \in N(v), s = 0, \dots, M\} \cup \{(u, \hat{v})_s : \hat{v} \in N(u), s = 0, \dots, M\} \end{aligned}$$

We define a subset of V', denoted V'_0, by $V'_0 = \{v_0 : v \in A\} \cup \{v_M : v \notin A\}$. We now define a class of subgraphs of G' that will be used to formulate our sufficient conditions for existence of cohesive equilibria.

Definition 7.7. *A subgraph $\tilde{G} \subseteq G'$ is of* class $\mathcal{T}$ *if the neighbor sets satisfy the following properties. For each node v_m, the neighbor set is a set $\mathcal{T}(v_m)$ satisfying*

$$\begin{aligned} \sum_{u \in N(v)} K_{uv} \left[\sum_{u_l \in \mathcal{T}(v_m)} \alpha_1(m, l) \right] > \sum_{u \in N(v)} K_{uv} \left[\sum_{u_l \in N(v_m)} \alpha_1(m, l) \right] \\ - \left(|\omega_v| - \overline{K}_v \beta_1(m, M) \right), \quad (7.19) \end{aligned}$$

where $\overline{K}_v = \sum_{u \in N(v)} K_{uv}$, $\beta_1(m,l)$ *is defined as in Section 7.2.3, and* $\alpha_1(m,l) = \beta_1(m,l) - \beta_1(m,l+1)$. *Furthermore, for each* $u \in N(v)$, *if* $u_l \notin \mathcal{T}(v_m)$, *then* $(u,v)_{m-l-1} \in \mathcal{T}(v_m)$. *Finally,* $v_{m'} \in \mathcal{T}(v_m)$ *for all* $m' < m$.

For each node $(u,v)_r$, *the neighbor set* $\mathcal{T}((u,v)_r)$ *satisfies*

$$\sum_{(\hat{u},v)_s \in \mathcal{T}((u,v)_r)} K_{\hat{u}v}\alpha_3(s) > \sum_{(\hat{u},v)_s \in N((u,v)_r)} K_{\hat{u}v}\alpha_3(s) - \big(\tau_v - (\overline{K}_v - K_{uv})\beta_3(M) - K_{uv}\beta_2(r)\big)$$

$$\sum_{(u,\hat{v})_s \in \mathcal{T}((u,v)_r)} K_{u\hat{v}}\alpha_3(s) > \sum_{(\hat{u},v)_s \in N((u,v)_r)} K_{\hat{u}v}\alpha_3(s) - \big(\tau_u - (\overline{K}_u - K_{uv})\beta_3(M) - K_{uv}\beta_2(r)\big)$$

where $\alpha_3(s) = \beta_3(s) - \beta_3(s+1)$. *Finally, for each* $l = 0, \ldots, r$, *exactly one of* u_l *or* v_{r-l-1} *is in* $\mathcal{T}((u,v)_r)$, *and* $\{(u,v)_{r'} : r' < r\} \subseteq \mathcal{T}((u,v)_r)$.

We now define a sufficient condition for existence of cohesive equilibria based on the class $\mathcal{T}$ subgraphs.

Theorem 7.6. *In the algorithm of Section 7.2.3,* $v \in S$ *if and only if, for each class* $\mathcal{T}$ *subgraph* $\tilde{G}$, *there exists a directed path from at least one node in* V_0' *to* v_{M-1}. *We have* $(u,v) \in S$ *if and only if, for each class* $\mathcal{T}$ *subgraph* $\tilde{G}$, *there exists a directed path from at least one node in* V_0' *to* $(u,v)_{M-1}$.

Proof. We first prove that if $v \in S$, then there exists a directed path from at least one node in V_0' to v_{M-1}, and that if $(u,v) \in S$, then there exists a directed path from at least one node in V_0' to $(u,v)_{M-1}$, in each class $\mathcal{T}$ subgraph of V'. Our approach is to prove by induction on k that, at each iteration k, the nodes

$$T[k] = \{v_m : m \geq x_v[k]\} \cup \{(u,v)_r : r \geq x_{uv}[k]\}$$

are connected to V_0' in each class $\mathcal{T}$ subgraph. At iteration $k = 0$, $T[0] = V_0'$.

Suppose that, at iteration k, node v satisfies $x_v[k] = x_v[k-1] - 1$. Suppose first that

$$\sum_{u \in N(v)} K_{uv}\beta_1(x_v[k-1], x_u[k-1]) > |\omega_v| \tag{7.20}$$

holds. Setting $m = x_v[k-1]$, Eq. (7.20) can be rewritten as

$$\sum_{u \in N(v)} \sum_{l=x_u[k-1]}^{M-1} K_{uv}\alpha_1(m,l) > -\overline{K}_v\beta_1(m,M) + |\omega_v|,$$

which is equivalent to

$$\sum_{u_l \in N(v_m) \cap T[k]} K_{uv}\alpha_1(m,l) \geq -\overline{K}\beta_1(m,M) + |\omega_v|.$$

Now, if $\tilde{G}$ is a class $\mathcal{T}$ subgraph, then we have that $\mathcal{T}(v_m)$ satisfies (7.19). Combining these equations yields

$$\sum_{u_l \in N(v_m) \cap T[k]} K_{uv}\alpha_1(m,l) + \sum_{u_l \in \mathcal{T}(v_m)} K_{uv}\alpha_1(m,l) > \sum_{u_l \in N(v_m)} K_{uv}\alpha_1(m,l).$$

Since $K_{uv}\alpha_1(m,l) \geq 0$ for all u, v, m, and l, we must have $T[k] \cap \mathcal{T}(v_m) \neq \emptyset$. By inductive hypothesis, each node in $T[k]$ is connected to a node in V_0', and hence v_m is connected to V_0'.

Now suppose that $x_u[k-1] + x_{uv}[k-1] < x_v[k-1]$. Let $l = x_u[k-1]$ and $r = x_{uv}[k-1]$. By induction, both u_l and $(u,v)_r$ are connected to V_0'. By definition of class $\mathcal{T}$ subgraph, either $u_l \in \mathcal{T}(v_m)$ or $(u,v)_{m-l-1} \in \mathcal{T}(v_m)$ and $(u,v)_r$ is connected to $(u,v)_{m-l-1}$. Hence v_m is connected to V_0' in this case.

If at iteration k, an edge (u,v) satisfies $x_{uv}[k] = x_{uv}[k-1] - 1$, then we have

$$\beta_2(x_{uv}[k-1]) + \sum_{\hat{u} \in N(v) \setminus \{u\}} \beta_3(x_{\hat{u}v}[k-1]) \geq \frac{|\omega_v|}{K} \tag{7.21}$$

if (7.6) holds (the case where (7.7) holds is similar). The expression can be rewritten as

$$\sum_{(\hat{u},v)_s \in N(v_m) \cap T[k]} K_{\hat{u}v}\alpha_3(s) + \sum_{(\hat{u},v)_s \in \mathcal{T}((u,v)_r)} K_{\hat{u}v}\alpha_3(s) > \sum_{(\hat{u},v)_s \in N((u,v)_r)} K_{\hat{u}v}\alpha_3(s),$$

which implies that $T[k] \cap \mathcal{T}((u,v)_r) \neq \emptyset$. By induction $T[k]$ is connected to V_0', and hence $(u,v)_r$ is connected to V_0' as well.

Finally, if $x_u[k-1] + x_v[k-1] < x_{uv}[k-1]$, let $l = x_u[k-1]$ and $m = x_v[k-1]$. We have that either $u_l \in \mathcal{T}((u,v)_r)$ or $v_{r-l-1} \in \mathcal{T}((u,v)_r)$ and v_m is connected to v_{r-l-1}. Since u_l and v_m are both connected to V_0' by induction, $(u,v)_r$ is connected to V_0'. This completes the proof that if $v \in S$ (resp. $(u,v) \in S$), then V_0' is connected to v_{M-1} (resp. $(u,v)_{M-1}$) in any class $\mathcal{T}$ subgraph of V'.

We now show that if v_{M-1} is connected to V_0' in every class $\mathcal{T}$ subgraph of V', then $v \in S$. We first assume that $v \notin S$, and then construct a class $\mathcal{T}$ subgraph in which V_0' is

not connected to v_{M-1}. In particular, we construct the set of nodes that are connected to v_{M-1}, denoted $\mathcal{D}$, and prove that $\mathcal{D} \cap V_0' = \emptyset$.

Initialize the set $\mathcal{D} \subseteq V'$ as $\mathcal{D} = \{v_{M-1}\}$. Since $v \notin S$, we have that

$$\sum_{u \in N(v)} K_{uv} \beta_1(M, x_u^*) < |\omega_v|,$$

and hence

$$\sum_{u \in N(v)} \sum_{l=x_u^*}^{M-1} K_{uv} \alpha_1(M, l) < |\omega_v| - \overline{K}_v \beta_1(M, M).$$

Letting $\mathcal{T}(v_{M-1}) = \{u_l \in N(v_{M-1}) : l < x_u^*\}$, we have that $\mathcal{T}(v_{M-1})$ satisfies (7.19). Finally, for all $u_l \notin \mathcal{T}(v_m)$ with $u \in N(v)$, add $(u, v)_{m-l-1}$ to $\mathcal{T}(v_m)$; since $x_v^* > m$, $x_{uv}^* > m - l - 1$.

Setting $\mathcal{D} = \mathcal{D} \cup \mathcal{T}(v_{M-1})$, we have that $\mathcal{D}$ consists of nodes u_l that satisfy $x_u^* > l$ and $u_l \notin V_0'$. Proceeding via induction, we select a node $u_l \in \mathcal{D}$ such that the neighbor set $\mathcal{T}(u_l)$ has not been assigned yet. We have that u_l satisfies

$$\sum_{s \in N(u)} \sum_{l'=x_s^*}^{M-1} K_{su} \alpha_1(l, l') < |\omega_v| - \overline{K}_u \beta_1(l, M).$$

By a similar argument to the above, we select $\mathcal{T}(u_l) = \{s_{l'} \in N(u_l) : l' < x_s^*\}$. We then set $\mathcal{D} = \mathcal{D} \cup \mathcal{T}(u_l)$, noting that the properties $x_s^* > l'$ for $s_{l'} \in \mathcal{D}$ and $\mathcal{D} \cap V_0' = \emptyset$ have been preserved.

If all nodes $u_l \in \mathcal{D}$ have had $\mathcal{T}(u_l)$ assigned, select a node $(u, v)_r \in \mathcal{D}$ such that $\mathcal{T}((u, v)_r)$ has not been assigned. By induction, $r < x_{uv}^*$, and hence

$$\sum_{\hat{u} \in N(v) \setminus \{u\}} K_{\hat{u}v} \beta_3(x_{\hat{u}v}^*) > |\omega_v| - K_{uv} \beta_2(r)$$

and

$$\sum_{\hat{v} \in N(u) \setminus \{v\}} K_{u\hat{v}} \beta_3(x_{u\hat{v}}^*) > |\omega_u| - K_{uv} \beta_2(r).$$

We therefore set $\mathcal{T}((u, v)_r) = \{(\hat{u}, v)_s : s < x_{\hat{u}v}^*\} \cup \{(u, \hat{v})_s : s < x_{u\hat{v}}^*\}$, and note that $\mathcal{T}((u, v)_r)$ satisfies the criteria for a class $\mathcal{T}$ subgraph. Finally, since $r < x_{uv}^*$, for each l either $x_u^* > l$ (in which case, add u_l to $\mathcal{T}(v_m)$) or $x_v^* > r - l$ (in which case, add v_{r-l}

to $\mathcal{T}(v_m)$). If all nodes in $\mathcal{D}$ have had their neighbor sets assigned, then the procedure terminates.

At the end of this procedure, we have a class $\mathcal{T}$ subgraph $\tilde{G}$ and a set of nodes $\mathcal{D}$, which consists of all nodes that are connected to v_{M-1} in $\tilde{G}$. Since $\mathcal{D} \cap V_0' = \emptyset$ by induction, V_0' is not connected to v_{M-1}. This proves the other direction of the theorem. □

The following corollary gives explicit conditions for existence of node and edge cohesive equilibria based on the connectivity of class $\mathcal{T}$ subgraphs.

Corollary 7.3. *Suppose that a class $\mathcal{T}$ subgraph of G' is chosen at random, according to a probability distribution that assigns nonzero weight to each class $\mathcal{T}$ subgraph. If each node $\{v_{M-1} : v \in V\}$ is connected to V_0' with probability one, then a node cohesive equilibrium exists. If each node $\{(u,v)_{M-1} : (u,v) \in E\}$ is connected to V_0' with probability one, then an edge cohesive equilibrium exists.*

The following known result will lead to a submodular constraint for existence of cohesive equilibria.

Lemma 7.7 ([68])**.** *Let $G = (V, E)$ be a directed graph, and let $A \subseteq V$ and $U \subseteq V$. Define $f(A)$ to be the expected number of nodes in U that are connected to A in a random subgraph of G with known distribution. Then $f(A)$ is submodular as a function of A.*

Combining Lemma 7.7 with Corollary 7.3 yields the following.

Lemma 7.8. *Let A denote the set of input nodes, and let $r_1(A)$ denote the expected number of nodes in $\{v_{M-1} : v \in V\}$ that are connected to V_0' in a randomly chosen class $\mathcal{T}$ subgraph. Then $r_1(A)$ is submodular as a function of A. If $r_1(A) = n$, then a node cohesive equilibrium exists.*

Proof. The submodularity of $r_1(A)$ follows from Lemma 7.7. If $r_1(A) = n$, then for any class $\mathcal{T}$ subgraph of G', all nodes in $\{v_{M-1} : v \in V\}$ are connected to at least one node in A. The existence of an edge cohesive equilibrium follows from Theorem 7.6. □

Analogously, if we define $r_1'(A)$ to be the expected number of nodes in $\{(u,v)_{M-1} : (u,v) \in E\}$ that are connected to V_0' in a randomly chosen class $\mathcal{T}$ subgraph, then $r_1'(A) = p$,

where $p = |E|$ as in Section 7.1.1, implies the existence of an edge cohesive equilibrium, and $r_1'(A)$ is submodular.

7.3.2 Submodular Constraint for Non-Existence of Non-Cohesive Equilibria

In this section, we present a submodular constraint on the set of input nodes for guaranteeing non-existence of non-node cohesive and non-edge cohesive equilibria. As in Section 7.3.1, our approach is to construct an augmented graph and prove that the desired property is equivalent to connectivity of a random subgraph of the augmented graph.

Let $\beta_i^\lambda(m,l)$ be defined as in Section 7.2.5. We define $\alpha_1^\lambda(m,l) = \beta_1^\lambda(m,l) - \beta_1^\lambda(m,l+1)$ and $\alpha_i^\lambda(r) = \beta_i^\lambda(r) - \beta_i^\lambda(r+1)$ for $i = 2,3$. Let $M \in \mathbb{Z}^+$. The vertex set of the augmented graph is given by

$$V'' = \{v_m : m = 0,\dots,M\} \cup \{(u,v)_r : (u,v) \in E, r = 0,\dots,M\}.$$

The nodes v_m represent whether node v has all stable equilibria in the region $|\theta_v| \le \frac{\pi m}{M}$, while the nodes $\{(u,v)_r\}$ represent whether all stable equilibria satisfy $|f(\theta_v - \theta_u)| \le \frac{\pi r}{M}$.

The edge set E'' of the graph is defined by assigning the following neighbor sets to each node. For each node v_m, the neighbor set is given by

$$N(v_m) = \{v_{m'} : m' < m\} \cup \{u_l : u \in N(v), l = 0,\dots,M\} \cup \{(u,v)_r : u \in N(v), r = 0,\dots,m\}.$$

The nodes $(u,v)_r$ have neighbor sets defined by

$$\begin{aligned} N((u,v)_r) = {} & \{(u,v)_{r'} : r' < r\} \cup \{(\hat{u},v)_s : \hat{u} \in N(v) \setminus \{u\}, s = 0,\dots,M\} \\ & \cup \{(u,\hat{v})_s : \hat{v} \in N(u) \setminus \{v\}, s = 0,\dots,M\} \\ & \cup \{u_l : l = 0,\dots,r\} \cup \{v_m : m = 0,\dots,r\} \end{aligned}$$

Define

$$V_0'' = \{v_0 : v \in A\} \cup \{v_M : v \notin A\} \cup \{(u,v)_M : (u,v) \in E\}.$$

We now define a *class $\mathcal{U}$ subgraph* of the augmented graph $G'' = (V'', E'')$. We will show that sufficient conditions for non-existence of non-cohesive stable equilibria can be formulated based on the connectivity of randomly-chosen class $\mathcal{U}$ subgraphs of G''.

Definition 7.8. *A subgraph $\tilde{G} = (\tilde{V}, \tilde{E})$ is a class $\mathcal{U}$ subgraph of G'' if $\tilde{V} = V''$ and the neighbor sets satisfy the following properties. Let $\Psi = \{\lambda_1, \ldots, \lambda_R\} \subset [0, \infty]$. For each node $v_m \in V''$ and $\lambda \in \Psi$, the neighbor set $\mathcal{U}(v_m)$ satisfies*

$$\sum_{u_l \in \mathcal{U}(v_m)} K_{uv}\alpha_1^\lambda(m,l) \geq \sum_{u_l \in N(v_m)} K_{uv}\alpha_1^\lambda(m,l) + \overline{K}_v\beta_1^\lambda(m,M) - \tau_{\lambda,v}. \tag{7.22}$$

For each $u \in N(v)$ and $l = 0, \ldots, m$, if $u_l \notin \mathcal{U}(v_m)$, then $(u,v)_{m-l-1} \in \mathcal{U}(v_m)$. Finally, $\{v_{m'} : m' < m\} \subseteq \mathcal{U}(v_m)$.

For each node $(u,v)_r \in V''$, the neighbor set $\mathcal{U}((u,v)_r)$ satisfies

$$\sum_{\hat{u} \in N(v)\setminus\{u\}} \sum_{(\hat{u},v)_s \in \mathcal{U}((u,v)_r)} K_{\hat{u}v}\alpha_2^\lambda(s) \geq K_{uv}\beta_3^\lambda(0) + (\overline{K}_v - K_{uv})(\beta_2^\lambda(0) - \beta_2^\lambda(M)) - \tau_{\lambda,v} - K_{uv}\beta_3^\lambda(r) \tag{7.23}$$

$$\sum_{\hat{v} \in N(u)\setminus\{v\}} \sum_{(u,\hat{v})_s \in \mathcal{U}((u,v)_r)} K_{u\hat{v}}\alpha_2^\lambda(s) \geq K_{uv}\beta_3^\lambda(0) + (\overline{K}_v - K_{uv})(\beta_2^\lambda(0) - K_{uv}\beta_2^\lambda(M)) - \tau_{\lambda,u} - \beta_3^\lambda(r) \tag{7.24}$$

For each $l = 0, \ldots, r$, either $u_l \in \mathcal{U}((u,v)_r)$ or $v_{r-l-1} \in \mathcal{U}((u,v)_r)$. Finally, $\{(u,v)_{r'} : r' < r\} \subseteq \mathcal{U}((u,v)_r)$.

In the definition of the neighbor set $\mathcal{U}(v_m)$, each constraint (7.22) is related to one of the constraints (7.14) of Section 7.2.5. Similarly, the constraints (7.23) and (7.24) map to the constraints (7.15) and (7.16) of Section 7.2.5. Formally, this connection is established by the following theorem, which relates the non-existence of stable non-cohesive equilibria to the connectivity of class $\mathcal{U}$ subgraphs.

Theorem 7.7. *Consider the algorithm of Section 7.2.5. The final index value of node v, y_v^*, satisfies $y_v^* \leq m$ if and only if v_m is connected to V_0'' in all class $\mathcal{U}$ subgraphs.*

Proof. We first show that if $y_v^* \leq m$, then v_m is connected to V_0'' in all class $\mathcal{U}$ subgraphs. Our approach is to prove that, at each iteration k of the algorithm, $m = y_v[k]$ implies that v_m is connected to V_0'' and $r = y_{uv}[k]$ implies that $(u,v)_r$ is connected to V_0''. At iteration 0, the result follows from the definition of V_0'' and the initialization of the algorithm.

At iteration k, let

$$T[k] = \{v_m : m \geq y_v[k]\} \cup \{(u,v)_r : r \geq y_{uv}[k]\}.$$

Suppose that v satisfies $y_v[k] = y_v[k-1] - 1$ and let $m = y_v[k]$. If (7.14) holds, then we have

$$\sum_{u \in N(v)} K_{uv} \beta_1^\lambda(y_v[k-1], y_u[k-1]) > \tau_{\lambda,v}.$$

Applying a telescoping sum argument yields

$$\sum_{u \in N(v)} K_{uv} \left[\sum_{l=y_u[k-1]}^{M-1} \alpha_1^\lambda(m,l) + \beta_1^\lambda(m,M) \right] > \tau_{\lambda,v}.$$

By rearranging terms and using the definition of $T[k]$, we have

$$\sum_{u_l \in T[k] \cap N(v_m)} K_{uv} \alpha_1^\lambda(m,l) > \tau_{\lambda,v} - \overline{K}_v \beta_1^\lambda(m,M). \tag{7.25}$$

Adding (7.22) and (7.25) yields

$$\sum_{u_l \in T[k] \cap N(v_m)} K_{uv} \alpha_1^\lambda(m,l) + \sum_{u_l \in \mathcal{U}(v_m)} K_{uv} \alpha_1^\lambda(m,l) > \sum_{u_l \in N(v_m)} K_{uv} \alpha_1^\lambda(m,l).$$

Since $K_{uv}\alpha_1^\lambda(m,l) \geq 0$ and $(T[k] \cap N(v_m))$ and $\mathcal{U}(v_m)$ are both subsets of $N(v_m)$, we must have $T[k] \cap \mathcal{U}(v_m) \neq \emptyset$. Since $T[k]$ is connected to V_0'' by inductive hypothesis, v_m is connected to V_0'' in $\tilde{G}$.

If $y_{uv}[k-1] + y_u[k-1] < m$ for some $u \in N(v)$, then let $r = y_{uv}[k-1]$ and $l = y_u[k-1]$. By induction, both $(u,v)_r$ and u_l are connected to V_0''. Furthermore, by definition of $\mathcal{U}(v_m)$, either $(u,v)_r$ or u_{m-r-1} is in $\mathcal{U}(v_m)$. If $(u,v)_r \in \mathcal{U}(v_m)$, v_m is connected to V_0''. If $u_{m-r-1} \in \mathcal{U}(v_m)$, then $l \leq m-r-1$ implies that V_0'' is connected to u_l and u_l is connected to v_m. Hence v_m is connected to V_0''.

Now, suppose that an edge $(u,v) \in E$ satisfies $y_{uv}[k] = y_{uv}[k-1] - 1$. Let $r = y_{uv}[k]$. We show that $(u,v)_r$ is connected to V_0''. If (7.15) holds for some $\lambda \in \Psi$, then

$$K_{uv}\beta_2^\lambda(r) + \sum_{\hat{u} \in N(v) \setminus \{u\}} K_{\hat{u}v} \beta_3^\lambda(y_{\hat{u}v}[k-1]) > \tau_{\lambda,v}.$$

A telescoping sum argument implies that

$$K_{uv}\beta_2^{\lambda}(r) + \sum_{\hat{u}\in N(v)\setminus\{u\}} K_{\hat{u}v}\left[\sum_{s=y_{\hat{u}v}[k-1]}^{M-1} \alpha_3^{\lambda}(s)\right] > \tau_{\lambda,v}. \tag{7.26}$$

Adding (7.23) and (7.26) and rearranging terms gives

$$\sum_{\hat{u}\in N(v)\setminus\{u\}} \sum_{(\hat{u},v)_s\in T[k]} K_{\hat{u}v}\alpha_3^{\lambda}(s) + \sum_{\hat{u}\in N(v)\setminus\{u\}} \sum_{(\hat{u},v)_s\in\mathcal{U}((u,v)_r)} K_{\hat{u}v}\alpha_3^{\lambda}(s) > \sum_{\hat{u}\in N(v)\setminus\{u\}} \sum_{(\hat{u},v)_s\in N(v_m)} K_{\hat{u}v}\alpha_3^{\lambda}(s). \tag{7.27}$$

Hence $T[k]\cap\mathcal{U}((u,v)_r) \neq \emptyset$, and so $(u,v)_r$ is connected to V_0''. A similar argument establishes connectivity if (7.16) holds.

If $y_u[k-1] + y_v[k-1] < r$, let $y_u[k-1] = l$ and $y_v[k-1] = m$. Either $u_l \in \mathcal{U}((u,v)_r)$, in which case $(u,v)_r$ is connected to V_0'', or $v_{r-l-1} \in \mathcal{U}((u,v)_r)$. In the latter case, $y_v[k-1] \leq (r-l-1)$ implies that V_0'' is connected to v_m, which is in turn connected to $(u,v)_r$.

To show the other direction of the theorem, suppose that $y_v^* > m$. We will construct a class $\mathcal{U}$ subgraph $\tilde{G}$ such that v_m is not connected to V_0''. Let $\mathcal{D}$ be a subset of V'' and initialize $\mathcal{D} = \{v_m\}$. The set $\mathcal{D}$ will be equal to the set of nodes that are connected to v_m in $\tilde{G}$; we will grow $\mathcal{D}$ through an iterative procedure and prove that, upon termination of the procedure, $\mathcal{D}\cap V_0'' = \emptyset$, and hence v_m is not connected to V_0''.

First, we define $\mathcal{U}(v_m) = \{u_l : l < x_u^*\}$. We will show that $\mathcal{U}(v_m)$ satisfies (7.22). Since $m < x_v^*$, we have

$$\sum_{u\in N(v)} K_{uv}\beta_1^{\lambda}(m, x_u^*) < |\omega_v|.$$

Writing $\beta_1^{\lambda}(m, x_u^*)$ as a telescoping sum gives

$$\sum_{u\in N(v)} K_{uv}\left[\sum_{l=x_u^*}^{M-1} \alpha_1^{\lambda}(m,l) + \beta_1^{\lambda}(m,M)\right] < |\omega_v|.$$

Adding

$$\sum_{u\in N(v)} \sum_{l=0}^{x_u^*-1} K_{uv}\alpha_1^{\lambda}(m,l)$$

to both sides and rearranging terms gives

$$\sum_{u\in N(v)}\sum_{l=0}^{x_u^*-1} K_{uv}\alpha_1^\lambda(m,l) > \sum_{u\in N(v)}\sum_{l=0}^{M-1} K_{uv}\alpha_1^\lambda(m,l) - \left(|\omega_v| - \overline{K}_v\beta_1^\lambda(m,M)\right),$$

implying that (7.22) holds with $\mathcal{U}(v_m) = \{u_l : l < x_u^*\}$.

In order to fully define $\mathcal{U}(v_m)$, for each $l = 0,\dots,m$, if $l > y_u^*$, we add $(u,v)_{m-l-1}$ to $\mathcal{U}(v_m)$. Since $y_v^* > m$,

$$m < y_v^* < (y_{uv}^* + y_u^*) < y_{uv}^* + l,$$

and hence $y_{uv}^* > m - l > m - l - 1$. Set $\mathcal{D} = \mathcal{D}\cup\mathcal{U}(v_m)$.

We proceed by induction. At each step, we have a set $\mathcal{D}$ consisting of elements that are connected to v_m in $\tilde{G}$, do not belong to V_0'', and are of the form u_l for some $l < y_u^*$ or $(u,v)_r$ for $r < y_{uv}^*$. Select an element $u_l \in \mathcal{D}$ and choose its neighbor set as $\mathcal{U}(u_l) = \{w_s : w \in N(u), s < y_w^*\}$. By the argument given above, $\mathcal{U}(u_l)$ satisfies (7.22). Set $\mathcal{D} = \mathcal{D}\cup\mathcal{U}(u_l)$.

If neighbor sets have been assigned to all nodes $u_l \in \mathcal{D}$, select a node $(u,w)_r \in \mathcal{D}$. The neighbor set $\mathcal{U}((u,w)_r)$ is defined by

$$\mathcal{U}((u,w)_r) = \bigcup_{\hat{w}\in N(u)\setminus\{w\}} \{(u,\hat{w})_s : s < y_{u\hat{w}}^*\} \cup \left(\bigcup_{\hat{u}\in N(w)\setminus\{u\}} \{(\hat{u},w)_s : s < y_{\hat{u}w}^*\}\right)$$

By induction, $r < y_{uw}^*$. Hence (7.15) does not hold, and

$$\sum_{\hat{w}\in N(u)\setminus\{w\}} K_{u\hat{w}}\beta_2^\lambda(y_{u\hat{w}}^*) < \tau_{u,\lambda} - K_{wu}\beta_3^\lambda(r).$$

Employing a telescoping sum gives

$$\sum_{\hat{w}\in N(u)\setminus\{w\}} K_{u\hat{w}}\left[\sum_{s=y_{u\hat{w}}^*}^{M-1}\alpha_2^\lambda(s) + \alpha_2^\lambda(M)\right] < \tau_{\lambda,u} - K_{uw}\beta_3^\lambda(r).$$

Adding

$$\sum_{\hat{w}\in N(u)\setminus\{w\}}\sum_{l=0}^{y_{\hat{u}w}^*-1} K_{u\hat{w}}\alpha_2^\lambda(s)$$

to both sides and rearranging terms yields

$$\sum_{\hat{w}\in N(u)\setminus\{w\}}\sum_{l=0}^{y_{u\hat{w}}^*-1} K_{u\hat{w}}\alpha_2^\lambda(s) > \sum_{\hat{w}\in N(u)\setminus\{w\}}\sum_{s=0}^{M-1} K_{u\hat{w}}\alpha_2^\lambda(s) - \left(\tau_{\lambda,u} - K_{uw}\beta_3^\lambda(r)\right),$$

and so $\mathcal{U}((u,w)_r)$ satisfies (7.23). The proof that (7.24) holds is similar. We then set $\mathcal{D} = \mathcal{D} \cup \mathcal{U}((u,w)_r)$ and continue.

At the end of the inductive procedure, we have a set $\mathcal{D}$ consisting of all nodes that are connected to v_m. By induction, $\mathcal{D} \cap V_0'' = \emptyset$, and so we have constructed a class $\mathcal{U}$ subgraph in which V_0'' is not connected to v_m. □

Theorem 7.7 gives conditions on the stable equilibria of (7.2) based on the connectivity of class $\mathcal{U}$ subgraphs of G''. The following corollary gives conditions for non-existence of non-cohesive equilibria based on Theorem 7.7.

Corollary 7.4. *Let $m = \lfloor \frac{M\gamma}{\pi} \rfloor$. Then all stable equilibria are node-cohesive if v_m is connected to V_0'' in all class $\mathcal{U}$ subgraphs. All stable equilibria are edge-cohesive if $(u,v)_m$ is connected to V_0'' in all class $\mathcal{U}$ subgraphs.*

The following corollary restates these conditions in terms of random class $\mathcal{U}$ subgraphs.

Corollary 7.5. *Let $m = \lfloor \frac{M\gamma}{\pi} \rfloor$. Suppose that a class $\mathcal{U}$ subgraph is chosen at random from a probability distribution that assigns nonzero weight to each class $\mathcal{U}$ subgraph. Then all stable equilibria are node-cohesive if v_m is connected to V_0'' for all $v \in V$ with probability 1, and all stable equilibria are edge-cohesive if $(u,v)_m$ is connected to V_0'' for all $(u,v) \in E$ with probability 1.*

Define $r_2(A)$ to be the expected number of nodes in $\{v_m : m = \lfloor \frac{M\gamma}{\pi} \rfloor\}$ that are connected to V_0'' when the input set is A, and define $r_2'(A)$ to be the expected number of nodes in $\{(u,v)_m : m = \lfloor \frac{M\gamma}{\pi} \rfloor\}$ that are connected to V_0'' when the input set is A. By Lemma 7.7, $r_2(A)$ and $r_2'(A)$ are submodular as functions of A. The following lemma gives sufficient conditions for non-existence of non-cohesive stable equilibria based on $r_2(A)$ and $r_2'(A)$.

Lemma 7.9. *All equilibria are node-cohesive if $r_2(A) = n$. All equilibria are edge-cohesive if $r_2'(A) = p$.*

Proof. By Corollary 7.5, all stable equilibria are node-cohesive if all nodes in $\{v_m : m = \lfloor \frac{M\gamma}{\pi} \rfloor\}$ are connected to V_0'' with probability 1 in any random class $\mathcal{U}$ subgraph. Equivalently, all stable equilibria are node-cohesive if the expected number of nodes that are connected

to V_0'' in any randomly chosen class $\mathcal{U}$ subgraph is N, i.e., $r_1(A) = n$. The proof that all equilibria are edge-cohesive if $r_2'(A) = p$ is similar. □

Having derived constraints for existence of cohesive equilibria and non-existence of non-cohesive equilibria, we next combine these constraints to formulate selecting a set of input nodes to achieve practical synchronization as an optimization problem. We present algorithms that exploit the submodular structure of these constraints to provide provable optimality bounds.

7.3.3 Selecting Inputs for Practical Synchronization

By combining the conditions derived in Sections 7.3.1 and 7.3.2, we have that the nodes achieve practical node synchronization if $r_1(A) + r_2(A) = 2n$ and practical edge synchronization if $r_1'(A) + r_2'(A) = 2p$.

We consider two problems based on these constraints. In the first problem, we select the minimum-size set of input nodes in order to guarantee practical node (resp. edge) synchronization. In the second problem, we select a set of up to k input nodes in order to maximize the level of synchronization.

In the case of ensuring practical node synchronization, the first problem can be formulated as

$$\begin{aligned} &\text{minimize} && |A| \\ &\text{s.t.} && r_1(A) + r_2(A) = 2n \end{aligned} \tag{7.28}$$

The problem of guaranteeing practical edge synchronization has an analogous formulation with $r_1'(A)$ and $r_2'(A)$ replacing $r_1(A)$ and $r_2(A)$, respectively.

The solution to (7.28) can be approximated by initializing the set $A = \emptyset$ and, at each iteration, adding the node v to A that maximizes $r_1(A \cup \{v\}) + r_2(A \cup \{v\})$. The algorithm terminates when $r_1(A) + r_2(A) = 2n$. The following lemma gives an optimality bound for this approach.

Lemma 7.10. *Let A_{k-1} denote the set of input nodes at the second-to-last iteration of the algorithm, let A denote the set that is returned, and let A^* denote the minimum-size set of*

inputs to satisfy the conditions of Lemmas 7.8 and 7.9. Then

$$\frac{|A_k|}{|A^*|} \leq 1 + \ln\left(\frac{n}{r_1(A) + r_2(A) - r_1(A_{k-1}) - r_2(A_{k-1})}\right).$$

Proof. The lemma follows from the submodularity of r_1 and r_2, and Theorem 1 of [136]. □

The problem of selecting up to k input nodes in order to maximize node synchronization can be formulated as

$$\begin{aligned} &\text{maximize} && r_1(A) + r_2(A) \\ &\text{s.t.} && |A| \leq k \end{aligned} \tag{7.29}$$

In order to approximate (7.29), a greedy procedure analogous to the algorithm for (7.28) is as follows. Initialize the set $A = \emptyset$. At each iteration, add the node v to A that maximizes $r_1(A \cup \{v\}) + r_2(A \cup \{v\})$. The algorithm terminates after k iterations, when $|A| = k$. The optimality bounds of this algorithm is described by the following lemma.

Lemma 7.11. *Let A^* denote the optimal solution to (7.29), and let $\hat{A}$ denote the set of inputs chosen by the greedy algorithm. Then $r_1(\hat{A}) + r_2(\hat{A}) \geq \left(1 - \frac{1}{e}\right)(r_1(A^*) + r_2(A^*))$.*

Proof. The proof follows from submodularity of $r_1(A) + r_2(A)$ and [96, Theorem 4.1]. □

We observe that, since the number of class $\mathcal{T}$ and $\mathcal{U}$ subgraphs is large in general, the objective functions $r_1(A)$, $r_2(A)$, $r_1'(A)$, and $r_2'(A)$ can be approximated by generating a set of random class $\mathcal{T}$ and $\mathcal{U}$ subgraphs according to some probability distribution.

7.4 Numerical Results

We investigated our approach through a numerical study. The goal of our study was to analyze the impact of the external inputs on the oscillator dynamics, as well as to investigate the properties of the inputs chosen by our algorithm. We considered synchronization in three application domains: (i) synchronization of phase angles in the IEEE 14 Bus power system test case [1], (ii) time synchronization in wireless networks [9], modeled as a geometric random graph, and (iii) synchronization of firing rates in the *C. Elegans* neuronal network [132]. The results of each study are summarized as follows. In all cases, the constant $M = 10$ and the number of random subgraphs used to compute $r_1(A)$ and $r_2(A)$ was equal to 20.

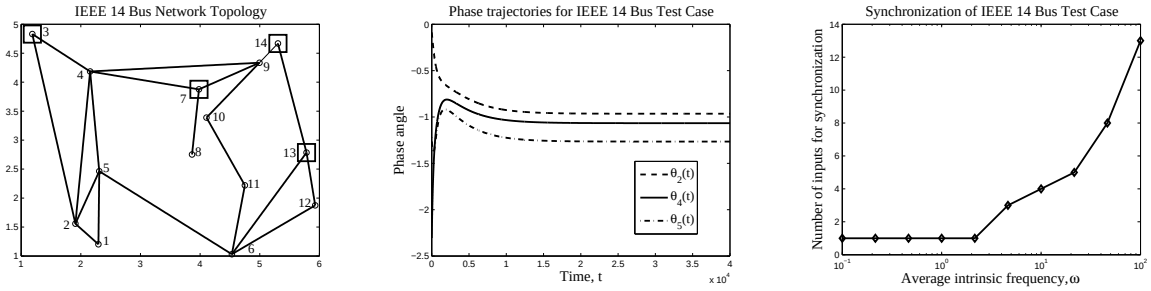
7.4.1 Phase Synchronization in Power System

Synchronization plays a vital role in the power system, in which stable operation requires buses and generators to maintain the same frequency and a relative phase difference of no more than $\pi/2$ on each transmission line (edge) [14]. In order to evaluate our approach to synchronization of power systems, we consider the first-order model (7.2) with negligible resistance on the transmission lines. The input nodes represent generators that are fixed to a common reference phase and frequency in order to restore stability to the grid. We studied phase synchronization in the power system using the IEEE 14 Bus case study [1], which provides a network topology with $n = 14$ nodes along with the line impedances of the edges. We defined the coupling coefficient $K_{uv} = 1/\eta_{uv}$, where η_{uv} is the magnitude of the impedance of edge (u, v). The intrinsic frequencies were chosen according to a zero-mean Gaussian distribution with variance ranging from 0.01 to 100 in different trials. The goal was to ensure that the phase angles satisfied $|\theta_v - \theta_u| < \frac{\pi}{2}$ (i.e., edge-cohesiveness with $\gamma = \pi/2$), consistent with the goal of guaranteeing power system stability [14].

Figure 7.1(a) illustrates input selection for a given set of intrinsic frequencies. The variance of the intrinsic frequencies was equal to 10. The dark squares indicate input nodes selected by our algorithm ($|A| = 4$). We observe that each non-input node is at most two hops away from an input node, which suggests that centrally located nodes are more likely to be candidates for inputs. On the other hand, nodes with high degree were not necessarily chosen as inputs. Additional information beyond the network topology, including the intrinsic frequencies and the coupling coefficients between nodes, is incorporated into the input selection process.

In Figure 7.1(b), trajectories of nodes 2, 4, and 5 from Figure 7.1(a) are shown. The trajectories of these neighboring nodes converge to within the desired bound of $\frac{\pi}{2}$ from each other, and synchronize to the frequency of the input nodes ($\omega_0 = 0$; by Lemma 7.1, an arbitrary initial frequency could be chosen, but we selected $\omega_0 = 0$ for clarity of the figure). Practical edge synchronization occurs in spite of the fact that the initial phases of neighboring nodes 2 and 5 differed by more than $\pi/2$.

Figure 7.1(c) shows the number of input nodes required to achieve practical synchro-



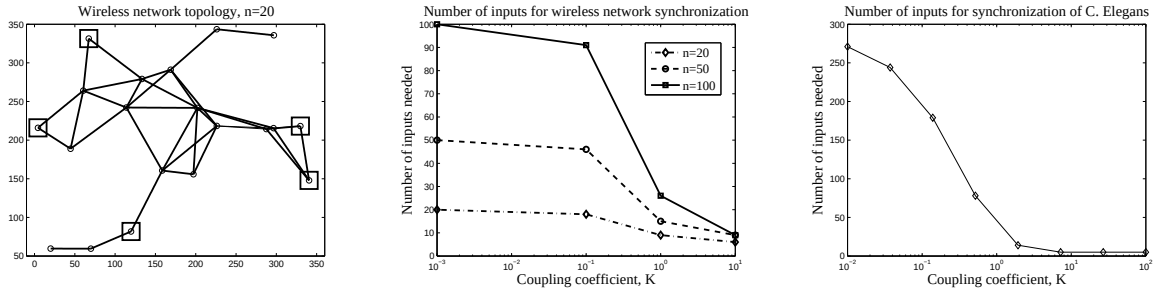
(a) (b) (c)

Figure 7.1: Synchronization case study using the IEEE 14 Bus case study [1]. Coupling coefficients were chosen to be equal to the inverse of the magnitudes of the line impedances, while the intrinsic frequencies were chosen from a Gaussian distribution. (a) Illustration of 14-bus network and inputs needed for one trial when the variance of the intrinsic frequencies was equal to 10. The black squares indicate input nodes. (b) Sample trajectories for nodes 2, 4, and 5 in the 14-bus network. Node trajectories converge to within the desired phase difference ($\gamma = \pi/2$) of each other. (c) Number of inputs required for practical edge synchronization as a function of the variance of the intrinsic frequencies. As the variance increases, the frequencies of neighboring nodes diverge and hence more inputs are needed to ensure synchronization.

nization for different intrinsic frequencies. When the variance of $\{\omega_v : v \in V\}$ is low, only one input node is required to achieve synchronization. For variance near 10, an intermediate number of inputs is required for synchronization, while for large variance in the intrinsic frequencies all nodes must be selected as input nodes to guarantee synchronization.

7.4.2 Time Synchronization

We analyzed time synchronization in wireless networks, in which the input nodes represent anchor nodes with clocks that are set to the reference time. We modeled the wireless network as a geometric random graph. Each node was assumed to share an edge with other nodes within a range of 100m. The simulation was run with the number of nodes $n = \{20, 50, 100\}$. The nodes were assumed to be deployed uniformly at random over a square region, with area chosen so that each node had three neighbors on average. Nodes were assumed to have i.i.d. Gaussian intrinsic frequency with unit variance and zero mean. We computed the number of input nodes required to ensure practical node synchronization with $\gamma = \pi/4$ of



(a) (b) (c)

Figure 7.2: Numerical results for time synchronization and neuronal synchronization. (a) Illustration of inputs chosen for a wireless network, modeled as a geometric random graph, when $K = 0.7197$, $n = 20$, and the input nodes are indicated by squares. The set of input nodes is based on the network topology and intrinsic frequencies of the nodes. (b) Number of input nodes required for time synchronization. The number of nodes is a decreasing function of the coupling coefficient. The reduction in the fraction of input nodes required for synchronization is most significant for larger network size. (c) Number of input nodes required for synchronization of the *C. Elegans* network ($n = 279$) when the intrinsic frequencies have Gaussian distribution with unit variance. The number of inputs is a decreasing function of the coupling coefficient.

each other. The edges had identical coupling with $K_{uv} = K$ for all $(u, v) \in E$, with K in the set $\{0.01, 0.01, 0.1, 1, 10\}$.

Figure 7.2(a) shows one example of a geometric random graph with $n = 20$ and $K = 0.7197$ for all links. The input nodes are indicated by black squares. As in the power system case study, most non-input nodes are within two hops of an input node, and high-degree nodes are not necessarily chosen as inputs. Two adjacent nodes are both chosen as inputs, due to the high intrinsic frequencies of those nodes.

The number of nodes required for synchronization is shown in Figure 7.2(b). The number of nodes required for synchronization is a decreasing function of the coupling coefficient, since a stronger coupling implies that input nodes are able to steer their neighbors into the cohesive region. We observe that the fraction of input nodes required for synchronization is smaller for larger networks.

7.4.3 Firing Rate Synchronization in Neuronal Network

Synchronization of groups of neuronal cells to a common frequency plays a role in information storage and processing for motor control [25] and memory [70]. We analyzed the impact of input nodes on synchronization using the *Caenorhabditis Elegans* (roundworm) neuronal network [132]. The number of nodes is $n = 279$. The intrinsic frequencies were independently from a zero-mean, unit-variance Gaussian distribution. All links had the same coupling coefficient K, which varied from $K = 0.001$ to $K = 10$. The goal was to achieve practical node synchronization with $\gamma = \pi/4$.

We observed that the number of input nodes required decreases as the coupling coefficient increases. The reduction in the number of nodes required was larger than in the random geometric graph (Section 7.4.2), even though the neuronal network contained more nodes. An explanation for this phenomenon is that the neuronal network has higher degree, as well as additional links between different regions of the network. This more connected network topology enables synchronization of more nodes and at a lower coupling coefficient.

7.5 Conclusions and Future Work

In this chapter, we considered the problem of ensuring practical synchronization of phase-coupled oscillators from any initial state by introducing external input nodes. We studied two synchronization problems. In the first problem, the goal is for all node phases to converge to within a bound of a given reference phase, while also converging to the same frequency. In the second problem, the relative phase differences between neighboring nodes must converge to a given bound, while all nodes converge to the same frequency.

We derived sufficient conditions for achieving both synchronization goals. We proved that each synchronization criterion is achieved if there exists at least one equilibrium that lies within the desired region (either all nodes within a given bound of the reference phase, or all relative differences within a given bound), and no stable equilibria outside the desired region. We then presented a threshold-based condition for existence of an equilibrium within the desired region, as well as an efficient algorithm for verifying existence of such an equilibrium. Finally, we derived a class of conditions for non-existence of stable equilibria outside the

desired region, and proposed an algorithm for verifying non-existence of undesired equilibria for a given set of input nodes.

We formulated a submodular optimization approach for selecting a set of input nodes to guarantee synchronization. To achieve a submodular formulation, we first constructed an augmented network graph. We identified two classes of subgraphs (denoted class $\mathcal{T}$ and class $\mathcal{U}$) of the augmented graph. We proved that the sufficient condition for existence of a desired equilibrium is equivalent to connectivity between each non-input node and the input set in all class $\mathcal{T}$ subgraphs. We then showed that the sufficient conditions for non-existence of undesired equilibria are equivalent to connectivity between each non-input node and the input set in all class $\mathcal{U}$ subgraphs.

We mapped this condition to submodular constraints on the set of input nodes. Based on the submodular formulation, we proposed efficient algorithms with provable optimality bounds for selecting a set of up to k nodes in order to maximize the level of synchronization in the network, as well as selecting the minimum-size input set to guarantee a desired level of synchronization.

Our approach was validated through a numerical study of synchronization in power system, wireless, and neuronal networks. Our numerical study supports the intuition that the number of input nodes required decreases as the coupling between neighboring nodes grows stronger. In addition, we found that centrally located nodes are often chosen as inputs, so that the maximum distance between a node and the input set is small.

7.5.1 Future Work

The preliminary results on achieving synchronization in complex networks by introducing input nodes could be extended in several directions. In what follows, we summarize these directions of future research.

Generalized oscillator models: We conjecture that the intuition behind our threshold-based approach - namely, that synchronized oscillators will pull their neighbors towards synchronization via local coupling - extends to other types of oscillators. Two such models that may be investigated are the second-order Kuramoto model [43] and the pulse-coupled

oscillator model [91]. The second-order Kuramoto model describes the second derivative of the node phases as well, and provides a close approximation of power system dynamics [43]. Pulse-coupled oscillators have been proposed as an alternative model of neuronal firing, and have also been used to design time synchronization protocols [61]. Generalizing our framework to incorporate these oscillator models would facilitate synchronization in a broader class of applications.

Maximizing synchronization rate: The results presented thus far guarantee asymptotic practical synchronization. In applications such as power systems, however, synchronization must be achieved within a desired time in order to prevent transient instabilities from causing larger-scale outages. Selecting input nodes that maximize the rate of synchronization, in addition to asymptotic global synchronization, is an open problem. Analysis of the homogeneous case, in which all oscillators have (approximately) the same frequency, would be a first step that still gives real-world applicability.

Synchronization via feedback control: In the approach of this chapter, each input node is fixed to a desired frequency and phase. It may be possible to achieve synchronization with fewer input nodes by introducing a time-varying phase at each input node, based on state feedback from other nodes. While achieving synchronization through feedback control has been studied for a simple two-node coupling [117], feedback synchronization of arbitrary networks of oscillators has received little attention in the literature.

Chapter 8

DISTRIBUTED ONLINE SUBMODULAR MAXIMIZATION

In the previous chapters, we focused on design problems where a subset of nodes must be selected by an external entity with global knowledge of the network topology in order to optimize performance and/or controllability. Large-scale distributed systems, however, typically operate for extended periods in the absence of such a centralized entity. Algorithms for optimal design that require only local information, and can be performed by nodes with communication, computation, and storage constraints, are therefore needed for practical design of networked control systems.

In this chapter, we study distributed, online submodular maximization. In addition to the problem of selecting leader nodes, our techniques have applications in wireless network design problems such as multi-carrier wireless scheduling [5], wireless content delivery through distributed caching [57], sleep-wake scheduling of sensors [73], and data forwarding [142].

Our distributed submodular maximization approach is based on two basic insights. First, when the objective function is submodular, locally optimal solutions, in which node follows an optimal strategy given the actions of the other nodes, are within a provable bound of the global optimum [96]. Second, even when the objective function is time-varying and unknown, each node can approximate the optimal strategy over time via prediction and learning algorithms [26].

We formulate algorithms for both constrained and unconstrained distributed submodular maximization. Under our approach, a subset of nodes (e.g., a set of nodes to act as caches [57], leaders [35], or active sensors [73]) is selected by each node deciding, in a distributed fashion, whether to join the set based on the current and previous values of the objective function. In the case of unconstrained optimization, we introduce a probabilistic algorithm in which each node decides, at each tick of its internal clock, whether to join or exit the set

based on the previously observed utility. We prove that, over a sufficiently long period of time, the expected utility of our algorithm is within a factor of 1/3 of the optimum.

In the case of constrained optimization of a monotone submodular function, we propose a probabilistic algorithm in which, at each tick of its internal clock, each node decides whether or not to join the set by exchanging with a uniformly randomly chosen member of the set, based on the previously observed utility. The expected utility of our algorithm for the constrained case converges to within a factor of 1/2 of the optimum, without requiring time synchronization, node broadcast, or global information. Through experiments, we evaluate both the unconstrained and constrained algorithms in the application of sensor selection using real-world data [2], and find that the utility achieved by our approach is within ten percent of the best centralized algorithms.

This chapter is organized as follows. Section 8.1 states our network model and gives relevant background. Sections 8.2 and 8.3 present our distributed algorithms for unconstrained and constrained submodular maximization, respectively. Section 8.4 analyzes the complexity of our proposed algorithms. Section 8.5 evaluates our approach through numerical case study. Section 8.6 concludes the chapter and describes open problems.

8.1 Network Model and Background

In this section, we state our assumptions on the computation capabilities of the distributed nodes and give needed background on multi-armed bandit prediction algorithms.

8.1.1 Network Model

We consider a set of n nodes indexed in the set $V = \{1, \ldots, n\}$. The nodes form a connected communication network in order to exchange information. The total number of nodes n is unknown to each individual node, although each node has knowledge of its one-hop neighbors. Each node is assumed to be capable of executing probabilistic polynomial-time algorithms, and can choose to join or leave the set S at any time. We let S_t denote the set at time $t \geq 0$. Each node is assumed to have an independent Poisson clock with rate 1, so that the expected number of clock ticks in $[0, T]$ is $O(T)$. The Poisson assumption is used to analyze our algorithm in Lemma 8.2 of Section 8.3.

The objective function $f_t : 2^V \to \mathbb{R}_{\geq 0}$ quantifies the overall utility of the group of nodes at time t. The function f_t is assumed to be nonnegative and submodular at each time t (in Section 8.3, we make the additional assumption that f_t is monotone). Furthermore, we assume that there exists $K > 0$ such that $f_t(U) \leq K$ for all t and $U \subseteq V$.

Each node is assumed to have knowledge of the current value of $f_t(S_t)$ (a distributed algorithm for computing the objective function used in our numerical case study is given in Section 8.5). The nodes do not have knowledge of the set S_t itself, including the cardinality of S_t, or the value of $f_t(A)$ for any $A \neq S_t$. We assume that the nodes have a distributed protocol that enables any node to send a message to a uniformly random node in S_t. This can be achieved by having the nodes in S_t maintain a distributed hash table, which would require each node in S_t to store the identities of $O(\log |S_t|)$ other nodes in S_t, and would incur a routing overhead of $O(\log |S_t|)$ [124].

8.1.2 Background on Multi-Armed Bandit Algorithms

A multi-armed bandit algorithm is defined as follows [26]. Let $\mathcal{A}$ represent a set of actions. Define $\ell_1, \ell_2, \ldots,$ to be a sequence of benefit functions $\ell_j : \mathcal{A} \to \mathbb{R}_{\geq 0}$. The functions ℓ_j can be arbitrary; in particular, they can depend on the prior actions $\{a_m \in \mathcal{A} : m < j\}$.

Definition 8.1. *A multi-armed bandit (MAB) algorithm $\mathcal{P}$ is a probabilistic algorithm that takes as input at time step j a sequence of past actions $a_1, a_2, \ldots, a_{j-1}$ and a sequence of past benefits $\ell_1(a_1), \ell_2(a_2), \ldots, \ell_{j-1}(a_{j-1})$, and outputs an action a_j.*

An example of a MAB algorithm, denoted Exponential Weighted Average (EWA) [26], that will be used as a subroutine in our distributed submodular maximization approach is given as Algorithm 1.

The actions chosen by EWA provide higher expected utility over time than choosing any fixed action at each time step, as described by the following.

Proposition 8.1 ([26], Theorem 6.10)**.** *When the number of actions $|\mathcal{A}| = N$, the algorithm*

Algorithm 15 Prediction algorithm for computing an action a_j based on previous observed gains.

procedure EWA($\mathcal{A}$)
  Input: Set of actions $\mathcal{A}$, $|\mathcal{A}| = N$
  Initialization: $w_{i,0} \leftarrow 1$, $p_i \leftarrow \frac{1}{N}$ for all $i \in \mathcal{A}$
    Choose parameters β, η, $\gamma \in [0, 1]$
  for Each round j **do**
    Select action $a_j \in \mathcal{A}$ from distribution $\mathbf{p}$, receive gain ℓ_j
    for $i = 1, \ldots, N$ **do**
      if $a_j = i$ **then**
        $\ell'_i \leftarrow (\ell_j + \beta)/p_i$
      else $\ell'_i \leftarrow \beta/p_i$
      end if
      $w_i \leftarrow w_i \exp(\eta \ell'_i)$
    end for
    $\mathbf{p} \leftarrow (\mathbf{1}^T \mathbf{w})^{-1} \mathbf{w}$
  end for
end procedure

EWA satisfies

$$\mathbf{E}\left(\sum_{j=1}^{M}\ell_j(a_j)\right)+O\left(\sqrt{(N\ln N)M}\right)\geq\max_{a\in\mathcal{A}}\left\{\mathbf{E}\left(\sum_{j=1}^{M}\ell_j(a)\right)\right\}. \tag{8.1}$$

8.2 Proposed Online Algorithm for Distributed Unconstrained Submodular Maximization

In this section, we present distributed algorithms for solving problems of the form

$$\text{maximize}\quad \textstyle\int_0^T f_t(S_t)\,dt \tag{8.2}$$

If the function f_t is monotone for all t, then the solution is $S_t \equiv V$. For the non-monotone case, our approach is based on the known fact that a locally optimal solution to $\max\{f(S) : S \subseteq V\}$, i.e., a solution satisfying $f(S\cup\{a\}) \leq f(S)$ for all $a \notin S$ and $f(S-\{a\}) \leq f(S)$ for all $a \in S$, is within a provable $O(1)$ bound of the global optimum [49]. In the distributed online setting of (8.2), the nodes cannot verify whether local optimality holds. By choosing whether to join S_t at each epoch using the EWA algorithm, however, the nodes can guarantee convergence in expectation to a local optimum by Proposition 8.1.

We first define time-varying sets $A_t \subseteq V$ and $B_t \triangleq V \setminus A_t$, as well as a parameter $\delta \in [-1, 1]$; we show that $\delta = 1/3$ provides an approximation factor of $1/3$. Nodes in A_t are in the set S_t with probability $p = \frac{1+\delta}{2}$, while nodes in B_t are in the set with probability $q = \frac{1-\delta}{2}$. Each node i implements the EWA algorithm discussed in Section 8.1.

The algorithm, which we denote **Distributed-Unconstrained-Optimization (DUO)**, is as follows. At the j-th tick of its Poisson clock, node i runs the algorithm EWA, with input $(a_1^{(i)}, \ell_1), \ldots, (a_{j-1}^{(i)}, \ell_{j-1})$, where $a_m^{(i)}$ is the action taken by node i at time m, $\ell_m = \int_{T_{m-1}}^{T_m} f_t(S_t)\,dt$ and T_m is the time of the m-th clock tick. If the algorithm returns 1, then i joins S_t with probability $p = \frac{1+\delta}{2}$. If EWA returns 0, then i joins S_t with probability $q = 1 - p$.

The following results describe the local optimality of DUO.

Proposition 8.2. *For any $\delta \in [-1, 1]$, the sets $\{S_t : t \in [0, T]\}$ chosen by DUO satisfy*

$$\mathbf{E}\int_0^T f_t(S_t)\,dt + O(\sqrt{2T\ln 2}) \geq \max\left\{\mathbf{E}\int_0^T f_t(R(A_t - \{i\}, \delta))\,dt, \mathbf{E}\int_0^T f_t(R(A_t \cup \{i\}, \delta))\,dt\right\}$$

Algorithm 16 Algorithm for node $i \in V$ to decide membership in S_t

procedure DUO($\mathcal{C}$,δ)
  Input: Poisson clock $\mathcal{C}$ with rate 1, parameter $\delta \in [-1, 1]$
  Output: Time-varying set S_t
  for Each clock tick **do**
    $\ell_j \leftarrow \int_{T_{i(j-1)}}^{T_{ij}} f_t(S_t)\ dt$
    $a_j \leftarrow EWA((a_1, \ell_1), \ldots, (a_{j-1}, \ell_{j-1}))$
    if $a_j = 1$ **then**
      $S_{T_{ij}} \leftarrow S_{T_{ij}}$ with probability p, $S_{T_{ij}} \leftarrow S_{T_{ij}} \setminus \{i\}$ with probability $(1-p)$
    else
      $S_{T_{ij}} \leftarrow S_{T_{ij}}$ with probability $(1-p)$, $S_{T_{ij}} \leftarrow S_{T_{ij}} \setminus \{i\}$ with probability p
    end if
  end for
end procedure

for each $i \in V$.

Proof. Suppose that each node follows the DUO algorithm. Let S_t denote the set at time t, and define $\hat{S}_t = S_t \setminus \{i\}$ (note that $S_t = \hat{S}_t$ if $i \notin S_t$). Furthermore, suppose that the Poisson clock ticks of node i occur at times $(T_1, \ldots, T_M)$. Let $\hat{a}_j$ denote the action of node i at the j-th clock tick.

Define a related multi-armed bandit problem with M time steps as follows. The action space $\mathcal{A} = \{0, 1\}$, while the gain $\hat{\ell}_j$ from action $a_j \in \mathcal{A}$ at step j is given by

$$\hat{\ell}_j = \begin{cases} \int_{T_j}^{T_{j+1}} f_t(\hat{S}_t \cup \{i\})\ dt, & a_j = 1 \\ \int_{T_j}^{T_{j+1}} f_t(\hat{S}_t)\ dt, & a_j = 0 \end{cases}$$

Suppose that the EWA algorithm is run with this action space and set of gains, using the same parameters as in the subroutine of DUO, outputting actions $(a_1^*, \ldots, a_M^*)$. First, note that the expected gain of the algorithm is

$$\sum_{j=1}^{M} \ell_j(a_j = a_j^*) = \mathbf{E} \int_0^T f_t(S_t)\ dt, \tag{8.3}$$

since a_j^* and $\hat{a}_j$ are obtained by running the same randomized algorithm with the same parameters and inputs. Second, note that if $a_j \equiv 1$ for $j = 1, \ldots, M$, then

$$\sum_{j=1}^{M} \ell_j(a_j = 1) = \sum_{j=1}^{M} \int_{T_j}^{T_{j+1}} f_t(\hat{S}_t \cup \{i\})\ dt = \int_0^T f_t(S_t \cup \{i\})\ dt. \tag{8.4}$$

Similarly, if $a_j \equiv 0$ for $j = 1, \ldots, M$, then

$$\sum_{j=1}^{M} \ell_j(a_j = 0) = \sum_{j=1}^{M} \int_{T_j}^{T_{j+1}} f_t(\hat{S}_t)\ dt = \int_0^T f_t(S_t \setminus \{i\})\ dt. \tag{8.5}$$

Combining (8.3), (8.4), and (8.5) and applying Proposition 8.1 yields the desired result. □

The local optimality of the set S_t follows as a corollary to Proposition 8.2.

Corollary 8.1. *Let $v \in V$, and define $\mathcal{T} := \{t \in [0,T] : v \in A_t\}$. For any $v \in V$, $\delta \in [-1,1]$, the sets $\{S_t : t \in [0,T]\}$ chosen by DUO satisfy*

$$\max\Bigg\{\mathbf{E}\int_{\mathcal{T}^c} f_t(S_t \cup \{v\}) - f_t(S_t - \{v\})\ dt, \\ \mathbf{E}\int_{\mathcal{T}} f_t(S_t - \{v\}) - f_t(S_t \cup \{v\})\ dt\Bigg\} \le O\left(\frac{\sqrt{2T\ln 2}}{\delta}\right). \tag{8.6}$$

Proof. By Proposition 8.2, we have

$$\begin{aligned} O(\sqrt{2T\ln 2}) &\ge \mathbf{E}\int_{\{t:v\notin A_t\}} f_t(R(A_t \cup \{v\}, \delta)) - f_t(R(A_t, \delta))\ dt \\ &= \mathbf{E}\int_{\{t:v\notin A_t\}} [pf_t(S_t \cup \{v\}) + qf_t(S_t - \{v\}) - (qf_t(S_t \cup \{v\}) + pf_t(S_t - \{v\}))]\ dt \\ &= \delta\mathbf{E}\int_{\{t:v\notin A_t\}} f_t(S_t \cup \{v\}) - f_t(S_t - \{v\})\ dt. \end{aligned}$$

The proof of the second part of Corollary 8.1 is similar. □

Based on the local optimality of our approach, the overall optimality gap of DUO is given as follows. The proof is omitted due to space constraints.

Theorem 8.1. *Let $C = \arg\max\{\int_0^T f_t(S)\ dt : S \subseteq V\}$. When $\delta = \frac{1}{3}$, the sets $\{S_t : t \in [0,T]\}$ selected by DUO satisfy*

$$\mathbf{E}\int_0^T f_t(S_t)\ dt + O(3n\sqrt{2T\ln 2}) \ge \frac{1}{3}\int_0^T f_t(C)\ dt.$$

Proof. Set $p = 2/3$ and $q = 1 - p = 1/3$. From [49, Theorem 3.6], we have that

$$\begin{aligned}\frac{(f_t(R \cup (B_t \cap C)) + f_t(R_t \cap (B_t \cup C)))}{2} \geq \frac{2}{27} f_t(B_t \cap C) + \frac{2}{27} f_t((A_t \setminus C) \cup (B_t \cap C)) \\ + \frac{1}{27} f_t(B_t) + \frac{4}{27} f_t(C) + \frac{2}{27} f_t(B_t \cup C) \\ + \frac{4}{27} f_t(A_t \cup C) + \frac{4}{27} f_t(A_t \cap C) \\ + \frac{2}{27} f_t((A_t \cap C) \cup (B_t \setminus C)).\end{aligned}$$

Combining with Corollary 8.1 yields

$$\begin{aligned}\mathbf{E} \int_0^T f_t(S_t)\ dt + O\left(\frac{n\sqrt{2T \ln 2}}{\delta}\right) \geq \mathbf{E} \int_0^T \Bigg[\frac{2}{27} f_t(B_t \cap C) + \frac{2}{27} f_t((A_t \setminus C) \cup (B_t \cap C)) \\ + \frac{1}{27} f_t(B_t) + \frac{4}{27} f_t(C) \\ + \frac{2}{27} f_t(B_t \cup C) + \frac{4}{27} f_t(A_t \cup C) + \frac{4}{27} f_t(A_t \cap C) \\ + \frac{2}{27} f_t((A_t \cap C) \cup (B_t \setminus C)) \Bigg]\ dt. \qquad (8.7)\end{aligned}$$

We now divide the remainder into two cases, depending on whether

$$\int_0^T f_t(C)\ dt \geq \mathbf{E} \int_0^T f_t(B_t)\ dt \qquad (8.8)$$

holds.

Case 1: Eq. (8.8) holds: Adding $\frac{1}{9} \int_0^T f_t(B_t)\ dt$ to both sides of (8.7) yields

$$\begin{aligned}\mathbf{E} \int_0^T f_t(S_t)\ dt + O\left(\frac{n\sqrt{2T \ln 2}}{\delta}\right) \geq \mathbf{E} \int_0^T \Bigg[\frac{2}{27} f_t(B_t \cap C) + \frac{2}{27} f_t((A_t \setminus C) \cup (B_t \cap C)) \\ + \frac{4}{27} f_t(B_t) + \frac{4}{27} f_t(C) + \frac{2}{27} f_t(B_t \cup C) \\ + \frac{4}{27} f_t(A_t \cup C) + \frac{4}{27} f_t(A_t \cap C) \\ + \frac{2}{27} f_t((A_t \cap C) \cup (B_t \setminus C)) \Bigg]\ dt.\end{aligned}$$

Now, submodularity implies that

$$f_t(B_t) + f_t((A_t \setminus C) \cup (B_t \cap C)) \geq f_t((A_t \setminus C) \cup B_t) + f_t(B_t \cap C) \geq f_t(B_t \cap C)$$

and, similarly, that

$$\begin{aligned} f_t(B_t) + f_t((A_t \cap C) \cup (B_t \setminus C)) &\geq f_t(B_t \cap ((A_t \cap C) \cup (B_t \setminus C))) \\ &\quad + f_t(B_t \cup ((A_t \cap C) \cup (B_t \setminus C))) \\ &\geq f_t(B_t \cup ((A_t \cap C) \cup (B_t \setminus C))) \\ &= f_t(B_t \cup C). \end{aligned}$$

Applying these lower bounds gives

$$\mathbf{E} \int_0^T f_t(S_t) + \frac{1}{9} f_t(B_t)\ dt + O\left(\frac{n\sqrt{2T \log 2}}{\delta}\right) \tag{8.9}$$

$$\begin{aligned} &\geq \mathbf{E} \int_0^T \left[\frac{4}{27} f_t(A_t \cap C) + \frac{4}{27} f_t(B_t \cap C) \right. \\ &\quad \left. + \frac{4}{27} f_t(A_t \cup C) + \frac{4}{27} f_t(B_t \cup C) + \frac{4}{27} f_t(C)\ dt \right]. \end{aligned} \tag{8.10}$$

Applying submodularity yields

$$f_t(B_t \cap C) + f_t(A_t \cap C) \geq f_t(\emptyset) + f_t(C) \geq f_t(C),$$

and

$$f_t(A_t \cup C) + f_t(B_t \cup C) \geq f_t(V) + f_t(C) \geq f_t(C).$$

The right-hand side of (8.10) can then be bounded below by

$$\mathbf{E} \int_0^T f_t(S_t) + \frac{1}{9} f_t(B_t)\ dt + O\left(\frac{n\sqrt{2T \log 2}}{\delta}\right) \geq \frac{4}{9} \int_0^T f_t(C)\ dt. \tag{8.11}$$

To complete the proof, we combine (8.8) and (8.11) as

$$\begin{aligned} \mathbf{E} \int_0^T f_t(S_t)\ dt + \frac{1}{9} \int_0^T f_t(C)\ dt + O\left(\frac{n\sqrt{2T \log 2}}{\delta}\right) &\geq \mathbf{E} \int_0^T f_t(S_t)\ dt + \frac{1}{9} \int_0^T f_t(B_t)\ dt \\ &\geq \frac{4}{9} \int_0^T f_t(C)\ dt. \end{aligned}$$

Rearranging terms gives the desired result.

Case 2: Eq. (8.8) does not hold: Using nonnegativity of f_t, we bound the right-hand

side of (8.7) as

$$\begin{aligned}\mathbf{E}\int_0^T f_t(S_t)\ dt + O\left(\frac{n\sqrt{2T\log 2}}{\delta}\right) &\geq \mathbf{E}\int_0^T \left[\frac{2}{27}f_t(B_t\cap C) + \frac{1}{27}f_t(B_t) + \frac{2}{27}f_t(B_t\cup C)\right. \\ &\quad \left. + \frac{4}{27}f_t(A_t\cup C) + \frac{4}{27}f_t(A_t\cap C) + f_t(C)\ dt\right] \quad (8.12)\\ &\geq \mathbf{E}\int_0^T \left[\frac{2}{27}f_t(B_t\cap C) + \frac{2}{27}f_t(A_t\cap C)\right. \\ &\quad + \frac{2}{27}f_t(A_t\cup C) + \frac{2}{27}f_t(B_t\cup C) \\ &\quad \left. + \frac{5}{27}f_t(C)\ dt\right] \quad (8.13)\end{aligned}$$

where (8.12) follows from the fact that $f_t((A_t\backslash C)\cup(B_t\cap C)) \geq 0$ and $f_t((A_t\cap C)\cup(B_t\backslash C)) \geq 0$, and (8.13) follows from the assumption that (8.8) does not hold. Submodularity of f_t implies that

$$f_t(B_t\cap C) + f_t(A_t\cap C) \geq f_t(\emptyset) + f_t(C) \geq f_t(C)$$

and

$$f_t(B_t\cup C) + f_t(A_t\cup C) \geq f_t(V) + f_t(C),$$

which we combine with (8.13) to give

$$\begin{aligned}\mathbf{E}\int_0^T f_t(S_t)\ dt + O\left(\frac{n\sqrt{2T\log T}}{\delta}\right) &\geq \int_0^T \left(\frac{2}{27}+\frac{2}{27}+\frac{5}{27}\right) f_t(C)\ dt \\ &= \frac{1}{3}\int_0^T f_t(C)\ dt,\end{aligned}$$

completing the proof. □

Theorem 8.1 implies that the algorithm DUO achieves an $O(1)$ optimality gap in expectation, for the case where there is no constraint on the set S_t. The case where there is a constraint on the size of S_t is discussed in the following section.

8.3 Proposed Online Algorithm for Distributed Constrained Submodular Maximization

In this section, we present our distributed online algorithm for approximating the solution to

$$\begin{array}{ll}\text{maximize} & \int_0^T f_t(S_t)\ dt \\ \text{s.t.} & |S_t| \leq k\end{array} \quad (8.14)$$

which represents maximization of the time-varying submodular functions $f_t(S_t)$ subject to a cardinality constraint. We make the additional assumption that f_t is nondecreasing for all t, i.e., that $f_t(A) \leq f_t(B)$ for all $A \subseteq B$.

Our algorithm is a distributed implementation of the exchange heuristic [96]. In the offline, centralized version of the algorithm, if there exist nodes $u \in V \setminus S$ and $v \in S$ with $f(S) \leq f(S \cup \{u\} - \{v\})$, then the set S is updated to $S \cup \{u\} - \{v\}$ and we say that nodes u and v are exchanged. In our distributed version, at each tick of its internal Poisson clock, each node u probabilistically decides, using the EWA algorithm, whether to exchange with a node in S_t. If so, node u chooses a random node $v \in S_t$ using a distributed protocol and requests an exchange. Node v decides whether to accept the exchange using the EWA algorithm, and if v chooses to accept, then the set is updated to $S_t \cup \{u\} - \{v\}$.

In what follows, we first describe the algorithm, denoted **Distributed-Constrained-Optimization (DCO)**, used by each node to perform the decentralized optimization. We then analyze the optimality gap.

At each tick of its Poisson clock, node u chooses an action based on whether $u \in S_t$, where t is the current time. If $u \in S_t$, then u takes no further action. If $u \notin S_t$, then u decides to either join S_t or remain in $V \setminus S_t$. In the case where join is chosen, node u chooses a node v uniformly at random from S_t, as discussed in Section 8.1.1, and requests an exchange. If v accepts, then S_t is updated to $S_t - \{v\} \cup \{u\}$. Otherwise, S_t remains unchanged.

Nodes decide whether to join S_t, and whether to accept an exchange, based on EWA. The four actions at each time are $(join, accept)$, $(join, refuse)$, $(stay, accept)$, and $(stay, refuse)$. If $u \notin S_t$ and the action chosen by the algorithm is $(join, accept)$ or $(join, refuse)$, then u will attempt to join S_t. Otherwise, u will not attempt to join. If $u \in S_t$ and another node requests an exchange, then u will accept if the algorithm outputs $(join, accept)$ or $(stay, accept)$, and refuse otherwise. If a node makes a decision at time T and the next decision is at time T', then the payoff for the decision at time T is given by $\int_T^{T'} f_t(S_t)\, dt$.

The structure of the optimality gap analysis is as follows. We first prove a sequence of

Algorithm 17 Algorithm for node $u \in V$ to decide membership in S_t under cardinality constraint.

procedure DCO($\mathcal{C}$,δ)
  Input: Poisson clock $\mathcal{C}$ with rate 1
  Output: Time-varying set S_t
  for Each clock tick **do**
    if $u \notin S_t$ **then**
      Update EWA with gain $\int_{T_{j-1}}^{T_j} f_t(S_t)\ dt$
      if EWA outputs *join* **then**
        Send exchange request to randomly chosen $v \in S_t$
        if v accepts exchange **then**
          $S_t \leftarrow S_t - \{v\} \cup \{u\}$
        end if
      end if
    end if
    if $u \in S_t$ and receive exchange request from $u' \notin S_t$ **then**
      Update EWA with gain $\int_{T_{j-1}}^{T_j} f_t(S_t)\ dt$
      if EWA outputs *accept* **then**
        $S_t \leftarrow S_t - \{u\} \cup \{u'\}$
      end if
    end if
  end for
end procedure

lemmas, which imply that

$$\int_0^T f_t(S_t)\ dt + o(T) \geq \int_0^T \frac{1}{k} \sum_{v \in S_t} f_t(S_t - \{v\} \cup \{u\}) \tag{8.15}$$

for all $u \in V$. We then use this result to prove that, for any fixed set C with $|C| \leq k$,

$$\int_0^T f_t(S_t)\ dt + o(T) \geq \frac{1}{2} \int_0^T f_t(C)\ dt.$$

In order to prove (8.15), we divide the set S_t into k slots. At each time t, slot m contains one node from S_t, denoted v_t^m, with $v_t^m \neq v_t^{m'}$ for $m \neq m'$, and $S_t = \bigcup_{m=1}^k \{v_t^m\}$. Initially, the mapping between nodes and slots is arbitrary. If an exchange takes place at time t, with node u joining S_t and v exiting S_t, then v_t^m is updated to u, where m satisfies $v_t^m = v$. Using this definition, we have the following lemma.

Lemma 8.1. *For any $u \in V$ and $m \in \{1, \ldots, k\}$, define $T_m \triangleq \min\{t : v_t^m = u\}$. The sets $\{S_t : t \in [0, T]\}$ chosen by DCO satisfy*

$$\mathbf{E} \int_{T_m}^T f_t(S_t)\ dt + O(\sqrt{4T \ln 4}) \geq \mathbf{E} \int_{T_m}^T f_t(S_t - v_t^m + u)\ dt. \tag{8.16}$$

Proof. Consider a multi-armed bandit problem from the perspective of node u. Let $\hat{I}_t$ be a process with actions $(join, accept)$, $(join, refuse)$, $(stay, accept)$, and $(stay, refuse)$, which receives payoffs $\int_T^{T'} f_t(S_t)\ dt$. The outcome of the MAB problem is the random set sequence $\{\hat{S}_t^u : t \in [T, T']\}$, which denotes the set chosen by the nodes other than u based on the actions $\hat{I}_t$, along with the current state of the node u (the current slot of node u).

If u is in slot 0 and a *join* action is chosen, then the payoff is $\mathbf{E} \int_T^{T'} f_t(\hat{S}_t^u \cup \{u\})\ dt$. If *stay* is chosen, then the payoff is $\mathbf{E} \int_T^{T'} f_t(\hat{S}_t^u)\ dt$. If u is in slot m and accepts an exchange, then the payoff is $\mathbf{E} \int_T^{T'} f_t(\hat{S}_t^u)\ dt$. Finally, if u is in slot m and refuses the exchange, then the payoff is $\mathbf{E} \int_T^{T'} f_t(\hat{S}_t^u - \{v_t^m\} \cup \{u\})$.

Now, if at each iteration, node u follows EWA, then the total expected payoff is equivalent to the payoff from choosing $\hat{I}_t$ at each iteration, which is $\mathbf{E} \int_{T_m}^T f_t(S_t)\ dt$. On the other hand, if node u chooses $(join, refuse)$ at each time slot, then the expected payoff is $\mathbf{E} \int_{T_m}^T f_t(S_t - \{v_m^t\} \cup \{u\})\ dt$. Proposition 8.1 and the fact that there are 4 possible actions at each time step complete the proof. □

Lemma 8.1 implies that, for any node, the algorithm DCO provides higher overall utility than occupying slot m in the interval $[T_m, T]$. We next show that the result of Lemma 8.1 can be extended to $[0, T]$, due to the fact that $\mathbf{E}(T_m) < \infty$.

Lemma 8.2. *For any $u \in V$, $m \in \{1, \ldots, k\}$, the sets $\{S_t : t \in [0, T]\}$ chosen by DCO satisfy*

$$\mathbf{E} \int_0^T f_t(S_t)\ dt + O(\sqrt{4T \ln 4}) \geq \mathbf{E} \int_0^T f_t(S_t - \{v_t^m\} \cup \{u\})\ dt.$$

Proof. By Lemma 8.1,

$$\mathbf{E} \int_{T_m}^T f_t(S_t)\ dt + o(T) \geq \mathbf{E} \int_{T_m}^T f_t(S_t - v_t^m \cup \{u\})\ dt.$$

Hence, it suffices to show that

$$\mathbf{E} \int_0^{T_m} f_t(S_t)\ dt + o(T) \geq \mathbf{E} \int_0^{T_m} f_t(S_t - v_t^m \cup \{u\})\ dt,$$

which is equivalent to

$$\mathbf{E} \int_0^{T_m} (f_t(S_t - v_t^m \cup \{u\}) - f_t(S_t))\ dt \leq o(T).$$

Since $f_t(S) \leq K$ for all t and S,

$$\mathbf{E} \int_0^{T_m} (f_t(S_t - v_t^m \cup \{u\}) - f_t(S_t))\ dt \leq \mathbf{E} \int_0^{T_m} K\ dt = K\mathbf{E}(T_m)$$

and it therefore suffices to show that $\mathbf{E}(T_m) = o(T)$.

Let $\hat{T}_1, \ldots, \hat{T}_r, \ldots$ denote the sequence of times when a clock tick occurs and $u \notin S_t$. Let $X = \min\{r : v_{\hat{T}_r}^m = u\}$. Hence $\mathbf{E}(T_m) = \mathbf{E}(\hat{T}_X)$. Let A_i denote the event that node u chooses *join* at time $\hat{T}_i$.

This expectation can be written as the summation $\sum_{i=1}^{X} \mathbf{E}(\hat{T}_i - \hat{T}_{i-1})$. We have that $\mathbf{E}(\hat{T}_i - \hat{T}_{i-1}) = \mathbf{E}(\hat{T}_i - \hat{T}_{i-1}|A_i)Pr(A_i) + \mathbf{E}(\hat{T}_i - \hat{T}_{i-1}|A_i^c)Pr(A_i^c)$.

If A_i holds, then the time before the next decision to *join* or *stay* is equal to the time for another node to initiate an exchange and for node u to accept. Each node chooses to *join* or *stay* according to an independent Poisson process with unit rate. At each tick, the probability of choosing *join* is (by definition of algorithm EWA) bounded below by $\frac{\gamma}{2}$. The probability of initiating an exchange with node u is $\frac{1}{k}$, and so exchange requests

from each other node arrive according to a Poisson process with rate bounded below by $\frac{2}{k\gamma}$. Furthermore, since each node has an independent Poisson clock, exchange requests arrive at u according to a Poisson process with rate bounded below by $\frac{2(n-k)}{k\gamma}$. By definition of algorithm EWA, the probability that node u accepts each request is at least $\frac{\gamma}{2}$. Thus $\mathbf{E}(\hat{T}_i - \hat{T}_{i-1}|A_i) \leq \frac{4(n-k)}{\gamma^2 k} + 1$, where the 1 is the expected time to wait for the next clock tick after leaving S_t.

If A_i does not hold, the expected time for the next clock tick is 1. Hence $\mathbf{E}(\hat{T}_i - \hat{T}_{i-1}) \leq 1 + \frac{4(n-k)}{\gamma^2 k}$, and

$$\mathbf{E}(\hat{T}_X) \leq \mathbf{E}\sum_{i=1}^{X}\left(1 + \frac{4(n-k)}{\gamma^2 k}\right) = \mathbf{E}(X)\left(1 + \frac{4(n-k)}{\gamma^2 k}\right),$$

which is bounded above by $\frac{4k}{\gamma^2} + \frac{16(n-k)}{\gamma^4 k}$. Since this bound is independent of T, the $O(\sqrt{4T\log 4})$ term dominates asymptotically. ■

Proof of Proposition 8.3: By Lemma 8.2, for each slot m,

$$\mathbf{E}\int_0^T f_t(S_t)\ dt + O(\sqrt{4T\log T}) \geq \mathbf{E}\int_0^T f_t(S_t - v_t^m \cup \{u\})\ dt.$$

Summing both sides over m yields

$$\begin{aligned} k\left(\mathbf{E}\int_0^T f_t(S_t)\ dt + O(\sqrt{4T\log 4})\right) &\geq \mathbf{E}\sum_{m=1}^{k}\int_0^T f_t(S_t - v_t^m \cup \{u\})\ dt \\ &= \mathbf{E}\int_0^T \sum_{v\in S_t} f_t(S_t - \{v\} \cup \{u\})\ dt. \end{aligned}$$

Dividing by k gives the desired result. □

Lemmas 8.1 and 8.2 imply that, for each node, following DCO provides higher utility than remaining in any slot m for the entire interval $[0, T]$. Summing over the slots $m = 1, \ldots, k$ yields (8.15), as shown in the following proposition.

Proposition 8.3. *For any $u \in V$, the sets $\{S_t : t \in [0,T]\}$ chosen by DCO satisfy*

$$\mathbf{E}\int_0^T f_t(S_t)\ dt + O(\sqrt{4T\ln 4}) \geq \frac{1}{k}\mathbf{E}\int_0^T \sum_{v\in S_t} f_t(S_t - \{v\} \cup \{u\})\ dt. \tag{8.17}$$

By Proposition 8.3, the utility of the set sequence S_t chosen by DCO cannot be improved by replacing a randomly chosen node in S_t with any fixed node u. The following theorem uses this result to derive the optimality gap of DCO.

Theorem 8.2. *The sets $\{S_t : t \in [0,T]\}$ chosen by DCO satisfy*

$$\mathbf{E}\int_0^T f_t(S_t)\ dt + O(k\sqrt{4T\ln 4}) \geq \frac{1}{2}\int_0^T f_t(C)\ dt$$

for all C with $|C| \leq k$.

Proof. Monotonicity of f_t implies that

$$\mathbf{E}\int_0^T f_t(C) - f_t(S_t)\ dt \leq \mathbf{E}\int_0^T f_t(C \cup S_t) - f_t(S_t)\ dt.$$

Define a sequence of sets $U_0^t \subseteq U_1^t \subseteq \cdots \subseteq U_k^t$, with $U_0^t = S_t$, $U_k^t = S_t \cup C$, and $U_j^t - U_{j-1}^t = \{a_i^t\}$ for some $a_i^t \in C$. Then we have

$$\begin{aligned}
\mathbf{E}\int_0^T f_t(C) - f_t(S_t)\ dt &\leq \mathbf{E}\int_0^T \sum_{j=1}^k (f_t(U_j^t) - f_t(U_{j-1}^t))\ dt \\
&= \mathbf{E}\int_0^T \sum_{j=1}^k (f_t(U_{j-1}^t \cup \{a_j^t\}) - f_t(U_{j-1}^t))\ dt \\
&\leq \mathbf{E}\int_0^T \sum_{j=1}^k (f_t(S_t \cup \{a_j^t\}) - f_t(S_t))\ dt \qquad (8.18) \\
&= \mathbf{E}\int_0^T \sum_{a\in C} (f_t(S_t \cup \{a\}) - f_t(S_t))\ dt \\
&\leq \int_0^T \left[\sum_{a\in C} \mathbf{E}\left(\frac{1}{k}\sum_{v\in S_t} f_t(S_t \cup \{a\} - \{v\}) \right.\right. \qquad (8.19) \\
&\quad \left.\left. - f_t(S_t - \{v\})\right)\right]\ dt, \qquad (8.20)
\end{aligned}$$

where (8.18) and (8.20) follow from submodularity of f_t. We now use Proposition 8.3 to upper bound (8.20) as

$$\begin{aligned}
\mathbf{E}\int_0^T f_t(C) - f_t(S_t)\ dt &\leq \int_0^T \sum_{a\in C} \mathbf{E}\left(\frac{1}{k}\sum_{v\in S_t} f_t(S_t) - f_t(S_t - \{v\})\right) dt + O(k\sqrt{4T\log 4}) \\
&\leq \mathbf{E}\int_0^T f_t(S_t)\ dt + O(k\sqrt{4T\log 4}), \qquad (8.21)
\end{aligned}$$

where (8.21) follows from submodularity of f_t. The result follows after rearranging terms. □

Theorem 8.2 shows that, in expectation, algorithm DCO achieves the same optimality gap as the offline exchange heuristic [96]. This is in spite of the fact that, while the offline exchange heuristic is based on identifying a specific pair of elements u and v satisfying $f(S - \{u\} \cup \{v\}) \geq f(S)$, DCO operates by performing a sequence of random exchanges. The use of random exchanges therefore reduces the memory, computation, and information requirements of each node, without sacrificing worst-case optimality guarantees.

8.4 Performance Evaluation and Comparison

In what follows, we evaluate our proposed algorithms for constrained and unconstrained submodular maximization, and compare with the constrained submodular maximization algorithm **Distributed-online-greedy** (DOG) introduced in [56]. We consider the following criteria: (a) the computational assumptions of the nodes specified by the algorithms, (b) the asymptotic bounds that are achieved, (c) the rate at which the algorithms must be computed in order to reach a given value ϵ from the asymptotic bound, (d) the communication overhead, and (e) the storage overhead at each node.

Our analysis makes the same assumptions as that of [56] in order to guarantee consistency of the results, with one exception. While [56] assumed that the cost of broadcasting a message to all other nodes is 1, we assume that nodes communicate via multi-hop paths, leading to a cost of broadcast of $O(\frac{n}{\pi} - \sqrt{n})$ [21]. Similarly, we assume that the cost of sending a message to a single node is proportional to the network diameter, i.e., $O(\sqrt{n})$. The comparison results are summarized below in Table 8.1.

Table 8.1: Performance Comparison of Distributed Submodular Maximization Algorithms

Criterion	**DOG** [56]	**DCO**	**DUO**
Constraints	Cardinality k	Cardinality k	None
Optimality Gap	$(1 - 1/e) \approx 0.63$	0.5	0.33
Communication	$O\left(kn^2 \log n\right)$	$O(n\sqrt{n})$	0
Storage	$O(k)$	$O(1)$	$O(1)$

Computational assumptions: The DOG algorithm assumes that each agent is able to compute $f_t(S_t \cup \{v\}) - f_t(S_t)$ at each time t, that each agent can broadcast to all other agents, and that the agents have calibrated, synchronized clocks (these assumptions are enumerated explicitly in Section 4 of [56]). In contrast, our algorithms DUO and DCO assume that each agent has access to the current objective function value $f_t(S_t)$, which can be jointly computed by all agents through distributed algorithms, but does not have access to $f_t(S_t \cup \{v\})$. We also do not assume time synchronization, although we do assume that each agent has a Poisson clock with stationary mean.

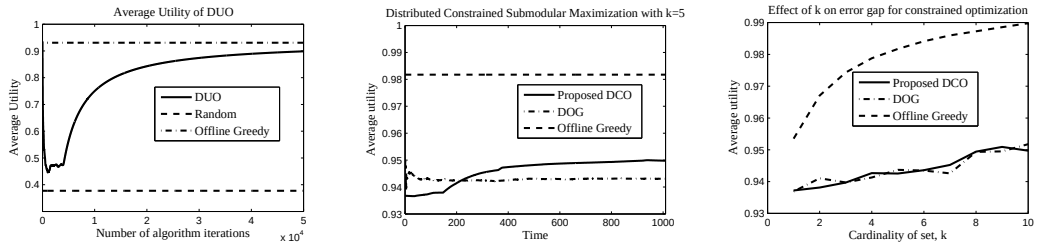
Asymptotic optimality gap: The DOG algorithm achieves an optimality gap of $(1 - 1/e) \approx 0.63$ in the limit as $T \to \infty$, while our constrained optimization approach has an asymptotic optimality gap of $\frac{1}{2} = 0.5$. In the unconstrained case, we achieve a bound of $\frac{1}{3}$, as proved in Theorem 8.2. Although there is currently no distributed algorithm for unconstrained submodular maximization to the best of our knowledge, the best known optimality gap for a centralized unconstrained submodular maximization algorithm is 0.5 [19].

Rate of convergence: As discussed in [56, Theorem 3], the DOG algorithm has deviation $O(k\sqrt{n \log nT})$ from its asymptotic bound after time T. By comparison, our approach has deviation $O(k\sqrt{T})$ by Theorem 8.2 in the constrained case and $O(n\sqrt{T})$ in the unconstrained case.

Based on the above, in what follows we assume that our algorithm is run with unit-rate Poisson clocks at each agent, while the DOG algorithm is run with rate $\frac{n \log n}{4 \log 4}$. This is based on the fact that the rate at which each algorithm is executed will be chosen in order to achieve a given rate of convergence to the asymptotic optimality gap.

Communication overhead: At each Poisson tick, each agent can decide to unicast a message to a random agent $v \in S_t$ requesting an exchange. Since agent v then responds, the expected overhead is bounded above by $O(\sqrt{n})$ each time an agent executes the algorithm (this is a loose bound, since it assumes that each agent always requests an exchange). Since each agent runs the algorithm with rate 1, the total communication overhead per unit time is $O(2n\sqrt{n})$.

The DOG algorithm has k iterations, each of which involves two stages of communication



(a) (b) (c)

Figure 8.1: Numerical evaluation of our proposed algorithms DUO and DCO for distributed online submodular maximization and comparison with DOG algorithm [56] on a network of 46 nodes [2]. The reduction in mean square error is used as a utility function. (a) Utility as a function of time for the case of unconstrained optimization. After an initial learning phase, the utility approaches the maximum possible value of 1, consistently outperforming random sensor selection. (b) Performance in the case where only $k = 5$ sensors can be active. Performance of the algorithm DCO is within 0.07 of the best achievable value of 1, and is close to the centralized and distributed greedy algorithms. (c) Increase in utility as a function of the number of sensors k. The increase from the DCO and DOG algorithms is roughly linear, while the increase of the centralized algorithm is concave.

between agents. In the first stage, one or more agents broadcast messages indicating their intention to join S_t; in [56, Theorem 2], the authors compute that the expected number of broadcasts is roughly 3 in the average case. In the second stage, the agent chosen to joint S_t broadcasts its updated weight value to the other agents. Hence the total broadcasts per iteration is $O(4)$. Furthermore, there are k iterations, and the algorithm is run at rate $O\left(n \log n\right)$, for a total overhead of $O\left(k\left(n - \sqrt{n}\right)\left(n \log n\right)\right)$.

The unconstrained algorithm incurs no communication overhead, since the lack of constraints implies that each agent does not have to notify any other agents before joining the leader set.

Storage overhead: The unconstrained algorithm requires each agent to maintain two weights, equal to the total loss from joining and not joining the set S_t. Hence the storage is $O(1)$. The constrained algorithm tracks the performance of 4 actions, enumerated in Section 8.3. The DOG algorithm incurs storage overhead of $O(k)$, since each agent must maintain a probability of joining the set S_t at each of k iterations.

8.5 Case Study: Distributed Sensor Scheduling

In this section, we evaluate our approach through a case study, based on the problem of determining the optimal set of nodes to activate in a sensor network. We first provide the details of our experimental setup. We then describe our approach for distributed computation of the objective function, followed by the simulation results.

8.5.1 Experimental Setup

We evaluate our proposed algorithms using Matlab. In this study, we consider the problem of deciding which sensors, out of a set of $n = 46$ sensors, should be active at a given time. To perform the study, we use the temperature monitoring data from the Intel Research Lab [2], which contains a time series of temperature measurements from a network of 46 sensors. We assumed that each temperature measurement corresponded to a sample from an underlying Gaussian random distribution μ_i corresponding to the node i. We then computed the covariance matrix K for the sensors. We let the utility function $f(S)$, which we assume to be fixed, be given as $f(S) = \frac{MMSE(\emptyset) - MMSE(S)}{MMSE(\emptyset)}$, where $MMSE(S)$ denotes the conditional variance of the measurements of each sensor, given knowledge of the measurements at locations S. The analysis considered both unconstrained and cardinality-constrained distributed optimization. For the EWA algorithm, we chose $\gamma = 0.01$, $\eta = 1/MMSE(\emptyset)$, and $\beta = 0.03$, based on the discussion of parameter selection in [26].

8.5.2 Sensing Model and Objective Function Definition

We assume that each sensor node $v \in V$ is capable of measuring a random variable X_v. The variables $\mathbf{X} \triangleq \{X_v : v \in V\}$ are jointly Gaussian with mean 0 and covariance matrix Σ. The precision matrix Q is defined by $Q \triangleq \Sigma^{-1}$. The objective function $f(S_t)$ can be written as

$$f(S_t) \triangleq \sum_{v \in V \setminus S_t} \mathrm{var}(X_v | X_{S_t}), \tag{8.22}$$

where $X_{S_t} \triangleq \{X_v : v \in S_t\}$ and

$$\mathrm{var}(X_v | X_{S_t}) = \int_{x \in \mathbb{R}^{|S_t|}} \mathrm{var}(X_v | X_{S_t} = x) p_{X_{S_t}}(x)\, dx.$$

This objective function measures the uncertainty of estimating the measurements at the inactive sensors, given the measurements at the active sensors. The function $f(S_t)$ was shown to be submodular as a function of S_t in [74].

To compute the objective function, a random process $z_v[m]$, $m = 1, 2, \ldots$, is simulated at each node $v \in V\ S_t$, such that the variance of $z_v[m]$ converges to $(Q_{S_t}^{-1})_{vv}$. The process is created by having each node maintain a time-varying state, which is equal to a linear combination of the states of its neighbors plus additional Gaussian noise. Each node can then estimate $\text{var}(X_v|X_{S_t})$ by computing the sample variance.

This approach assumes that nodes are capable of local synchronization, in which each node $v \in V \setminus S_t$ knows which messages to expect before its m-th iteration (in this case, the states $\{z_u[m-1] : u \in N(v) \setminus S_t\}$) [15, Ch. 1.4]. Local synchronization requires less overhead than global synchronization, which requires all nodes to agree on a common time index, and hence is a common assumption in sensor networks [99]. Furthermore, nodes in S_t do not send state values at any iteration, and are therefore not incorporated in state updates. Hence, each node $v \in V \setminus S_t$ does not require any knowledge of S_t, since these nodes are automatically excluded from computation of the algorithm.

A Lyapunov function argument shows that the variance σ_v^2 of node v converges to $\text{var}(X_v|X_{S_t})$; we omit the proof due to space constraints. It remains for each node to compute the average variance $f(S_t)$ over all the nodes $u \notin S_t$.

8.5.3 Experimental Results

For the unconstrained case, the overall utility function was equal to $f(S) - c|S|$, where c represents the cost of activating each sensor; we chose $c = \frac{1}{n}$, so that the total cost is equal to the fraction of sensors activated. For comparison, we also evaluated the performance of random sensor activation. As shown in Figure 8.1(a), the distributed online approach first experiences a rapid decrease in utility, as multiple nodes randomly decide to activate, thus increasing the cost. As the time progresses, these nodes observe the decrease in their utility and deactivate, eventually approaching a utility of 0.9. Since 1 is an upper bound on the maximum achievable utility, DUO is at least 0.9-optimal for this problem. At all time

periods, the algorithm DUO provides higher utility than random selection, and converges to a comparable utility to the offline greedy algorithm.

Figure 8.1(b) shows the utility of the constrained case as a function of time with $k = 5$. The algorithm DCO converges rapidly to a utility of 0.945. As a comparison, a centralized greedy algorithm provides utility of 0.97, while 1 is an upper bound on the utility that can be achieved. Moreover, DCO provides higher utility than the DOG algorithm. The average utility as a function of the constraint k is shown in Figure 8.1(c). Clearly, increasing the number of sensors that can be activated increases the utility. We observe a concave increase in utility for centralized greedy approach and a roughly linear increase for DCO and DOG. The rate of increase for the distributed algorithm is lower than that of the centralized case; however, when $k = 10$, DCO gives utility of 0.95 while the greedy algorithm gives 0.99.

8.6 Conclusions and Future Work

In this chapter, we studied the problem of submodular maximization in a distributed, online setting, in which each node must decide whether to join a given set in order to maximize a submodular objective function. We considered two versions of the distributed submodular maximization problem. In the first case, there are no constraints on which nodes can join the set. In the second case, the number of nodes in the set cannot exceed a fixed number k and the function to be maximized is assumed to be monotone.

For both cases, we introduced distributed algorithms for submodular maximization with $O(1)$ expected optimality gap. In the unconstrained case, our distributed algorithm achieves a 0.33 optimality gap, compared to the best centralized algorithm's 0.5 optimality gap. Our algorithm for maximization in the constrained case achieves a 0.5 optimality gap, compared to a (1-1/e) gap for centralized algorithms. Both algorithms can be computed by nodes with limited computation, communication, and storage capabilities, and without knowledge of the number of nodes, any cardinality constraint, or access to a value oracle of the objective function. An experimental study using sensor network data showed that both algorithms provide near-optimal empirical performance.

8.6.1 Future Work

Improved Optimality Gap: The optimality guarantee of $1/2$ achieved by our proposed algorithm DCO is less than the $(1-1/e)$ guarantee achieved by the greedy centralized algorithm. In our future work, we will investigate additional problem structure that may enhance the performance of local search algorithms such as DCO. One possible set of problems arise in facility location, where exchange-based algorithms are known to provide near-optimal solutions [7].

Matroid Constraints: While the approach outlined in this chapter provides algorithms for unconstrained and cardinality-constrained submodular maximization, matroid-constrained optimization remains an open problem. Generalizing the cardinality-constrained case from arbitrary exchanges to matroid exchange [77] is a promising approach to incorporate these more general constraints.

BIBLIOGRAPHY

[1] Data sheets for IEEE 14 bus system. http://shodhganga.inflibnet.ac.in/bitstream/10603/5247/18/19_appendix.pdf.

[2] Intel Berkeley Sensor Network Dataset. http://db.csail.mit.edu/labdata/labdata.html.

[3] J. A. Acebrón, L. L. Bonilla, C. J. P. Vicente, F. Ritort, and R. Spigler. The Kuramoto model: A simple paradigm for synchronization phenomena. *Reviews of Modern Physics*, 77(1):137, 2005.

[4] E. Altman and Z. Altman. S-modular games and power control in wireless networks. *IEEE Transactions on Automatic Control*, 48(5):839–842, 2003.

[5] M. Andrews and L. Zhang. Scheduling algorithms for multi-carrier wireless data systems. *Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 3–14, 2007.

[6] A. Arenas, A. Díaz-Guilera, J. Kurths, Y. Moreno, and C. Zhou. Synchronization in complex networks. *Physics Reports*, 469(3):93–153, 2008.

[7] V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pandit. Local search heuristics for k-median and facility location problems. *SIAM Journal on Computing*, 33(3):544–562, 2004.

[8] T. Başar and G. J. Olsder. *Dynamic Noncooperative Game Theory*, volume 200. SIAM, 1995.

[9] R. Baldoni, A. Corsaro, L. Querzoni, S. Scipioni, and S. T. Piergiovanni. Coupling-based internal clock synchronization for large-scale dynamic distributed systems. *IEEE Transactions on Parallel and Distributed Systems*, 21(5):607–619, 2010.

[10] B. Bamieh, M.R. Jovanovic, P. Mitra, and S. Patterson. Coherence in large-scale networks: Dimension-dependent limitations of local feedback. *IEEE Transactions on Automatic Control*, 57(9):2235 –2249, 2012.

[11] P. Barooah, N. M. da Silva, and J. P. Hespanha. Distributed optimal estimation from relative measurements for localization and time synchronization. *Distributed Computing in Sensor Systems*, pages 266–281, 2006.

[12] P. Barooah and J.P. Hespanha. Graph effective resistance and distributed control: Spectral properties and applications. *45th IEEE Conference on Decision and Control (CDC)*, pages 3479–3485, 2006.

[13] W. Basener, B. P. Brooks, and D. Ross. The Brouwer fixed point theorem applied to rumour transmission. *Applied Mathematics Letters*, 19(8):841–842, 2006.

[14] A. R. Bergen. *Power Systems Analysis*. Pearson Education India, 2009.

[15] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation*. Prentice Hall Inc., 1989.

[16] C. Bettstetter, G. Resta, and P. Santi. The node distribution of the random waypoint mobility model for wireless ad hoc networks. *IEEE Transactions on Mobile Computing*, 2(3):257–269, 2003.

[17] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Randomized gossip algorithms. *IEEE Transactions on Information Theory*, 52(6):2508–2530, 2006.

[18] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2009.

[19] N. Buchbinder, M. Feldman, J. S. Naor, and R. Schwartz. A tight linear time (1/2)-approximation for unconstrained submodular maximization. *IEEE 53rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 649–658, 2012.

[20] K. Cai and H. Ishii. Convergence time analysis of quantized gossip consensus on digraphs. *Automatica*, 48(9):2344–2351, 2012.

[21] T. Calamoneri, A. Clementi, M. Ianni, M. Lauria, A. Monti, and R. Silvestri. Minimum energy broadcast and disk cover in grid wireless networks. *Theoretical Computer Science*, 399:38–53, 2008.

[22] G. Calinescu, C. Chekuri, M. Pal, and Jan Vondrak. Maximizing a submodular set function subject to a matroid constraint. *SIAM Journal on Computing*, 40(6):1740–1766, 2011.

[23] M. Cao, A S. Morse, and B. D. O. Anderson. Reaching a consensus in a dynamically changing environment: A graphical approach. *SIAM Journal on Control and Optimization*, 47(2):575–600, 2008.

[24] Y. Cao, W. Yu, W. Ren, and G. Chen. An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial Informatics*, 9(1):427–438, 2013.

[25] M. Cassidy, P. Mazzone, A. Oliviero, A. Insola, P. Tonali, V. Di Lazzaro, and P. Brown. Movement-related changes in synchronization in the human basal ganglia. *Brain*, 125(6):1235–1246, 2002.

[26] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

[27] A.K. Chandra, P. Raghavan, W.L. Ruzzo, R. Smolensky, and P. Tiwari. The electrical resistance of a graph captures its commute and cover times. *Computational Complexity*, 6(4):312–340, 1996.

[28] Venkat Chandrasekaran, Pablo A Parrilo, and Alan S Willsky. Convex graph invariants. *SIAM Review*, 54(3):513–541, 2012.

[29] A. Chapman and M. Mesbahi. On strong structural controllability of networked systems: a constrained matching approach. *American Control Conference (ACC)*, pages 6126–6131, 2013.

[30] S. Chatterjee and E. Seneta. Towards consensus: Some convergence theorems on repeated averaging. *Journal of Applied Probability*, pages 89–97, 1977.

[31] C. Chekuri, J. Vondrak, and R. Zenklusen. Dependent randomized rounding via exchange properties of combinatorial structures. *51st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 575–584, 2010.

[32] L. M. Childs and S. H. Strogatz. Stability diagram for the forced Kuramoto model. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 18, 2008.

[33] N. Chopra and M. W. Spong. On exponential synchronization of Kuramoto oscillators. *IEEE Transactions on Automatic Control*, 54(2):353–357, 2009.

[34] A. Clark, B. Alomair, L. Bushnell, and R. Poovendran. Minimizing convergence error in multi-agent systems via leader selection: A supermodular optimization approach. *IEEE Transactions on Automatic Control*, 59(6):1480–1494, 2014.

[35] A. Clark, L. Bushnell, and R. Poovendran. Leader selection for minimizing convergence error in leader-follower systems: A supermodular optimization approach. *10th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*, pages 111–115, 2012.

[36] A. Clark, L. Bushnell, and R. Poovendran. On leader selection for performance and controllability in multi-agent systems. *51st IEEE Conference on Decision and Control (CDC)*, pages 86–93, 2012.

[37] A. Clark, L. Bushnell, and R. Poovendran. Joint leader and weight selection for fast convergence in multi-agent systems. *Proceedings of the American Control Conference (ACC)*, pages 3751–3757, 2013.

[38] Andrew Clark, Linda Bushnell, and Radha Poovendran. A supermodular optimization framework for leader selection under link noise in linear multi-agent systems. *IEEE Transactions on Automatic Control*, 59(2):283–297, 2014.

[39] J Corfmat and A.S. Morse. Structurally controllable and structurally canonical systems. *IEEE Transactions on Automatic Control*, 21(1):129–131, 1976.

[40] C. Daskalakis, A. Mehta, and C. Papadimitriou. A note on approximate nash equilibria. *Internet and Network Economics*, pages 297–306, 2006.

[41] J. P. Desai, J. P. Ostrowski, and V. Kumar. Modeling and control of formations of nonholonomic mobile robots. *IEEE Transactions on Robotics and Automation*, 17(6):905–908, 2001.

[42] J.-M. Dion, C. Commault, and J. Van Der Woude. Generic properties and control of linear structured systems: a survey. *Automatica*, 39(7):1125–1144, 2003.

[43] F. Dörfler and F. Bullo. Synchronization and transient stability in power networks and nonuniform Kuramoto oscillators. *SIAM Journal on Control and Optimization*, 50(3):1616–1642, 2012.

[44] F. Dörfler, M. Chertkov, and F. Bullo. Synchronization in complex oscillator networks and smart grids. *Proceedings of the National Academy of Sciences*, 110(6):2005–2010, 2013.

[45] P. G. Doyle and J. L. Snell. *Random Walks and Electric Networks*. Carus Mathematical Monographs, 1984.

[46] R. Eckhorn, R. Bauer, W. Jordan, M. Brosch, W. Kruse, M. Munk, and H.J. Reitboeck. Coherent oscillations: A mechanism of feature linking in the visual cortex? *Biological Cybernetics*, 60(2):121–130, 1988.

[47] M. Fardad, F. Lin, and M. Jovanovic. Algorithms for leader selection in large dynamical networks: Noise-free leaders. *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pages 7188–7193, 2011.

[48] U. Feige. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.

[49] U. Feige, V. Mirrokni, and J. Vondrak. Maximizing non-monotone submodular functions. *SIAM Journal on Computing*, 40(4):1133–1153, 2011.

[50] G. Filatrella, A. H. Nielsen, and N. F. Pedersen. Analysis of a power grid using a Kuramoto-like model. *The European Physical Journal B-Condensed Matter and Complex Systems*, 61(4):485–491, 2008.

[51] K. Fitch and N.E. Leonard. Information centrality and optimal leader selection in noisy networks. *52nd IEEE Conference on Decision and Control (CDC)*, pages 7510–7515, 2013.

[52] Y. Freund and R. E. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. *Computational Learning Theory*, pages 23–37, 1995.

[53] S. Fujishige. *Submodular Functions and Optimization*. Elsevier Science, 2005.

[54] Victor Gabillon, Branislav Kveton, Zheng Wen, Brian Eriksson, and S Muthukrishnan. Adaptive submodular maximization in bandit setting. *Advances in Neural Information Processing Systems*, pages 2697–2705, 2013.

[55] D. Goldin and J. Raisch. On the weight controllability of consensus algorithms. *IEEE European Control Conference (ECC)*, pages 233–238, 2013.

[56] D. Golovin, M. Faulkner, and A. Krause. Online distributed sensor selection. *Proceedings of the 9th ACM/IEEE International Conference on Information Processing in Sensor Networks*, pages 220–231, 2010.

[57] N. Golrezaei, K. Shanmugam, A. G. Dimakis, A. F. Molisch, and G. Caire. Femtocaching: Wireless video content delivery through distributed caching helpers. *Proceedings of IEEE Infocom*, pages 1107–1115, 2012.

[58] M. H. DeGroot. Reaching a consensus. *Journal of the American Statistical Association*, 69(345):118–121, 1974.

[59] Y. Hatano and M. Mesbahi. Agreement over random networks. *IEEE Transactions on Automatic Control*, 50(11):1867–1872, 2005.

[60] X. Hong, M. Gerla, G. Pei, and C.C. Chiang. A group mobility model for ad hoc wireless networks. *2nd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems*, pages 53–60, 1999.

[61] A.-S. Hu and S. D. Servetto. On the scalability of cooperative time synchronization in pulse-connected networks. *IEEE Transactions on Information Theory*, 52(6):2725–2748, 2006.

[62] S. Huang. Reprogramming cell fates: reconciling rarity with robustness. *Bioessays*, 31(5):546–560, 2009.

[63] 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, 2003.

[64] A. Jadbabaie, N. Motee, and M. Barahona. On the stability of the Kuramoto model of coupled nonlinear oscillators. *Proceedings of the American Control Conference (ACC)*, 5:4296–4301, 2004.

[65] D. Jakovetic, J. Xavier, and J. M. F. Moura. Weight optimization for consensus algorithms with correlated switching topology. *IEEE Transactions on Signal Processing*, 58(7):3788–3801, 2010.

[66] M. Ji, G. Ferrari-Trecate, M. Egerstedt, and A. Buffa. Containment control in mobile networks. *IEEE Transactions on Automatic Control*, 53(8):1972–1975, 2008.

[67] M. Ji, A. Muhammad, and M. Egerstedt. Leader-based multi-agent coordination: Controllability and optimal control. *American Control Conference (ACC)*, pages 1358–1363, 2006.

[68] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 137–146, 2003.

[69] H. K. Khalil and J.W. Grizzle. *Nonlinear Systems*. Prentice Hall Upper Saddle River, 2002.

[70] W. Klimesch. Memory processes, brain oscillations and EEG synchronization. *International Journal of Psychophysiology*, 24(1):61–100, 1996.

[71] H. Kori and A. S. Mikhailov. Entrainment of randomly coupled oscillator networks by a pacemaker. *Physical Review Letters*, 93(25):254101, 2004.

[72] A. Krause, B. McMahan, C. Guestrin, and A. Gupta. Selecting observations against adversarial objectives. *Advances in Neural Information Processing Systems (NIPS)*, pages 777–784, 2008.

[73] A. Krause, R. Rajagopal, A. Gupta, and C. Guestrin. Simultaneous optimization of sensor placements and balanced schedules. *IEEE Transactions on Automatic Control*, 56(10):2390–2405, 2011.

[74] A. Krause, A. Singh, and C. Guestrin. Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. *The Journal of Machine Learning Research*, 9:235–284, 2008.

[75] Andreas Krause and Daniel Golovin. Submodular function maximization. *Tractability: Practical Approaches to Hard Problems*, 3:19, 2012.

[76] Y. Kuramoto. Self-entrainment of a population of coupled non-linear oscillators. *International Symposium on Mathematical Problems in Theoretical Physics*, pages 420–422, 1975.

[77] J. Lee, V. Mirrokni, V. Nagarajan, and M. Sviridenko. Maximizing non-monotone submodular functions under matroid or knapsack constraints. *SIAM Journal on Discrete Mathematics*, 23(4):2053–2078, 2010.

[78] A. Leon-Garcia. *Probability, Statistics, and Random Processes for Electrical Engineers*. Pearson Prentice Hall, 2008.

[79] N. E. Leonard, T. Shen, B. Nabet, L. Scardovi, I. D. Couzin, and S. A. Levin. Decision versus compromise for animal groups in motion. *Proceedings of the National Academy of Sciences*, 109(1):227–232, 2012.

[80] D.A. Levin, Y. Peres, and E.L. Wilmer. *Markov Chains and Mixing Times*. American Mathematical Society, 2009.

[81] C.T. Lin. Structural controllability. *IEEE Transactions on Automatic Control*, 19(3):201–208, 1974.

[82] F. Lin, M. Fardad, and M. Jovanovic. Algorithms for leader selection in large dynamical networks: Noise-corrupted leaders. In *Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*. IEEE, 2011.

[83] F. Lin, M. Fardad, and M.R. Jovanovic. Optimal control of vehicular formations with nearest neighbor interactions. *IEEE Transactions on Automatic Control*, 57(9):2203–2218, 2012.

[84] B. Liu, T. Chu, L. Wang, and G. Xie. Controllability of a leader–follower dynamic network with switching topology. *IEEE Transactions on Automatic Control*, 53(4):1009–1013, 2008.

[85] Y.Y. Liu, J.J. Slotine, and A.L. Barabási. Controllability of complex networks. *Nature*, 473(7346):167–173, 2011.

[86] L. Lovász. Random walks on graphs: A survey. *Combinatorics, Paul Erdos is Eighty*, 2(1):1–46, 1993.

[87] C. C. McIntyre and P. J. Hahn. Network perspectives on the mechanisms of deep brain stimulation. *Neurobiology of Disease*, 38(3):329–337, 2010.

[88] M. Mesbahi and M. Egerstedt. *Graph Theoretic Methods in Multiagent Networks*. Princeton University Press, 2010.

[89] M. Mesbahi and F.Y. Hadaegh. Graphs, matrix inequalities, and switching for the formation flying control of multiple spacecraft. *American Control Conference (ACC)*, 6:4148–4152, 1999.

[90] D. C. Michaels, E. P. Matyas, and J. Jalife. Mechanisms of sinoatrial pacemaker synchronization: a new hypothesis. *Circulation Research*, 61(5):704–714, 1987.

[91] R.E. Mirollo and S.H. Strogatz. Synchronization of pulse-coupled biological oscillators. *SIAM Journal on Applied Mathematics*, pages 1645–1662, 1990.

[92] P. Monzón and F. Paganini. Global considerations on the Kuramoto model of sinusoidally coupled oscillators. *Proceedings of the 44th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pages 3923–3928, 2005.

[93] L. Moreau. Stability of continuous-time distributed consensus algorithms. *Proceedings of the 43rd IEEE Conference on Decision and Control (CDC)*, 4:3998–4003, 2004.

[94] Kazuo Murota. Refined study on structural controllability of descriptor systems by means of matroids. *SIAM Journal on Control and Optimization*, 25(4):967–989, 1987.

[95] G. L. Nemhauser and L. A. Wolsey. Best algorithms for approximating the maximum of a submodular set function. *Mathematics of Operations Research*, 3(3):177–188, 1978.

[96] G.L. Nemhauser, L.A. Wolsey, and M.L. Fisher. An analysis of approximations for maximizing submodular set functions - I. *Mathematical Programming*, 14(1):265–294, 1978.

[97] N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, 2007.

[98] G. Notarstefano, M. Egerstedt, and M. Haque. Containment in leader–follower networks with switching communication topologies. *Automatica*, 47(5):1035–1040, 2011.

[99] 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, 2004.

[100] R. Olfati-Saber. Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Transactions on Automatic Control*, 51(3):401–420, 2006.

[101] 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, 2004.

[102] A. Olshevsky. Minimum input selection for structural controllability. *arXiv preprint arXiv:1407.2884*, 2014.

[103] A. Olshevsky and J. N. Tsitsiklis. Convergence speed in distributed consensus and averaging. *SIAM Journal on Control and Optimization*, 48(1):33–55, 2009.

[104] J.G. Oxley. *Matroid Theory*. Oxford University Press, 1992.

[105] D. A. Paley, N. E. Leonard, R. Sepulchre, D. Grunbaum, and J. K. Parrish. Oscillator models and collective motion. *IEEE Control Systems*, 27(4):89–105, 2007.

[106] F. Pasqualetti, S. Martini, and A. Bicchi. Steering a leader-follower team via linear consensus. *Hybrid Systems: Computation and Control*, pages 642–645, 2008.

[107] F. Pasqualetti, S. Zampieri, and F. Bullo. Controllability metrics and algorithms for complex networks. *arXiv preprint arXiv:1308.1201*, 2013.

[108] S. Patterson and B. Bamieh. Leader selection for optimal network coherence. In *Proceedings of the 49th IEEE Conference on Decision and Control (CDC)*, pages 2692–2697. IEEE, 2010.

[109] S. Patterson, B. Bamieh, and A. El Abbadi. Convergence rates of distributed average consensus with stochastic link failures. *IEEE Transactions on Automatic Control*, 55(4):880–892, 2010.

[110] O. V. Popovych and P. A. Tass. Macroscopic entrainment of periodically forced oscillatory ensembles. *Progress in Biophysics and Molecular Biology*, 105(1):98–108, 2011.

[111] J.G. Proakis. Spread spectrum signals for digital communications. *Encyclopedia of Telecommunications*, 2001.

[112] A. Rahmani, M. Ji, M. Mesbahi, and M. Egerstedt. Controllability of multi-agent systems from a graph-theoretic perspective. *SIAM Journal on Control and Optimization*, 48(1):162–186, 2009.

[113] K.J. Reinschke and G. Wiedemann. Digraph characterization of structural controllability for linear descriptor systems. *Linear algebra and its applications*, 266:199–217, 1997.

[114] W. Ren. On consensus algorithms for double-integrator dynamics. *IEEE Transactions on Automatic Control (TAC)*, 53(6):1503–1509, 2008.

[115] W. Ren and R. W. Beard. *Distributed Consensus in Multi-vehicle Cooperative Control: Theory and Applications*. Springer, 2007.

[116] M. G. Rosenblum, A. S. Pikovsky, and J. Kurths. Phase synchronization of chaotic oscillators. *Physical Review Letters*, 76(11):1804, 1996.

[117] C.G. Rusin, H. Kori, I. Z. Kiss, and J. L. Hudson. Synchronization engineering: tuning the phase relationship between dissimilar oscillators using nonlinear feedback. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 368(1918):2189–2204, 2010.

[118] J. Ruths and D. Ruths. Control profiles of complex networks. *Science*, 343:1373–1376, 2014.

[119] V. S. Borkar, J. Nair, and N. Sanketh. Manufacturing consent. *Proceedings of the 48th Annual Allerton Conference on Communication, Control, and Computing*, pages 1550–1555, 2010.

[120] A. Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*. Springer, 2003.

[121] D. Shah. *Gossip Algorithms*. Now Publishers, 2009.

[122] J. S. Shamma and G. Arslan. Dynamic fictitious play, dynamic gradient play, and distributed convergence to Nash equilibria. *IEEE Transactions on Automatic Control*, 50(3):312–327, 2005.

[123] Y. Shoham and K. Leyton-Brown. *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press, 2009.

[124] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. *ACM SIGCOMM Computer Communication Review*, 31(4):149–160, 2001.

[125] M. Streeter and D. Golovin. An online algorithm for maximizing submodular functions. In *Carnegie Mellon University Technical Report*, 2007.

[126] M. Streeter and D. Golovin. An online algorithm for maximizing submodular functions. *Advances in Neural Information Processing Systems (NIPS)*, pages 1577–1584, 2008.

[127] Steven 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.

[128] A. Tahbaz-Salehi and A. Jadbabaie. A necessary and sufficient condition for consensus over random networks. *IEEE Transactions on Automatic Control*, 53(3):791–795, 2008.

[129] H.G. Tanner. On the controllability of nearest neighbor interconnections. *43rd IEEE Conference on Decision and Control (CDC)*, 3:2467–2472, 2004.

[130] L.N. Trefethen and D. Bau. *Numerical Linear Algebra*. Society for Industrial Mathematics, 1997.

[131] J. van der Woude and K. Murota. Disturbance decoupling with pole placement for structured systems: a graph-theoretic approach. *SIAM Journal on Matrix Analysis and Applications*, 16(3):922–942, 1995.

[132] L. R. Varshney, B. L. Chen, E. Paniagua, D. H. Hall, and D. B. Chklovskii. Structural properties of the Caenorhabditis Elegans neuronal network. *PLoS Computational Biology*, 7(2), 2011.

[133] Yongqiang Wang and Francis J Doyle. Exponential synchronization rate of Kuramoto oscillators in the presence of a pacemaker. *IEEE Transactions on Automatic Control*, 58(4):989–994, 2013.

[134] D. A. Wiley, S. H. Strogatz, and M. Girvan. The size of the sync basin. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 16, 2006.

[135] A. T. Winfree. Biological rhythms and the behavior of populations of coupled oscillators. *Journal of Theoretical Biology*, 16(1):15–42, 1967.

[136] L.A. Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2(4):385–393, 1982.

[137] L.A. Wolsey and G.L. Nemhauser. *Integer and Combinatorial Optimization*. Wiley-Interscience, 1999.

[138] C. Wu, Y. Xu, Y. Chen, and C. Lu. Submodular game for distributed application allocation in shared sensor networks. *Proceedings of IEEE Infocom*, pages 127–135, 2012.

[139] L. Xiao, S. Boyd, and S. Kim. Distributed average consensus with least-mean-square deviation. *Journal of Parallel and Distributed Computing*, 67(1):33 – 46, 2007.

[140] E. Yip and R. Sincovec. Solvability, controllability, and observability of continuous descriptor systems. *IEEE Transactions on Automatic Control*, 26(3):702–707, 1981.

[141] Y. Yuan, J. Liu, R. M. Murray, and J. Gonçalves. Decentralised minimal-time dynamic consensus. *Proceedings of the American Control Conference*, pages 800–805, 2012.

[142] Z. Zheng and N. B. Shroff. Maximizing a submodular utility for deadline constrained data collection in sensor networks. *10th IEEE International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*, pages 116–123, 2012.

Appendix A

LIST OF PUBLICATIONS

Journal Publications

1. A. Clark, L. Bushnell, and R. Poovendran. Minimizing Convergence Error in Multi-Agent Systems via Leader Selection: A Supermodular Optimization Approach. *IEEE Transactions on Automatic Control (TAC)*, vol. 59, no.6, pp. 1480–1494, June 2014.
2. A. Clark, L. Bushnell, and R. Poovendran. A Supermodular Optimization Framework for Leader Selection under Link Noise in Linear Multi-Agent Systems. *IEEE Transactions on Automatic Control (TAC)*, vol. 59, no. 2, pp. 283–297, February 2014.
3. P. Lee, A. Clark, L. Bushnell, and R. Poovendran. A Passivity Framework for Modeling and Mitigating Wormhole Attacks on Networked Control Systems. To appear in *IEEE Transactions on Automatic Control (TAC), Special Issue on Control of Cyber-Physical Systems*, 2014.
4. B. Alomair, A. Clark, J. Cuellar, and R. Poovendran. Statistical Framework for Source Anonymity in Sensor Networks. *IEEE Transactions on Mobile Computing (TMC)*, vol. 12, no. 2, pp. 248–260, February 2013.
5. B. Alomair, A. Clark, J. Cuellar, and R. Poovendran. Scalable RFID Systems: a Privacy-Preserving Protocol with Constant-Time Identification. *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, vol. 23, no. 8, pp. 1536–1550, August 2012.
6. B. Alomair, A. Clark, and R. Poovendran. The Power of Primes: Security of Authentication Based on a Universal Hash-Function Family. *Journal of Mathematical Cryptology*, vol. 4, no. 2, pp. 121–148, 2010.

Conference Publications

1. A. Clark, B. Alomair, L. Bushnell, and R. Poovendran. Distributed Online Submodular Maximization in Resource-Constrained Networks. In *Proceedings of the 12th IEEE International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, pp. 397–404, 2014. **Best Student Paper Award**.
2. A. Clark, K. Sun, and R. Poovendran. Effectiveness of IP Address Randomization in Decoy-Based Moving Target Defense. In *Proceedings of the 52nd IEEE Conference on Decision and Control (CDC)*, pp. 678–685, 2013.
3. Q. Zhu, A. Clark, R. Poovendran, and T. Başar. Deployment and Exploitation of Deceptive Honeybots in Social Networks. To appear in *Proceedings of the 52nd IEEE Conference on Decision and Control (CDC)*, pp. 212–219, 2013.
4. A. Clark, L. Bushnell, and R. Poovendran. Joint Leader and Weight Selection for Fast Convergence in Multi-Agent Systems. In *Proceedings of the IEEE American Control Conference (ACC)*, pp. 3751–3757, 2013.
5. A. Clark, Q. Zhu, R. Poovendran, and T. Başar. An Impact-Aware Defense Against Stuxnet. In *Proceedings of the IEEE American Control Conference (ACC)*, pp. 4070–4077, 2013.
6. P. Lee, A. Clark, L. Bushnell, and R. Poovendran. Modeling and Designing Network Defense against Control Channel Jamming Attacks: A Passivity-Based Approach. In *Workshop on Control of Cyber-Physical Systems, co-located with IEEE Conference on Information Sciences and Systems (CISS), 2013.* **Invited Paper.**
7. A. Clark, L. Bushnell, and R. Poovendran. On Leader Selection for Performance and Controllability in Multi-Agent Systems. In: *Proceedings of the 51st IEEE Conference on Decision and Control (CDC '12)*, pp. 86–93, 2012. **Finalist for Student Best Paper Award.**
8. Q. Zhu, A. Clark, R. Poovendran, and T. Başar. Deceptive Routing Games. In: *Proceedings of the 51st IEEE Conference on Decision and Control (CDC '12)*, pp. 2704–2711, 2012.
9. A. Clark, B. Alomair, L. Bushnell, and R. Poovendran. Leader Selection in Multi-

Agent Systems for Smooth Convergence via Fast Mixing. In: *Proceedings of the 51st IEEE Conference on Decision and Control (CDC '12)*, pp. 818–824, 2012.

10. A. Clark, Q. Zhu, R. Poovendran, and T. Başar. Deceptive Routing in Relay Networks. In: *Proceedings of the 3rd IEEE Conference on Game and Decision Theory for Security (GameSec '12)*, pp. 171–185 2012.
11. A. Clark, L. Bushnell, and R. Poovendran. A Passivity-based Framework for Composing Attacks on Networked Control Systems. In: *Proceedings of the 50th Allerton Conference on Communication, Control, and Computing (Allerton '12)*, 2012. **Invited Paper.**
12. A. Clark, L. Bushnell, and R. Poovendran. Leader Selection for Minimizing Convergence Error in Leader-Follower Systems: A Supermodular Optimization Approach. In: *Proceedings of the 10th IEEE International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, pp. 111–115, 2012. **Best Paper Award**
13. A. Clark, L. Bushnell, and R. Poovendran. Leader Selection Games Under Link Noise Injection Attacks. In: *Proceedings of the 1st ACM International Conference on High Confidence Networked Systems (HiCONS)*, pp. 31–40, 2012.
14. A. Clark and R. Poovendran. Maximizing Influence in Competitive Environments: A Game-Theoretic Approach. In: *Proceedings of the 2nd IEEE Conference on Decision and Game Theory for Security (GameSec)*, pp. 151–162, 2011.
15. A. Clark and R. Poovendran. A Submodular Optimization Framework for Leader Selection in Multi-Agent Systems. In: *Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pp. 3614–3621, 2011.
16. A. Clark, R. Hardy, and R. Poovendran. A Joint Performance-Vulnerability Metric Framework for Designing Ad Hoc Routing Protocols. In: *Proceedings of the IEEE Military Communications Conference (MILCOM)*, pp. 930–935, 2010.
17. B. Alomair, A. Clark, J. Cuellar, and R. Poovendran. Statistical Framework for Source Anonymity in Sensor Networks. In: *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, 2010.

18. B. Alomair, A. Clark, J. Cuellar, and R. Poovendran. Scalable RFID Systems: a Privacy-Preserving Protocol with Constant-Time Identification. In: *Proceedings of the The 40th Annual IEEE/IFIP International Conference on Dependable Systems and Networks-DSN'10* **Received the 2010 IEEE/IFIP William C. Carter Award.**
19. A. Clark and R. Poovendran. A Metric for Quantifying Key Exposure Vulnerability in Wireless Sensor Networks. In: *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC)*, 2010.
20. B. Alomair, K. Sampigethaya, A. Clark, and R. Poovendran. Towards Trustworthy Cryptographic Protection of Airplane Information Assets. In: *Proceedings of the AIAA Infotech@Aerospace Conference*, 2009.

Conference Submissions Under Review

1. A. Clark, B. Alomair, L. Bushnell, and R. Poovendran. A Submodular Optimization Approach to Global Practical Synchronization of Kuramoto Oscillators. Submitted to American Control Conference (ACC), 2015.
2. P. Lee, A. Clark, B. Alomair, L. Bushnell, and R. Poovendran. Jamming-Based Adversarial Control of Network Flow Allocation: A Passivity Approach. Submitted to American Control Conference (ACC), 2015.
3. P. Lee, A. Clark, L. Bushnell, and R. Poovendran. Passivity Framework for Composition and Mitigation of Multi-Virus Propagation in Networked Systems. Submitted to American Control Conference (ACC), 2015.

VITA

Andrew Clark received the BS degree in Electrical Engineering and the MS degree in Mathematics from the University of Michigan - Ann Arbor in 2007 and 2008, respectively. His research interests include performance and security of cyber-physical systems, modeling and design of adaptive and proactive network defenses, lightweight cryptography, and vulnerability metrics. He was co-author of the IEEE/IFIP William C. Carter award paper (2010), the WiOpt Best Paper (2012), and the WiOpt Student Best Paper (2014), and was a finalist for the IEEE CDC 2012 Best Student Paper Award. He received the University of Washington Center for Information Assurance and Cybersecurity Distinguished Research Award (2012) and the University of Washington Center Information Assurance and Cybersecurity Distinguished Dissertation Award (2014). He holds a patent on privacy-preserving constant-time identification in RFID systems.

INDEX