ON BEHAVIOUR-BASED NETWORK INFERENCE AND DISTRIBUTION-BASED NETWORK CONTROL

by

Dave McKenney

A thesis submitted to the
Faculty of Graduate and Postdoctoral Affairs
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

School of Computer Science

at

CARLETON UNIVERSITY

Ottawa, Ontario
September, 2017

Table of Contents

List of Tables

List of Algorithms

List of Figures

Abstract

Complex networks are prevalent structures throughout technological systems, and are also used to model many non-technological systems as well. Application domains that make use of networks range from financial systems, to biology/medicine, to online social networks and agent-based systems. There is a strong desire to control these types of systems, to avoid catastrophic failures, increase system stability, or achieve some known system goal.

The development of automated controllers for these types of systems is a complex problem that involves several key subproblems, including the selection of a control node set and the generation of control signals to be injected into the network. Previous research involving network control has typically assumed the underlying network connections are precisely known and has also taken a strictly vector-based view of system state. This thesis expands on the existing network control work in two significant directions.

First, this thesis investigates algorithmic, behaviour-based methods for predicting links within networks. This involves using transfer entropy measurements, calculated between time series of actions generated by participating agents. This approach could be used to predict underlying networks in unobservable problem domains (e.g., financial systems) or to identify links that are truly influential within observable problem domains (e.g., online social networks). A number of prediction algorithms are proposed and compared, several of which attain high levels of accuracy, even with a limited amount of available system information.

Second, this thesis eschews the traditional vector-based view of system state within network control problems, proposing a novel, distribution-based approach. One of the most studied control goals in the existing research has involved moving a system between two vector-based states. Distribution-based control, however, identifies state distributions as targets for control, which is arguably a more expressive and suitable approach for many problem domains. The effect of various network parameters on control success is investigated within a distribution-based control problem, with microscopic analysis of subset distributions being used to demonstrate why control is more difficult in certain scenarios. This information is used to inform the creation of new control node selection algorithms, with statistically significant improvements over the highest rated heuristic from previous research being realized.

Acknowledgements

To family, friends, colleagues, and everybody else who has influenced my life in one way or another to bring me to this point, thank you.

To Professor Tony White, thank you for your guidance, intellectual support, and seemingly unwavering confidence. While I wasn't always sure, I am happy that one of us was convinced I could do this.

List of Symbols

Symbol	Description	Page
α	Learning rate parameter within the SARSA algorithm	24
β	Randomness parameter used in Small World network generation	14
γ	Discount factor within the SARSA algorithm	24
τ	Random choice probability of an agent within an AMG	53
Δ_μ	Difference between target distribution mean and state distribution mean	89
Δ_σ	Difference between target distribution standard deviation and state distribution standard deviation	90
θ	Threshold value	86
$\theta{-}CAP$	$\theta-$Consensus Avoidance Problem	45
θ_g	Threshold value for a $\theta{-}CAP$ instance	45
θ_{TE}	Transfer entropy threshold value	59
A	A set of agents	22
A_{sample}	A sample taken from the population A	62
a	A single agent from the set of agents A	22
$ACC_h(v)$	Accessibility of vertex v given random walks of length h	30
ACT	The action set within the SARSA algorithm	22
AMG	Anti-majority game instance	50
B	Control signal matrix used in structural control theory	41
BGT	Control budget amount	90
$BC(v)$	Betweenness centrality of a vertex v	11
C	Set of game choices within an AMG	50
C_{pop}	Most popular choice in an AMG round	51
$C4.5$	An algorithm for constructing decision trees	67
$CMAC$	Cerebellar model arithmetic computer	25
COM	Communication strategy of an agent within an AMG	52
CON	Controller within a distribution-based network control problem	83

D	Matrix of all-pairs shortest paths for a network	95
D_S	State distribution	53
D_T	Target distribution	83
d	Degree parameter used in network generation algorithm	14
$damp$	Damping factor parameter within the PageRank calculation	12
$DC(v)$	Degree centrality of the vertex v	28
$deg(v)$	Degree of the vertex v	11
DS	Decision strategy of an agent within an AMG	52
E	Edge set within a graph or network	11
FAR	A heuristic-based control node set selection algorithm	45
FAR_{min}	A minimization-based variant of the FAR algorithm	118
G	Graph, generally consisting of a set of vertices and edges	13
$GEI(x)$	Global effective influence of a node x	56
$H(X)$	Shannon entropy of a time series X	17
$H(Y\|X)$	Conditional entropy of time series Y when given time series X	17
$H(x,y)$	Hellinger distance between the distributions x and y	85
H_{max}	Hellinger distance threshold	90
$I(Y;X)$	Mutual information between Y and X	18
$ID3$	An algorithm for constructing decision trees	67
$InFar$	An influence-weighted variant of the FAR algorithm	118
$InFar_{min}$	A minimization-based, influence-weighted variant of the FAR algorithm	119
k	Discrete domain of a probability function	85
KDE	Kernel density estimate	64
$L(x)$	Number of outgoing links of page x	12
$LEI(x,a)$	Local effective influence of node x on node a	56
M	Adjacency matrix representing a network	41
$M(x)$	Set of pages that link to page x in the PageRank calculation	12
MU	Memory update strategy of an agent within an AMG	52
N	Set of neighbouring vertices	52

$\mathcal{N}(x,y)$	A normal distribution with mean of x and standard deviation y	64
N_C	Set of control nodes within a distribution-based control problem	83
N_S	Set of sensor nodes within a distribution-based control problem	82
NCP	Network Control Problem	2
o	Order of the transfer entropy calculation (i.e., the time window)	18
$p(x)$	Probability of some event x occuring	17
$P(c)$	Payoff function within an AMG	50
P_F	Set of parameters that are free to vary	92
P_I	Domain of values for a parameter of interest	92
PM	Probability measure representing a distribution	85
$PW_h(v,t)$	Probability of reaching t using a self-avoiding random walk of h steps when starting from node v	30
$PR(x)$	PageRank value of the page/vertex x	12
Q	Estimated state/action value function used in SARSA	24
R	Reward function within the SARSA algorithm	22
ROC	Rate of change	83
S	State set within the SARSA algorithm	22
$s(v,t)$	State of node v at time t	83
$SARSA$	State-Action-Reward-State-Action reinforcement learning	23
SD	Standard deviation	103
SF	Scale Free network type	13
SW	Small World network type	14
T	State transition function within the SARSA algorithm	22
t	A reference to a particular time step	18
$TE_{X \to Y}$	Transfer entropy from a time series X to a time series Y	18
$TEGI(x)$	Transfer entropy-based global influence of x	56
$TELI(x,y)$	Transfer entropy-based local influence of x on y	56
$U(t)$	Utility function	45
V	Vertex set within a graph or network	11

Chapter 1

Introduction

Financial systems, social networks, information systems, and agent-based systems are just some of the social systems that are becoming increasingly prevalent throughout the world. Additionally, these are all examples of complex networks in which the flow of information causes changes in network state. Within these systems, the sharing of information drives changes to participants' state (e.g., opinions, outlooks, preferences) and actions. Over time, it is possible for these systems to naturally move toward, or be manipulated to, an undesirable point. This can result in negative consequences of varying severity, such as: financial system crashes (Sorkin, 2010), election manipulation, chatbot failures (Neff and Nagy, 2016), or undesirable product monopolies. To avoid these consequences, we should study the possibility of control within these systems to help us understand why they fail and also provide solutions to avoid or mitigate failures.

There are a number of important subproblems and questions that must be addressed when considering the control of complex networks[1]. The most important of these may be: what exactly does it mean to control a network (i.e., what are the goals of network control)? A large amount of existing network control research (e.g., Liu et al., 2011b) has focused on 'full state controllability' and 'structural control' (see Section 3.4.3 for full details), which views network control strictly as the ability to move a system from one state vector to another in an arbitrary amount of time; however, this is not the only possible view of network control. More recent work within the network control domain (e.g., Runka, 2016; Runka and White, 2015a; McKenney and White, 2016) has considered the behavioural aspects present within many networked systems, directing the focus toward a more practical view of network control. This opens up the possibility of many different types of network control problems involving failure avoidance, the limitation of state change, or other control goals. By moving beyond the full state controllability view of network control, new targets for control and state representations can also be realized, such as the aggregate measures used to determine failure in the work of Runka (2016). These types of representations and control

[1]Within this work, the terms 'network' and 'complex network' are used interchangeably.

problems are, arguably, significantly more applicable to many social control problems.

In solving any one specific control problem, there are several subproblems that must be considered. The first of these problems is determining the optimal control node set that can be selected within the system. This problem has been investigated in the existing structural control research, but there have also been arguments presented against the use of these solutions in many types of systems that do not fit well within the structural control framework (Cowan et al., 2012). The existing Network Control Problem (NCP) work has investigated a number of different control node set selection algorithms (Runka and White, 2015a; Runka, 2016), but there is still a need for more in depth analysis to determine the properties of an optimal control node set and identify methods for generating these sets. An additional problem that must be addressed within network control research is the selection of signals to inject into the system, or the selection of other control methods, such as the limitation of information flow, in order to achieve the desired control goals. The work of Runka (2016) uses neural networks successfully for this purpose, but there still exists a significant area of work that should aim to understand the effect of control signals within networked systems and develop improved control signal solutions.

One of the most significant contributions of previous structural control research has been the proposal of algorithmic methods to determine if a network is fully state controllable. As network control begins to move toward more practical applications involving behaviour and different types of control problems, it would be advantageous to develop similar tools that can answer questions like 'can the system be controlled within the given parameters?' or 'what is the probability of successful control?'. To answer these types of questions, an effort must be made to understand which properties of the systems under consideration affect control success likelihood, as well as the effect that control signals may have on a system's behaviour. For a failure avoidance problem, it may be beneficial to consider the system's natural momentum and the effect of the controller as opposing forces, which must be kept in balance to ensure successful control. In this case, developing methods for measuring the natural movement of the systems, as well as methods for measuring a controller's ability to affect this movement, are necessary. In understanding the dynamics of these systems in more detail, it may also be possible to develop alternative methods of network control, such as the modification of the network/system to increase controllability (e.g., by reducing or eliminating properties that lead to decreased success likelihood). Finally, existing research, and likely future research, often requires strong assumptions regarding system information, which may not be practical in many

applications. As an example, the structural analysis of Liu et al. (2011b) requires precise knowledge of the network connections present within a system. As knowledge relating to network connections, relationships, and influence plays such an important role in many proposed control solutions, it is important to develop methods for inferring as much of this information as possible in cases where it is not known precisely or is unobservable. Within many real-world systems, this is often the case, and even when links are known, the underlying properties of those links (relationship strength, frequency of communication, quality of communication, etc.) are generally unknown.

This thesis addresses several existing gaps within network control research, which are explained in the remainder of this chapter. Section 1.1 outlines the high level objectives of the thesis, while Section 1.2 contains the problem statement for this thesis. Section 1.3 discusses the scope of the thesis and details the questions that will be answered. These sections are followed by sections summarizing contributions (Section 1.4) and publications (Section 1.5). This chapter concludes by outlining the remaining structure of this document in Section 1.6.

1.1 Thesis Objectives

This thesis has two primary objectives. The first of these objectives is to demonstrate the viability of transfer entropy measures for the measurement of influence and prediction of who influences whom within social networks. The second primary objective of this work is to better understand what system properties have a strong effect on control success likelihood in order to support the development of more advanced network control solutions.

1.2 Problem Statement

Can observed activity traces be used to infer relationships between agents in a social network and is it possible to control the distributions of agent states within these networks?

1.3 Motivation

Section 1.1 outlined the two primary objectives of this thesis. To achieve these objectives, several relevant questions must be answered. The list below summarizes these key questions, while a short discussion follows that describes how each of these questions will be answered and why the

answers are important to the thesis objectives. Based on the objectives outlined above, the main questions that this thesis aims to answer are:

- Are transfer entropy influence measurements strongly correlated with known influence properties within a modelled system?
- Can transfer entropy measurements be used to predict who influences whom within a networked system?
- How do different amounts of available system knowledge affect the influence prediction accuracy using a transfer entropy approach?
- Can the distribution of state be controlled by a learned controller?
- What role, if any, do certain network properties and influence dynamics play in determining control success in a networked system?
- Can influence measurements be used to improve control node selection and control success in a network control problem?

The simulation of social network systems plays a large role within this thesis. As will be discussed throughout this thesis, simulations offer a number of benefits over real-world experimentation. One of the most significant of these advantages is that simulations rely on known models that produce specific types of behaviour. Within real-world systems, there are many possible confounding factors that can skew analysis. Within the simulations used here, the methods of influence and state change are known, which informs much of the analysis. Additionally, these simulated models allow different levels of abstraction to be applied, which can be used to simplify calculations, computation and analysis. For example, time and behaviour can be discretized within simulations, allowing the use of straightforward algorithms that would not be possible with real-world data. The networks considered within this thesis include those that are theoretically generated, as well as networks that are sampled from real-world social networks. So while the underlying system properties may be simplified to facilitate the research, many of the networks still reflect those that would be found in a practical application.

Within much of the existing network control research, it is assumed that the underlying network is known. This data is often extremely valuable within the domain of network control, as it provides details regarding the information flow within a system. In practice, however, this information may

not be readily available. Some networks may be difficult or impossible to observe, while other networks may provide a list of network connections, but lack any detail regarding the relationship between the end points of each link. For this reason, this thesis will investigate the use of transfer entropy for the measurement of influence and inference of network connections within a social system.

Within this thesis, it is assumed that a historical log of activity traces is maintained for each agent, which allows transfer entropy values between pairs of agents to be computed. The use of transfer entropy values to measure influence is investigated within the domain of the anti-majority game, which is formalized in Section 4.2. While existing work has considered transfer entropy-based influence identification using real-world data (Ver Steeg and Galstyan, 2013), the use of a theoretical dataset generated through simulation of the anti-majority game provides absolute knowledge of the ground truth relationships between each participant. This ground truth allows accurate theoretical measures of influence to be formulated based on the known influence dynamics of the system (see Section 4.3.1). These known measures, then, are used to quantify how closely transfer entropy-based influence measures come to replicating the real influence dynamics (Section 4.3.3). If high correlations between the theoretical influence measures and the transfer entropy influence measure can be identified, then the validity of transfer entropy as a form of influence measurement is verified.

In addition to the use of transfer entropy as an influence measure, a related problem involves predicting which participants affect the future actions of other participants within the system. Being able to accurately determine who influences whom within a networked system has several possible application areas, such as in marketing/advertising, or in network control problems such as the one investigated within Chapter 5. The first step in examining the ability to predict influential links within a network using transfer entropy values was to formulate the influence link inference problem (Section 4.4.1), which, within the anti-majority game model being considered here, is equivalent to predicting the communication network. As mentioned above, the use of a theoretical model with known properties is advantageous in this case, as it allows the ground truth to be used for analysis of link prediction accuracy. This is an improvement over the existing transfer entropy influence work, which only included manual identification of a few of the most likely links within the network. This thesis proposes (Section 4.4.2) and evaluates the accuracy of (Section 4.4.3) several methods for predicting influence links, requiring varying degrees of network/system knowledge. While the ultimate goal of this type of research would be the creation of a highly accurate prediction method that requires absolutely no prior system knowledge, practical applications in real-world

systems will present varying amounts of available system knowledge. For this reason, comparing multiple predictive mechanisms that require different degrees of knowledge, as is done here, is a valuable contribution because it provides a more thorough understanding of the challenges in creating accurate predictions within different system types.

A significant portion of the existing network control research, has focused on moving a system between arbitrary state vectors (Liu et al., 2011b; Ruths and Ruths, 2014). The NCP work of Runka (2016) expanded upon this vector-to-vector formulation by defining a control utility function based on an aggregate measure of the system state. This thesis moves beyond a vector-space model, proposing a distribution-based control solution (see Section 5.2), in which the target for control is no longer a vector, but is represented by the distribution of state values. It can be argued that this approach is more applicable to social systems, where the overall system state properties (i.e., the distribution of state) are more important than reaching any specific state vector. Within this thesis, a distribution-based failure avoidance control problem is proposed. The main objective of this control problem is to hold the overall state distribution within a certain threshold distance from a specified ideal normal distribution. This type of problem could be applicable to social systems in which the goal is to avoid consensus and/or extremism. The distribution-based control work presented within this thesis represents one possible solution to a single distribution-based control problem. While many other learning approaches, distribution targets, control problems and other variations exist, these are not investigated within this thesis. Instead, this thesis focuses on other subproblems within the network control domain (e.g., control node selection and controllability analysis), while also demonstrating the feasibility of controlling the state distribution within a networked social system.

An important question to answer when considering a network control problem is: given the parameters and constraints, how effectively can the system be controlled? Structural control theory research has contributed several methods for answering this question, from a purely structural perspective, but this structural approach does not account for agent behaviour and relationship dynamics within a system. Early NCP work has proposed a type of control problem that includes these behavioural factors (Runka and White, 2015a; Runka, 2016), but that work was mainly focused on the formulation of the control problem itself and initial investigations into the performance of different control node selection algorithms. This thesis aims to identify and investigate specific network properties that negatively impact the likelihood of control success in a significant way. In doing so, this thesis moves toward a method for estimating control success probability within NCP-type problems

through system analysis. Another important factor when estimating the likelihood of control success is a measurement of the effect that a controller is capable of exerting on the system. Measuring the effect of a controller, however, is considered beyond the scope of this thesis and is not considered here.

Controller success is investigated solely within the proposed distribution-based control problem, which is applied within the domain of the real-valued voter model used within the original NCP work (see Section 2.5 for details). In order to analyze the role of network properties and influence dynamics in determining control success, learned controllers are applied within simulations of this problem across a wide variety of theoretical and real-world networks. It is important to note that the learning of control signals is not viewed as a major contribution of this thesis, so an in-depth analysis of the efficiency and effectiveness of the reinforcement learning approach is not presented. Using the results of these simulations, however, specific network/influence properties that are most likely to have an adverse effect on network control success can be identified. These results are used to motivate the development of improved control node selection algorithms, several of which are proposed and compared within this thesis.

1.4 Summary of Contributions

The contributions of this work include:

- Proposal of global (system level) and local (node level) influence measurements using transfer entropy and a comparison of these new measurements to theoretical influence measures (Sections 4.3.2 and 4.3.3)
- Development and analysis of methods for identifying influence links within networks based on transfer entropy values (Sections 4.4.2 and 4.4.3)
- Formalization of the anti-majority game (Section 4.2), a communication-based game that can be used to investigate influence and control within social systems
- Definition of distribution-based control, a novel view of network control problems that eschews the traditional state vector network control approach (Section 5.2)
- Investigation of the use of distribution-based control for the real-valued voter model (Sections 5.3 and 5.4)

- Identification of possible causal relationships between network properties, control node influence and control success (Sections 5.4.2-5.4.4)
- Development of influence-weighted control node selection algorithms which improve on the previous best performing solution investigated in the original NCP research (Section 5.5)

1.5 List of Publications

The publications that have originated from this dissertation to date include:

- Using Transfer Entropy for Influence Measurement and Network Prediction, *SocialCom, ASE Eighth International Conference on Social Computing*, pp. 1-14, 2015.
- Observations on the role of influence in the difficulty of social network control, *ASONAM, IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 1294-1301, 2016.
- Selecting Transfer Entropy Thresholds for Influence Network Prediction, *Social Network Analysis and Mining*, 7(1), 2017.
- Towards Distribution-Based Control of Social Networks, *Computational Social Networks* (in submission).

1.6 Thesis Organization

The remainder of this thesis is organized as follows. Chapter 2 presents introductory background material to facilitate understanding of the contributions of this thesis. This is followed by Chapter 3, which discusses existing research within the relevant areas and identifies the gaps within this research that this thesis intends to fill. Chapter 4 introduces transfer entropy influence measurements and provides experimental analysis regarding the accuracy of influence and network prediction using the proposed measures. Following this, Chapter 5 begins with the definition of distribution-based control, before moving towards experimental results regarding network control success within a distribution-based control problem. Chapter 5 also provides analysis to determine which system properties are strongly correlated with reduced network control success and proposes/compares new control node

selection algorithms. Finally, the thesis concludes with a summary of the most significant results and a discussion of future research directions related to the thesis topics presented in Chapter 6.

Chapter 2

Background

2.1 Introduction

This chapter introduces several background topics used within this thesis. These background topics include social network analysis measures, randomly generated and sampled networks, transfer entropy, the real-valued voter model, and reinforcement learning. Those familiar with any of these topics can skip the corresponding sections, as the information presented here does not cover any of the main contributions of this thesis. Those who are unfamiliar with any of these topics, though, may use this chapter to gain an initial understanding of these topics that will aid in comprehending the work presented in the remainder of this thesis. Section 2.2 introduces several social network analysis measures that are used within this thesis. Section 2.3 describes the process used to generate the theoretical (Section 2.3.1) and sampled real-world (Section 2.3.2) network instantiations that are used within this work. Section 2.3 also summarizes some of the network properties of each of these network types and outlines the naming scheme used to identify a few of the specific network classes/parameters throughout this work. Section 2.4 describes the mathematical formulation of the transfer entropy measurement and discusses some of the important computation details that must be considered when calculating transfer entropy values. The real-valued voter model, which is used extensively within Chapter 5, is discussed within Section 2.5. Finally, Section 2.6 briefly introduces the topic of reinforcement learning and discusses the approximation method used to handle the continuous state space that is required within the network control work presented later in this thesis.

2.2 Social Network Analysis Measures

Throughout this work, a number of commonly used social network analysis measurements are used to compare the various network types and instantiations. This section will briefly explain the meaning of each of these measures and provide details of their calculation. Additional information regarding these measures and other network analysis can be found in the work of Jackson (2010).

2.2.1 Degree

Typically, social networks are viewed as graphs consisting of a set of vertices[1], V, which are connected by a set of edges[2], E. The degree of a vertex/node within one of these networks represents the number of edges that are incident to that vertex. In the case of directed graphs, in which edges have a defined start and end vertex, this can be further subdivided into the in-degree and out-degree of the vertex, representing the number of incoming and outgoing edges respectively. Additionally, the average degree is an aggregate measure calculated over all vertices within the network, as in Equation 2.1, where $deg(v)$ represents the degree of the vertex v.

$$\text{Average Degree} = \frac{\sum_{v \in V} deg(v)}{|V|} \tag{2.1}$$

2.2.2 Diameter

The diameter of a network measures the longest shortest path between any pair of vertices within the network. This can be calculated by finding the shortest path between each possible pair of vertices within the network and taking the largest of these values. The diameter of a network that is not strongly connected (i.e., there exists a pair of vertices that cannot be connected via a simple path) is considered infinite. This thesis only considers graphs that are strongly connected.

2.2.3 Betweenness Centrality

Centrality measures produce a ranking of the importance of vertices within a network. The meaning of 'importance' varies depending on the specific centrality measure being considered. A common centrality measure that is used throughout this work is betweenness centrality, which is generally viewed as a measure of the importance of a node within the network in propagating information between nodes along the shortest path. More precisely, the betweenness centrality of a vertex, v, measures the percentage of all shortest paths between pairs of nodes within the network that pass through v. The betweenness centrality measure of a vertex v can be calculated using Equation 2.2.

$$BC(v) = \sum_{s \neq v \neq t \in V} \frac{\#\text{ shortest paths from s to t}}{\#\text{ shortest paths from s to t that contain v}} \tag{2.2}$$

[1]The term vertex and node are used interchangeably within this thesis.
[2]The term edge and link are used interchangeably within this thesis.

2.2.4 Clustering Coefficient

The clustering coefficient of a vertex/node measures how closely that vertex's neighbours (those directly connected to the vertex) are to being a clique (complete graph). In other words, the clustering coefficient of a vertex represents the percentage of the edges within a complete graph formed among itself and its neighbours that actually exist within the network. A high local clustering coefficient represents a tightly knit group of vertices within the network. Social networks, and small world networks in particular, have been shown to possess a higher overall clustering coefficient that randomly generated graphs with a similar average degree. While the clustering coefficient of a single vertex gives information related to that vertex's neighbourhood, averaging the clustering coefficient across all vertices within a graph gives an aggregate measure of the tightness of communities within the graph overall. Within this work, unless otherwise specified, the use of the term clustering coefficient refers to the average clustering coefficient calculated over all vertices present within a network.

2.2.5 PageRank

The PageRank algorithm (Page et al., 1999) is a method that was originally designed to measure the importance of websites (vertices) within the World Wide Web (network) using link analysis. Within a directed graph, the PageRank algorithm treats the incoming edges of a vertex as votes of support from the source vertices of those edges. The algorithm defines PageRank recursively, so that the PageRank value of a vertex, v, depends on the PageRank values of the supporting vertices that link to v. In each recursive step, the PageRank value of a vertex, v, is spread evenly to v's neighbours through its outgoing links. If a sink node (a node with no outgoing links) exists within the network, it is possible that this node will accumulate all of the PageRank value over time. To address this problem, a damping factor, *damp*, is included within the algorithm to avoid this problem. So, for a set of n pages p_1,p_2,...,p_n, the equation used to update the PageRank (PR) values for a specific page during each recursive step is given by Equation 2.3, where $M(p_x)$ represents the set of pages that link to p_x, and $L(p_x)$ represents the number of outgoing edges from p_x. This equation is generally repeatedly applied to the pages within the graph until the PageRank values do not change significantly, based on a given threshold value.

$$PR(p_i) = \frac{1-damp}{n} + damp \sum_{p_j \in M(p_i)} \frac{PR(p_j)}{L(p_j)} \tag{2.3}$$

Another way to view PageRank is using the random surfer model, which assumes there is a user that starts at a randomly selected vertex and randomly chooses to follow edges between vertices over time. The damping factor, then, represents the probability at any given time that the user will stop randomly following links and restart their behaviour from another randomly chosen vertex. The PageRank values of a page, when the values are normalized across all pages, can be viewed as representing the probability that the random surfer will be on that page at any given point in time. These values can be used to rank the importance of pages, or graph vertices in general, and has also been applied to measure the influence within networks, as is discussed in Section 3.2.

2.3 Random and Sampled Networks

The networks connecting agents within simulations are an important component of the research presented within this thesis. Some of the considered networks are based off of theoretical network types, and are generated using an algorithmic approach, while others are sampled from real-world online social networks. As mentioned previously, the networks used within this thesis can be viewed as a graph $G=(V,E)$, consisting of a set of vertices, V, and a set of connecting edges, E. The edges within the networks considered here are all bidirectional/undirected and a self-loop edge is included for each node. Additionally, all networks considered here consist of a single strongly connected component. In general, ten instantiations of each network type were generated and used throughout experimentation. The only exception to this case is the Facebook networks, which were limited by the available data to only 8 instantiations. This remainder of this section will describe the network generation procedure for the theoretical network types and the network sampling approach used for the real-world network types.

2.3.1 Theoretical Networks

Three theoretical network types are investigated within this work: Barabási-Albert scale free networks (Barabási and Albert, 1999), Erdös-Rényi random networks (Erdös and Rényi, 1959), and Watts-Strogatz small world networks (Watts and Strogatz, 1998). Instantiations of each of these network types can be created using algorithmic generators. More information about each type of network, including the generation method, is included below:

- Scale Free (SF): It has been shown that many real-world networks have a degree distribution

similar to a scale free power-law distribution (Jackson, 2010). Barabási and Albert (1999) proposed a preferential attachment model that can be used to generate these types of networks, which is used here. Within this work, the actual network generation was performed using the network library JGraphT (JGraphT, 2015). The algorithm for generation begins with a network consisting of a single self-connected node. The algorithm then repeatedly adds a new node, x, to the network and connects that node to each previously existing node, y, within the network with a probability proportional to node y's degree. More precisely, the probability of adding an edge between x and y within the network is $\frac{deg(y)}{\sum_{v \in V} deg(v))}$. Since the desire is to form strongly connected networks, this edge addition process is repeated in cases where no edges were formed between the new node and existing nodes within the network.

- Random: Given a desired number of vertices and edges, an Erdös-Rényi random network is defined as a single instantiation containing the correct number of vertices/edges selected randomly from all possible instantiations that fit the constraints. To generate this type of network, an initial graph with the desired number of vertices and no interconnecting edges is created. Edges are then randomly inserted into the graph between two randomly selected nodes until the desired average degree is reached. An alternative generation method for this type of graph involves the specification of an edge probability, p, which specifies the probability of adding each possible edge. A network can then be generated by probabilistically deciding whether each possible edge should be included within the network or not. The naming scheme used to refer to random networks throughout this document is Random-d, where d represents the targeted average degree that was used when generating the instantiation. This work applied the edge-based probability generation method, which resulted in networks with an average degree very close, but not exactly equal, to the target average degree.

- Small World (SW): Small world graphs have also been used to model real-world social networks. A method for generating networks with small world properties, which are characterized by high clustering and short average path lengths, was proposed by Watts and Strogatz (1998). This generation method requires two input parameters: the desired average degree, d, and the parameter β, which must be within the interval of [0,1]. The algorithm for generating the network begins by creating a regular ring lattice network, in which each node is connected to the $\frac{d}{2}$ nodes on either side within the ring lattice. Following this process, each edge present

in this regular ring lattice network is 'reconfigured' with probability β. In this case, reconfiguring an edge involves changing one of its end points to a randomly selected node within the network. This has the effect of moving the graph slightly toward a more random structure, which maintains the high level of clustering while decreasing the average path length. The parameter β determines how far the network is expected to move away from the lattice structure. With a β value of 0, the network remains a regular lattice, while a β value of 1 produces a completely random network. The small world networks used within Chapter 4 all have an average degree of 8, while the small world networks considered in Chapter 5 have an average degree of 4. The β value used for all small world networks is either 0.1 or 0.2. The naming scheme used to refer to specific small world networks within the remainder of this thesis is Small-β, which indicates what specific β value was used in creating the network instantiation.

2.3.2 Sampled Real-World Networks

In addition to the theoretical networks described above, this work also considers networks that are sampled from real-world social network datasets from Google+ (G+), Facebook, and Twitter. All of the network datasets used within this thesis are available online (Leskovec and Krevl, 2014). While Twitter network datasets with 100 nodes were readily available, the Google+ and Facebook datasets contained a variable, and in some cases prohibitively large, number of nodes. In order to generate 100 node networks from the available datasets, a sampling method was used to create a number of specific instantiations for each of these real-world network datasets. While a number of graph sampling methods exist, the specific method used here is the exploration-based method described in Algorithm 1. This random walk method was originally described by Leskovec and Faloutsos (2006).

To sample n nodes from a network using this random walk method, an initial seed node is originally selected from the source graph and added to the result graph. A random walk is then performed on the source graph, beginning from this seed node, with newly visited nodes being added to the result graph. After each random walk step, with probability of 0.15, the random walk begins again from the original seed node. This process is repeated until n nodes have been added to the resulting graph. After this node sampling process has completed, existing edges are added between the sampled nodes if they existed in the original graph.

Ideally, the macroscopic network properties of the sampled network would be very close to those found in the original source network. For Twitter and Facebook networks, this was found to

Algorithm 1 Pseudocode of the network sampling process

Require: $G_O = (V_O, E_O)$, the original network

Require: $size$, the target size

procedure SAMPLENETWORK(G_O, $size$)
    $V_R \leftarrow \{\}$
    $E_R \leftarrow \{\}$
    $seed \leftarrow$ randomly selected vertex from V_O
    $cur \leftarrow seed$
    while $|V_R| < size$ **do**
        $V_R \leftarrow V_R \cup cur$
        $chance \leftarrow$ random integer in the range [1,100]
        if $chance \leq 15$ **then**
            $cur \leftarrow seed$
        else
            $cur \leftarrow$ randomly selected neighbour of cur from G_O
    for $v1 \in V_R$ **do**
        for $v2 \in V_R$ **do**
            if edge between $v1$ and $v2$ exists in E_O **then**
                Add edge between $v1$ and $v2$ to E_R
    $G_R \leftarrow (V_R, E_R)$
    return G_R

be almost true, while samples generated from G+ networks often had skewed network properties relative to the original graph. To address this problem, a second network type, called G+Similar, was also sampled. To generate G+Similar instantiations, the sampling process described above was repeated until the average degree and clustering coefficient within the resulting graph were found to be within the range of 90% and 110% of the original network values.

2.4 Transfer Entropy

Transfer entropy, originally formulated by Schreiber (2000), is an information theoretic measure that quantifies the directed transfer of information between two random processes. In other words, transfer entropy measures the increase in certainty about a following value from a time series X when the past values of X and another time series Y are known. Information theoretic measures, specifically those that are entropy-based, have started to receive more attention in the domain of network analysis, some examples of which include the work of Quax et al. (2013) and Quinn (2014). Ver Steeg and Galstyan (2013) have also used transfer entropy to measure the influence of users within a Twitter dataset, as discussed in more detail in Section 3.2.4. Using transfer entropy to measure the influence of one user on another is advantageous, as it only requires a time series of states and does not rely on network knowledge or any specific underlying model. This section explains the mathematical derivation of transfer entropy using discrete, single dimension time series data, similar to the type of data that is generated by the anti-majority game used as a model within this thesis (see Section 4.2). It should be noted, however, that it is possible to estimate transfer entropy using discrete or continuous data, over single or multi-dimensional time series, with varying values of the time window parameter, o (see Ver Steeg and Galstyan, 2013; Kaiser and Schreiber, 2002, for details). A more detailed and general description of transfer entropy can be found in the original transfer entropy work of Schreiber (2000). Following the description of the transfer entropy calculation, Section 2.4.1 includes details of the parameters used in the transfer entropy calculations completed as part of this work, as well as the bootstrap approach that has been applied to reduce possible small sample bias.

To explain the calculation of the transfer entropy from one time series to another, several intermediary calculations can be used. Given a time series of discrete values produced by a random process, X, sample probabilities of each behaviour $x \in X$ can be computed. The Shannon entropy, $H(X)$, of the process that generated the time series can then be calculated using Equation 2.4. This simple entropy value represents the expected amount of information required to encode a new value of X.

$$H(X) = -\sum_{x \in X} p(x) \log p(x) \tag{2.4}$$

Another measure required for the derivation of the transfer entropy is conditional entropy. Conditional entropy measures the amount of information required to encode a decision from one agent, Y, when the decision of another agent, X, is known. The conditional entropy, $H(Y|X)$, is calculated

using Equation 2.5.

$$H(Y|X) = -\sum_{x \in X, y \in Y} p(x,y) \log \frac{p(x,y)}{p(x)} \tag{2.5}$$

To further explain the conditional entropy, $H(Y|X)$, consider two extreme scenarios. The first case arises when Y is completely determined by X, in which case the conditional entropy of Y given X is 0. The other extreme is when Y and X are completely independent, in which case $H(Y|X) = H(Y)$. The difference between $H(Y)$ and $H(Y|X)$, then, can be viewed as measuring the degree to which the two processes are dependent. This is exactly what the mutual information measurement, calculated using Equation 2.6, is used for.

$$I(Y;X) = H(Y) - H(Y|X) \tag{2.6}$$

The problem with mutual information, at least in terms of identifying influence, is that it is symmetric ($I(Y;X) = I(X;Y)$). The mutual information, then, only indicates a dependence between the two variables, but is not capable of inferring any causation.

Schreiber (2000) notes, however, that if it is assumed that the system can be approximated by a stationary Markov process of order o, where the order represents the number of past states that can affect the following state (in this work, only $o = 1$ is considered), then the transition probabilities of that Markov process can be used to compute a directional measure of dependence, called transfer entropy. When considering the transition probability at a time step t, it would be expected that Equation 2.7 would hold true for two independent processes, X and Y.

$$p(x_{t+1}|x_t) = p(x_{t+1}|x_t, y_t) \tag{2.7}$$

The transfer entropy, as shown in Equation 2.8, uses the Kullback-Leibler divergence to quantify the level to which this assumption of independence is broken.

$$TE_{Y \to X} = \sum p(x_{t+1}, x_t, y_t) \log \frac{p(x_{t+1}|x_t, y_t)}{p(x_{t+1}|x_t)} \tag{2.8}$$

In other words, this equation is calculating the reduction in uncertainty in the next value of X when previous values of both X and Y are known. Applying this to time series data from an information-based social system, the transfer entropy from the time series of an agent Y's actions to the time series of an agent X's is the reduction in uncertainty of the behaviour of X when we know the past behaviours of X and Y. The logical argument, then, is that if Y influences X significantly, the transfer entropy from Y to X will be significantly higher than if Y does not influence X.

2.4.1 Transfer Entropy Computation Details

The anti-majority game model (Section 4.2) that transfer entropy calculations are applied to within this thesis has known properties, which provides a number of advantages when selecting an algorithm and parameters for the calculation of transfer entropy values. First, since the game model uses a discrete action set, the basic transfer entropy calculation defined by Schreiber (2000) can be used. This eliminates the need to use more complex algorithms and estimators necessary for calculation using continuous data, which can affect the overall accuracy. In addition to this, since it is known that agents within the system only have a single step memory, it is also known that the system is a Markov process of order 1. This allows the parameter $o=1$ to be used in the transfer entropy calculation from Equation 2.8 confidently.

An additional consideration when calculating the transfer entropy between two finite time series is the effect that the finite sample size may have on the entropy value. As entropy values are theoretically defined for time series of infinite size, measurements calculated from finite samples are inherently biased estimates. This effect has been demonstrated in previous works, such as Marschinski and Kantz (2002). To address this problem, a bootstrapping approach is recommended to produce a less biased estimate. This work uses the bootstrap procedure proposed by Dimpfl and Peter (2013) and outlined in Algorithm 2, which is designed to preserve the important temporal dynamics of the time series under consideration, is applied. Using this method, the biased transfer entropy, $TE_{X\to Y}$ between two time series is computed as usual. In addition to this, the transfer entropy value between a shuffled time series, X^1, and the original destination Y is also computed. The shuffled series, X^1, is generated randomly using the same transition probabilities of the original time series X, which preserves the time dynamics of the source time series while removing the temporal relationship between the source and destination time series. The value of $TE_{X^1\to Y}$ represents the amount of the original transfer entropy caused by small sample bias and an average of these values calculated across a number of randomly generated bootstrap time series is subtracted from the original biased transfer entropy to produce the 'effective transfer entropy' from X to Y[3]. The effect that this bootstrap approach has on the transfer entropy value is demonstrated in Figure 2.1. This figure shows the average percentage of the original biased transfer entropy remaining as the number of bootstraps increases, calculated over 10000 randomly generated time series pairs. These time series were created with a length of 1500 values and a domain of 5 possible values to resemble the

[3]Within this thesis, $T_{X\to Y}$ is used to refer to this effective value

Algorithm 2 Calculation of unbiased transfer entropy

Require: X, the source time series

Require: Y, the destination time series

Require: $VALS$, set of possible values in time series

Require: $BOOT$, number of bootstrap time series

procedure EFFECTIVETRANSFERENTROPY(X, Y, $VALS$, $BOOT$)

▷ Calculate biased transfer entropy

    $TE_{X\to Y}$ = compute TE from X to Y using Equation 2.8

▷ Compute transition probabilities from X

    for $i \in VALS$ **do**

        $SUM_i = 0$

        for $j \in VALS$ **do**

            $COUNT_{i,j} = 0$

    for i from 1 to $|X|$ **do**

        $COUNT_{X_{i-1},X_i} = COUNT_{X_{i-1},X_i} + 1$

        $SUM_{X_{i-1}} = SUM_{X_{i-1}} + 1$

    for $i,j \in VALS$ **do**

        if $SUM_i > 0$ **then**

            $TransProb_{i,j} = \frac{COUNT_{i,j}}{SUM_i}$

        else

            $TransProb_{i,j} = 0$

▷ Compute average TE for shuffled variants of X, maintaining transition probabilities

    $BIAS_{X\to Y} = 0$

    for i from 1 to $BOOT$ **do**

        $X^1 \leftarrow$ empty time series

        Add random selection from $VALS$ to X^1

        while $|X^1| < |X|$ **do**

            $LAST$ = last value in X^1

            $NEXT$ = probabilistically choose value using $TransProb_{LAST,v\in VALS}$

            Add $NEXT$ to X^1

        $BIAS_{X\to Y} = BIAS_{X\to Y} + TE_{X^1\to Y}$

    $BIAS_{X\to Y} = \frac{BIAS_{X\to Y}}{BOOT}$

    return $MAX(0, TE_{X\to Y} - BIAS_{X\to Y})$

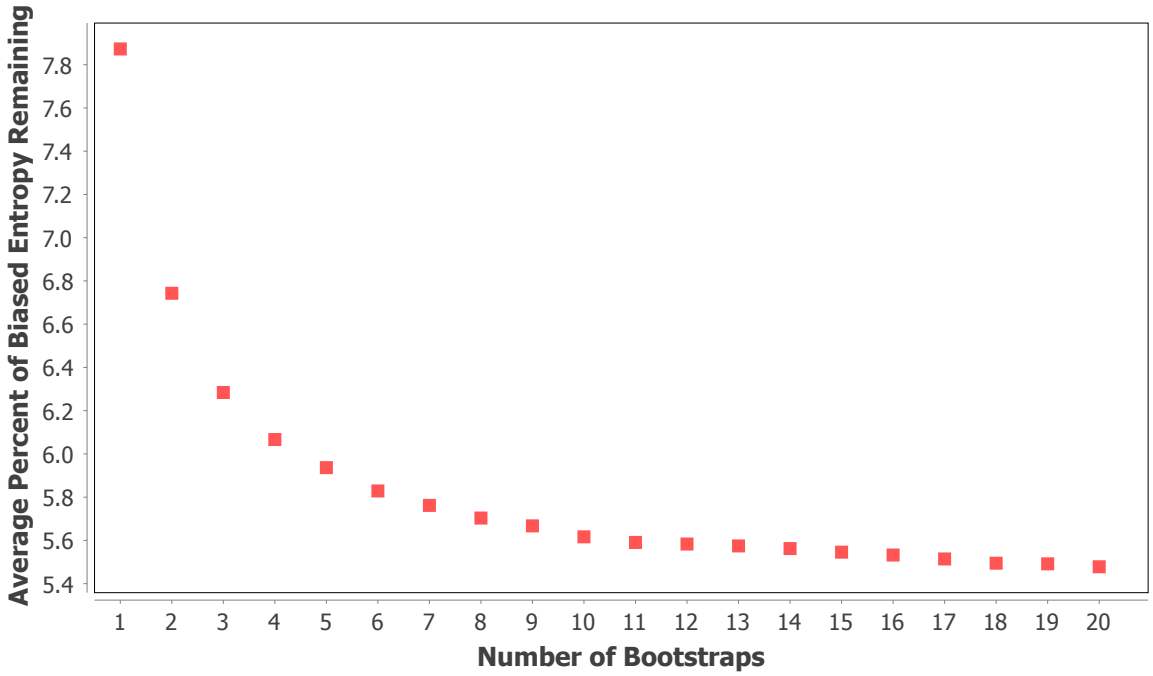


Figure 2.1: Average percent of biased transfer entropy remaining as the number of bootstrap samples used increases

time series that are used later in this work that are most likely to be affected by the small sample bias.

Figure 2.1 shows that even when using only a single bootstrap sample, it can be expected that over 90% of the biased entropy value will be removed. As the number of bootstrap samples increases, the overall improvement of the effective transfer entropy decreases rapidly, to the point that there is less than 0.5% improvement between 10 and 20 samples. For this reason, the results presented in this work use effective transfer entropy values computed using the average of ten additional random samples and all effective transfer entropy values are given a lower bound of 0.

2.5 Real-valued Voter Model

Voter models have been used extensively in existing social network research to model the change of agents' state/opinion over time (Even-Dar and Shapira, 2007; Runka, 2016; Runka and White, 2015a). A discrete binary-valued voter model was used within the previous network control work of Runka (2016). Within this binary voter model, each agent probabilistically changes its state

Algorithm 3 Pseudocode of the state update process within the binary voter model

Require: G is the communication network

Require: *state* is a list of the current state values of each agent

```
procedure UpdateStateDiscrete(G, state)
    for a in A do
        neighbour ← randomly selected neighbour of a
        state_a ← state_neighbour
```

(0 or 1) at each time step. The new state of the agent is chosen by randomly selecting the state of one of that agent's neighbours, as outlined in Algorithm 3. This model is guaranteed to converge to a single value in $O(|V|^5)$ time and is expected to converge with high probability within $O(|V|^3 log(|V|))$ times, where V represents the vertex/agent set of the networked system under consideration (Holley and Liggett, 1975).

Runka (2016) also considered the possibility of applying network control within the domain of a real-valued voter model, which is the type of model used within this thesis. Within this real-valued voter model, the state values are bounded within the range $[-1.0,1.0]$ and, during each time step, each agent moves its state value toward a randomly selected neighbour's state value by a specific step parameter value. This state update process is formalized in Algorithm 4. The natural momentum of this model, like the discrete case, is to have all values converge toward a single value.

2.6 Reinforcement Learning

Reinforcement learning is a method of machine learning, based on the principles of behavioural psychology, that attempts to learn an optimal action policy capable of maximizing the received reward of a rational agent. Reinforcement learning approaches have been previously applied within many control problems, the most well-known of which is probably the pole-balancing problem (Sutton, 1984). Generally, a reinforcement learning problem occurs within a stochastic environment that can be specified as a tuple (S,ACT,T,R) consisting of:

- S: The set of states the environment can be in. In the most basic sense, this is often a discrete set of states; however, it is possible to discretize continuous state environments, as will be explained later in this section.

Algorithm 4 Pseudocode of the state update process within the real-valued voter model

Require: G is the communication network

Require: *state* is a list of the current state values of each agent

Require: *step* is a parameter specifying the rate of change of agent state values

procedure UPDATESTATEREAL(G, *state*, *step*)
  for a in A **do**
    neighbour ← randomly selected neighbour
    $sign \leftarrow 0.0$
    if $state_{neighbour} > state_a$ **then**
      $sign \leftarrow 1$
    else if $state_{neighbour} < state_a$ **then**
      $sign \leftarrow -1$
    $state_a \leftarrow state_a + (step \times sign)$
    $state_a \leftarrow max(-1.0, state_a)$
    $state_a \leftarrow min(1.0, state_a)$

- ACT: The set of actions that the agent can choose from.
- T: A probabilistic state transition function, $P(s_{t+1}|s_t,a_t)$, which models the probability of moving to a future state when selecting a specific action from the current state.
- R: The expected reward function, $E(r_t|s_t,a_t)$, which estimates how much reward the agent receives when it takes a specific action from a specific state.

The goal of a reinforcement learning agent, then, is to learn an optimal action selection policy, $Policy(s_t,a_t)$, that maximizes the reward the agent receives. This is accomplished through repeated interaction with the environment and the use of reinforcement learning algorithms to update the various transition, reward, and action selection functions.

The specific reinforcement learning algorithm used within this thesis is an on-policy algorithm called SARSA, or State-Action-Reward-State-Action. On policy, in the case of reinforcement learning, means that learning is carried out using the agent's action policy, even during exploration. The SARSA algorithm attempts to learn the value of certain state/action pairs within the environment through repeated experiments. As the name implies, the learning process used to update the

Algorithm 5 Pseudocode of a learning episode using SARSA reinforcement learning

Require: S the set of states

Require: ACT the set of possible agent actions

Require: $Q(s,a)$ the current learned values for each state/action pair

Require: α the learning rate

Require: γ the discount factor, used to determine the importance of future rewards relative to immediate rewards

```
1: procedure LearningEpisode
2:     s_t ← a random starting state
3:     a_t ← an action selected using the current policy and Q values (e.g., greedy or ε-greedy)
4:     while the episode is not over do
5:         Execute action a
6:         s_{t+1} ← resulting state
7:         r_t ← reward from environment
8:         a_{t+1} ← an action selected using the current policy and Q values (e.g., greedy or ε-greedy)
9:         Q(s_t,a_t) ← Q(s_t,a_t) + α[r_t + γQ(s_{t+1},a_{t+1}) − Q(s_t,a_t)]
10:        s_t ← s_{t+1}
11:        a_t ← a_{t+1}
```

estimated state values relies on information regarding an initial state/action pair, the reward gained through taking this action from the initial state, the following state the agent will be in, and the following action the agent takes from this second state. More precisely, the estimated state/action value function $Q(s_t,a_t)$, is updated after each action using the equation $Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha[r_t + \gamma Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)]$. This update process is repeated over time as the agent interacts with the environment, as outlined in Algorithm 5. It should be noted that Algorithm 5 demonstrates the update procedure for a single learning episode. Within this thesis, as is typical with reinforcement learning in general, a number of learning episodes are executed in order to produce a better overall result.

Within many real-world problems, including the network control problem investigated within this thesis, the state space is continuous in nature. In continuous cases, or even in cases in which the state and/or action spaces are very large, applying these straightforward reinforcement learning algorithms is not computationally feasible. Instead, generalization methods must be used to approximate the value of specific states based on the value of similar states. There are a number

of methods available for generalization within reinforcement learning problems (see Sutton and Barto, 1998; Siginam, 2012, for thorough discussions), but this work uses tile-based coding, which is commonly used in reinforcement learning applications due to the balance between computational cost and representational power (Sherstov and Stone, 2005). In the most basic sense, a *tiling* is used to divide a large or continuous state space into several non-overlapping *tiles*. A continuous state value, then, corresponds to a single tile within this tiling, and that tile can be used to approximate the value of all other states that also map to the same tile. Using a single tiling, the reinforcement learning update process will update the value of all other states that map to that specific tile. Within this thesis, a cerebellar model arithmetic computer (CMAC) tiling approach (Albus, 1975) is applied, which uses a number of different tilings covering the same state domain. Each of the tilings is skewed slightly relative to the others, which allows the update process to update the values of states depending on how similar they are to the current state (i.e., how many tilings they match on). When the tilings are generated effectively, this allows for more accurate state value updates and improves the overall learning process. This CMAC tiling, combined with an online learning algorithm such as SARSA, has been shown to produce successful learning results when applied to several control problems with large/continuous state spaces (Sutton, 1996).

2.7 Summary

This chapter introduced background information relating to several topics that will help understand the remaining work presented within this thesis. The following chapter will discuss existing research that is related to the work presented within this thesis.

Chapter 3

Related Work

3.1 Introduction

The research presented within this thesis is centered around problems involving influence measurement, network prediction/inference, and network control. This chapter will discuss the existing literature on these topics, focusing on the types of problems within these domains that are most relevant to the work presented in the remainder of this thesis. The influence measurement research presented within this thesis deals with the use of historical action traces within a communication-based game to estimate participant influence. Section 3.2 will discuss a number of influence measurement approaches that have been previously investigated, ranging from early work involving structural network analysis to more recent work involving action or behaviour-based influence measurements. While, the majority of existing network prediction/inference work that has been previously undertaken focuses on the prediction of the time evolution of network structure, the work within this thesis is concerned with inferring the network connections of an unchanging network after observing the network for a period of time. Section 3.3 discusses both the problem of predicting the time evolution of networks and inferring entire networks given historical traces, but more detail is included for the latter problem. Discussion regarding the existing research within the network control domain is presented in Section 3.4. Section 3.4.1 briefly discusses some of the existing network control goals, while different mechanisms for achieving control are outlined in Section 3.4.2. It should be noted that, as the field is so broad, these sections do not represent an exhaustive list of all possible network control goals/mechanisms. Instead, the focus is on the approaches that are most common in the existing literature and those that are most strongly related to the work presented within this thesis. Section 3.4.3 discusses a specific type of network control, which is concerned with full state controllability. While the study of this type of control has received a significant amount of recent attention, there are several criticisms/limitations of this view on network control, which are presented in Section 3.4.4. The Network Control Problem (NCP), which was partially motivated by the criticisms presented in Section 3.4.4, is discussed

in Section 3.4.5. This includes discussion of the existing NCP work involving the FAR heuristic for control node selection, as well as the initial experiments involving the control of a discrete voter model. Finally, this chapter concludes with Section 3.5, which identifies the key points from the analysis of these works that have motivated the research presented within this thesis.

3.2 Influence Measurement in Networks

Determining the influence properties of nodes within networks is beneficial for many practical applications, including: financial applications (Pan et al., 2012), predicting information propagation (Barbieri et al., 2012, 2013), maximizing advertising efficiency (Kiss and Bichler, 2008), and, as will be discussed in detail in Chapter 5, network control. For this reason, a significant amount of research has been dedicated to developing different methods and algorithms for measuring influence within networks. Within this section, these existing methods will be analyzed and compared. While the definition of influence can vary widely across different domains, within this work, influence is viewed as the ability of one agent to affect the future behaviour of another agent within the same system. This definition is consistent with the general definition of social influence proposed by Raven (1964), as well as the definition used in other influence measurement works, such as that of Hajian (2011). While many of the research works discussed here share a similar view of influence, some (e.g., Page et al., 1999) may not have been originally designed to measure the same type of influence. To further categorize the existing work, Section 3.2.1 outlines several ways in which influence measures can be classified. While many solutions can be modified to fit different classifications, the analysis presented here is carried out using the specification/intent of the original algorithms.

3.2.1 Influence Measure Classification

In order to better categorize and compare the existing literature, this section defines two methods of classifying influence measure: type and method. The type of an influence measure specifies the influence relationship that it is measuring, and can be divided into:

Global The algorithm measures a node's influence on the entire network or produces an aggregate value.

Local The algorithm measures influence between specific pairs of nodes in a directional or symmetric way. One argument in favour of measuring this type of influence is that it provides

information about the relationship between two nodes, but can also be easily converted to a global measure through aggregation. For example, a sum of an agent's local influence over all other nodes, could be considered an aggregate value that represents that agent's global influence. Alternatively, a global structural measure of influence could be applied to a weighted network created using the local influence measures. In contrast, calculating specific influence measures from global measures may not be possible.

Topical The algorithm measures influence across a variety of topics. This category can be further divided into topical measures that are global or local.

The method of influence measurement describes how the measurement is calculated, and can be divided into:

Structural The influence measurement is calculated using only structural properties of the specified network connecting the nodes.

Behavioural The influence measurement uses recorded behaviour of nodes, generally over a time period or repeated simulations, to determine a measure of influence.

Mixed The influence measure is calculated using some degree of both structural and behavioural analysis.

The analysis presented within the following subsections will typically argue in favour of measures which are based on agent behaviour and capable of producing both global and local measurements, while arguing against the use of structure-based measurements, at least in the classical sense of analysing networks defined by homogeneous network links.

3.2.2 Structural Measures

Some of the earliest methods for determining influence within networks relied solely on analysis of the structural properties of the networks. One of the most commonly used classes of structural measure are the centrality measures, many of which are described and compared in detail by Landherr et al. (2010). Of these centrality measures, the easiest to calculate is the degree centrality of a node, which is simply equal to the number of links the node possesses, as shown in Equation 3.1.

$$DC(v) = deg(v) \tag{3.1}$$

A slightly more complex and more frequently used centrality measure is the betweenness centrality, calculated using Equation 3.2.

$$BC(v) = \sum_{s \neq v \neq t \in V} \frac{\#\text{ shortest paths from s to t}}{\#\text{ shortest paths from s to t that contain v}} \tag{3.2}$$

Assuming that information only travels through the shortest paths within a network, the betweenness centrality measures the ability of a node to affect the information flow between pairs of nodes within the network. By considering an epidemiological model, Piraveenan et al. (2013) define percolation centrality as a measure similar to betweenness centrality, where the source and target nodes s and t are limited to nodes that are 'infected'. The percolation centrality, then, is both state and time dependent. The idea of using epidemiological models to estimate influence has been used in other works as well, which are discussed in more detail in Section 3.2.3. An additional measure of note is the eigenvector centrality, which accounts for the centrality measures of neighbouring nodes using the assumption that a node connected to important nodes is generally more important itself. An estimate of a node's eigenvector centrality is produced by the well-known PageRank algorithm of Page et al. (1999), which is described in Section 2.2.5.

While these centrality measures have received significant attention in domains involving epidemiological analysis and general network analysis, there are a number of criticisms regarding their use for influence measurement. One criticism of centrality measures is that they produce an overall ranking of the importance of a node within the network, instead of a numerical quantification of influence, and these rankings are capable of significantly underestimating the effect of lower ranked nodes on information flows within a network (Bauer and Lizier, 2012; Šikić et al., 2013). Additionally, centrality measures, and structural measures in general, do not consider behaviour and, therefore, are not actually measuring influence as it is defined in Section 3.2. The work of Borgatti (2005) and Borgatti and Everett (2006) demonstrate that what various centrality measures really represent is the role of a node in random walks within the network under certain properties. An example of this concept in practice is the random surfer model used within the PageRank algorithm (Page et al., 1999). It is easy to argue, however, that the information dynamics in real networks are not random in nature, but are determined by the relationships between, and behaviour of, nodes within the system. So while the structural properties of a node within a network can estimate the node's importance within the network, it is the actual dynamics of behaviour and information flow within the system that determine the a node's level of influence. This statement has been verified in existing literature

where structural and behavioural measures within real-world datasets are compared. Cha et al. (2010) compared the number of followers (degree centrality, a structural measure) to the number of retweets (a behavioural measure) in a Twitter dataset and found that there is not necessarily a strong correlation. In addition to this, Kwak et al. (2010) observed similar rankings of users on Twitter when using degree centrality and PageRank, but significantly different ranking when considering retweets. Finally, these measures produce an estimate of a node's *global* importance within the network and do not allow the *specific* importance of a node in relation to another to be measured. This specific relationship information, however, could be useful in applications such as network control.

3.2.3 Mixed Measures

To improve upon purely structural approaches to influence quantification, some researchers have combined structural solutions with information propagation models (e.g., Kempe et al., 2005). A large amount of this research takes an epidemiological perspective, treating 'infected' individuals as those who have been affected by a piece of information and measuring influence based on the ability of a node to infect others within the system. While these works use many similar ideas when compared to the previously discussed structural approaches, such as random walks and centrality measures, the inclusion of propagation models and temporal dynamics produce solutions that at least consider the effect of behaviour within the network.

One of the earliest works to consider this type of model was that of Kempe et al. (2005), which attempted to select a set of nodes that maximized the spread of a 'contagion' representing some idea or innovation. In this case, the influence of a node is represented by its ability to spread the contagion to other nodes, which is calculated through the simulation of a cascade model. A similar approach is applied by Bauer and Lizier (2012), using random walks on a graph from infected individuals. These random walks differ from the purely structural random walks related to centrality measures because of the use of a susceptible-infected-recovered model, which captures possible temporal dynamics within the system.

One measure that has emerged from this type of work is called *accessibility* and was originally defined by Travençolo and Costa (2008). To determine a node's accessibility for a time step interval h, an entropy calculation is performed with the probabilities of reaching other nodes in the network using a self-avoiding random walk on the network of length h. This calculation is shown in Equation 3.3, where $PW_h(v,t)$ is the probability of reaching t using a self-avoiding random walk

of h steps starting from v and treating $log\ 0=0$.

$$A_h(v)=-\sum_{t\neq v\in V} PW_h(v,t)\ log\ PW_h(v,t) \tag{3.3}$$

The main argument in favour of this measure made by Travençolo and Costa is that information leaving nodes with low accessibility must follow a limited number of pathways, and therefore cannot reach as much of the network quickly when compared to a node with high accessibility. This point has been investigated further by Viana et al. (2012), who found that the nodes possessing the highest accessibility values required the smallest number of walks to reach all nodes within the network. Further analysis by da Silva et al. (2012) found a strong correlation between a node's accessibility value and the prevalence of epidemics initiated at that node in several types of theoretical networks. This same work found that the correlation between epidemic prevalence and several purely structural measures, including betweenness centrality, depended on the class of network used. The *expected force* measure, proposed by Lawyer (2015), is calculated in a similar way to accessibility but uses the rate at which nodes are becoming infected after h steps within the entropy calculation instead.

As these measures incorporate the notion of propagation dynamics, it can be argued that they are an improvement over strictly structure-based influence measures. These solutions, however, still rely on random walks and/or the idea of identical network links. The main criticism then is that these solutions are still only estimating the expected influence measure under these assumptions and not truly incorporating the behaviour of individuals within the system.

As Twitter emerged as a popular social network, it enabled researchers to investigate the use of behaviour in influence analysis to a greater extent. One way in which this work progressed was in the development of topical models of influence and information propagation. One of the earliest works to consider this type of problem is that of Tang et al. (2009), which experimented with a topical propagation model on several real-world networks to determine experts on particular topics. This idea was extended and applied to real-world datasets by Weng et al. (2010) and Sung et al. (2013) through the application of topic modelling to Twitter content. These works found that using a model that combines structural information, in the form of the Twitter follower graph, with similarity information determined through topical analysis, produced more accurate recommendations and predictions of the level of interaction between users than a purely structural approach. While these topical works still rely heavily on structural analysis (i.e., PageRank calculations), the increased accuracy could be attributed to the fact that they move away from the view that all links within a network

represent an identical relationship. Instead, they acknowledge and account for the ideas that:

1. Some relationships are stronger than others.

2. Even between two individuals, the relationship may be stronger in one direction than the other.

3. There can also be additional context (topics) present that further determine the relationship.

The work of Cataldi et al. (2013) and Cataldi and Aufaure (2014) extended the existing topical influence work by incorporating behaviour in the form of Twitter 'retweets' (equivalent to the broadcast of a message received from another user). In this case, instead of only considering the follower graph and topical similarity between users, a new graph is constructed for each defined topic where directed edges from a node s to another node t exists if and only if s has retweeted something from t relating to the topic. To determine the most influential users for each topic, then, a PageRank calculation is performed on the constructed retweet graph. While the presented results appear to identify intuitively influential users within the investigated domains (e.g., Bill Simmons, a sports writer, at the top of the sports domain), they also highlight one of the difficulties in investigating influence measurement using real-world datasets – we do not have a ground truth determining who is actually the most influential in these real-world cases. This criticism motivates the use of a theoretical model with known properties in the influence measurement work presented in Chapter 4.

The work of Hajian and White (2011) moves away from a topical view of influence while continuing to investigate the use of behavioural measures in influence ranking. The behavioural component of influence, in this case, is based on the proportion of followers that retweet, comment, or like each post made by a user. A measure of the overall effect a user has on neighbouring users within the network, referred to as the magnitude of influence, is then calculated by aggregating this measure over all posts made by a user. The influence rank value of each user within the system is then determined through iteration of a PageRank algorithm combining the influence rank of a user, that user's followers, and that user's magnitude of influence. This approach is in line with the observations made by Klemm et al. (2012), who noted that the effect a user has in a network is determined not only by structural positioning within the network, but also by the dynamics present. As an example of this reasoning, consider a node that is connected to two nodes with very large degree within a network. Using purely structural influence measures, the two large nodes would generally have a high level of influence, while the node of degree two would have a low level of influence. If this small node is capable of affecting the behaviour of its large neighbours, however,

it would be expected to have a high level of influence as well – it influences the influencers. This idea, which is not considered in many of the previously discussed measurements, is a significant factor in the algorithms proposed by Hajian and White (2011).

While the mixed approaches that have been discussed represent significant improvements over the purely structural measures, there are still areas in which they can be improved. For example, many of the mixed solutions presented earlier in this section only incorporate the idea of behaviour in limited ways, such as the assumption of a basic information diffusion model (e.g., the independent cascade model Kempe et al., 2003, 2005). These approaches do not actually take into account any sort of observed behaviour of network entities over a time period. In addition to this, most of these measures produce global influence values and, in many cases, it would be difficult to determine specific influence levels between pairs of nodes using the same approach. While the work of Cataldi et al. (2013) and Hajian and White (2011) have begun to explicitly include behaviour of users in influence measurement, including some local measures of influence between users, the following subsection will discuss methods for influence measurement that rely solely on behavioural analysis.

3.2.4 Behavioural Measures

As has been mentioned previously, many of the influence measurement approaches discussed so far make strong assumptions regarding the flow and effect of information within a network. However, many authors have noted that, while the links of many networks are easily identifiable in a binary sense (existent or not), the underlying properties of these connections are generally unknown (Gomez Rodriguez et al., 2010). Šikić et al. (2013) has also presented evidence that it is these unknown link properties that truly determine the behaviour and information dynamics within a system, noting that a node that is highly influential under one set of 'spreading parameters' may be significantly less influential under a different parameter set. The works discussed in the remainder of this section attempt to address this problem by computing influence measurements based on the temporal patterns of actions made by members of a network, without making assumptions regarding how information flows within the network.

Many of the existing behaviour-based influence measurement solutions rely on the analysis of cascade behaviour over time. In these cases, a dataset is constructed from observed user/action/time tuples and analysis on the temporal relationships between users and actions is used to predict influence relationships. Using a similar objective to the work of Kempe et al. (2005), Goyal et al.

(2010) attempt to estimate the probability of a node performing an action (becoming infected) from a set of initial nodes performing the same action by calculating the probability that the target nodes follows a source node in the user/action/time tuple log. These measurements can be transformed into a global influence measure for a source node by calculating the expected number of infected nodes generated by a cascade initiated at the source node. Subbian et al. (2013) uses a frequent itemset mining approach for identifying influencers using the same type of action log. In this case, itemsets consist of a set of users ordered by the time in which they performed a specific action (earliest acting users first). Sets that occur frequently across the set of actions, as determined by a threshold value, are used to calculate the influence of a node by considering how many others frequently appear after the node. As with the algorithm of Goyal et al. (2010), this produces a global measure of a node's influence within the network and does not provide specific information regarding what nodes immediately follow the actions of a source node.

The work of Tsai et al. (2014) combines the idea of weighting with the frequent item mining approach of Subbian et al. (2013) to predict influence leaders (those that are likely to initiate action cascades). This approach produced accurate leader predictions when compared to some other simple leader predicting heuristics (e.g., those with at least 10 actions that have had at least 10 following actions). However, as with many other approaches discussed previously, this solution only identifies a few globally influential nodes and does not provide a method for identifying the influence relationships between pairs of nodes within the network.

Ver Steeg and Galstyan (2013) proposed the use of transfer entropy for measuring the level of influence between users on Twitter. By classifying the content of users' tweets using topic models, Ver Steeg and Galstyan use transfer entropy calculations to measure the expected information transfer (i.e., influence) from a source user's tweets to a target user's tweets. Upon calculating all pairs of transfer entropy values, a threshold value is selected manually to determine which links to include in the predicted network (those with transfer entropy values higher than the threshold). As there is no known ground truth underlying the interactions on the Twitter dataset, the authors identify possible links between predicted network neighbours using available information, such as profile descriptions. Interestingly, several of the identified influence links are not present in the Twitter follower graph, providing further evidence that simple network analysis can produce inaccurate results. The transfer entropy approach proposed by Ver Steeg and Galstyan offers several benefits when considering the problem of influence analysis. First, it is completely behaviour

driven, requiring no knowledge or estimates of the underlying network to compute the transfer entropy values. In addition to this, it produces specific influence measures between pairs of nodes, which could possibly be aggregated to form a global influence measurement.

3.3 Network Prediction and Inference

The main focus of existing network prediction and inference literature involves predicting the time evolution of a network. In general, the link prediction problem proposed originally by Liben-Nowell and Kleinberg (2007), involves observing the communication within a network up to a time t and predicting the links of the same network at some later time $t+w$. A common application of this problem is the recommendation of new connections within online social networks, but it would also be useful in any case where the future of the network may be of interest (e.g., forecasting the controllability of a system or predicting the future influence of network members). The work of Wang et al. (2015) divides the various approaches that have been applied within the domain of the link prediction problem into five main categories:

Node Methods Node methods for the link prediction problem work on the premise that similar nodes are more likely to be connected. In the most basic sense, a node-based prediction method may assign a probability of connection between two nodes based on some measure of similarity of the two nodes. Within online social networks, the similarity between two nodes may be calculated using available information, such as interest overlap (Anderson et al., 2012) or profile similarity (Akcora et al., 2013).

Topological Methods Topology-based prediction methods perform analysis of the network structure surrounding the nodes of interest. Liben-Nowell and Kleinberg (2007) proposed several of these methods, including those based on the number of common neighbours between two nodes and the Jaccard's coefficient computed over the neighbour sets of the two nodes. This idea has also been extended to involve information regarding the paths between nodes within the network as well (Lü et al., 2009).

Random Walk Methods Random walks are often used to model behaviour within social networks (e.g., Page et al., 1999) and have also been applied for the link prediction problem. For example, the hitting time (Fouss et al., 2007) is used as a measure of similarity by computing the expected number of steps to travel from one node to another. As with the study of

centrality measures, changing the properties of the random walk allows for different measures to be calculated and compared. A full discussion of this is available in Wang et al. (2015).

Social Theory Methods These methods attempt to incorporate social ideas, such as triadic closure and community, into the link prediction problem. Some of these methods can be considered a combination of the node and topology-based prediction methods described above, in that they combine personality (node similarity) and structural (topological similarity) measures (Valverde-Rebaza and de Andrade Lopes, 2013).

Learning Methods Learning-based methods have applied various forms of machine learning to the link prediction problem, often using the information used in the other prediction approaches as features for a learning algorithm. As an example, the work of Bringmann et al. (2010) attempts to learn the rules of network evolution through frequent itemset mining. A more thorough discussion of these methods is available in Wang et al. (2015).

The previously discussed link prediction problem involves predicting the time evolution of a network. A related, though different, problem involves predicting all of the links in a network. Within this thesis, this problem will be referred to as the link inference problem. In the case of unobserved networks, the link inference problem may be a useful starting point for influence measurement, control analysis, or the link prediction problem described above. It is also possible that the link inference problem could be applied to observable networks in which the link properties are unknown, incomplete, or possibly misleading. Less research exists for link inference problems than for link prediction, but a few works have attempted to address the problem.

There is a small amount of research that has traced communication and behaviour for the purpose of network inference. For example, De Choudhury et al. (2010) analyzed email records over an extended time period with the purpose of predicting the friendships of users. Using self-reported ground truth, it was discovered that the optimal threshold on the number of emails required to predict a friendship was between 5 and 10 emails per year. In a similar study, Eagle et al. (2009) identified methods for predicting self-reported friendships using historical mobile phone data (both calls and location). This type of analysis has even been extended outside of human social networks, being used to identify relationships between animals through the observation of historical location data (Psorakis et al., 2012). While the methods in these works are tailored specifically for the type of data that they consider, they do motivate the use of historical communication in inferring networks.

The main focus of research on the link inference problem has been focused on the analysis of cascade data. Within this type of model, it is assumed that a number of cascades (modelled as a virus propagating through a network) are observed and the infection times of each node within each cascade are recorded. Using this data, it is possible to infer network structure and/or link properties. Some of the earliest works within this area (i.e., Gomez Rodriguez et al., 2010; Myers and Leskovec, 2010) attempt to determine the most likely propagation tree or adjacency matrix (which can represent the network connections) through probabilistic analysis of the infection times over a number of cascades. These works, however, require assumptions to be made regarding the underlying diffusion properties driving the cascade and, in one of the works, an estimate of the number of edges within the graph. The importance of the accurate measurement of these items is demonstrated by a decrease in accuracy when the solution proposed in Gomez Rodriguez et al. (2010) are applied to real-world data, as well as the wide variance in precision/recall realized with different estimates of the number of edges. Gomez Rodriguez et al. (2011) extended this work by defining a continuous diffusion process and removing the need to assume anything related to the underlying diffusion dynamics. Later work in this area aimed to improve the inference algorithms and provide analysis of the lower bound and expected number of cascades that must be observed (Abrahao et al., 2013; Daneshmand et al., 2014; Netrapalli and Sanghavi, 2012). For example, Abrahao et al. (2013) proposed an algorithm which only considers the first step in which propagation occurs from the host to some other node. As the cascade model used within that work limits the source of an infection to a single node, the first propagation must occur over a real link within the network. The analysis provided by Abrahao et al. (2013) considers the worst case number of cascades that must be observed for this algorithm to predict the network with high probability, finding a bound of $\Theta(n\Delta log n)$, where n and Δ represent the number of nodes and the largest node degree respectively. While the existing research has developed a strong foundation for network inference within this type of cascade model, it still requires a problem domain that matches with the repetitive cascade model. In other network inference problems, such as the opinion model investigated within this thesis (Chapter 5), it is difficult to see how this model could be applied.

3.4 Network Control

Networks can represent the underlying structure of many types of systems, including social networks, financial systems, electronic systems, and biological systems. As such, the term network

control can relate to a wide variety of approaches and applications. Within the domain of network control, there are a variety of goals, including but not limited to: maximizing the adoption of a product or idea, minimizing the spread of information or contagion, limiting the rate of change of opinions, balancing demands, and achieving synchronization. Sections 3.4.1 and 3.4.2 will briefly discuss some of the most frequently investigated goals and control mechanisms within the network control domain. Following this, Sections 3.4.3 to 3.4.5 will provide a more in-depth analysis of the network control research that is most strongly linked to the work presented within this thesis.

3.4.1 Network Control Goals

As mentioned previously, there are a wide variety of network control goals that have been investigated. One of the earliest of these goals was to achieve synchronization of networked oscillators. Within this type of problem, an oscillator influences the phase of other oscillators it is connected to within a network (Arenas et al., 2008). Research related to synchronization of these types of networks has attempted to understand what leads to synchronization (i.e., identical phases) of these networked oscillators. As noted by Liu et al. (2011a), this problem is related to the consensus problems that are investigated within swarm literature. Several works have investigated the relationship between network parameters (e.g., path length, degree distribution, centrality measures) and the difficulty of achieving synchronization (Chen and Duan, 2008; Nishikawa et al., 2003). For example, Zhao et al. (2006) found that increasing the average path length or the degree distribution variance led to an increase in synchronization difficulty. Additionally, control methods have been developed to attempt to achieve synchronization within these networks by directly influencing the phases of oscillators within the network (Yu et al., 2009; Song and Cao, 2010).

Another general type of network control goal involves optimizing the flow within a network. Early works regarding information flows within this area proposed methods for maximizing information flow through networks (Kempe et al., 2003), which has important applications in areas such as advertising. A related problem involves minimizing the flow through a network, which is seen in problems related to epidemiology and immunization (Gallos et al., 2007; Hartvigsen et al., 2007). The control of network flows has also received attention in large scale social systems. A prime example of this is the intelligent control of vehicle traffic networks. This research often involves the optimization of traffic signals, which exert control forces on the vehicle flows within the traffic network (McKenney, 2011; Cools et al., 2013). Another area of application for network flow control is within

smart power grids (Molderink et al., 2010). This can be viewed in a similar way to traffic network optimization, as the efficient control and management of power grids can lead to increased efficiency.

An additional type of network control goal can be viewed as achieving a state-to-state transition. These types of investigations typically view network state as a vector and network behaviour over time as a linear system (Liu et al., 2011b). The control goal, then, is to move from one specific state vector to another. This type of control goal is the focus of the structural control theory that will be discussed in more detail in Section 3.4.3.

Finally, network control has also aimed to maintain the system state, prevent the state from reaching a point in which there are significant negative consequences, or maximize an overall utility function. This type of general problem was defined with the NCP work of Runka (2016), who investigated a consensus avoidance network control problem. The definition of the NCP, however, only specified the use of a utility function to determine controller success, which does not preclude additional problem types. This existing failure avoidance control problem, and the NCP in general, are discussed in more detail in Section 3.4.5. A similar type of problem is the foundation of the distribution-based control work included within this thesis (Chapter 5).

3.4.2 Types of Control Mechanisms

Just as there are a wide variety of possible network control problems and goals, there are different approaches to achieving control that have been investigated. Pinning control is an important control method that has inspired much of the existing network control work covered here. Pinning control involves the use of 'pinner' nodes, which are capable of influencing a number of nodes within the network. These pinner nodes are used to guide the direction of the system to achieve the control goal. Pinning control has been used extensively to solve problems of network synchronization (Wang and Chen, 2002; Yu et al., 2013). Many of the early pinning control works relied on the availability of extensive network knowledge, but localized and adaptive methods have also been considered (DeLellis et al., 2010; Su et al., 2013).

The idea of pinning control can be seen as a strong influence on more recent network control approaches, such as signal injection. Arguably, injecting signals into nodes within the network is very similar to using pinner nodes to affect other nodes within the network. Achieving network control through signal injection, however, is slightly different, in that the controller typically sets the state of control nodes within the network directly (this difference is categorized as indirect

vs. direct control by Runka, 2016). Control through signal injection is the focus of the structural control theory work discussed in Section 3.4.3, where a control vector representing the inserted signals is one of the components of the linear equation used to model the system. This type of control was also used by Runka (2016) to achieve network control in a NCP instance and is similarly used within this thesis (Chapter 5).

While the previously discussed control methods generally involve modifying the state of elements within the network, another method for achieving network control involves modifying the network itself. The structure of a network can, and often does, have a strong effect on the dynamics of its associated system (Boccaletti et al., 2006). By modifying the network, then, the behaviour of the system is also modified, which presents an opportunity to achieve control. In control problems related to network flow, the addition or removal of edges can drastically change the properties of the system. As an example, consider the changes that may arise with the addition of road segments to a traffic network or the removal of communication links within a computer network (Zou et al., 2004). Network modification, however, has also been used to both achieve control and improve the possibility of control in synchronization problems (Funato et al., 2006; Chen and Duan, 2008). Instead of modifying the structure of the network itself, it is also possible to exert control within a system by modifying the flow properties of the connections with the network. Perhaps one of the most obvious applications of this type of control would be the optimization of traffic signals, which control flow through junctions within traffic networks (McKenney, 2011; Cools et al., 2013). But beyond this, another interesting form of control that could be used to achieve control through flow modification is market-based control (Clearwater, 1996), which uses markets (e.g., auctions, exchanges) to effectively allocate the use of resources within systems. This type of control has already been investigated within power grids (Bompard and Han, 2013), traffic systems (Vasirani and Ossowski, 2009), and communication networks (Kuwabara et al., 1996; Miller et al., 1996).

3.4.3 Structural Network Control

A large proportion of existing network control research has involved the optimization of the controller configuration (i.e., the set of nodes used to control the network). Much of this research has applied the idea of structural controllability, originally defined by Lin (1974), and specifically has considered the problem of full state controllability. A system, such as a complex network, is said to be fully state controllable if it is possible to move the system from any initial state $\boldsymbol{x}$ to any other possible

state in finite time (Paraskevopoulos, 2001). One of the most influential and cited works involving the idea of full state controllability in networks is that of Liu et al. (2011b), which proposed methods for measuring the controllability of directed networks modelled by linear time-invariant systems. These linear time-invariant systems can be described by the differential equation shown in Equation 3.4, where $\boldsymbol{x}$ is a vector representing the state of the system's nodes at time t, M is a matrix representing the connection weights between nodes in the network, B is a matrix representing the controlled nodes in the system, and $\boldsymbol{u}$ represents the input vector supplied by the controller.

$$\frac{d\boldsymbol{x}(t)}{dt} = M\boldsymbol{x}(t) + B\boldsymbol{u}(t) \tag{3.4}$$

Liu et al. identified that Kalman's rank condition can be used to determine if a system defined by Equation 3.4 is fully controllable. Specifically, the system is fully state controllable if the controllability matrix in Equation 3.5 has full rank (i.e., $rank(CM) = n$, where n is the number of nodes in the network). In addition to this, Liu et al. also identified an algorithm for calculating the minimum driver node set required to achieve full state controllability within a network, based on a maximum matching within the network. While this initial work was limited to networks with specific properties (directed, no self-links), it has since been extended and applied to arbitrary network structures by Yuan et al. (2013).

$$CM = B, MB, M^2B, ..., M^{n-1}B \tag{3.5}$$

The work of Liu et al. (2011b) has been followed by a variety of further evolutions and developments. One of the first of these works was that of Liu et al. (2012), which proposed a method for measuring the control centrality of nodes within the network. In this case, the control centrality of a node measures the proportion of the network that can be controlled by that node. In analyzing the distribution of control centrality values within different network types, Liu et al. demonstrated that the control centrality distribution is largely determined by the degree distribution, which indicates a strong link between the structure of the network and the number of controllers required. In a similar fashion, Jia and Barabási (2013) analyzed all possible minimal control node sets computed using the strategy proposed by Liu et al. (2011b) and identified relationships between structural properties of the nodes and their frequency of occurrence within the minimal control sets. In doing so, Jia and Barabási found that the likelihood of a node being included in a minimal control node set decreases as the in-degree of that node increases. In other words, the most likely nodes to be included in a control set are those of low in-degree. Based on

these identified links between structure and controllability, other works have moved to classify or measure the controllability of networks based on structural properties (Jia et al., 2013; Pósfai et al., 2013). While these works identified correlations and relationships between network structures and control nodes on a network-level of abstraction, they did not identify what types of sub-structures were most important in determining what nodes are selected as controllers. This problem, however, was addressed by the works of Ruths and Ruths (2014) and Campbell et al. (2015), which aimed to analyze local network structures and their effect on control node sets. These works divide an entire network into a number of 'stems', which represent directed non-looping paths that would each require a control node. In doing so, they identify two major influences on the structure of stems:

Source Nodes These nodes have no incoming links and, as such, must lie at the start of a stem and be controlled directly.

Sink Nodes These nodes have no outgoing links and, therefore, must lie at the end of a stem.

The linkage between these two node types and the number of stems/controllers within a network is proposed as a reasoning for the results of the earlier work, which found a strong correlation between the degree distribution and the number of control nodes. In addition to these two node types, Ruths and Ruths (2014) also define a network dilation as any point in which a stem must be split in order to reach all nodes within the network (which is a requirement for full state controllability). It was found, however, that these dilations contributed significantly less to the number of required controllers than source/sink nodes. This type of analysis, which focuses on the effect of localized network interactions on controllbaility motivates part of the empirical controllability analysis presented within this thesis.

3.4.4 Criticisms of Structural Network Control

Previously discussed works on structural network control have produced interesting contributions related to the analysis of full state controllability (e.g., identifying control node sets and analyzing the effect of different network properties), but there are a number of criticisms to be made when the practical application of these algorithms within the network control domain are considered. A discussion of some of these criticisms is included below:

Linear Time-Invariant System Structural controllability, as proposed by Lin (1974), assumes that the system under control is modelled by a linear, time-invariant system. While this

assumption should hold strongly in some types of networks and systems (e.g., with electronics), it would be expected to be violated frequently in most social systems, which include randomness, imperfection and non-linear dynamics. In addition to this, Zhao et al. (2015) and Cowan et al. (2012) have demonstrated that straying from this assumption can lead to unexpected and spurious results when applying structural control theory algorithms to systems containing self-loops (i.e., individual dynamics), leading to networks that are theoretically controllable with a single control input. While recent work has started to work toward generalizing the structural control theories to non-linear systems (Liu and Barabási, 2016), the question of whether these theories will be applicable in social network control still remains.

Full State Controllability When considering practical social network control, it can be argued that the requirements of full state controllability are too strong in some cases and too weak in others. For example, requiring that the system can move between any two given states is unnecessarily strong, as the system will never be required to be in many states. This point has been previously identified in the work of Gao et al. (2014), who proposed the idea of target control in networks, which aims to ensure full state controllability over only a subset of nodes in the network. In addition to this, requiring that the system can be moved between two states within finite time can be viewed as too weak. When a system in which there are associated payoffs, consequences, or failure conditions associated with states, it may be required to move the state within some time period. This idea has also recently been investigated to some degree from a structural control perspective through the introduction of the idea of the 'energy' required to move a system (Yan et al., 2015; Li et al., 2015; Bof et al., 2016). In essence, the energy required is related to the number of changes that must be made to the state of control nodes to achieve the desired network control goal. This measure, however, is still fairly theoretical in nature and the analysis to this point, it seems, has been limited to identifying relationships between network structure, control time and control energy. A further requirement of full state controllability that can be considered too weak is the requirement that the system can move between any two states, without consideration of the intermediary states that may be required. Again, considering a practical control problem, in which some states may have significant consequences, the state the system is in at any point in time can be very important, meaning the intermediary states encountered while moving the system between two states cannot be ignored.

Lack of Effective Control Signals The second subproblem (after control node selection) within the NCP involves the search for effective control signals to achieve the desired network control goal. This poses a problem when considering the linear time-invariant systems modelled by Equation 3.4, as determining the correct control signals would require accurate measures of the values within the interconnection matrix M. While some research has been undertaken that attempts to estimate the values based on the system's overall behaviour (Barzel et al., 2015), this is a difficult problem that will require significantly more work. This problem motivates the transfer entropy work within this thesis, as the values produced through transfer entropy analysis could be used to infer the connection weights of M.

3.4.5 The Network Control Problem

The network control problem work of Runka (2016) put forward a generalized view of practical network control problems using signal injection. This involves subproblems that include the optimization of control node sets, as well as the generation of control signals to insert into the network. This subsection will discuss the identified relationships between the NCP and existing network control problems, as well as the results found in the initial NCP applied within the domain of a discrete voter model.

Other Controller Configuration Problems

Selecting the optimal set of control nodes, as defined by the NCP, has been investigated for a number of different problems. The immunization problem (Cohen et al., 2003; Pastor-Satorras and Vespignani, 2002) involves the selection of a set of nodes to immunize against some contagion that is propagating through the network, with the ultimate goal of minimizing the spread of the contagion. As noted by Runka and White (2015a), the NCP equivalent of this problem is selecting the set of control nodes which minimizes the spread of contagion when the state of the controlled nodes is constantly set to 'uninfected'. A similar problem is that of influence maximization (Domingos and Richardson, 2001; Kempe et al., 2003), which attempts to identify the set of nodes that maximizes the spread of a specific state through the network. Again, the equivalent NCP definition is to associate the selected nodes as controllers and leave the state of each controller as the desired state to propagate through the network.

A number of controller configuration heuristics based on those used in these previous works

were compared by Runka (2016). Additionally, Runka also proposed a new configuration heuristic called FAR, which iteratively selects control nodes in a way that maximizes the minimum distance between the newly added control node and all previously selected control nodes (i.e., it selects the node 'farthest' from the other control nodes). The motivation behind the development of the FAR heuristic was to produce a control set that minimizes the time to diffuse information through the network from the controllers. As with many of the influence measurement algorithms discussed in Section 3.2, most of these control configuration heuristics are structural in nature. While the FAR heuristic was shown to be superior to many others, including a solution generated through evolutionary computation (Runka and White, 2015b), the reason for this high level of performance has not yet been investigated in detail. The analysis presented in Chapter 5 of this thesis provides some evidence in support of the FAR's heuristics success, while several improvements to the original FAR heuristic are also proposed and compared (Section 5.5).

Control Signal Generation

While controller selection has been investigated quite thoroughly within the existing literature, the problem of selecting control signals has received significantly less attention. The formalization of the NCP, however, has provided a means for investigating the problems of controller selection and control signal generation simultaneously. The first problem instantiation created specifically for use within the NCP is the $\theta-$Consensus Avoidance Problem ($\theta-CAP$) problem (Runka and White, 2015a; Runka, 2016). Within the discrete, 2-valued $\theta-CAP$ problem, nodes within the network take one of two states: 0 or 1. The goal of the controller within the $\theta-CAP$ problem is to maintain the state of the system such that the proportion of nodes in either state does not exceed a specified threshold θ_G. This is modelled via the utility function that the controller must attempt to minimize, shown in Equation 3.6, where V^1, V^0 represent the sets of nodes with state equal to 1 or 0 respectively and V represents the entire set of nodes.

$$U(t) = \begin{cases} 0, \text{ if } \frac{||V^1(t)| - |V^0(t)||}{|V|} \leq \theta_G \\ 1, \text{ otherwise} \end{cases} \tag{3.6}$$

This type of utility function represents an aggregate of the overall system state and is quite different from the typical start/end state goals considered within the structural network control research. This type of problem definition may be more suitable for many social problems, where the interest

is not necessarily in moving from one state to another, but is instead focused on maintaining some overall system state and/or behaviour (e.g., consensus avoidance within the $\theta{-}CAP$ problem).

The previously discussed structural network control works (Section 3.4.3) have represented the system's state as a vector consisting of the state of each node within the network. When considering systems within the structural control framework, this vector-based view of state is necessary. The first problem within the NCP framework that considers controller behaviour, however, begins to move away from the use of vectors by using an aggregate of the vector state to determine a controller's success. It can be expected that further NCP problems will continue this trend, which may require different views of what it means to control a system.

3.5 Summary

This chapter discussed the existing research within the areas of influence measurement, network prediction/inference, and network control. Based on this discussion, a number of important questions that this thesis will address can be identified. The remainder of this section will discuss these questions and identify where in the thesis each question is addressed.

Within the domain of influence measurement, it was noted that structural influence measures may be misleading due to unknown link properties. An ideal network influence measure should be based on the actions/behaviour of the participants of the network. Additionally, the ideal influence measure should be capable of measuring both the global influence (i.e, network-wide) and the local influence (i.e., node-to-node) of any particular node. Within Chapter 4, this thesis will investigate the use of transfer entropy for the measurement of global and local influence of nodes within a simulated network-based game.

When considering the problem of network inference, one particular method involving the analysis of cascades has been investigated thoroughly within the existing literature. This cascade approach, however, is not necessarily applicable to many other scenarios that cannot successfully be modelled using the idea of cascades. The network inference work presented in Chapter 4 combines transfer entropy measurements and a threshold selection approach to perform network inference. This approach is applicable to a wide range of scenarios in which the states of entities within the network change over time. It is also possible that this transfer entropy approach could be applied within a cascade model as well.

As mentioned in Section 3.4.5, the first network control problem defined within the NCP

framework started to diverge from the vector space model used within the structural control theory work presented in Section 3.4.3 by using an aggregate measure of state within a utility function. Within this thesis, the work presented in Chapter 5 continues this trend by investigating the control of state distributions. The use of distributions as targets for control is arguably more powerful/expressive than a vector-based approach, while also being more applicable to the practical problems that will evolve from the original NCP work.

Finally, while the existing NCP work demonstrated that the FAR heuristic algorithm for control node selection was superior to all other considered solutions, the reason for this success has not been investigated in detail. The analysis of the distribution-based control problem included within Chapter 5 contributes a better understanding of what network properties most significantly affect the control success within an NCP-type problem. In a way similar to the work of Jia et al. (2013) and Pósfai et al. (2013) within the structural control literature, Chapter 5 of this thesis will consider the effect of various network parameters on network control success. Additionally, this thesis will look at the effect of local structures and interactions on network control success, which is similar to the structural control analysis performed by Ruths and Ruths (2014) and Campbell et al. (2015). In doing so, empirical evidence in support of the FAR heuristic's previous success is presented, while possible improvements are also identified, implemented and investigated (Section 5.5).

Chapter 4

Influence Analysis and Network Inference using Transfer Entropy

4.1 Introduction

The definition of influence presented in Section 3.2 involves the ability of one entity to affect the future behaviour of another entity within a system. As discussed in Chapter 3, there are many different methods for measuring influence within social systems. Many of these methods, however, neglect this behavioural view of influence and only rely on the analysis of networks with homogeneous nodes/links. An important criticism of these approaches, then, is that nodes/links within social systems are generally not homogeneous (Alarcón-del Amo et al., 2015) and measuring influence without considering the behavioural relationships present within the system can lead to misleading results.

Many systems, especially social networks, produce observable behaviour (e.g., action logs) which can be used to measure influence more accurately. This chapter answers a number of important questions related to the use of transfer entropy (see Section 2.4 for an introduction) for the identification and measurement of influence relationships within networks through the analysis of the actions taken by agents within a simple game model. The chapter begins with Section 4.2, which defines a simulated game that is used to generate the data used for transfer entropy analysis later in the chapter. Following this, Section 4.3 describes how transfer entropy values calculated over time series data can be used to measure both local and global influence. Section 4.3.3 provides comparisons between the transfer entropy-based influence measurements proposed in Section 4.3.2 and two other influence measures to determine how strongly transfer entropy influence measures correlate with these other measures. Following this influence measurement analysis, Section 4.4 considers the problem of infering influential links using transfer entropy values. This problem is defined as the influence link inference problem in Section 4.4.1. Section 4.4.2 describes how the link inference problem could be solved using transfer entropy measurements, proposing a general algorithmic approach and several methods for implementing this algorithmic approach in practice. Section 4.4.3 provides discussion that answers a number of important questions regarding the accuracy of the proposed influence prediction solutions. Section 4.4.3 describes the absolute best case

prediction accuracy that could be realized using the proposed transfer entropy prediction approach for each network instantiation that is considered. Section 4.4.3 also performs classification based on the best case accuracy values to determine which network properties most strongly determine the success or failure of the proposed algorithms in performing link prediction. The discussion then considers how accurate each specific prediction algorithm performs on each of the considered network instantiations and concludes with an overall ranking of the proposed prediction algorithms.

4.2 Anti-Majority Game Model

To investigate the use of transfer entropy for influence analysis, a simple game model with known behaviour and influence properties is used. The model is based on the minority game model proposed by Challet and Zhang (1998), which is itself related to the El Farol Bar problem (Arthur, 1994). In the original minority game model, an odd number of players must simultaneously and independently select one of two possible choices at each turn. The players are then divided based on choice with the majority group 'losing' the turn, while the minority group 'wins'. The goal of each player taking part in the game, then, is to predict the minority choice in the upcoming round based on previous events. It has been proposed that this simple model can be used to explain some aspects of financial markets, such as significantly large fluctuations in prices without any external 'shock' affecting the system (Buchanan, 2012). It is this link to financial, social, and resource-based systems, which this line of research aims to ultimately understand and effectively control, that motivates the use of a similar model within this thesis. A control approach similar to the one proposed and examined in Chapter 5 could be applied to this model, leading to the development of a controller capable of limiting or reducing fluctuations within similar resource-based systems. This research, then, could ultimately have applications in financial and resource-based systems (e.g., market-based control).

Through the addition of communication to the basic model, the minority game becomes a social system in which influence can be investigated. Much of the research relating to influence measurement discussed in Chapter 3 (e.g., Ver Steeg and Galstyan, 2013) uses real-world social network data. Within this thesis, a fabricated dataset is used, largely due to the lack of ground truth in many real social datasets. While the connections of a network may be included within an existing real dataset, the strength and influence properties of these connections are generally unknown. In addition to this, there are external influences within real-world data that may also affect participants within the system. When using a mathematically specified model, such as the one considered within this

chapter, there is complete knowledge regarding the network used to share information, the information that is shared between entities within that network, and how decisions are made based on that information. This allows the accuracy of any influence-related predictions to be measured precisely.

The simple minority game is augmented here to create the anti-majority game, which simulates how an information network may be used by autonomous agents to make choices while playing. The main changes to the original game include the addition of communication between players, resulting in dependent, or group-based decision making, as well as the inclusion of more than two choices. As there are more than two choices, there is no longer a guarantee of a strict minority/majority, which is why the term anti-majority game is used here. Arguably, the decisions of participants within financial (and many other) systems are not made independently. For example, friends likely share tips, strategies, or ideas relating to the stock market with each other. This model, then, may more accurately model these types of systems, in which communicated information affects the future decisions of participants.

More formally, the anti-majority game can be described as a tuple $AMG = (G,C,P(c),A)$, consisting of a network graph (G), set of game choices (C), payoff function ($P(c)$), and agent set (A). The following subsections outline the various components of the augmented game model, describe the implementation of each of these components used within this work, and explain further modifications that could be introduced for more complex analysis. In addition to these explanations, Algorithm 6 describes the high level process of playing the anti-majority game.

4.2.1 Choices

Instead of using only two choices, as in the original specification of the minority game, the anti-majority game used within this thesis has a set of choices, C, that can be selected by agents in the simulation. A payoff function, $P(c)$, is used to determine how much a player receives as reward/penalty when c is selected in a game round.

4.2.2 Payoffs

As there are more than two choices in this game, a strict majority is not guaranteed as it was in the original minority game. Instead, the 'losing' choice is the choice - or choices in the case of a tie - selected by the largest number of agents (i.e., the most popular choice). Assuming the set of majority choices is C_{pop}, each agent that made a choice $c \in C_{pop}$ receives a payoff (penalty) of

Algorithm 6 Pseudocode of anti-majority game simulation process

Require: G is the communication network

Require: C is the set of available game choices

Require: $P(c)$ is the payoff function

Require: A is the agent set

Require: r is the number of rounds the game should run

```
1: procedure AMG(G,C,P(c),A,r)
2:     round ← 0
3:     while round<r do
4:         for a in A do
5:             a updates memory using memory update algorithm a.MU
6:         for a in A do
7:             a selects choice using decision strategy a.DS
8:         C_pop ← most popular choice(s)
9:         for a in A who chose C_pop do
10:            Assign penalty to a using payoff function P(c)
11:        for a in A who did not chose C_pop do
12:            Assign reward to a using payoff function P(c)
13:        for a in A do
14:            a shares information using communication strategy a.COM
15:        round←round+1
```

$-P(c)$. For all other choices $c \notin C_{pop}$, the agents that selected c receive a payoff (reward) of $P(c)$. Within the model used for analysis here, all choices are given a static payoff of 1.

4.2.3 Network

An underlying communication network connects each agent to a subset of all other agents taking part in the game. This network can be modelled by a graph, $G=(V,E)$, where the vertex set, V, consists of all agents within the game and each edge e in E indicates the ability to communicate between the two agents connected by e. While the basic formulation of the model used within this work uses an undirected graph, resulting in bidirectional communication across each edge in E, a

directed graph could also be used to model relationships in which communication is unidirectional. It should be noted that, as the only form of influence within the system is the messages received from neighbouring nodes in this communication network, this network also represents the influence network we wish to infer in later sections.

4.2.4 Agents

An agent within the simulations of the anti-majority game can be defined as a tuple $a = (N,COM,DS,MU)$, which consists of the following components:

N - The neighbours of the agent. This represents the set of agents that the agent can communicate with directly. As the simulations used within this thesis only involve undirected network edges, this set consists of all other agents that are directly connected to the agent by an edge in the graph G.

COM - The communication strategy which governs the communication of the agent. The communication strategy used by agents in the simulations analyzed within this thesis is described under the Agent Communication heading below.

DS - The decision strategy the agent uses to make a choice within the game. The decision strategy used by agents in the simulations analyzed within this thesis is described under the Agent Decisions heading below.

MU - The algorithm which determines how the agent's memory is updated within the simulation. This must specify how received messages from other agents are processed, as well as any memory updates that must be made over time. The algorithm used by all agents within the simulations considered here is described under the Agent Memory Update heading below.

While there are many possible strategies and variations that could be investigated, this is considered beyond the scope of this thesis. Within this work, only a homogeneous network of agents using the communication, decision and memory update strategies described in the following three subsections is considered.

Agent Communication

The communication strategy, COM, for an agent within the anti-majority game can be defined as a function that maps current agent/game state to a set of messages which will be communicated to neighbouring agents. The agent definition within this work requires an agent to send its choice and payoff information from the most recent round to every other agent it shares a connection with. Each agent also receives the same type of message from itself, allowing it to remember its last choice and payoff. While all messages sent from an agent in the simulations considered here are sent to all neighbouring agents, the definition of the anti-majority game does not preclude the use of directed messaging to a subset of an agent's neighbours.

Agent Memory Update

When an agent receives a message containing choice/payoff information from a neighbour, that choice and payoff information is added to memory. In addition to this, each agent applies the same memory update rule at the start of each game round. This update process involves the elimination of any stored information from rounds other than the previous round. This limits the length of the memory of agents within these simulations to a single round.

Agent Decisions

Similar to the communication strategy, the decision strategy, DS, for an agent within the anti-majority game can be defined as a function that maps current agent/game state to one of the possible choices, C, that can be made within the game. The decision strategy applied by all agents within this work is outlined in Algorithm 7. Each agent iterates through every piece of choice/payoff information it has stored in memory from the previous game round. The average payoff for each choice is computed and these numbers are treated as expected values for the following round. Each agent, then, chooses in the next round, one of the choices with the highest average payoff from the previous round based on the information available to that agent. The only time this strategy is changed is when the agent only has a single possible choice (i.e., it only has information regarding one choice), in which case it may choose a random choice with probability $\tau = 0.1$. This small probability of making a random choice was incorporated into the model to prevent all agents from converging to a single choice over time. The precise value of τ used here was determined through initial

Algorithm 7 Basic agent decision strategy

Require: Memory containing last round choice/payoff information

Require: τ probability of using a random choice

procedure MAKECHOICE(Memory, τ)
  for choice c in Memory **do**
    $\mathbb{E}(c) \leftarrow$ average payoff of c
  $C_{max} \leftarrow$ set of choices with maximum $\mathbb{E}(c)$
  $next\_choice \leftarrow$ one of C_{max} chosen randomly
  if $|C_{max}|=1$ **then**
    $next\_choice \leftarrow$ random choice from C with probability τ
  return $next\_choice$

experiments, which found that 0.1 was a high enough value to prevent convergence of agents toward a single state while still allowing agents to make information-based decisions in almost all cases.

4.2.5 Experimental Data

In order to generate a transfer entropy dataset to be used for the influence measurement research presented in this chapter, the anti-majority game model was simulated with varying parameters. Throughout each simulation, the choices made by each agent were recorded, which resulted in a number of time series representing each agent's actions throughout the simulation. More precisely, each agent's time series data is an ordered set of choices made in each round throughout the simulation's duration. This time series information can then be used to calculate and analyze transfer entropy values between different agents' time series. To reduce the affect of small sample bias in the transfer entropy values, the bootstrapping approach described in Section 2.4.1 was used with 10 bootstrap samples. The varying game parameters that were used to generate the data are described in more detail below:

- Network: The networks used include both artificially generated networks and networks sampled from real-world datasets. The artificially generated networks were generated using either the scale-free (Barabási and Albert, 1999), random (Erdös and Rényi, 1959), or small world (Watts and Strogatz, 1998) models and consisted of 400 nodes each. The networks sampled from real-world datasets consisted of 100 nodes extracted from Google+, Facebook or Twitter

networks. For all network types other than Facebook, 10 network instantiations were created and analyzed. As discussed in Section 2.3, the Facebook dataset only provided 8 total networks, so in this case, there were 8 instantiations analyzed. For more information on the networks, as well as the generation and sampling methods used, see Section 2.3 and Appendix A.

- Rounds: The number of game rounds that were used in calculating the transfer entropy values was varied between 500 and 10000. In general, the accuracy of the results improve as the number of game rounds increases, as the transfer entropy values converge toward their true value.

- Choices: The number of choices available to agents within the game. While data was produced using 5, 10 and 20 game choices, early analysis found that, given enough time to converge, 20 game choices produced the most accurate results. For this reason, most of the results presented in the remainder of this chapter use scenarios with 20 game choices.

4.3 Influence Measurement

The first problem that will be investigated here involves the use of transfer entropy for the purposes of influence measurement. This section will analyze the relationship between transfer entropy values, calculated between pairs of time series data produced by agents within the anti-majority game, and two other measures of influence. As the information and diffusion properties of the underlying game model are known, it is expected that the two types of influence measures used should accurately reflect the real influence of agents within the system. A strong correlation between the measurements derived from the transfer entropy values, then, should indicate that these measures also produce an accurate view of influence while only considering the history of behaviour within the game. Unless otherwise stated, all results discussed within the remainder of this section involve game scenarios with 20 game choices and transfer entropy values calculated after 5000 game rounds.

4.3.1 Influence Baselines

Two influence baselines will be used for comparison with the transfer entropy-based measures. The first of these measures is PageRank, which is a proven method of network structure analysis for influence measurement (Page et al., 1999). While the use of purely structural measures, such as PageRank, was criticized in Chapter 3, this criticism stemmed from the fact that the link properties of networks are generally not known in practice. Here, however, it is known that all links

are identical within the network, which means that the PageRank algorithm should produce an accurate measure of influence. The second influence measurement is derived from the algorithmic properties which determine agent behaviour within the anti-majority game model and is referred to here as the 'effective influence'. Each agent, a, makes its next choice based on the information it received from its $degree(a)$ neighbours about the previous round. As each neighbour contributes one piece of information, and all information is weighted equally, each neighbour's contribution can be weighted as $\frac{1}{degree(a)}$. This value can be viewed as a local effective influence measure of an agent x on its neighbour a, which will be referred to as $LEI(x,a)$ and is defined by Equation 4.1.

$$LEI(x,a) = \begin{cases} \frac{1}{degree(a)}, & a \in N(x) \\ 0, & \text{otherwise} \end{cases} \tag{4.1}$$

To produce a global influence measurement from this information, the local influence values associated with each of an agent's neighbours can be summed to produce the global effective influence, $GEI(x)$, as in Equation 4.2.

$$GEI(x) = \sum_{a \in N(x)} \frac{1}{degree(a)} = \sum_{a \in N(x)} LEI(x,a) \tag{4.2}$$

4.3.2 Transfer Entropy Influence Measures

Using the time series data generated through a simulation, a transfer entropy value is calculated between each pair of agents within the network under consideration. These transfer entropy values can be used to generate both local and global influence measurements. The transfer entropy-based local influence of an agent x on another agent y ($TELI(x,y)$) is simply represented by $TE_{x \to y}$, which is the transfer entropy value from the time series of x to that of y. Similar to what is done with the global effective influence measure $GEI(x)$, discussed previously, a transfer entropy-based global influence measure, $TEGI$, for a node x, can be defined as the sum of the local influence of x on all other agents within the system, as in Equation 4.3.

$$TEGI(x) = \sum_{y \in A} TE_{x \to y} \tag{4.3}$$

4.3.3 Influence Measure Comparison

Table 4.1 shows the mean and standard deviation of the Pearson's correlation coefficient between the $TEGI(x)$ and $PR(x)$ measures over each type of network. In general, this table demonstrates

Table 4.1: Pearson's correlation (PC) between $TEGI(x)$ and $PR(x)$ for different network types

Network Type	**Mean PC**	**Std. Dev. PC**
Scale-Free	0.947	0.016
Random-8	0.781	0.032
Random-16	0.849	0.014
Small-0.1	0.542	0.034
Small-0.2	0.622	0.028
Twitter	0.870	0.062
G+	0.712	0.174
G+Similar	0.629	0.107
Facebook	0.720	0.144

a relatively high linear correlation between the two values. The networks with the lowest correlation values belong to the two types of small world networks, which may be at least partially attributed to the low variance in node degree within these small world networks. This hypothesis is supported by a Pearson's correlation coefficient of 0.73 between the standard deviation of node degree relative to average node degree (i.e., $\frac{\sigma_{degree}}{\mu_{degree}}$) and the linear correlation between the $TEGI(x)$ and $PR(x)$ measures. Still, a strong linear correlation is realized between the two values overall, with an average correlation coefficient value of 0.771 across all considered network instantiations.

In comparing the transfer entropy influence measurements to the effective influence measures, both the global and local influence measures can be considered. Table 4.2 shows the Pearson's correlation coefficients between the global transfer entropy and effective influence measurements. As with the PageRank comparison above, a strong linear correlation is recognized across most network types considered. Once again, the lowest correlation values are found in the small world networks with low degree variance. When considering the local influence measures, correlations remain strong, as can be seen in Table 4.3. In fact, bolstered by the increased correlation values realized for small world and random network types, the overall average correlation in the local case increases to 0.856, compared to a 0.727 overall average in the global case.

The comparisons made within this section demonstrate that the transfer entropy influence measures are strongly correlated with measures that are expected to accurately represent influence within the anti-majority game model. In this case, then, transfer entropy has been shown to be an effective behaviour-based influence measurement that does not require any structural analysis or assumptions regarding link/node properties. In cases where the behavioural properties of the system are unknown and heterogeneous, it is expected that transfer entropy influence measurements should

Table 4.2: Pearson's correlation between $TEGI(x)$ and $GEI(x)$ for different networks

Network Type	**Mean PC**	**SD PC**
Scale-Free	0.936	0.011
Random-8	0.856	0.017
Random-16	0.835	0.015
Small-0.1	0.534	0.028
Small-0.2	0.602	0.020
Twitter	0.901	0.067
G+	0.824	0.135
G+Similar	0.703	0.082
Facebook	0.747	0.123

Table 4.3: Pearson's correlation between $TELI(x,a)$ and $LEI(x,a)$ for different networks

Network Type	**Mean PC**	**SD PC**
SF	0.948	0.006
Random-8	0.953	0.002
Random-16	0.873	0.003
Small-0.1	0.865	0.002
Small-0.2	0.866	0.003
Twitter	0.808	0.103
G+	0.803	0.064
G+Similar	0.820	0.031
Facebook	0.747	0.087

continue to be accurate while structure-based measures such as PageRank will not perform as well.

4.4 Predicting Influence Relationships

While the previous section compared transfer entropy influence measures to other baseline influence measurements, this section will deal with the problem of identifying which pairs of agents share an influence relationship. The work of Ver Steeg and Galstyan (2013) demonstrated that some of the most likely connections could be identified by looking at outliers within the distribution of transfer entropy value pairs. This section will expand on this previous work by considering just how accurately the actual links of a network, which represent influence relationships in the anti-majority game model, can be predicted using various methods.

4.4.1 The Influence Link Inference Problem

Given a set of action logs belonging to agents within an anti-majority game simulation using an unknown network, the link prediction problem is a binary classification problem with the goal of correctly predicting whether a link exists or not between each possible pair of agents. In the case of the anti-majority game defined within this work, the links of the network represent the only form of influence within the system. Other than the small chance of randomly selecting a choice to prevent convergence, external influence is non-existent. As has been discussed previously, the lack of external influence present in this artificially generated dataset is an advantage that is not available when analyzing real-world data.

The remainder of this section will discuss solutions to this problem that are generated using transfer entropy values calculated from the time series that represent the action logs of the agents. While the work of Ver Steeg and Galstyan (2013) noted that some likely influence relationships could be identified using outlier transfer entropy values, the algorithms presented in the remainder of this section improve upon this existing work by predicting the entire influence network.

4.4.2 Transfer Entropy Threshold Selection

To create an influence-based solution to the link prediction problem discussed above, a method must be proposed to answer the question 'should a link between i and j be predicted?' for each pair of nodes i and j in the network. Within this thesis, this question is answered by algorithmically selecting a transfer entropy threshold value, θ_{TE}. A link is then predicted between any pair of nodes i and j where $TE_{i \to j} \geq \theta_{TE}$ or $TE_{j \to i} \geq \theta_{TE}$. This prediction process is formalized in Algorithm 8.

The list of possible threshold value choices used within this work is a discrete set of 100 values uniformly distributed between 0.0 and the maximum observed transfer entropy value between any two time series within the simulation. In this case, the use of 100 values was made to balance the trade-off between the computational complexity required to compute predicted networks and network measures at each possible threshold and the precision of the ultimate result, which could be increased by considering a larger number of possible thresholds. A uniform distribution of these threshold values was used because it requires no assumptions or further analysis regarding the distribution of transfer entropy values and/or network metrics. As many of the network metrics change monotonically in relation to the threshold, an improved search method could focus the search on more precise thresholds based on the metric accuracy (e.g., similar to a binary search), but this is expected to

Algorithm 8 Predicting a network based on a transfer entropy threshold value

Require: Threshold value to consider, θ_{TE}

Require: Transfer entropy measurements between all possible agent pairs, TE

```
procedure ThresholdPredict(θ_TE, TE)
    PredictNet ← new empty network
    for agent i in A do
        for agent j in A do
            if TE_{i→j} ≥ θ_TE then
                Add edge between i and j to PredictNet, if not already present
    return PredictNet
```

produce only marginal gains in prediction accuracy and is considered outside the scope of this thesis.

To select the value of θ_{TE} from a list of possible values, several algorithmic methods are investigated, which require varying degrees of system or network knowledge. The accuracy of the predictions made using each of these methods will be compared to each other and across the various considered network types and instantiations in Section 4.4.3. Each of the algorithmic methods are discussed in more detail below.

Property Assumption

Within some systems, certain network properties may have known or expected values. In other systems, it may also be possible to estimate the value of certain network properties or determine that a network property falls within some interval. For example, it could be known, assumed, or estimated based on available system information, that the average degree of the network is 5 or that the diameter of the network is 8. Assuming the value of a specific network property is known, how can this information be used to enable network prediction? This subsection discusses a method that uses an assumed network property value to select a final transfer entropy threshold value for network prediction.

To predict a network using a property assumption, a property of the network is assumed to have a specific value. The value of this property can be calculated within any network that has been predicted using a specific transfer entropy threshold value. By comparing the network property's known value to the value of the same metric in a network predicted using a specific transfer entropy

Algorithm 9 Predicting network based on target property value

Require: Set of threshold values to consider, TH

Require: Property of interest, $Prop_{int}$

Require: Target value of property of interest, $Target$

Require: Transfer entropy measurements between all possible agent pairs, TE

```
1: procedure PropertyPredict(TH, Prop_int, Target, TE)
2:     BestThresholds ← {}
3:     MinError ← ∞
4:     for each value t in TH do
5:         PredictNet ← ThresoldPredict(t, TE)
6:         PredictProp ← value of Prop_int in PredictNet
7:         error ← | PredictProp − Target|
8:         if error < MinError then
9:             BestThresholds ← {t}
10:            MinError ← error
11:        else if error = MinError then
12:            BestThresholds ← BestThresholds ∪ t
13:    FinalThreshold ← median value of BestThresholds
14:    return ThresoldPredict(FinalThreshold, TE)
```

value, a measure of error can be calculated. This error calculation process can be carried out for each possible transfer entropy threshold value, and the threshold that minimizes the error between the known value and predicted network value can be selected. In cases where the error is minimized at more than one threshold value, which is common for some properties such as diameter, the median threshold value among the set of minimizing values is chosen. The process of selecting a threshold value and predicting a network via property assumption is formalized in Algorithm 9.

Within this thesis, the use of only a single property value is considered. It is possible, however, that combining multiple properties (i.e., using a weighted combination of multiple errors) may lead to more accurate results than relying only on a single property value assumption. In addition to this, only cases where the target value is known exactly are considered. This type of analysis, however, is not limited to scenarios in which an exact network property is known. In cases where this information is purely an estimate, the accuracy of prediction will be affected by the magnitude

Algorithm 10 Predicting network based on sampled agent data

Require: Set of threshold values to consider, TH

Require: Set of sample agents, A_{sample}

Require: Property of interest, $Prop_{int}$

Require: Transfer entropy measurements between all possible agent pairs, TE

1: **procedure** SAMPLEPREDICT(TH, A_{sample}, $Prop_{int}$, TE)

2: $Target \leftarrow$ calculate property of interest over A_{sample}

3: **return** PropertyPredict(TH, $Prop_{int}$, $Target$, TE)

of the estimation error. In other cases, it may be known that the network property is within some interval. The same algorithm, then, could be applied at different values within the known interval and the prediction results could be used to select a final threshold (e.g., through a consensus-based algorithm modification). These ideas, however, are not investigated in detail within this thesis.

Network Sampling

The previously discussed property assumption prediction method assumes that the value of at least one network property is known. When the system under consideration is difficult or impossible to observe, which is arguably the case when considering influence relationships, requiring the value of a network property to be known precisely may not be a feasible approach. For practical applications, then, it is important to identify methods that may be more feasible in these types of systems. So how can network prediction be achieved without knowing any global properties of the network under consideration?

This section proposes a solution to this question that relies on network sampling. While it may be infeasible to define, either using system knowledge or through calculation, a global value for any network property, it may be feasible to produce an estimate of a network property by receiving some network information from a small sample of agents/participants within the network. As described in Algorithm 10, once a network property has been estimated using sampled data, the previously discussed property assumption method can be used to make a prediction.

Unless otherwise stated, the results relating to sample-based methods included within this thesis use a sample size of 5% of network nodes. Additionally, all sample-based results are calculated using the average result found over 100 randomly selected samples. Sample sizes of 1% and 10%

were also considered, with marginal differences being realized. The F-score using a 5% sample was found to be, on average, 84.1% of the maximum possible threshold-based F-score (see Section 4.4.3 for details of the best case threshold-based predictions), while the average using 1% and 10% samples were 80.2% and 84.8% respectively. While these results indicate a larger sample can lead to improved performance, whether the additional observation is worth performing will depend strongly on the application domain and the associated cost of gaining additional information. As an in-depth analysis of the effect of sample size is considered beyond the scope of this thesis, detailed results for sample sizes of 1% and 10% of nodes are omitted. Further details regarding the three sample-based prediction methods that are considered here are included below:

- Average Degree: The average degree of the sampled set of nodes is computed to determine the expected average degree of the network. The error between the predicted average degree and the expected average degree is computed for each threshold value. If a single threshold minimizes the error, that threshold is selected for use in prediction. If multiple thresholds minimize the error, the median of these threshold values is used for prediction.

- Degree Distribution: A histogram of degree values is created based on the sample and a similar histogram is created for the predicted network at each threshold. The earth-mover's distance (Rubner and Tomasi, 2001) between the sample and predicted histograms is used as a minimization target, with the threshold producing the histogram closest to that of the sampled node histogram being used for final prediction.

- Edge: This solution assumes that edge information is available from a sampled set of nodes. For each possible threshold value, a network is predicted. The precision and recall of this prediction across the sampled nodes' edges are computed and the F-score is calculated from these values. The threshold that produces the highest F-score, which represents the most accurate prediction of the sampled nodes' identified connections, is used to produce the final network prediction.

It should be noted that these three measures do not have to be measured separately. For example, sampling the degree distribution would also provide information about the average degree, and vice versa. Additionally, performing edge-based sampling would provide the information to generate average degree and degree distribution estimates. While it is not investigated in detail here, it

is possible that combining these methods, which share the same information requirements, into a single threshold selection, could allow for improved prediction accuracy.

Kernel Density Estimation

Both of the previously discussed methods have required some form of network or system knowledge to perform network predictions. Is it possible, however, to produce predictions using only the transfer entropy measurements (i.e., without relying on any available system information)? As was observed by Ver Steeg and Galstyan and confirmed through analysis of the data generated as part of this thesis, the distribution of transfer entropy values tends to have a peak around 0 created by the large number of unconnected pairs of nodes. The transfer entropy values for pairs of nodes that are connected, then, tend to lie outside of this large mass. While the exact distribution of transfer entropy values is network dependent, it would be expected that a similar pattern would be present in all networks of sufficiently low density. In all of these networks, there will be a large proportion of non-links, and the transfer entropy values of these non-links are expected to form the large central mass around 0. This section will investigate a method of prediction that makes use of this expected distribution, while not relying explicitly on any actual network or system knowledge.

Kernel density estimation (KDE) is a method for estimating the density function of random variables. In this case, KDE is used to estimate the density of the transfer entropy values at different thresholds. As is common in practice, the kernel used for estimation here is the normal distribution $\mathcal{N}(0.0,1.0)$. The threshold value with the highest estimated density is viewed as being the center of the peak representing the non-linked pairs of nodes. The KDE threshold selection algorithm, described in Algorithm 11, works by finding this center and following the KDE curve downward as the threshold value increases. Once the estimated density has reached a local minimum, the current threshold is selected and network prediction is made. Ideally, this will lead to the selection of a threshold that is above any value close to the peak value while including most of the values that are outliers to the main mass.

Figure 4.1 demonstrates an example of how this KDE method would select a threshold. Within this figure, and in most of the cases that are investigated within this thesis, there are several threshold values that could be considered local minima. All of the predictions made using the KDE method within this work follow Algorithm 11, selecting the last of these possible minimum threshold values. It is, however, possible that the various minimum threshold values could be

Algorithm 11 Predicting network using kernel density estimation

Require: Set of threshold values to consider, sorted in ascending order, TH

Require: Transfer entropy measurements between all possible agent pairs, TE

```
1: procedure KDEPredict(TH, TE)
2:     KDE ← kernel density estimate for all values in TH
3:     Density_max ← −1
4:     for t in TH do
5:         if KDE_t ≥ Density_max then
6:             Density_max ← KDE_t
7:     Density_min ← Density_max
8:     for t in TH starting at Density_max do
9:         if KDE_t ≤ Density_min then
10:            Density_min ← KDE_t
11:        else
12:            return ThresoldPredict(Density_min, TE)
13:    return ThresoldPredict(Density_min, TE)
```

used to identify an ideal threshold range or estimates of upper/lower bounds for various network properties. This information could then be used to apply modified versions of the previously discussed prediction approaches without relying on actual network knowledge or sampling.

4.4.3 Prediction Accuracy Analysis and Discussion

This section will investigate the performance of the previously described prediction approaches on the influence link inference problem. The section will begin with an analysis of the best case performance of the transfer entropy threshold prediction method across all of the scenarios considered. Following this, the prediction performance results of each of the proposed methods are discussed and compared.

Best Case Prediction Accuracy

Due to differences in network properties and the random decisions present within the anti-majority game, the accuracy of the transfer entropy measurements, and in turn, the threshold approach, will vary across scenarios and will also vary as the number of game rounds in a specific scenario

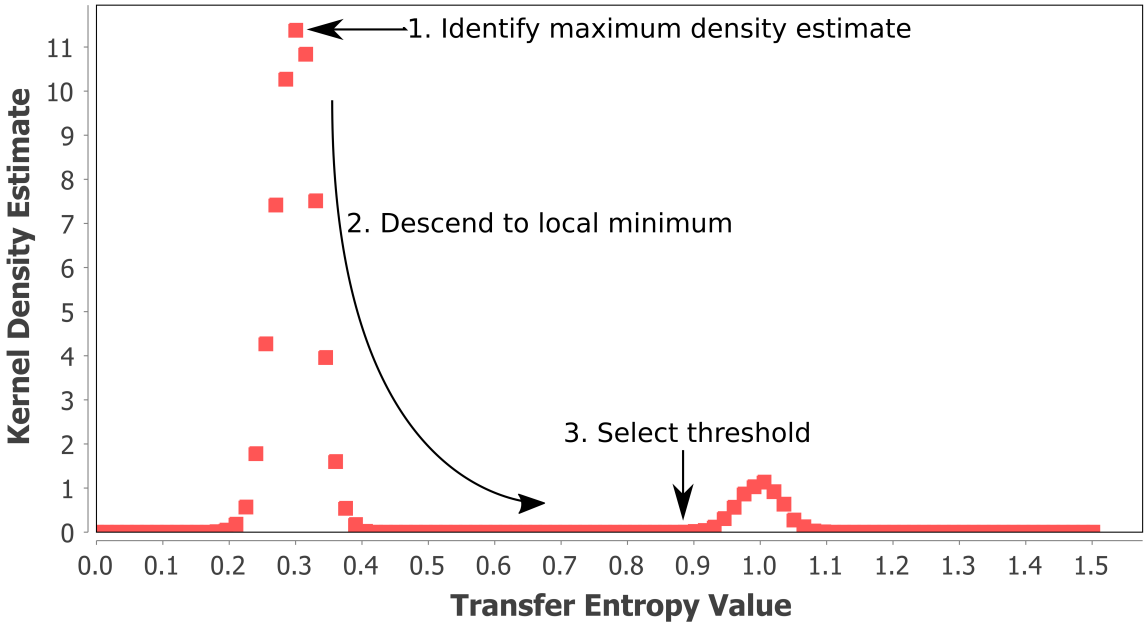


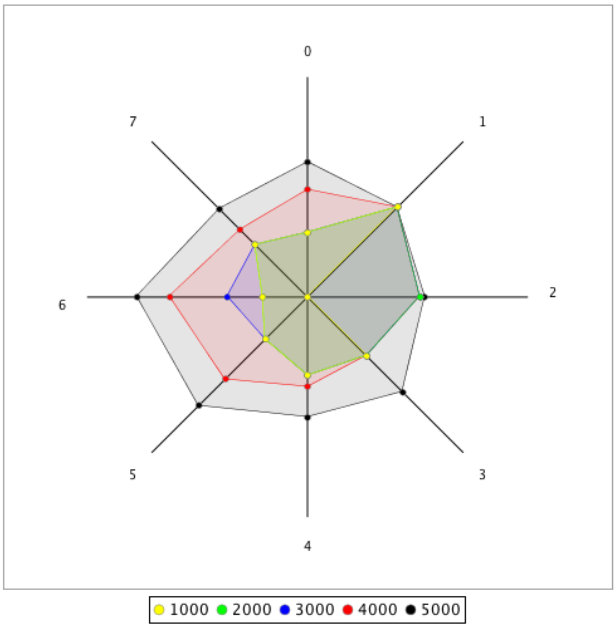
Figure 4.1: Example of kernel density estimation being used to determine a threshold

changes. An important question that will be answered here, then, is what is the best case prediction accuracy that can be realized using the threshold selection approach for each network instantiation? Answering this question for each instantiation will provide information that will be used to answer a second important question: are there any specific network properties that contribute strongly to the success or failure of prediction using the transfer entropy threshold selection prediction method? The answer to this question will not only help understand the prediction accuracy results presented in the remainder of the discussion section, but may also indicate which types of networks are most suitable for influence prediction using the methods proposed in this chapter.

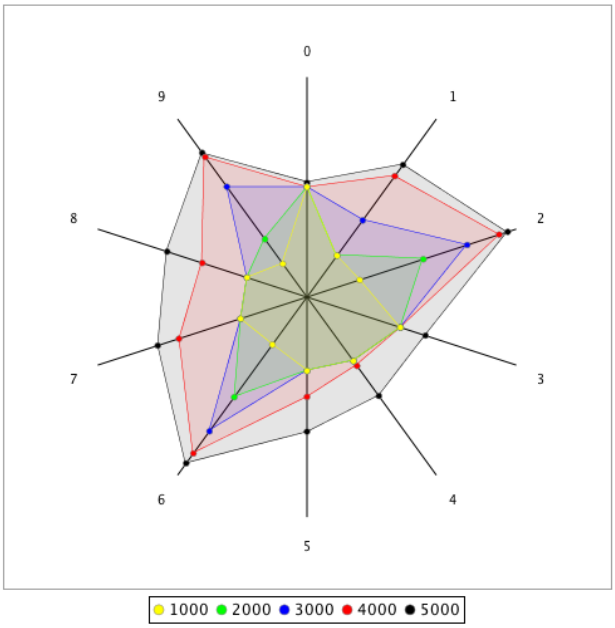
It is expected that, as the number of game rounds considered increases, the measured transfer entropy values will begin to converge toward their true values and, in turn, should increase the maximum attainable prediction accuracy using the threshold approach or any other prediction method. This hypothesis is supported by Figures 4.2 and 4.3, which show the maximal F-score attainable via the transfer entropy threshold approach, as the number of games rounds increases, for real-world and theoretical networks respectively. As the dataset provides ground truth network knowledge for each instantiation, these maximum F-score values are obtained by predicting networks for a given scenario at each possible threshold and using the known network to determine the F-score of each prediction. The threshold value that produces the highest F-score, then, represents the peak prediction

accuracy that can be achieved using the threshold prediction approach for that particular scenario. Within these two figures, each spoke represents an instantiation of that network type and the points on the spokes represent the maximum F-score at any of the 100 thresholds considered. The general trend in both of these figures seems to indicate that the maximum possible accuracy, as measured by the F-score, is monotonically increasing as the number of game rounds considered increases.

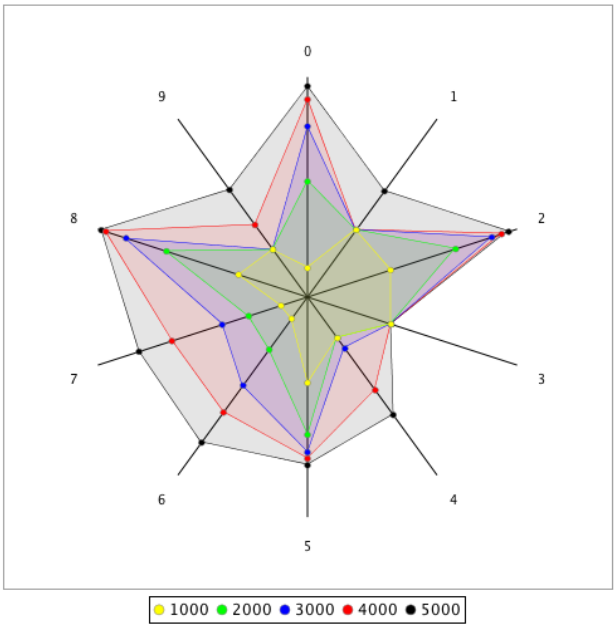
These two figures, and especially the real-world network results in Figure 4.2, demonstrate that there is noticeable variance across the different network instantiations in both the overall prediction accuracy and the rate at which the prediction accuracy increases relative to the number of game rounds. The theoretical networks (SF, Random, SW) tend to produce the possibility of high accuracy quickly, relative to the real-world networks. To investigate the cause of these accuracy differences, the networks were compared across game scenarios with 20 choices and 5000 rounds. These values were used because 20 game choices appeared to produce the most accurate predictions when given the necessary number of rounds to converge and the best case accuracy does not increase quickly enough beyond 5000 game rounds to warrant the additional computational complexity in computing transfer entropy values over a higher number of rounds. Network instantiations which resulted in a best possible F-score greater than or equal to 0.95 were considered successes and others were considered failures. Based on this division, the C4.5 decision tree construction algorithm (Quinlan, 1993), which is an extension of the earlier ID3 algorithm (Quinlan, 1986), was used to classify the successful networks based on the following network properties: average degree, clustering coefficient, diameter, average path length, and density. The results of this algorithm produced a simple classification, stating that networks with a density greater than 0.03 are expected to be unsuccessful, while those with a lower density are expected to be successful. Based on this classifier, 51 out of 54 successful networks and 33 out of 34 unsuccessful networks were identified correctly, for a weighted F-score of 0.955. Networks in which less than 3% of the total possible links exist, then, tend to have the possibility for high prediction accuracy after 5000 game rounds have been considered. This result intuitively makes sense when the method for calculating transfer entropy is considered. The transfer entropy value from an agent i to an agent j is increased when j's choice in the following round is the same as i's choice in the current round. When an agent has many neighbours, though, it is more likely that the agent's game choice will not be in line with several of its neighbours, which has a net effect of reducing the transfer entropy values from those neighbours. Further analysis could consider this relationship in more detail and could also consider more



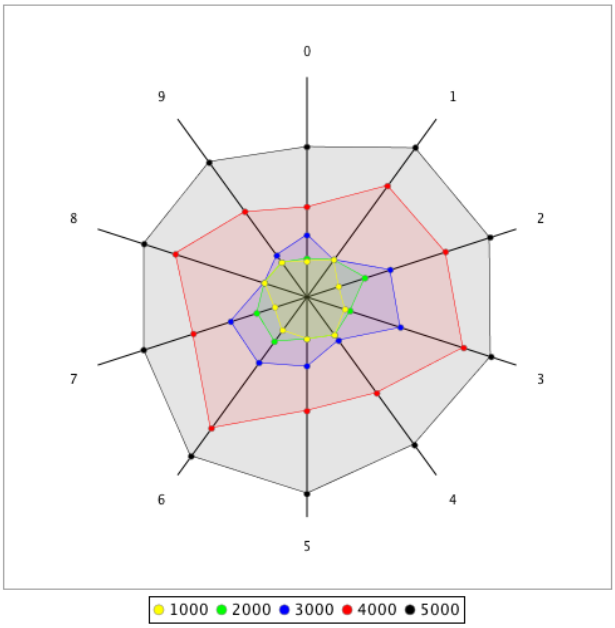
(a) Facebook



(b) Twitter

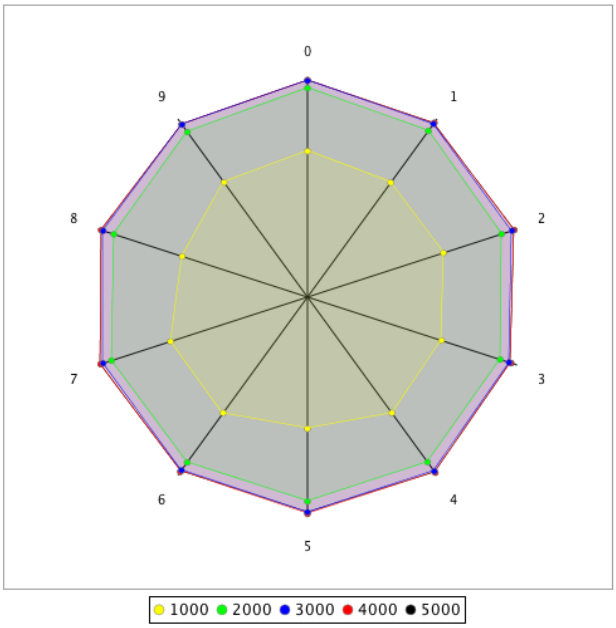


(c) G+

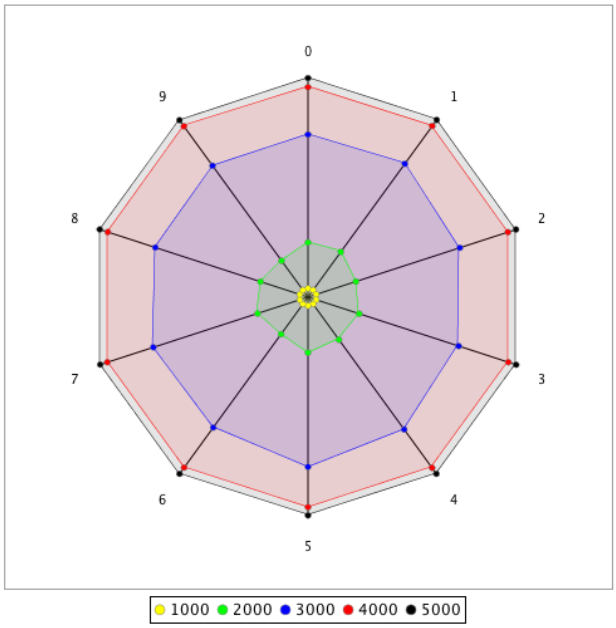


(d) G+Similar

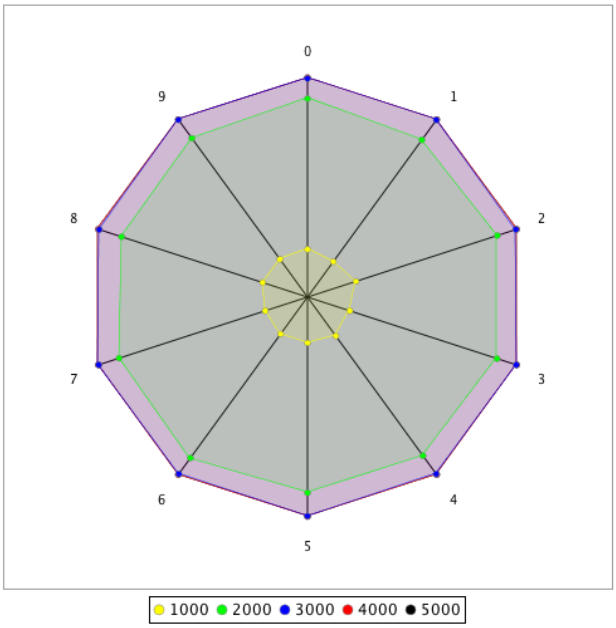
Figure 4.2: Best possible F-score that can be achieved using threshold approach, even with perfect network knowledge, on different real-world networks as number of game rounds increases – end of spokes represent a perfect F-score of 1



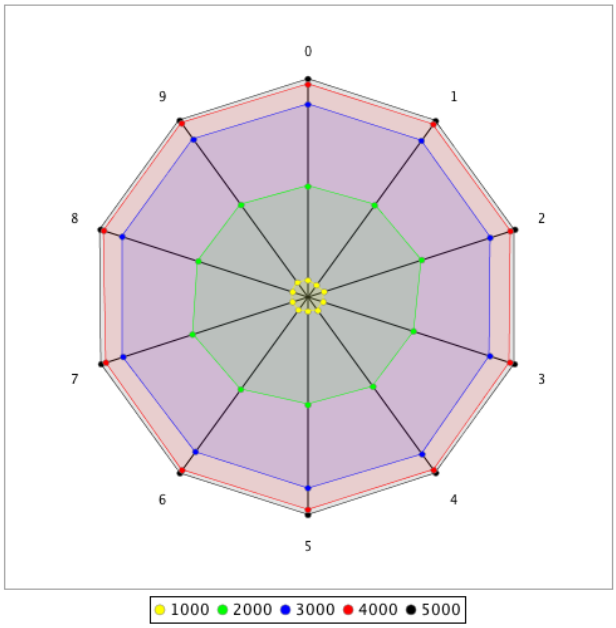
(a) SF



(b) Small-0.1



(c) Random-8



(d) Random-16

Figure 4.3: Best possible F-score that can be achieved using threshold approach, even with perfect network knowledge, on different theoretical networks as number of game rounds increases – end of spokes represent a perfect F-score of 1

advanced methods than the threshold approach, but these ideas are beyond the scope of this thesis.

Algorithmic Prediction Accuracy

This section will present the prediction accuracy results using each prediction method across the different networks types and instantiations. The previous analysis demonstrated that there was significant variance in the best case prediction accuracy for different network types, and even different instantiations of the same network type. When evaluating the predictive accuracy of the algorithmic approaches, it is undesirable to have the evaluation skewed by the limitations of the transfer entropy values and/or the threshold-based prediction approach. For this reason, these prediction accuracy results are presented relative to the best case accuracy values for each network instantiation. By comparing the prediction accuracy of the algorithmic approaches to these peak accuracy values, the variance between network instantiations should be reduced.

Figures 4.4-4.10 include radar plots comparing the prediction accuracy using each algorithmic prediction method to the best possible accuracy using the threshold approach for the various network types. As mentioned previously, these predictions were all made using data produced from anti-majority games using 20 choices and 5000 rounds. The following analysis will refer back to these results when discussing the performance of the different prediction algorithms and, ultimately, a final ranking of the overall performance of all of the proposed solutions will be presented.

While Figures 4.4-4.10 allow for easy comparison of the prediction algorithms across each specific network type and that type's associated instantiations, it is difficult to compare the overall performance of the algorithms in this format. Table 4.4 shows the mean and standard deviation of the relative prediction accuracy values for each prediction algorithm, calculated over all network instantiations considered. This table also includes information for different sample sizes considered for the sample-based prediction methods. The 1% sample size for degree distribution prediction is excluded, as this would involve histograms computed over a single value (equivalent to 1% degree sampling). This table can be viewed as a summary of the algorithms' overall performance across all network types.

Assumed Degree/Diameter

The first prediction methods that will be discussed are those using assumed network property information. When the target degree of the network is known, the prediction accuracy is nearly

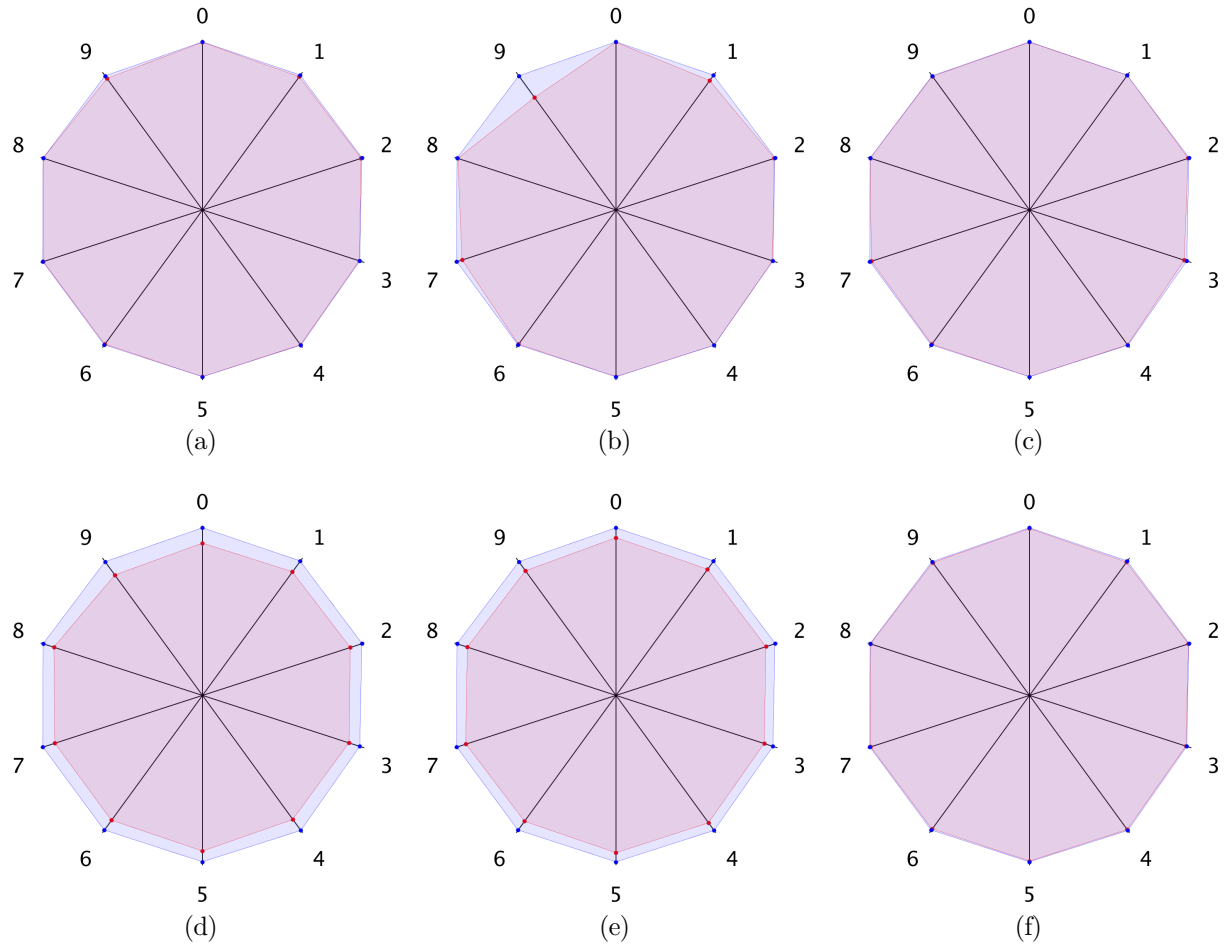


Figure 4.4: Prediction accuracy (red) relative to the best threshold accuracy (blue) for SF networks using either a) Assumed Degree, b) Assumed Diameter, c) Kernel Density Estimation, d) Sampled Degree, e) Sampled Degree Distribution, f) Sampled Edges

identical to the best possible accuracy across almost all network instantiations, as evidenced by the 98.8% average relative success for this method in Table 4.4. The use of diameter, in comparison, has much larger variance in accuracy (18.4% overall standard deviation), especially on the Facebook and G+ networks. A likely explanation of this accuracy difference is the difference in sensitivity to changes in the threshold of the average degree and diameter measures. A slight change in threshold will generally cause a small, but measurable change in average degree, allowing the prediction to be made at a single threshold that minimizes the error. The diameter of the network, though, can remain stable across a wider range of threshold values, as adding/removing links does not necessarily change the network diameter. Diameter-based prediction, then, must select one of

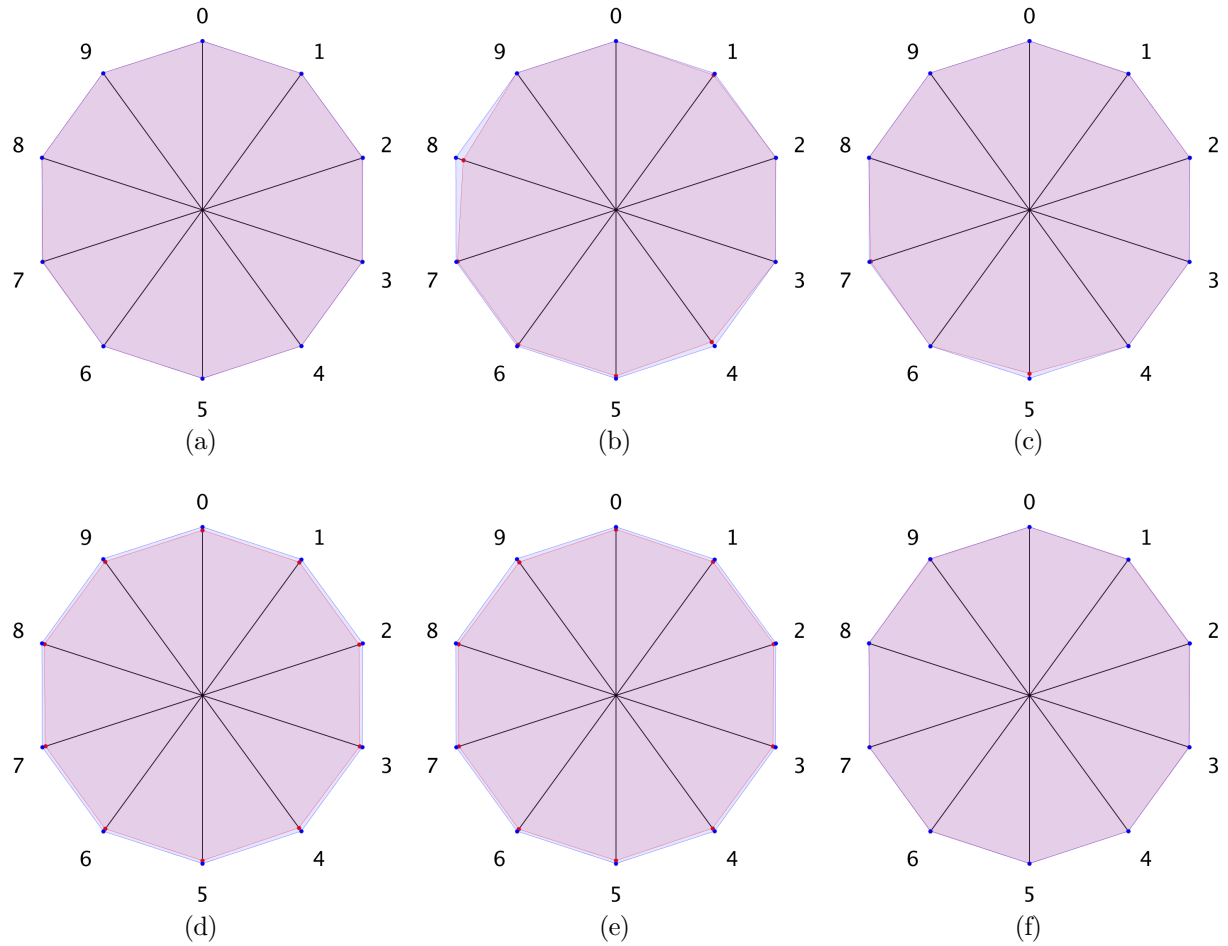


Figure 4.5: Prediction accuracy (red) relative to the best threshold accuracy (blue) for Random-16 networks using either a) Assumed Degree, b) Assumed Diameter, c) Kernel Density Estimation, d) Sampled Degree, e) Sampled Degree Distribution, f) Sampled Edges

several possible thresholds that minimize the error. As described in Section 4.4.2 and Algorithm 9, the final choice is made by selecting the median threshold from the set of error minimizing values. It is possible that the accuracy of diameter-based prediction could be improved by combining it with other measures, but this is left as future work.

Sampled Network

While the assumed degree method discussed above produced prediction results that were near optimal, it requires fairly strong assumptions about available network knowledge. To improve on this, prediction methods that generated estimates based on sampling data from the network were

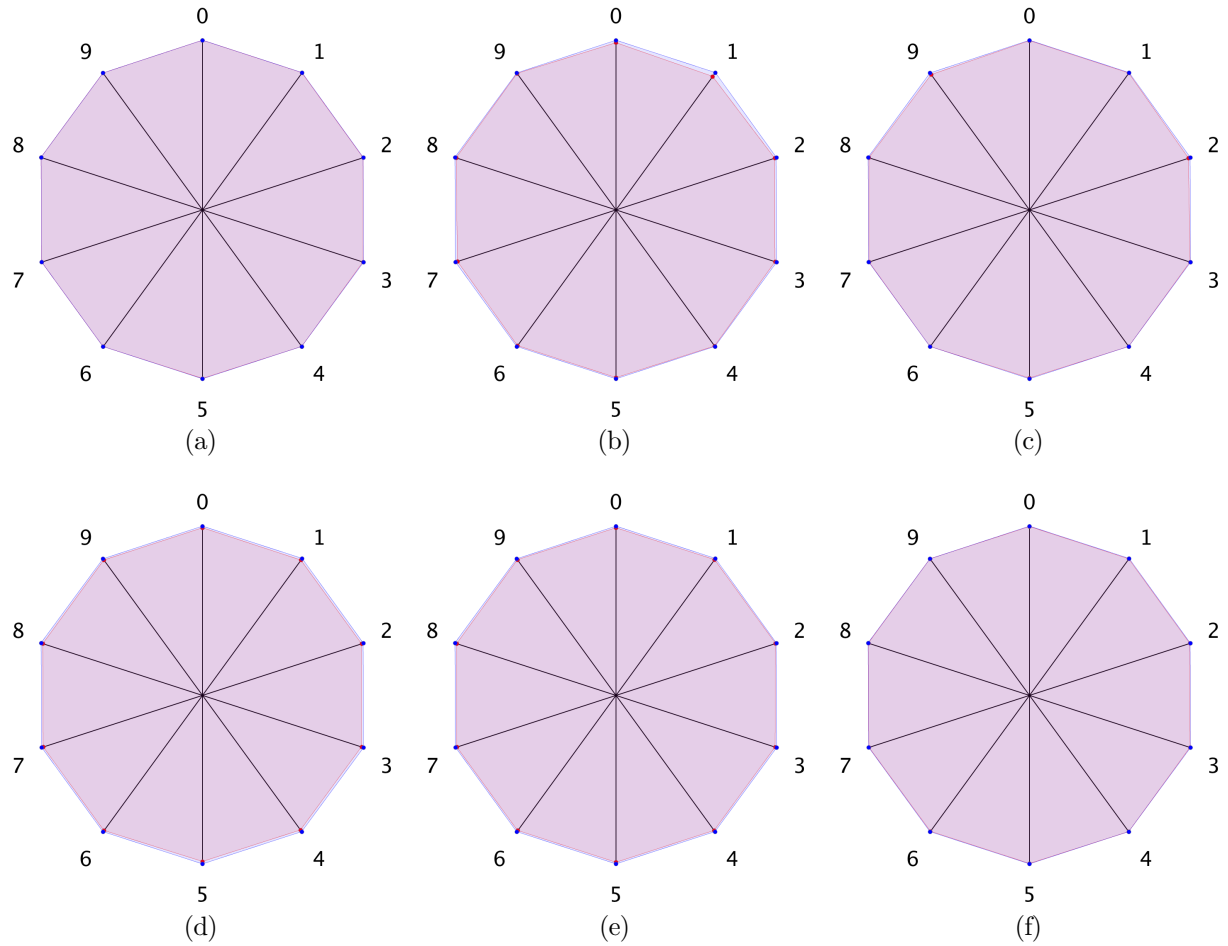


Figure 4.6: Prediction accuracy (red) relative to the best threshold accuracy (blue) for Small-0.2 networks using either a) Assumed Degree, b) Assumed Diameter, c) Kernel Density Estimation, d) Sampled Degree, e) Sampled Degree Distribution, f) Sampled Edges

investigated. Instead of assuming some network parameter value is known, a sample of the network is used to inform the prediction algorithm. Overall, the three sampling methods investigated performed well, often coming very close to optimal prediction accuracy. Of the three methods, edge sampling appears to produce near optimal accuracy on the most consistent basis, producing an F-score equal to 99.0% of the best possible F-score on average. The difference between edge-based sampling and the degree-based sampling methods are most obvious within the G+ and Twitter network instantiations, where the accuracy of the sampled average degree and degree distribution methods is lower for several network instantiations. The robustness of the sampling techniques in the presence of outliers may explain why edge sampling produces better prediction accuracy overall.

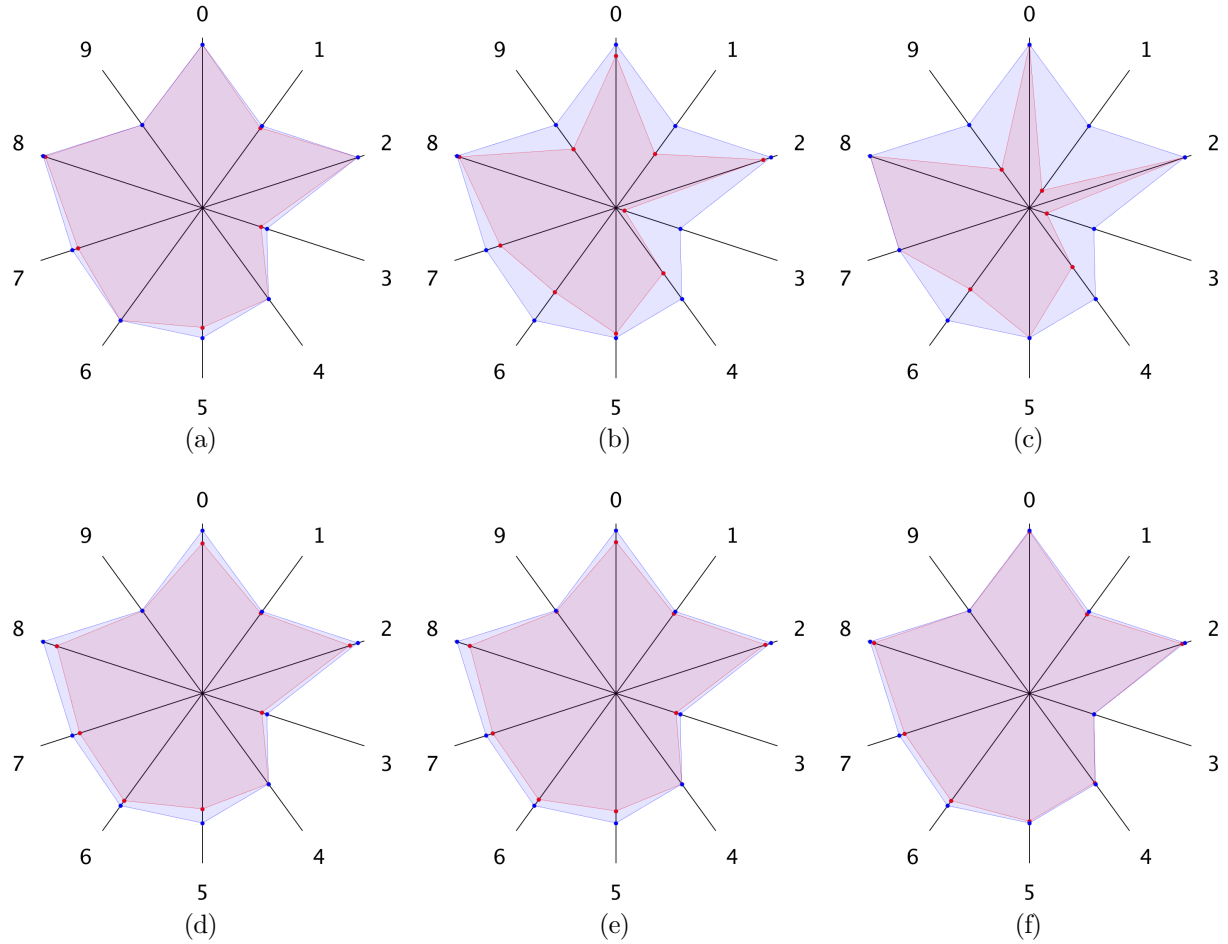


Figure 4.7: Prediction accuracy (red) relative to the best threshold accuracy (blue) for G+ networks using either a) Assumed Degree, b) Assumed Diameter, c) Kernel Density Estimation, d) Sampled Degree, e) Sampled Degree Distribution, f) Sampled Edges

When considering an average degree estimated using a small sample, an extreme outlier could significantly affect the accuracy of the sample measurement. When using edge sampling, however, each observation is weighted equally when it is categorized as a true/false positive/negative. So, even if one node's information is severely skewed, the magnitude of the error in that node's information does not adversely affect the sample accuracy to the same extent that it would with average degree computation. In addition to this, edge sampling also provides n measurements per sampled node, where n is the number of nodes in the network. This may also improve the robustness of the measurements by increasing the size of the data considered, even though the node sample size remains the same.

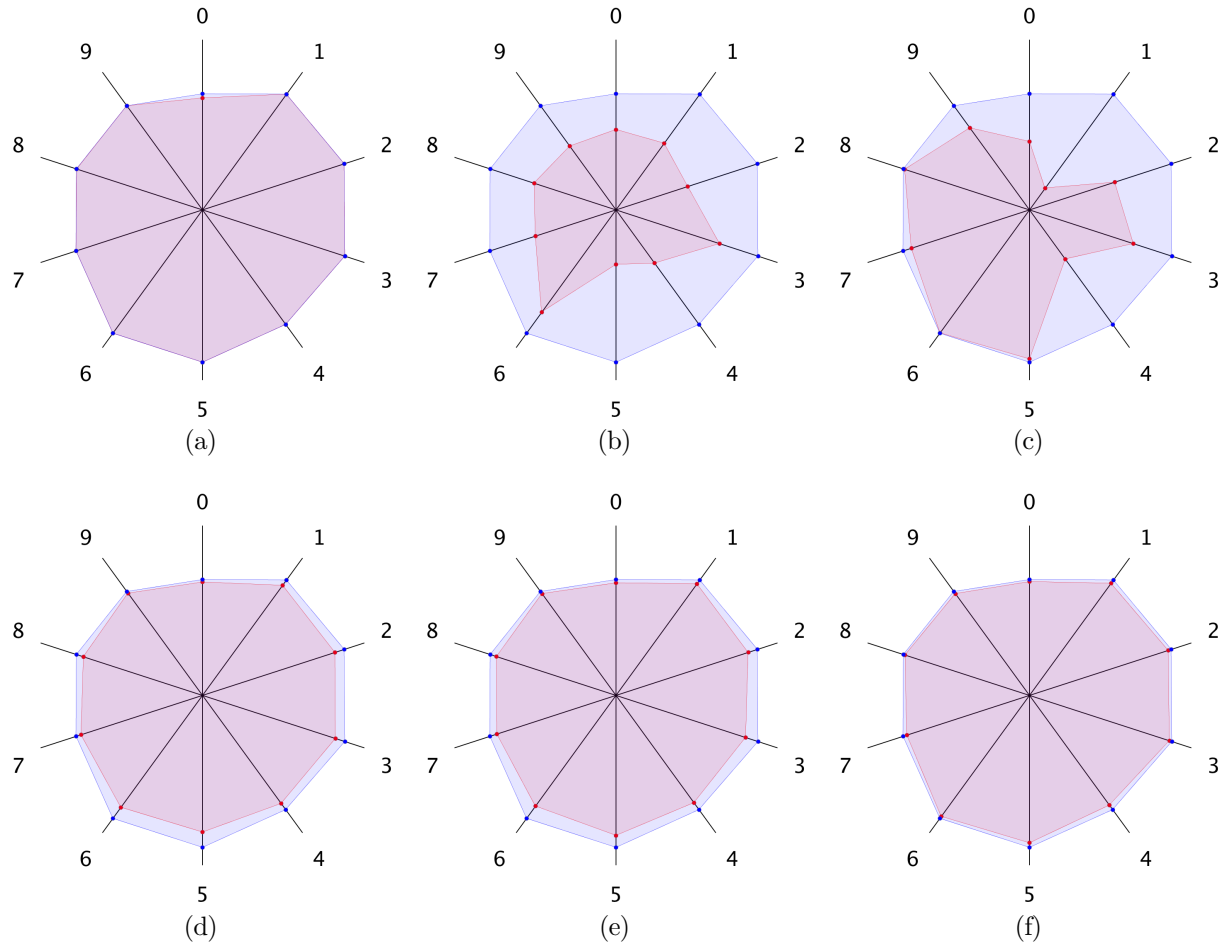


Figure 4.8: Prediction accuracy (red) relative to the best threshold accuracy (blue) for G+Similar networks using either a) Assumed Degree, b) Assumed Diameter, c) Kernel Density Estimation, d) Sampled Degree, e) Sampled Degree Distribution, f) Sampled Edges

Kernel Density Estimate

The use of kernel density estimation is the first and only link prediction method investigated here that requires absolutely no knowledge of any network or node properties. Instead, the KDE prediction method operates only on the transfer entropy values and can be seen as predicting a true influence network that is completely unknown a priori. Across all of the theoretical network instantiations, the KDE prediction method produces near optimal accuracy levels. The accuracy of KDE across the real networks, however, is much less consistent. This inconsistency is responsible for the high standard deviation and, at least relative to the other prediction methods, low average accuracy of predictions using the KDE method. Many of the scenarios in which KDE produces

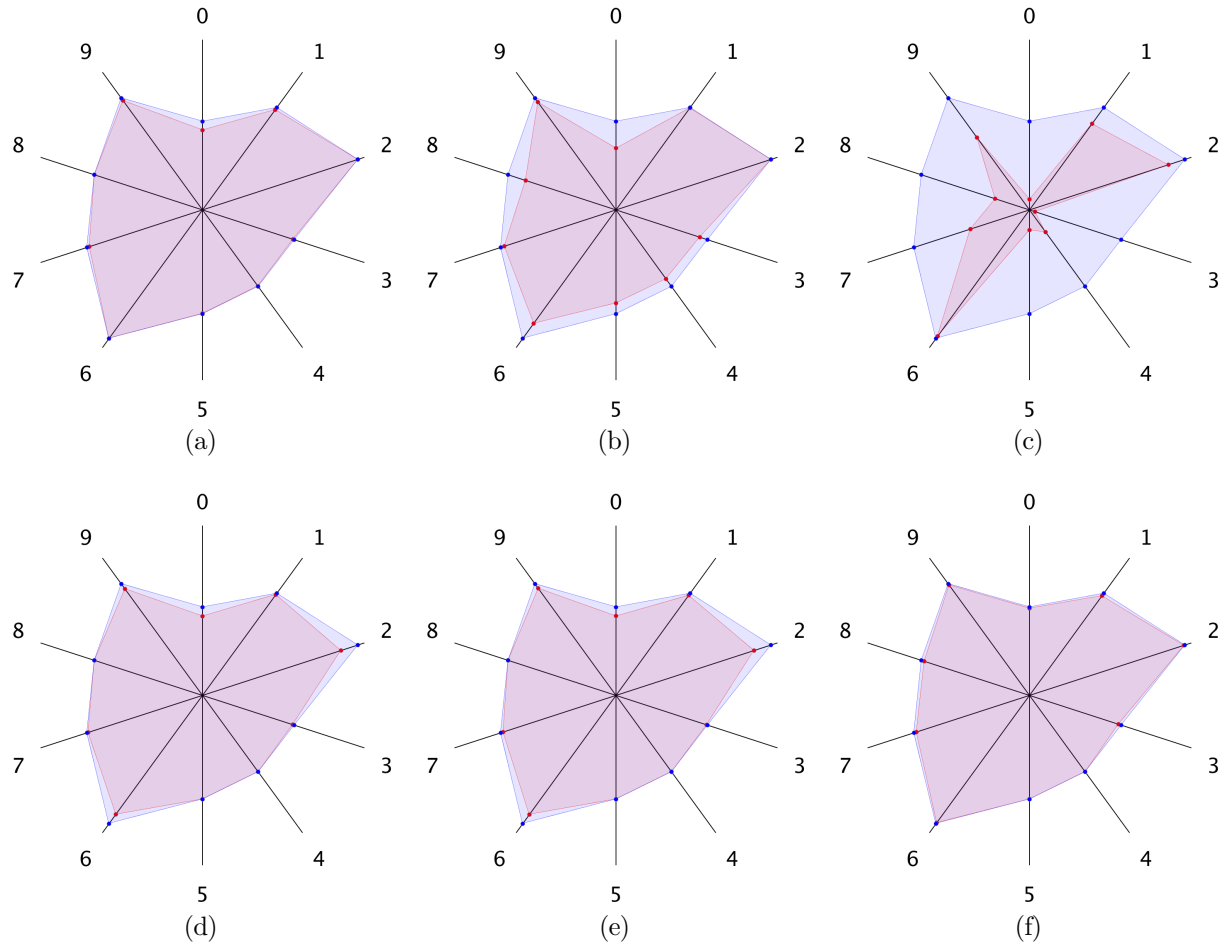


Figure 4.9: Prediction accuracy (red) relative to the best threshold accuracy (blue) for Twitter networks using either a) Assumed Degree, b) Assumed Diameter, c) Kernel Density Estimation, d) Sampled Degree, e) Sampled Degree Distribution, f) Sampled Edges

the lowest accuracy appear to also be scenarios in which the optimal accuracy values are also low. It is possible that the KDE method, which attempts to descend a smooth slope, may struggle in scenarios where the 'peak' is not as well defined as it may be in other cases. While not considered here, it may be possible to improve on the basic KDE method by implementing additional smoothing, considering more complex strategies for finding local minima/maxima, or combining the value estimated via KDE with other possible methods discussed in this chapter.

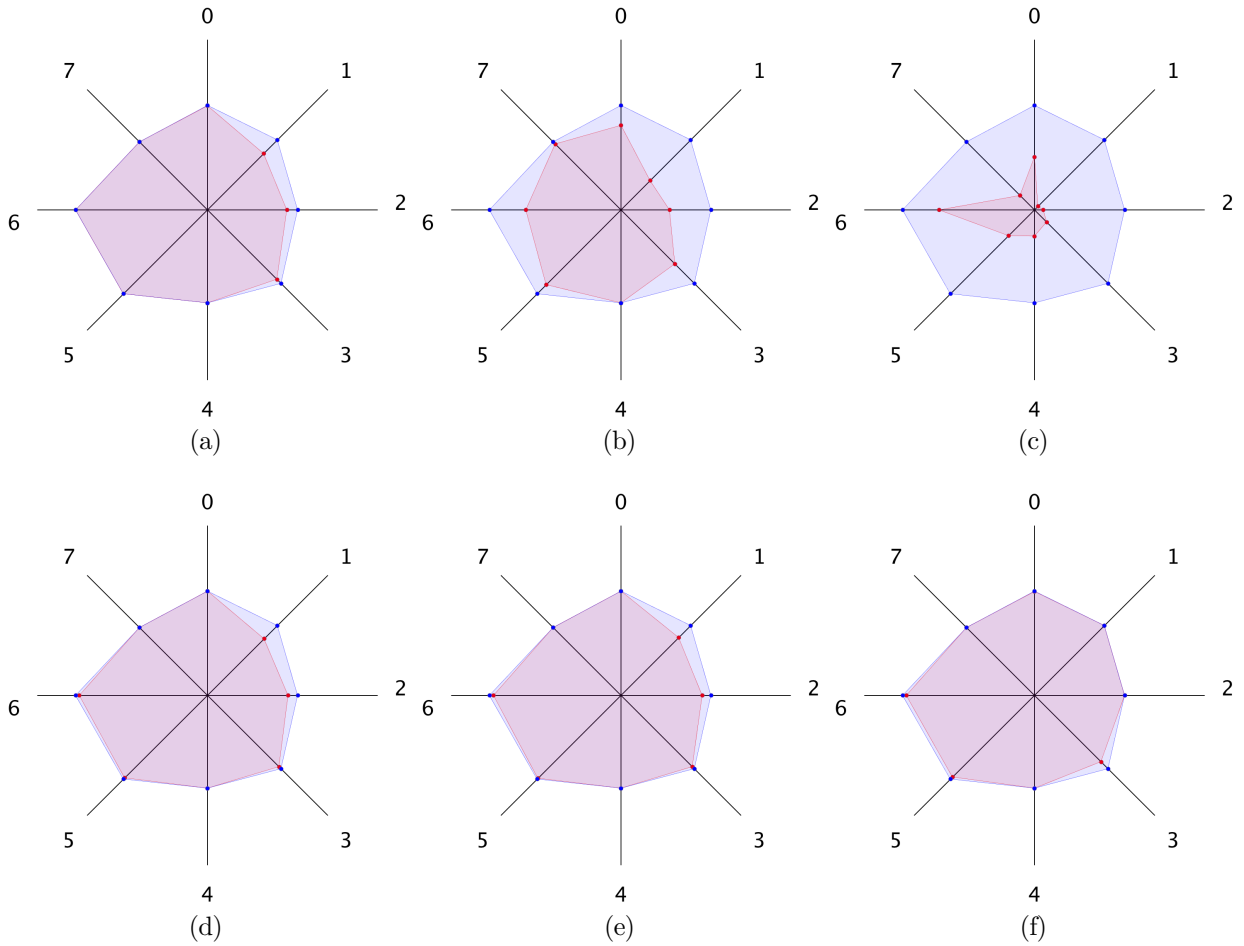


Figure 4.10: Prediction accuracy (red) relative to the best threshold accuracy (blue) for Facebook networks using either a) Assumed Degree, b) Assumed Diameter, c) Kernel Density Estimation, d) Sampled Degree, e) Sampled Degree Distribution, f) Sampled Edges

Overall Algorithm Ranking

The previous analysis compared the overall accuracy of each prediction method separately and explained what factors most strongly contributed to the experimental results. This section will use statistical significance testing to produce an overall ranking of all of the proposed algorithmic prediction approaches. The exact statistical significance test used in this case was a paired T-test, with an α value of 0.05, calculated using the prediction accuracy over all network instantiations. Table 4.5 provides a summary of all of the algorithms, where an X within a cell represents that the algorithm specified in the cell's row produced a statistically significant increase in prediction accuracy when compared to the algorithm in the cell's column.

Table 4.4: Mean and standard deviation of prediction accuracy relative to the maximum attainable prediction accuracy for each algorithmic prediction method

Algorithm	**Mean**	**Standard Deviation**
Assumed Degree	98.8%	3.11%
Assumed Diameter	87.7%	18.4%
Sampled Degree (1% of nodes)	90.7%	7.08%
Sampled Degree (5% of nodes)	95.9%	3.61%
Sampled Degree (10% of nodes)	97.0%	3.04%
Sampled Degree Distribution (5%)	96.3%	3.23%
Sampled Degree Distribution (10%)	97.5%	2.83%
Sampled Edges (1%)	97.3%	3.44%
Sampled Edges (5%)	99.0%	1.49%
Sampled Edges (10%)	98.7%	2.42%
Kernel Density Estimate	79.0%	31.5%

Table 4.5: Statistically significant difference in prediction accuracy using different prediction algorithms – an X represents that a paired T-test ($\alpha = 0.05$) found a statistically significant increase in prediction accuracy when using the row's algorithm when compared to the column's algorithm

	Significantly Different Prediction Accuracy					
	Assumed Degree	**Assumed Diameter**	**Sampled Degree**	**Sampled Distribution**	**Sampled Edges**	**KDE**
Assumed Degree	—	X	X	X		X
Assumed Diameter		—				X
Sampled Degree		X	—			X
Sampled Distribution		X	X	—		X
Sampled Edges		X	X	X	—	X
KDE						—

The most important result that can be extracted from Table 4.5 is that both assumed degree prediction and sampled edge prediction produce statistically significant increases in accuracy when compared to the four other methods. At the same time, neither of these two prediction methods are found to produce higher overall accuracy than the other. Based on the information requirements of these two prediction methods, the edge-based sampling method could be selected as the best predictor overall. This is because it is capable of producing similar accuracy results

using information from only 5% (or even 1%) of the network's nodes, as the assumed degree approach produces with the exact average degree measure calculated over the entire network. When considering only the overall prediction accuracy, these results would rank the KDE prediction approach as the approach that performed least favourably. At the same time, though, the KDE approach is the only prediction method that operated with absolutely no network knowledge. So this approach would still have merit in applications where little is known about the desired network. Furthermore, as mentioned in Section 4.4.2, it may be possible for future work to improve on the KDE method and possibly increase the prediction accuracy to rival the other methods.

4.5 Summary

This chapter investigated the use of transfer entropy measurements to solve problems related to quantifying and predicting influence within networks. The chapter defined the anti-majority game (Section 4.2) and fabricated dataset (Section 4.2.5) that were used to perform in-depth analysis of these two problems. A method for measuring both local and global agent influence using transfer entropy analysis was proposed (Section 4.3) and it was demonstrated that these measures were strongly correlated with both the PageRank and an influence measure based on the mathematical specification of the anti-majority game (Section 4.3.3). Following this influence measurement analysis, the influence link inference problem was defined (Section 4.4.1) and algorithmic solutions to this problem were proposed (Section 4.4.2). Analysis of the described prediction algorithms in Section 4.4.3 demonstrated that in many networks, and especially those with a low density, a high level of prediction accuracy was achievable using transfer entropy values, with the average best case prediction F-score over all network instantiations being 0.87. Comparisons of the different prediction algorithms investigated found that the two most successful prediction algorithms used the known average degree of the network and sampled edge information from the network. The sample-based prediction algorithm, however, requires significantly less network knowledge to perform prediction, making it the preferred solution in any practical application where exact network information is not available. While the kernel density estimate prediction method produced the lowest overall prediction accuracy, it operates without any network knowledge and was still within 79% of the best case accuracy on average.

There are two main conclusions that can be drawn from this chapter. First, when enough action data is available, transfer entropy measurements can be used to measure an agent's influence on other agents or the network as a whole. Second, these values can also be used to predict which

links are truly influential within a network with a relatively high level of accuracy in many cases. The next chapter will investigate the problem of distribution-based network control, and a large part of the discussion presented in that chapter will involve the role of control node influence in determining the success/failure of a network control solution. Specifically, it is argued that when attempting network control, influence measurements should be included when selecting a control node set. The work that has been presented in this chapter, then, could be used to further improve the selection of a control node set in practical applications by allowing for more accurate, behaviour-guided, influence measurement.

Chapter 5

Distribution-based Network Control

5.1 Introduction

The discussion in Chapter 3 analyzed the existing signal injection network control research, most of which has involved the analysis of networks from a structural control viewpoint (e.g., Liu et al., 2011b; Ruths and Ruths, 2014). The main criticisms of the structural control framework presented in Chapter 3 identified full state controllability as a common network control goal. The work related to full state controllability, however, has not considered the actual selection of control signals to achieve successful control. Acknowledging these shortcomings, the Network Control Problem (NCP) was proposed by Runka (2016), who formalized a type of control problem that uses a utility function to determine a controller's success. The use of a utility function allows for a richer specification of the goals of a controller when compared to the full state controllability goal of structural network analysis research. In addition to this, the NCP stresses the importance of both the controller configuration and the selection of control signals within a control problem, which represents a significant step toward practical network controller development.

As the structural control framework uses a linear time-invariant system model and investigates full state system controllability, the system state is represented by a state vector. In practical applications, however, those wishing to achieve control of a system may be more interested in the overall state properties of the system, as opposed to reaching a single specific state in the vector space. This may be especially true when considering control of social networks, where problems such as the avoidance of flocking or opinion consensus are more concerned with overall properties of the system than any specific state vector. The existing NCP research made some progress in this sense through the use of an aggregate measure of the system's state to determine a controller's effectiveness. However, this chapter will propose a different approach to network control using the idea of state distributions as control goals. Arguably, using a distribution allows for a more expressive and general form of defining (un)desirable system properties than is possible with a single vector or the simple vector aggregations used in the existing NCP research.

In addition to the definition of distribution-based network control, this chapter will also further investigate controller configuration algorithms within the proposed distribution-based control problem. The work of Runka proposed the FAR heuristic, described in Section 5.3.3, and found that it outperformed all of the other heuristics investigated when overall control success was considered, including an algorithm based on the structural network control framework. A question that has not been investigated in detail within the existing work, however, is why this heuristic performed so well and how that may inform future control selection algorithm development. The later sections of this chapter will analyze the success of controllers across a wide range of scenarios and identify possible factors that most strongly determine a controller's success. Based on the conclusions of this analysis, improvements on the existing FAR heuristic are proposed and the performance of these algorithms is compared. Ultimately, the results presented in this chapter argue two important points:

- Influence Dynamics - The influence dynamics within a networked system play an important role in that system's behaviour and, therefore, should be considered as an important factor when selecting a set of control nodes.
- Optimization - While heuristic approaches such as FAR allow for rapid generation of control node sets, the results presented here will demonstrate that performing more computationally complex optimizations can allow for significantly improved controller performance.

5.2 Distribution-based Control System

There are several mandatory and optional components involved in formulating a distribution-based control system. Figure 5.1 shows the basic components and information flow present in a distribution-based control system. A list of these components and a short description of each are included below. It should be noted that these components are defined to match the distribution maintenance type of problem used within this thesis. Other components may be necessary to achieve other types of distribution-based control, such as those outlined in Section 5.2.2.

- Network (G) - The network/graph connecting entities within the system.
- Sensor Nodes (N_S) - A set of nodes within the network which provide input regarding the network state to the controller. Within this work, it is assumed that the state of all nodes within the system is observable, though this is unlikely to be the case in a real-world

application. From a practical viewpoint, one possible benefit of using distributions to represent system state is that the parameters of the system's state distribution can be estimated based on data sampled from the sensor nodes. When using a vector-based state representation, this estimation would likely be significantly more difficult without some knowledge regarding relationships between nodes in the network (i.e., which nodes often share similar state values).

- Control Nodes (N_C) - A set of control nodes within the network, the state of which can be set at each time step. This set of nodes represents the interface which is used by the control system to affect the network state.

- Target Distribution (D_T) - The defined ideal distribution of the system. In a distribution maintenance problem, the control system attempts to keep the system's state distribution close to this target. Other types of distribution-based control problems could also be defined, such as those that allow the distribution to move from a target distribution as long as the velocity of that movement is below a threshold (i.e., a distribution drift problem). Within this work, however, the focus is on preventing the distance between the state and target distributions from exceeding some threshold. Additionally, it is assumed within this work that the target distribution remains the same throughout the learning and control evaluation process.

- State Distribution (D_S) - A measure of the current distribution of the state, composed of the state value, $s(v,t)$, of each sensor node within the system. Both the state and target distributions can be represented by either a parametrized distribution (e.g., a normal distribution with specific mean and variance) or discretized to form a histogram.

- Rate of Change (ROC) Analysis, *optional* - In the case of parameterized distributions, it is also possible to estimate the rate of change of the state distribution parameters over time using techniques such as alpha-beta (Brookner, 1998) or Kalman (Zarchan and Musoff, 2005) filtering. This estimate can allow the 'velocity' of the system to be quantified, which could improve the performance of a control system by producing more accurate prediction of the future state of the system.

- Controller (CON) - The controller is responsible for taking the state information as input and producing the control signals for each of the control nodes within the network as output.

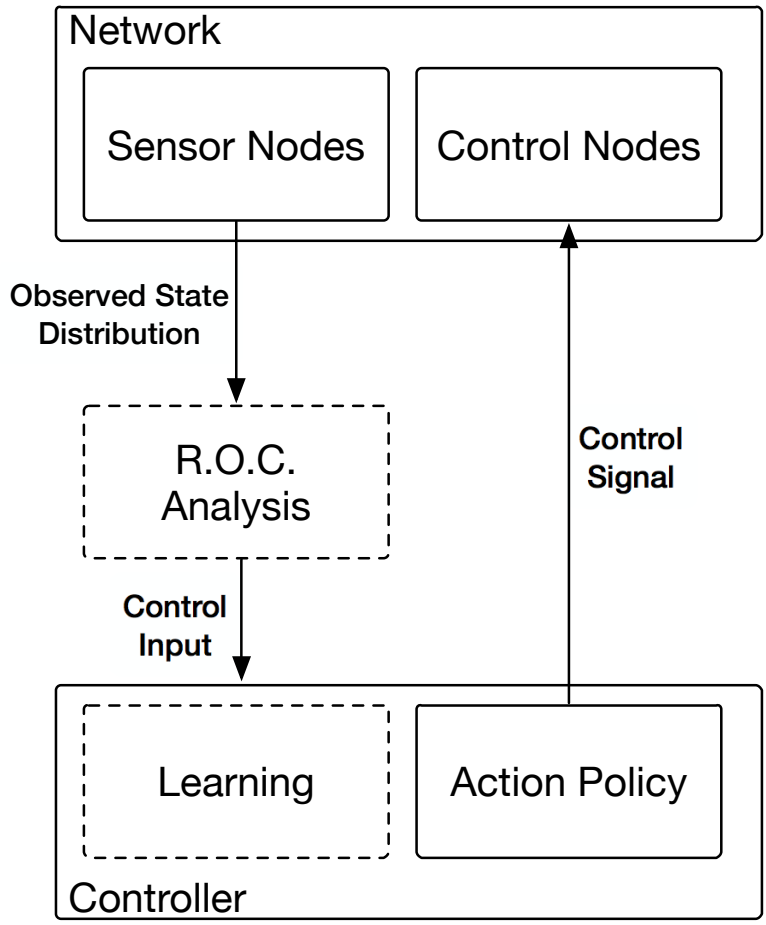


Figure 5.1: General components and information flow within a distribution-based control system

Within this work, reinforcement learning (see Sutton and Barto, 1998, for a thorough introduction) is used to generate a policy of signal selection based on the state distribution parameters. More details regarding the process used for learning the control policy are included in Section 5.3. Neural controllers have also been used in previous network control research (e.g., Runka, 2016) and the definition of a distribution-based control system presented here does not preclude the use of neural controllers or any other type of learning mechanism in the future.

5.2.1 Comparing Distributions

As with control systems that rely on the use of state vectors, a method for distribution comparison is required to achieve distribution-based control. Distribution comparison is necessary for a number

of reasons, such as determining if the controller has failed (i.e., in failure avoidance or distribution maintenance problems, see Section 5.2.2), determining a system's velocity by comparing distributions at different time points, or determining how far away the system is from some other state. While there are a number of methods for comparing distributions, such as the Kullback-Leibler divergence and the total variation distance, the Hellinger distance is used here. One of the main reasons for this is that the Hellinger distance is easily calculated on both continuous and discrete distribution types. Additionally, and most importantly, the Hellinger distance is bounded within the range [0,1] and fulfills the properties of a metric.

In the continuous case, the Hellinger distance between two probability measures PM_1 and PM_2 can be calculated as in Equation 5.1, where f and g are the probability density functions of PM_1 and PM_2 respectively.

$$\begin{aligned} H^2(PM_1,PM_2) &= \frac{1}{2}\int\left(\sqrt{f(x)}-\sqrt{g(x)}\right)^2 dx \\ H^2(PM_1,PM_2) &= 1-\int\sqrt{f(x)g(x)}dx \\ H(PM_1,PM_2) &= \sqrt{1-\int\sqrt{f(x)g(x)}dx} \end{aligned} \tag{5.1}$$

For two discrete probability distributions PM_1 and PM_2 defined over a common domain k, the Hellinger distance can be computed using Equation 5.2. Within this work, it is assumed that the domain k does not change over time, although this definition does not preclude changes in domain.

$$H(P,Q) = \frac{1}{\sqrt{2}}\sqrt{\sum_{i=1}^{k}\left(\sqrt{p_i}-\sqrt{q_i}\right)^2} \tag{5.2}$$

5.2.2 Distribution-based Control Problems

Like the original NCP work of Runka, this thesis considers a single type of failure avoidance problem. Using the idea of a utility function from the NCP, though, and including the idea of distributions, there are a number of different problem types that could be defined. The list below includes several of these types and discusses how they may be formulated from a distribution-based control perspective.

- Distribution Maintenance: A failure avoidance type of problem in which the controller attempts to maintain the state distribution such that it does not drift too far away from a target distribution. This is the type of problem investigated within this thesis, where a

target distribution of $\mathcal{N}(0.0,0.05)$ is used to represent an opinion control problem in which the desire is to maintain a distribution balanced around neutral. In this case, the distance between any current state distribution and the target distribution can be calculated and compared to a threshold value to determine if the controller has failed. This type of problem is best suited for scenarios in which the system has a known 'ideal' distribution. Beyond the more abstract opinion control problem investigated here, this problem type could, for example, be applied to Internet chatbots. In this case, a target distribution may be applied to messages received by a chatbot, preventing it from being manipulated through spam messages of a particular opinion (e.g., the failure of Microsoft's Tay chatbot, Neff and Nagy, 2016). In a similar fashion, this type of control problem could also be used for the regulation of the distribution of sentiment within a system.

- Limiting Change: In other cases, it may be more important to limit the rate of change of the state distribution over time. For example, rapid changes in the distribution of opinions in the stock market could shock the system and lead to panic. In this case, the distance between state distributions at any two time points can be compared to ensure the rate of change is within acceptable bounds. This idea is captured in Equation 5.3, which computes the average Hellinger distance over some interval of time i and compares it to a threshold value θ. The system could be said to be in a state of control failure, then, any time the average rate of change exceeds the specified threshold value. Financial or market-based domains are possible areas in which this type of control may be applied.

$$\frac{H(X_t,X_{t+i})}{i} \leq \theta \tag{5.3}$$

- Distribution Movement: In some cases, the goal may be to move the system from the current distribution to another. Problems of this type could define utility functions based on the time taken to move between distribution or the ability of the system to stay 'close' to the current/next distribution. This may also be posed as a multi-step process in which the system must move through or close to a series of distributions. This could be applied, for example, within agent-based conversation systems where the desire is to move conversations across specific topic or sentiment distributions over time.

In addition to these new problem types, it is possible to restate existing control problems, such as the $\theta-CAP$ proposed in the original NCP work of Runka, using a distribution-based control

Table 5.1: Relationship between fraction of 0/1 state nodes and the Hellinger distance from a uniform discrete distribution over the set {0,1}

$\|V^1\|$	$\|V^0\|$	$\frac{\|\|V^1\|-\|V^0\|\|}{\|V\|}$	$H(V^1,V^0)$
0	100	1.0	0.54
10	90	0.8	0.32
20	80	0.6	0.23
30	70	0.4	0.15
40	60	0.2	0.07
50	50	0.0	0.0
60	40	0.2	0.07
70	30	0.4	0.15
80	20	0.6	0.23
90	10	0.8	0.32
100	0	1.0	0.54

approach. For example, consider the values in Table 5.1, which includes data points showing the number of 0/1 nodes ($|V^0|$ and $|V^1|$), the proportion value that would have been used in the original $\theta-CAP$ definition, and the associated Hellinger distance between the discrete distribution of 0/1 values and a uniform distribution over the set {0,1}. In this case, the Hellinger distance can be substituted for the proportion value, and an equivalent threshold value can be selected. If the original control goal was to ensure that less than 60% of nodes had either possible value at one time, a Hellinger threshold of 0.07 could be used.

5.2.3 Controllability Analysis

One of the most significant contributions of the existing structural network control research is the development of methods for analysing the controllability of a system based on the structural properties of the network. However, it has been previously demonstrated that this structural analysis does not necessarily produce accurate results when applied to a practical control problem, such as those considered here (Runka, 2016). But if we consider a model that estimates a system's behaviour, we can produce some rudimentary analysis on system controllability within a distribution-based control system. By repeatedly simulating the model from a starting state, or many starting states, and measuring the distance between subsequent states at different time steps, we can produce an estimate of the distribution of distance values between the initial and resulting states for a time interval. This distribution can be used as a measure of the speed with which the system tends to change.

For example, consider Figure 5.2, which shows the distribution of Hellinger distance values over a single time step within a real-valued voter model simulation where no control is applied. Within this example system, prolonged control with a Hellinger threshold of less than 0.03 cannot be expected. In fact, in almost 10% of cases, this threshold is exceeded in a single step, giving the controller no opportunity for control. Controlling the system to a threshold of 0.1 or 0.05, however, may be possible depending on how consistently the Hellinger distance moves over multiple steps and how effective the control system is in changing the state distribution. However, it is still necessary to create a method that can accurately measure the ability of a controller to modify the state distribution. While this is not explored in depth within this thesis, it would appear that the influence of the control nodes would play an important role. Essentially, the control nodes need to be capable of exerting enough force on the system at each time step to counteract the natural tendency of the system to move away from the target distribution. This type of analysis is related to some of the existing structural control theory research that deals with analysis of control node sets (Jia and Barabási, 2013; Ruths and Ruths, 2014; Campbell et al., 2015). For example, the results of Ruths and Ruths (2014) could be viewed as identifying the importance of long 'stems', which represent a chain of nodes that can be controlled (i.e., influenced) by a single control node, in minimizing the number of control nodes required in the system. The results discussed in Section 5.4 address the issue of controller influence in relation to control success in more detail. In addition to this, Section 5.5 provides a comparison between control set selection algorithms that either acknowledge or ignore the underlying influence model to make decisions.

5.3 Learning Control of the Real-Valued Voter Model

As described in Section 2.5, the real-valued voter model that is used here allows nodes to take on a continuous value in the range of $[-1.0,1.0]$. To simplify learning, the states that the control nodes can be set to are limited to a discrete set consisting of values between -0.5 and 0.5 in 0.05 increments, for a total of 21 possible values. These values were selected to minimize the size of the search space while still providing a suitable set of actions to move the system in any direction deemed to be desirable. The real-valued voter model requires agents to move their state toward a neighbour's state. As the target distribution used within this work is a normal distribution with a mean of 0 and standard deviation of 0.05, it was deemed unnecessary to include values beyond ten standard deviations from the mean, since it is unlikely that a control node would need to use a value outside of this range to move all of its neighbours in a specific direction. Rinforcement learning

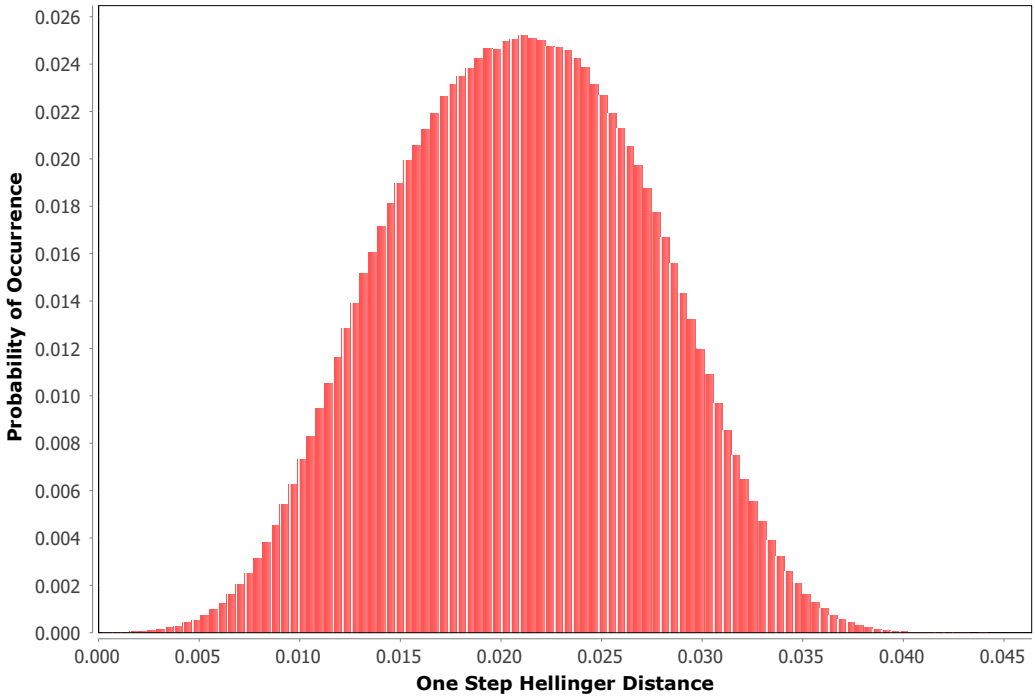


Figure 5.2: Example distribution of single step Hellinger distance values under the real-valued voter model without control

is used to learn the control signal selection policy. More precisely, the control policy is learned using a gradient-descent SARSA algorithm (Rummery and Niranjan, 1994) with a CMAC tiling (Albus, 1975) for function approximation of the real-valued distribution parameters. These are both commonly used solutions within the reinforcement learning domain and are described in more detail in Section 2.6. To limit the size of the action space, which greatly affects the computational requirements of the algorithm implementation, the same signal is inserted into each controller. Otherwise, the action space grows exponentially with the number of nodes within the control set, making learning significantly harder and more computationally expensive. As a comparison, using the single signal approach results in a constant sized action space of 21 actions, regardless of the number of control nodes. The separate signal approach, however, leads to an action space size of 9261 for 3 controllers and 4084101 for 5 controllers. More generally, the action space consists of x^y possibilities, where x and y represent the number of available signal choices and the number of controllers respectively. The state space for the problem was represented by two values:

- Δ_μ - The difference between the ideal target distribution mean and the observed state distribution mean.

- Δ_σ - The difference between the target distribution standard deviation and the standard deviation of the observed state values.

Several important system parameters were varied throughout the simulation process to generate a more diverse set of control success data. These parameters affect the overall difficulty of the control problem in a number of ways, such as enforcing tighter bounds on the state distribution or limiting the capabilities of the control system overall. Description of these important parameters and the values that were investigated are included below:

- Hellinger Threshold (H_{max}): This value is one of the most direct ways of modifying the overall difficulty of the control problem. Lower H_{max} values represent control problems in which the distribution can not move as far from the defined target. For example, consider Figure 5.2, which plots the probability of the system moving specific Hellinger distances in a single simulated step. Decreasing the H_{max} parameter increases the probability that the system will fail in a single step before the controller can adjust its signal. In cases where this one-step failure probability is non-zero, the probability will strongly affect the overall expected control success within the network. As mentioned above, if the H_{max} value is too low, it is practically impossible to control the system for any prolonged period of time. Within the simulations conducted as part of this thesis, threshold values from the set {0.05,0.06,0.07,0.08,0.09,0.1} were used. These values were selected as they present a range of difficulty within the proposed model that ranges from practically impossible (0.05) to commonly possible (0.1). Using Figure 5.2 as a reference point, this range of thresholds represents scenarios in which the uncontrolled system could fail in as little as two steps (0.05) or could run for many steps before failure (0.1). Unless otherwise specified, the results presented here omit the threshold value of 0.05, as the scenarios that used this threshold value were entirely uncontrollable with the given budget constraints.

- Budget (BGT): The budget of the controller, which defines the total cost that the control node set can have. Here, as in the original NCP research, the cost of a node is equivalent to that node's degree within the network. Like the H_{max} value, it is expected that lowering the budget available to the controller will increase the difficulty of the problem, as the control nodes will have less nodes to communicate with directly. However, due to the single signal control approach used here, increasing the budget may not always increase the control success. While an in-depth analysis is not provided here, it has been found in trial experiments that including

a large budget (i.e., more control nodes) can actually cause the system to fail, due to the lack of precision of the controller. Therefore, it is expected that a more computationally expensive multi-signal control approach would alleviate this problem and ensure that increasing the budget available would decrease problem difficulty. As the total number of edges within the system varies greatly across the different network types investigated, the budget is represented as a percentage of the overall total network edges. Budget values of 0.5%, 1%, and 2% of the total network edges are used here. These values were selected because they represent an interesting area of the control space. With an average success rate of 16%, 53%, and 60% for budgets of 0.5%, 1%, and 2% respectively, these values capture a wide range of control difficulty from 0% control success with a 0.5% budget in G+Similar networks to over 95% control success with a 2% budget in small world (Small-0.1) networks.

- Network (G): As would be expected, the network that connects nodes within the system can significantly affect the controllability of the system. Here, 100 node networks are used that have either been generated algorithmically using theoretical network class definitions or sampled from real-world social network datasets (Leskovec and Krevl, 2014). A more complete description of the different network types and generation methods is included in Section 2.3.

- Control Node Set (N_C): The set of nodes that are used to achieve control, selected by a control set selection algorithm. In the initial analysis presented within this chapter, all algorithms investigated are based off of the FAR heuristic. This heuristic was originally proposed for the NCP by Runka (2016) and is explained in detail within Section 5.3.3. Later analysis within this chapter (Section 5.5) will consider multiple FAR variants.

For each combination of network, Hellinger threshold, budget and controller set, up to 250 learning episodes were simulated for learning purposes, each starting from a randomly generated system state within a Hellinger distance of 0.01 of the target distribution and ending if the distance between the state and target distribution exceeded the specified Hellinger threshold. Throughout training, the action policy was made progressively more greedy, which is necessary in many control applications due to the poor performance that can result from the selection of random actions. More specifically, a Boltzmann exploration policy was used with the temperature parameter of the Boltzmann distribution being halved after every 25 training episodes and reaching a final value of 0.0001 after 250 episodes. Based on preliminary experiments, high initial and low final temperature

values were necessary to ensure satisfactory early exploration while avoiding the detrimental affects of random actions on a controller's performance later in the learning process. If the controller was capable of controlling the network for 50000 steps in ten consecutive learning episodes, training was terminated early. Otherwise, all 250 episodes were used for learning the control policy.

After training was completed, the learned control policy was evaluated using the same system parameters over a set of 250 testing episodes starting from pre-computed initial states, each of which was within a Hellinger distance of 0.01 of the target distribution. Within these episodes, if the controller was capable of successfully maintaining the state distribution within the permitted range for 50000 time steps, the controller is said to have successfully controlled the testing instance. In the case of the test episodes, a strictly greedy action selection policy was used, with no further exploration/learning being undertaken. Unless otherwise specified, the results presented here represent averages computed over all 250 testing episodes.

5.3.1 Ranking Control Success Across the Parameter Space

With many system parameters that can vary across different simulations (e.g., Hellinger threshold, controller budget, network), it can be difficult to compare the control success across values of any one particular parameter. For this reason, many of the results presented later in this chapter make use of a ranking approach to aggregate the effect of a large set of free parameters while focusing on a specific parameter of interest. This allows general conclusions to be drawn about a specific parameter, such as the network instantiation or control set selection algorithm. Additionally, this ranking approach weights all experimental scenarios evenly, which provides a less biased view of performance. Algorithm 12 describes the ranking algorithm that is used within this thesis. Given a set of parameter values, the control success can be measured based on the simulation results. To generate a ranking based on a single parameter then, all combinations of the other parameters, which are free to vary, can be considered. Using this (possibly large) set of combinations, each value of the parameter of interest is considered and the control success using each of these values in combination with the current values of the free parameters is measured. The control success within a system using each of the possible values for the parameter of interest can then be compared to produce a ranking of the values across that particular free parameter set. The rank of each of these values is added to a running total of ranks for that particular value. In the event of a tie, each tied value receives the same rank and additional ranks are skipped. The following examples demonstrate

Algorithm 12 Pseudocode for generating rank-based comparisons across a single parameter

Input: P_I - The set of values the parameter of interest can take

Input: P_F - The parameters that are free to vary. This is all other parameters that are not the parameter of interest (i.e., budget, network and instantiation, Hellinger threshold, control selection algorithm)

```
for each possible value, val, in P_I do
    RankCount_val ← 0
for each possible combination, P_F_Instance, of values of P_F do
    for each possible value, val, in P_I do
        Score_val ← ControlSuccess(val, P_F_Instance)
    Cur_Rank ← 1
    while Cur_Rank ≤ |P_I| do
        for each possible value, val, in P_I do
            if Score_val is highest value still unassigned, including ties then
                RankCount_val ← RankCount_val + Cur_Rank
        Cur_Rank ← Cur_Rank + (# ranks assigned)
```

the mapping between control success values and rankings for a specific free parameter value set:

- $\{0.75,0.50,0.40,0.30\} \rightarrow \{1,2,3,4\}$
- $\{0.75,0.50,0.50,0.30\} \rightarrow \{1,2,2,4\}$
- $\{0.75,0.50,0.75,0.50\} \rightarrow \{1,3,1,3\}$
- $\{0.60,0.60,0.60,0.60\} \rightarrow \{1,1,1,1\}$

Once this calculation is completed for all combinations of free parameter values, the final ranking scores of each possible value of the parameter of interest can be compared. In this case, values with a lower ranking total can be viewed as producing better control success than those with higher ranking totals.

5.3.2 Control Node Selection

The work of Runka (2016) demonstrated that the method used to select control nodes can have a significant effect on control success. The next section discusses the FAR heuristic, which was found to be the most successful control set selection heuristic by Runka. Following this, the control

success using the FAR heuristic across a wide range of scenarios is investigated and specific factors affecting control success are investigated. This analysis leads to the criticisms of the FAR heuristic and proposed improvements presented in Sections 5.4.5 and 5.5 respectively.

5.3.3 The FAR Heuristic

The FAR heuristic was originally proposed by Runka (2016) and was shown in the original NCP research to outperform many other heuristics, including those derived from the structural control theory literature and those generated through evolutionary computation. The main goal of the FAR heuristic, which is described in Algorithm 13, is to distribute control nodes throughout the network in a way that maximizes the distances between the control nodes. After starting with a single seed node, the FAR heuristic iteratively chooses the next control node to be the one with the largest minimum distance to any current control node. This process is repeated until the budget has been fulfilled. The distance between nodes, in this case, is measured by the number of edges on the shortest path between two nodes. This selection process is repeated until the control budget has been reached or no more nodes are eligible for selection. As the algorithm selects greedily following the initial seed node, the resulting control node set and the overall control success is sensitive to this seed node selection. Previously, there has been no investigation regarding the potential difference in control set quality when selecting one node as the seed versus another, but the results presented within this chapter demonstrate that the seed that is used may have a significant effect on control performance. As there are many possible control sets that can be computed using FAR for any one network, where control success results are presented concerning the FAR heuristic, they are generally computed as an average over all possible control sets generated using all possible seeds (with duplicate sets eliminated).

5.4 Network Controllability Using the FAR Heuristic

The previous NCP work of Runka compared many control set selection heuristics across several network types and determined that the best performance was produced by the FAR heuristic. The FAR heuristic, outlined in Algorithm 13, is conceptually quite simple and can produce control sets quickly. While the FAR heuristic was shown to outperform many others, including control sets that were evolved using an evolutionary algorithm, the reason why this is true was not investigated in detail. In addition to this, the existing NCP work does not thoroughly investigate why certain network types/instantiations are easier to control than others. This section will compare control

Algorithm 13 Selection of control nodes using FAR heuristic

Input: D - matrix of network's all-pairs shortest paths

Input: V_U - copy of set of network's nodes

Input: BGT - control set budget

Input (optional): V_I - a seed node from the network

Output: V_C - set of control nodes, initially empty

```
function FarSelect(D,V_U,V_C,V_I, BGT)
    if V_I = null then
        V_I ← randomly selected seed node from V_U
    V_C ← V_C ∪ V_I
    V_U ← V_U \ V_C
    while |V_C| < BGT do
        v ← (v_j : argmin_i(D_ij) ≥ argmin_i(D_ik), ∀k ∈ V_U, ∀i ∈ V_C)
        V_C ← V_C ∪ v
        V_U ← V_U \ v
```

success using the FAR heuristic across a range of both theoretical networks and networks sampled from the real-world. In addition to this, the relationship between various properties of these networks and the success of a learned controller in controlling a system on these networks will be investigated. The main questions that this work will address are:

- Which network classes are easiest/hardest to control?
- What network properties out of the several considered here are most important in determining control success?
- Why do these network properties have a significant affect on the overall control success?
- When considering these important properties, why does the FAR heuristic produce successful controllers?
- Based on this information, how can the FAR heuristic be improved to produce more effective control node sets?

Following this analysis, modifications to the FAR algorithm which incorporate the recommendations generated from answering the previous questions will be presented and compared in Section 5.5.

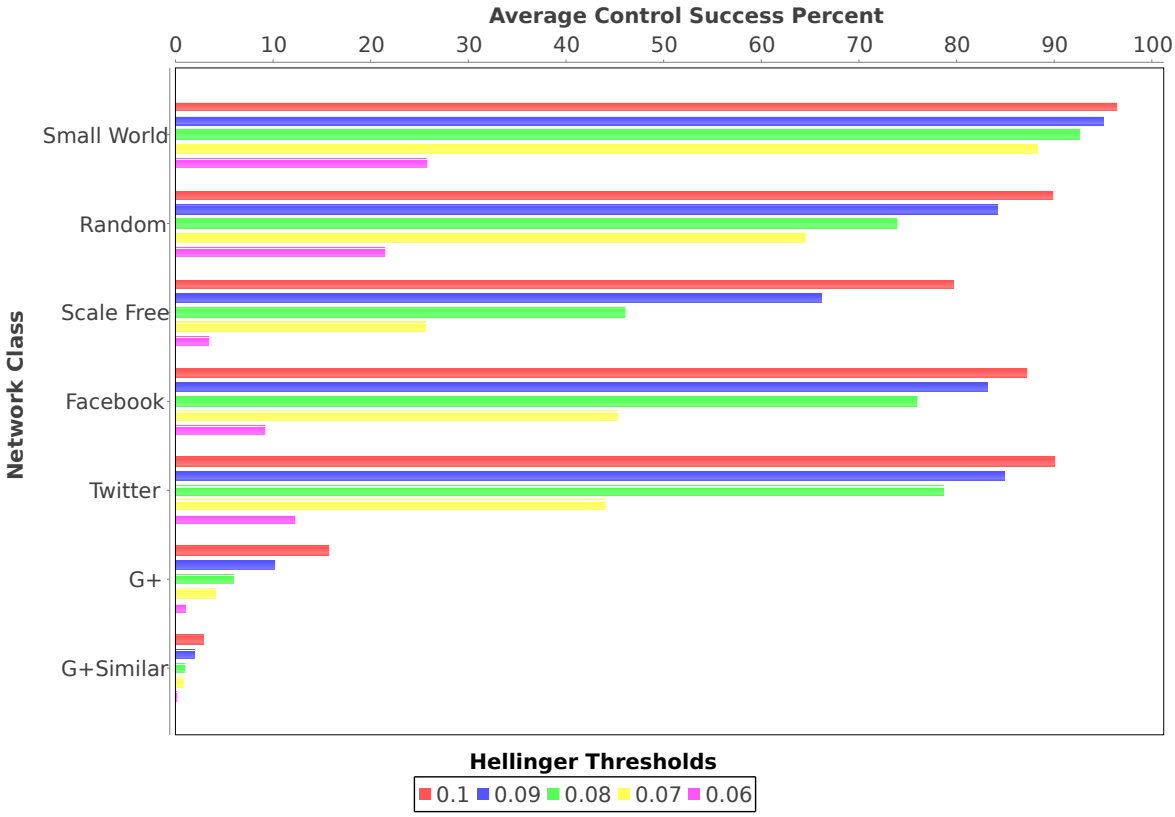


Figure 5.3: Average control success for each network class with different Hellinger thresholds

5.4.1 Network Comparison

To determine which networks are the most difficult to control, the average control success of each network class will be compared for varying Hellinger threshold values, using a constant budget of 1% of the network's edges. As mentioned in Section 5.3.3, the FAR heuristic can produce many control sets, depending on the seed node selected. For this reason, the results presented here represent the average control success across all of the different control sets that can be generated using different seed nodes within the network. Figure 5.3 shows the average control success across all instantiations and control sets for each network class using different Hellinger threshold values. From this figure, it is easy to identify the two types of G+ networks as being the most difficult to control, having a low control success even when using the largest Hellinger threshold. In addition to this, the rate that control success decreases can also be considered. In this case, the Scale Free network class appears to also be more difficult to control than others, as the control success appears to drop off more rapidly than other network types as the Hellinger threshold decreases.

Table 5.2: Summary statistics for the control success ranking of each network class investigated using the FAR heuristic – complete rankings for each instantiation can be found in Appendix B

Network Type	Mean	Median	Min	Max
Small-0.1	5.5	5.5	1	10
Random-3.1	20.4	20	11	31
Facebook	28.25	26.5	16	41
Twitter	29.7	27.5	12	47
Scale Free	39.7	39.5	32	49
G+	57.8	59	46	67
G+Similar	59.8	59.5	51	68

To further demonstrate the difference in controllability, the ranking procedure described in Section 5.3.1 can be used to rank the control success of each network instantiation relative to the others across all budget/threshold combinations. Table 5.2 includes summary statistics, including the mean, median, minimum and maximum rankings of the instantiations of each network class considered, where a lower rank represents more successful control in a network[1]. The ranking results in Table 5.2 further confirm the results from Figure 5.3. Both of the G+ network classes are ranked the lowest overall and have almost no overlap with other network classes when the minimum and maximum rankings for each class are considered. In addition to this, the Scale Free network class also represents the 3rd highest mean and median rankings.

While these results are helpful in determining which of the network types investigated were least/most difficult to control, they do not provide any direct evidence as to why some are harder to control than others. The following subsection will further consider the relationship between various network parameter values and control success. These results will identify what network properties are most likely to negatively affect control success.

5.4.2 Network Properties and Control Success

While the previous subsection identified which network classes were more/less controllable overall, the goal of this subsection is to answer the question: which network properties are most likely to determine a network's controllability? To answer this question, the analysis will focus on correlations between specific network properties and control success across all of the considered network instantiations. In addition to this, a decision tree algorithm will be applied to discretized

[1]The full ranking results of every network instantiation considered, are included in Appendix B.

Table 5.3: Summary statistics for the control success of each network class investigated ($BGT = 1\%$, $H_{max} = 0.08$)

Network	**Instantiations**	**Mean**	**Median**	**Min**	**Max**
Random-3.1	10	73.84	73.13	49.67	87.60
Scale Free	10	45.97	51.14	7.53	61.33
Small-0.1	10	92.57	93.49	76.89	96.36
Facebook	8	75.91	82.33	41.13	94.93
G+	10	5.91	3.18	0.00	21.62
G+Similar	10	0.85	0.12	0.00	3.33
Twitter	10	78.73	84.62	57.95	93.30

network property and control success values, with the goal of identifying which combination of network properties have a negative effect on control success. To simplify the analysis, the results presented here will only consist of scenarios using a budget of 1% of the network's total edges and a Hellinger threshold of 0.08. These values were selected because they lie around the area in which control seems to become significantly more difficult, which makes subtle differences between the networks more noticeable than in scenarios that are easier/harder to control overall.

Table 5.3 includes the summary statistics for each network class when these specific scenarios are considered. This table demonstrates that, even within the same class of network, there can be significant variation in control success. An important question to consider, then, is what network properties within each class of network are most strongly related to the overall control success?

The first thing that will be considered in answering this question is the correlation between difference network parameters and control success across each type of network and across all networks in general. Table 5.4 shows the Pearson and Spearman correlations for various network properties across each network class, as well as the correlations over all networks investigated. When considering specific network classes, each of the network properties considered show at least one example of being highly correlated with control success, but maximum betweenness and diameter appear in significant levels most frequently. As would be expected based on the class-specific results, the diameter of the network and maximum node betweenness within the network show the highest levels of correlation with control success when all networks are considered. One interesting note from this table is that, while diameter does not present a strong correlation to success for the G+ network types, these network types also have the largest diameters and lowest control success of all networks in general, which is in line with the strong negative correlation found when all networks

Table 5.4: Pearson and Spearman correlations between network control success rate and different network properties, calculated across all instantiations of each network type ($BGT = 1\%$, $H_{max} = 0.08$, correlations greater than 0.5 bolded)

Network Type	Diameter	Average Degree	Max Degree	Average Betweenness	Max Betweenness	Clustering Coefficient
Random-3.1	**-0.70/-0.57**	0.35/0.33	0.18/0.04	-0.47/-0.48	-0.45/**-0.75**	-0.06/0.20
Small-0.1	-0.40/-0.28	N/A	-0.14/-0.37	**-0.53**/-0.21	**-0.53/-0.56**	**-0.85/-0.53**
Scale Free	0.14/0.03	**0.55/0.74**	**-0.92/-0.95**	**0.67**/0.30	**-0.79/-0.83**	**-0.63/-0.73**
Facebook	**-0.87/-0.73**	**0.57**/0.45	0.48/0.48	**-0.76**/-0.43	**-0.90/-0.67**	0.40/0.26
G+	-0.29/-0.32	-0.13/-0.24	-0.32/-0.43	-0.03/0.02	-0.37/-0.14	-0.29/-0.43
G+Similar	-0.12/-0.13	-0.10/-0.14	-0.09/-0.03	-0.12/-0.01	-0.21/-0.20	-0.03/-0.07
Twitter	**-0.57/-0.56**	**0.78/0.79**	0.35/0.37	**-0.67/-0.68**	**-0.73/-0.77**	0.46/**0.60**
All	**-0.67/-0.55**	0.10/0.03	-0.14/-0.29	-0.33/-0.20	**-0.70/-0.75**	0.24/0.21

are considered. While these results present a starting point in identifying which network properties may be most strongly linked to control success, a number of the properties may be correlated with each other as well, which could confound the analysis. For example, the maximum betweenness and node degree within the Scale Free class of networks share a Pearson's correlation coefficient of 0.94.

To further investigate how the network property information can be used to predict network control success, the C4.5 decision tree learning algorithm was applied to the data. Before using this algorithm, each of the properties and control success frequencies were discretized into three categories as follows:

- Low: The value falls into the bottom 25 percent of the values across all networks.
- Average: The value lies between the bottom 25 percent and the top 25 percent of the values across all networks.
- High: The value lies within the top 25 percent of the values.

The output of the decision tree algorithm, considering only the FAR scenarios with $H_{max} = 0.08$ and a 1% edge budget, is shown in Figure 5.4. The output of a second decision tree algorithm, which considered all threshold and budget combinations, is included in Figure 5.5. The resulting trees show that, as may be expected from the information in Table 5.4, the network diameter is the first value used to predict control success, which implies that diameter divides the group of all networks the best when considering control success. Additionally, the second split in all cases uses the maximum node degree within the network. The accuracy of these classifiers in deciding the control success over all network instantiations was 69% and 79% respectively. It should also be noted that

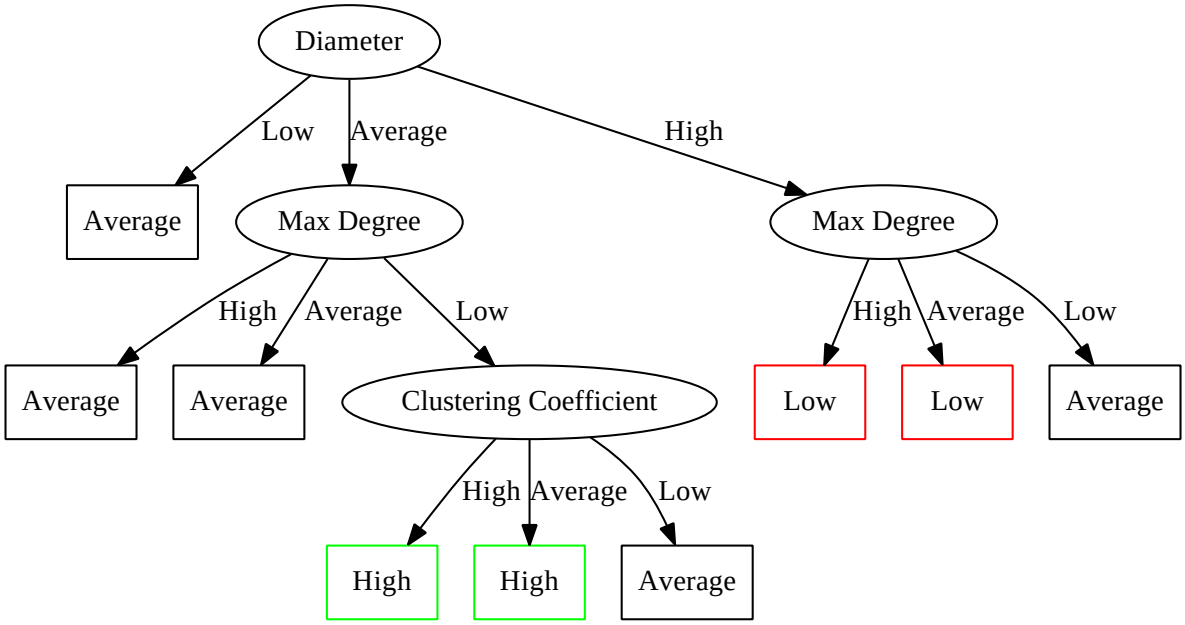


Figure 5.4: Decision tree output for predicting control success (shown in rectangles) using the FAR heuristic, $H_{max}=0.08$ with a 1% edge budget

all classifications were within one step of their true value. This means that no instantiations with low control success were classified as having high control success, and vice versa. Based on these findings, the following two subsections will present arguments as to why the maximum degree node and network diameter are so strongly related to the overall network control success.

5.4.3 Scale Free Networks and Maximum Degree Node

The analysis within the preceding section identified the maximum node degree within a network as being a strong predictor of network control success. This section will answer a question related to this finding: why are networks containing a node with very large degree so difficult to control? To answer this question, this section will begin by focusing on the underlying influence dynamics within the system and identifying how large nodes could affect control success based on these influence dynamics. Following this, a microscopic analysis is performed, which involves dividing the network into those nodes that are attached to the largest node and those that are not. This microscopic analysis will demonstrate that the presence of a large degree node can have a significant affect on the ability of a controller to successfully control the network. As scale free networks had the strongest correlation between the maximum degree node and control success, the investigation

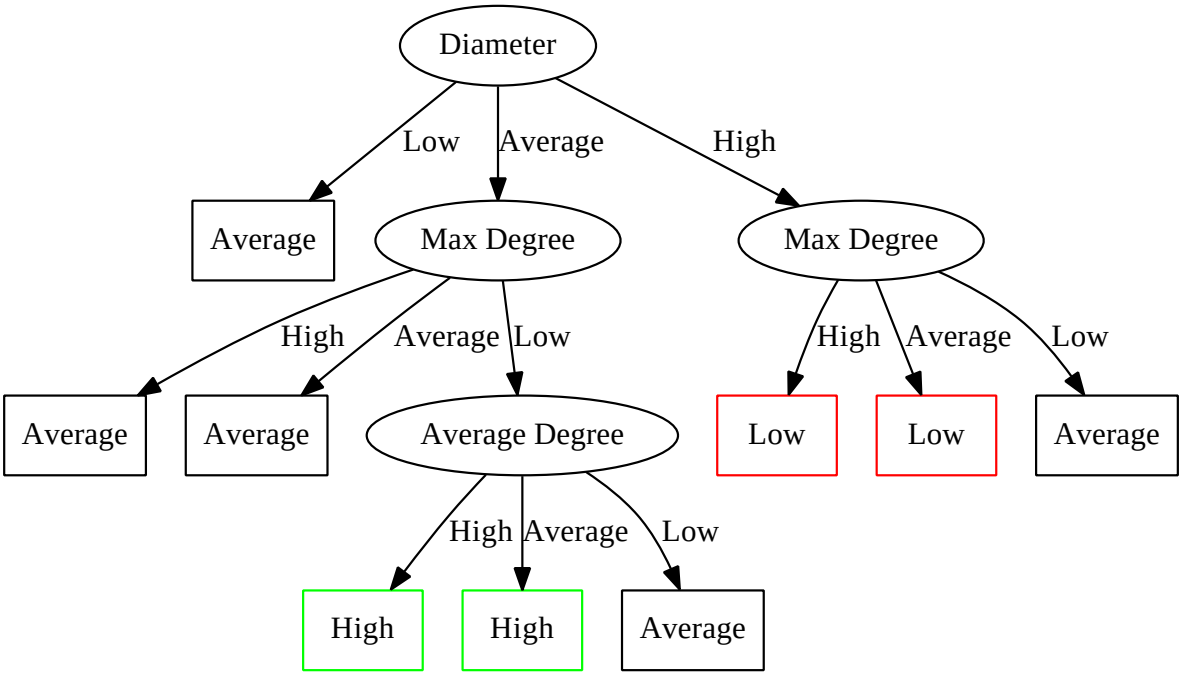


Figure 5.5: Decision tree output for predicting control success (shown in rectangle boxes) using FAR heuristic over all H threshold and budget choices

in this section will be limited to analysis of the scale free network instantiations. Figure 5.6 shows the average control success for each of the ten scale free network instantiations, computed over scenarios using a Hellinger threshold of 0.08 and a budget of 1% of the network edges. The analysis within this section will focus specifically on instantiation #0, which has the largest maximum degree and the lowest control success of the scale-free network instantiations.

As discussed previously, the presence of a large degree node within the network can have a significant negative impact on overall control success. What can the underlying influence model of the system under consideration tell us about the effect of these large degree nodes? While this work, especially the analysis of previous research presented in Chapter 3, has argued against the use of node degree as a measure of influence in general, the use of a theoretical agent model here results in specific influence properties within the system that are related to a node's degree. The description of the real-valued voter model presented in Section 2.5, specifically Algorithm 4, demonstrate that at each time step within the simulation, each node moves its opinion state value towards (i.e., is influenced by) the state value of one of its randomly chosen neighbours. The probability that a node i influences its neighbour j at any time step, then, is equal to $\frac{1}{deg(j)}$, where

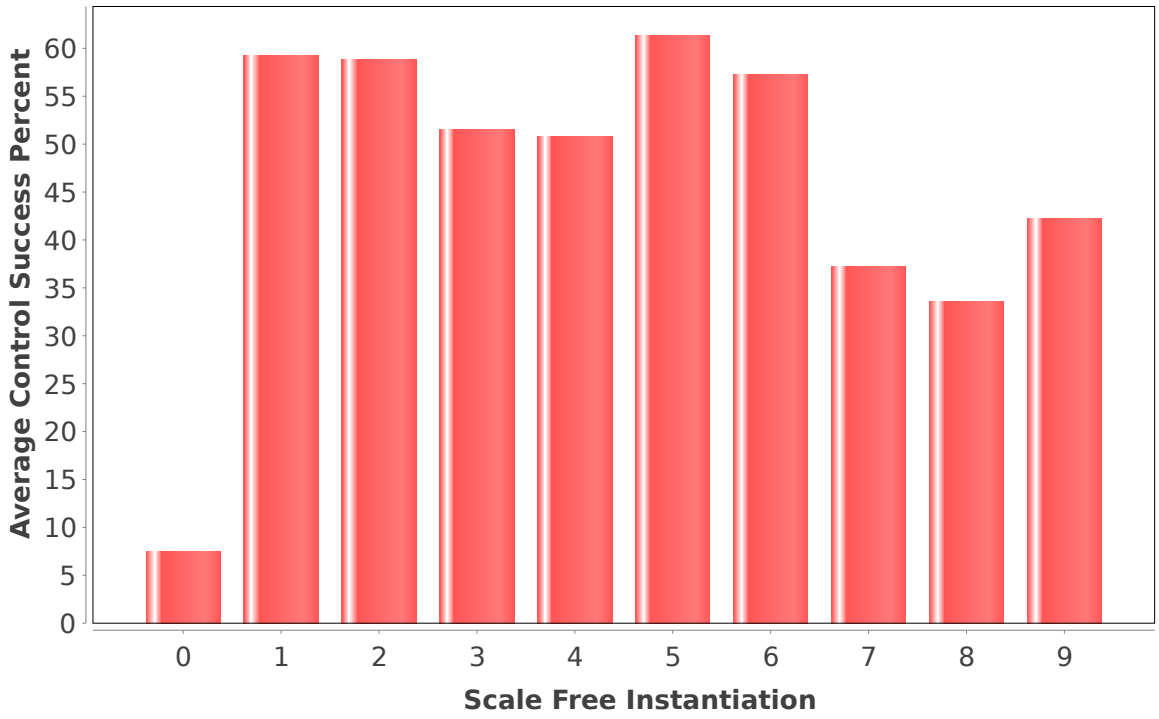


Figure 5.6: Average control success for each Scale Free network instantiation using a Hellinger threshold of 0.08 and a budget of 1%

$deg(j)$ represents the degree of node j. Based on this known property, it can be concluded that the higher the degree of j is, the less likely it is that any one of its neighbours will influence it during a particular step. By the same argument, as the degree of a node increases, it is more likely to influence nodes within the network, as each outgoing edge represents the possibility to influence another (although the exact amounts depend on the neighbour degrees as well). Within scale free networks, this effect may be strengthened, as large degree nodes (which have many opportunities to influence) are often connected to nodes with small degree (which are easily influenced).

So, as the degree of a node, n, increases, it becomes harder for neighbours to influence node n, while node n also tends to becomes more influential. But, an important question to consider is: does this affect the overall control success in a negative way? To answer this question, the change of the state distribution parameters over time will be considered in both controlled and uncontrolled scenarios. More specifically, the change in these parameters will be considered within the following three sets of nodes:

- Large Node Neighbourhood: This set consists of the largest node in the network and all

of its direct (i.e., 1 hop) neighbours.

- Other Nodes: Any node that is not present in the large node neighbourhood set.
- All Nodes: The union of the two sets above, or in other words, the entire network.

If the effect of the controller on the overall system behaviour is minimal within the large node neighbourhood set, it can be concluded that the inability to influence this large node and its neighbourhood, combined with the large node's ability to affect other nodes, is negating the effect of the controller.

The target distribution of state values within the real-valued voter model used here has two parameters: the mean and the standard deviation (SD). Through experimentation, it was discovered that the natural momentum of the real-valued voter model, when uncontrolled, is for the state values to converge toward a single value, causing the SD to move toward 0. The mean, on the other hand, often did not move significantly from its original value. For this reason, only the SD parameter is considered within this analysis and the term 'distribution parameters' is used interchangeably here to refer to the SD.

Figure 5.7 shows the average standard deviation of state values within the three node sets over 25 steps of simulation where no control actions are taken. Each of these values represents the average SD value within that set of nodes over 1000 executions of an uncontrolled simulation. On average, the SD of each set decreases at close to the same rate over these 25 simulation time steps, with the large node neighbourhood moving slightly faster. Again, using the influence property analysis discussed above, it is not surprising that the large degree node neighbourhood converges faster than other areas of the network, as the large degree node has a relatively large probability of influencing a large number of nodes, making it more likely for those nodes to move toward the large node's state value. When the uncontrolled scenarios are compared to scenarios with control actions being applied by learned controllers, as shown in Figure 5.8, an interesting result emerges. In the controlled case, the large node neighbourhood converges in almost an identical way to the uncontrolled case, whereas the other nodes maintain a higher level of variance to counteract the low SD of the large node neighbourhood, allowing the overall SD to stay near the defined target of 0.05 (the average absolute difference between the uncontrolled and controlled cases is included in Figure 5.9). In these cases, then, the controller is really only attempting to counteract the affect of the large node neighbourhood by increasing the variance elsewhere in the network. As the large

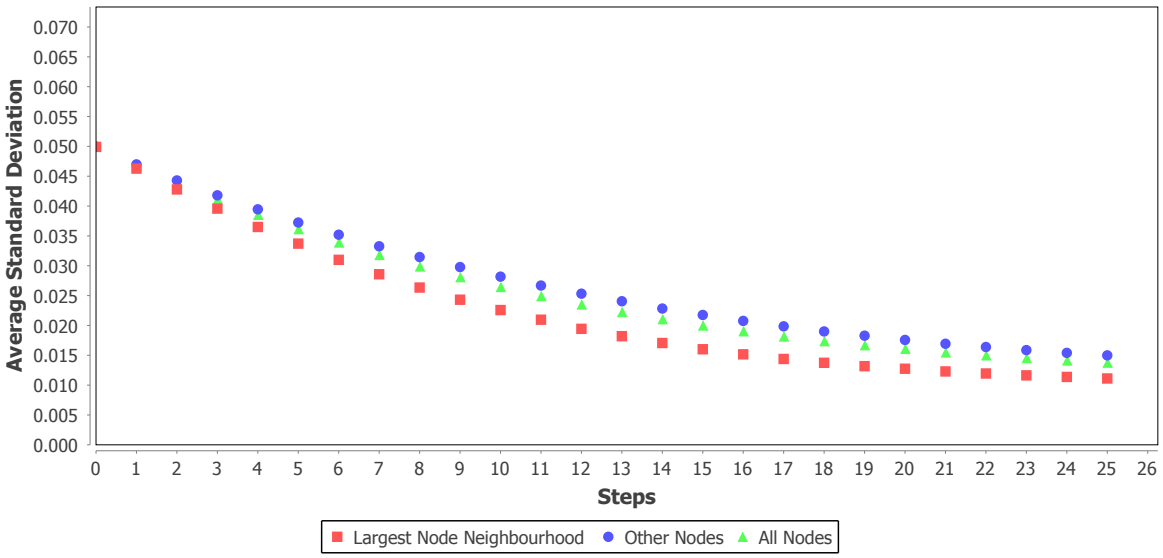


Figure 5.7: Average standard deviation of state values within different node sets over time without control

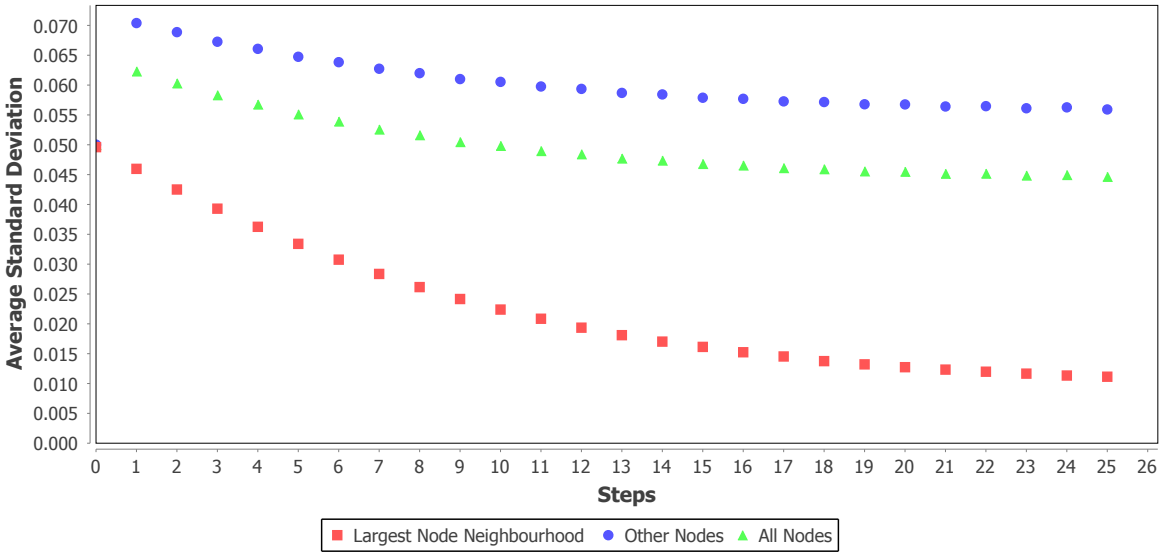


Figure 5.8: Average standard deviation of state values within different node sets over time with control

degree neighbourhood increases in size, however, the number of nodes available to counteract the effect decreases, resulting in a higher rate of control failure.

The previous results demonstrated that large degree nodes negatively affect control success by preventing controllers from affecting the state of the system within their neighbourhood. One

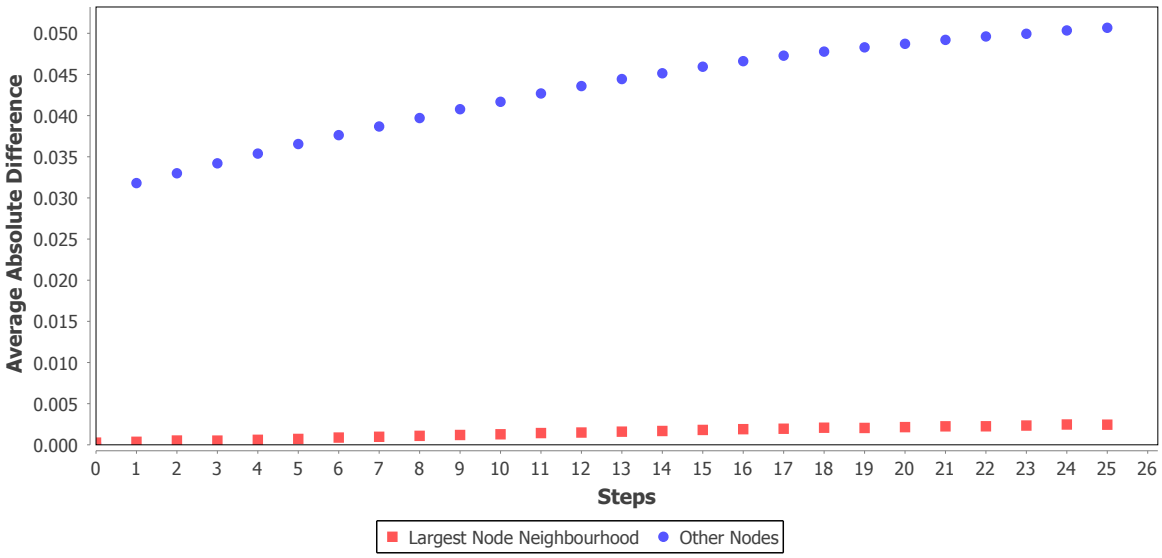


Figure 5.9: Average absolute difference (controlled case vs. uncontrolled case) in state distribution value within the large node neighbourhood and other node sets over time

important question that will now be considered is: are there specific control node sets that are capable of affecting the state of the large node neighbourhood? Based on the known influence properties of the system, the control nodes most likely to be capable of affecting the state within the large node neighbourhood are nodes that are within this neighbourhood. To determine if this is the case, the behaviour of the state distribution parameters over time will again be compared, this time using control sets that have a direct connection to the large neighbourhood and control sets that do not. In doing so, the effect of each of these control sets can be measured to determine which types of control sets, if any, are capable of affecting the large node neighbourhood.

Using a similar methodology to the one used previously, the uncontrolled case was simulated for 100 time steps with 1000 repetitions and the average state distribution parameter value within the large node neighbourhood was recorded at each step. For each of the controller sets generated by the FAR heuristic using different seed nodes, a controller was learned and the same average state distribution parameter values were calculated with control actions being applied. The average absolute difference between the controlled and uncontrolled cases over all steps was then calculated and recorded for each controller configuration. Table 5.5 includes a comparison showing the average and maximum difference in the average state parameter value between the uncontrolled case and either:

- Uncontrolled: A case in which the simulation was run a separate time without control,

Table 5.5: Average and maximum difference between the large degree neighbourhood state parameter value without control and either a second uncontrolled simulation, or control with/without a direct connection to the large degree node

Controller	**Average Difference**	**Max Difference**
Uncontrolled	0.00155	0.00434
Connected	0.03218	0.03912
Not Connected	0.00179	0.00416

meaning any difference realized is due to random chance.

- Not Connected: Cases where the control set does not have a direct connection to the largest degree node.

- Connected: Cases where the control set has a direct connection to the largest degree node.

The small difference between the distributions in the uncontrolled and not connected scenarios demonstrate that in cases where the control set is not directly connected to the node with largest degree, the control actions have almost no influence on the large node neighbourhood, which means all of the nodes in that neighbourhood are effectively uncontrolled. As was mentioned previously, as the size of this neighbourhood grows, the proportion of nodes in the network that are effectively uncontrolled increases and, at the same time, the number of nodes available to counterbalance the effect created by the uncontrolled nodes decreases. These two factors combine to greatly decrease the controllability of the network.

The previous analysis has provided considerable evidence in support of the claim that large node neighbourhoods negatively affect control success. But do control sets that have the ability to influence the large node neighbourhood (i.e., those that are connected to the large node) have a higher rate of control success than those that do not? When the control success of connected/non-connected control sets were compared, the median steps to failure and the percentage of tests successfully controlled were found to be 34226/20212 and 46.2%/30.5% respectively. This represents an overall improvement of over 50% in both the median steps to failure and the probability of successful control when using control sets capable of influencing the neighbourhood of the largest node in the network.

Based on this information, another important question to answer is: does the FAR heuristic favour control sets that have a connection to the largest degree node in the network? Unfortunately, the FAR heuristic does *not* seem to favour connected control sets. In total, 61 of the 100 generated

control sets fall into the category of 'not connected' and are, therefore, incapable of having a significant affect on the distribution of values in the large node neighbourhood. This information would indicate that a control node set selection algorithm that acknowledges the underlying influence dynamics of the system could result in improved network control success when compared to the FAR heuristic and other heuristics that FAR has been shown to outperform.

The previous results and discussion have demonstrated that large degree nodes can effectively block their own neighbourhood from being influenced. An additional argument in favour of including possible influence and information flow analysis in control set selection is that these nodes may affect control success further by blocking the ability of control signals to flow through the network. The easiest way for information (e.g., control signals) to flow between nodes within a network is via the shortest path. So how many of the shortest paths within each scale-free network must attempt to flow through a large degree node? Table 5.6 includes the betweenness centrality rank and the percentage of shortest paths that pass through the largest node in each of the scale free network instantiations. The percentage of shortest paths passing through these nodes is very high, particularly for networks that had lower levels of control success (i.e., instantiations 0, 7, 8, and 9, see Figure 5.6). As the preceding discussion provided evidence to support the claim that it is difficult to influence large degree nodes, it is expected that the effect of control signals will essentially be blocked by these large nodes within the networks. These nodes, then, act as a wall along a large percentage of the shortest paths between nodes, blocking the influence of the control nodes from flowing to other areas of the network. For this reason, the effect of a control signal in these networks will take longer to propagate to some nodes than a structural analysis of the network would predict. As the FAR heuristic only accounts for network location, and not the actual potential for information flow, this effect would not be considered when selecting a control node set. An influence-based control set selection algorithm, however, could account for these effects and possibly produce more effective control sets.

So while it is clear that the nodes that have the largest degree in these scale free networks can have a significant negative effect on network control success, a final question to consider is: what happens if the large degree nodes are controlled directly? To answer this question, the FAR heuristic was used to select a single control set containing the largest degree node within each scale-free network instantiation. Unfortunately, this approach seems to result in an even lower probability of successful control. None of the 10 control node sets considered produced a successful controller in the case of $H_{max} = 0.08$, which can be compared to the 71% sucess rate found with

Table 5.6: Betweenness centrality measurements of the node of largest degree across the scale free network instantiations

Instantiation	Betweenness Rank	Percent of Shortest Paths
#0	1	61.7
#1	3	25.2
#2	1	42.4
#3	1	40.8
#4	1	39.5
#5	1	27.9
#6	1	36.1
#7	1	57.1
#8	1	54.6
#9	1	67.3

other control sets. While the sample size in this case is small, it is likely that using a node with such a large degree causes too much change within the system, leading to control failure.

5.4.4 G+ Networks and Diameter

The results presented in Section 5.4.2 found that the two most important network properties in relation to network control success were the maximum node degree and the diameter. The previous section provided analysis relating to the maximum node degree in scale free networks. This section will investigate the role of network diameter in determining the poor network control success of G+ network types, as these networks presented the largest diameters and lowest control success.

To get an initial idea of why increasing network diameter may negatively impact control success in such a significant way, Figures 5.10 and 5.11 show visualizations of two graph instantiations - one instantiation of a G+ network (Figure 5.11), which was very hard to control, and one instantiation of a Twitter network (Figure 5.10), which was significantly more controllable. The graph layout in both of these figures is accomplished using the NEATO package of the graph visualization software package GraphViz (Gansner and North, 2000), which uses a spring-based model to distribute nodes in the network in a way that represents their distance from each other within the network. Within these figures, there is a noticeable structural difference between the two network instantiations. The nodes within the Twitter network are tightly clustered, representing a short distance within the network between many nodes. Within the G+ network, however, a large portion of the nodes are clearly spread across a large space, representing a significant distance between many nodes. If one imagines control

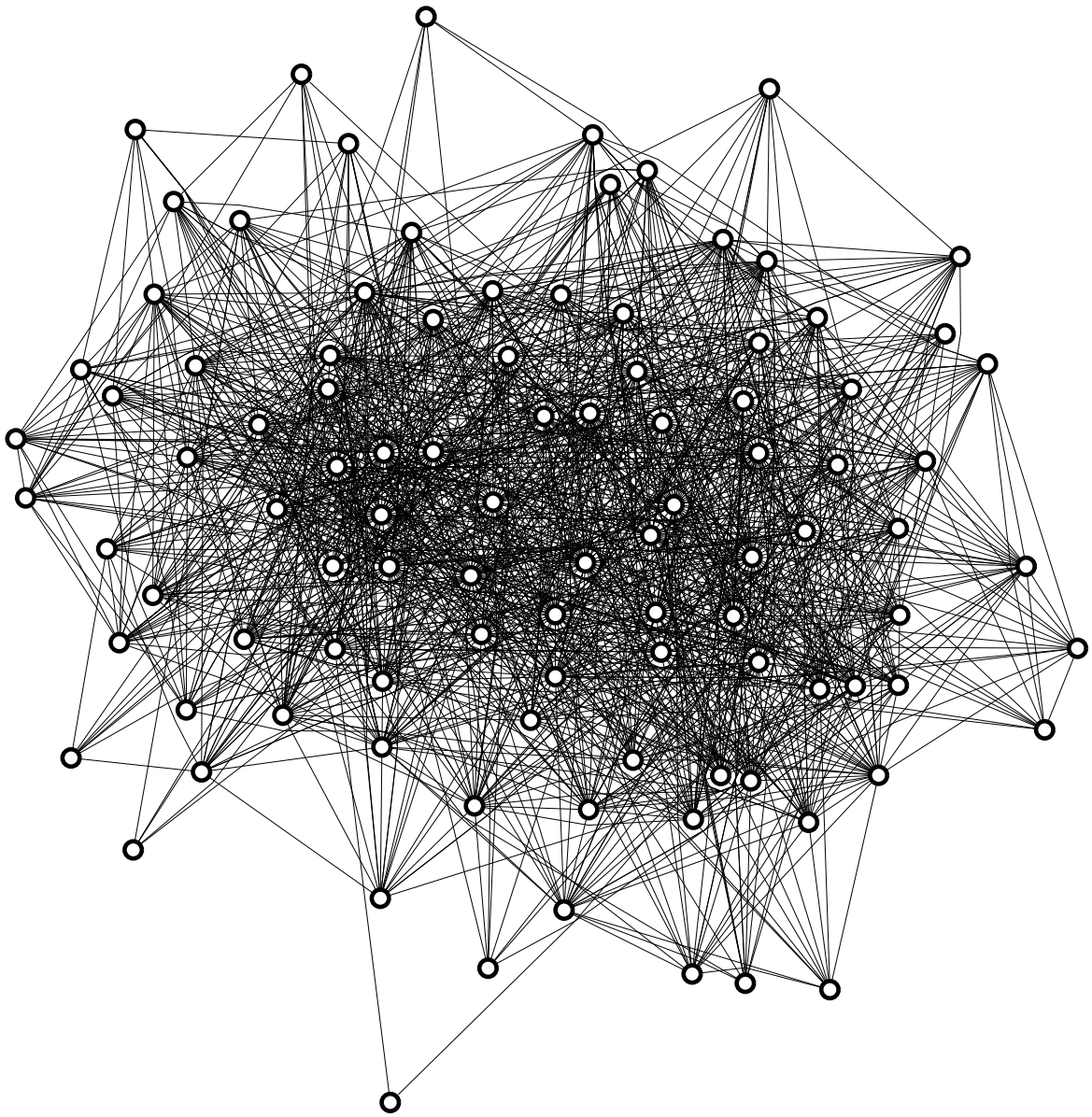


Figure 5.10: NEATO graph layout of a Twitter network instantiation which has been found to be relatively easy to control

signals propagating probabilistically across the two networks from a small number of control nodes, it would seem that these signals would take significantly longer to spread in the G+ network than they would in the Twitter network. The remainder of this section will investigate this idea in more detail by investigating the role of diameter in relation to network controllability of some G+ networks.

To investigate the effect of a large diameter on control success, a similar methodology to that

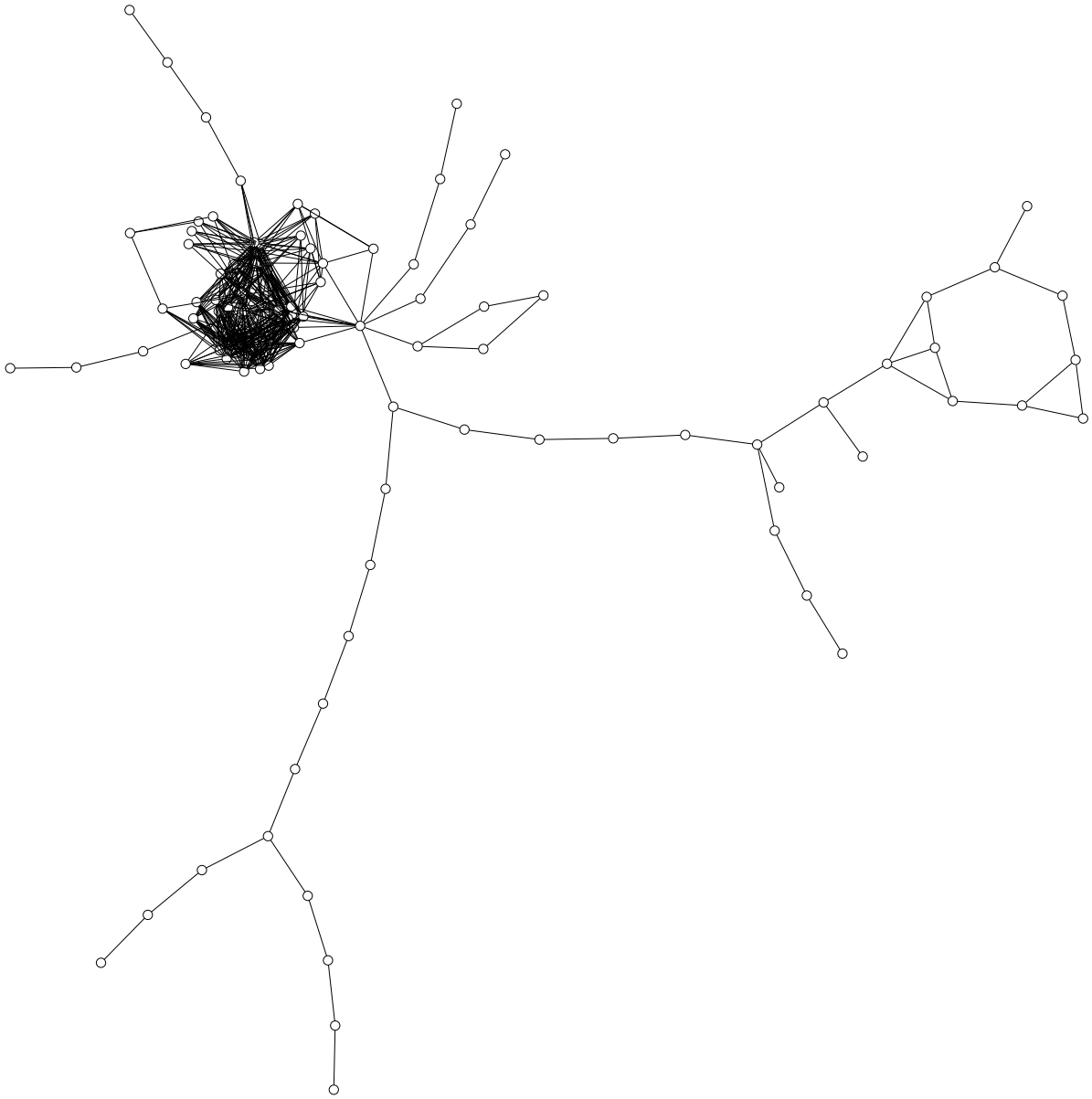


Figure 5.11: NEATO graph layout of a G+ network instantiation that was found to be one of the most difficult networks to control

used in investigating the effect of large degree nodes (Section 5.4.3) will be used. This approach involves considering the expected value of a state distribution parameter within subsets of the network's nodes over time. Each of the expected values represents the average state distribution parameter at that time step calculated over 1000 independent simulations. The subsets of nodes used within this analysis are defined below:

- Close Nodes: Those nodes that are within three hops of any control node within the network.
- Distant Nodes: Those nodes that are greater than three hops away from all control nodes within the network.
- All Nodes: All nodes within the network.

In this case, the limit of three hops was selected experimentally, as it divided the network's nodes most effectively into 'influenced' and 'non-influenced' groups. Interestingly, though, the number three also arises in real-world studies of humans, as in the work of Fowler and Christakis (2008), which found 'happiness' spreads up to three steps within social networks. By dividing the network into distant and close nodes, the effect of controllers on the two groups can be measured and compared to an uncontrolled scenario. If, like the large degree node issues discussed in Section 5.4.3, the controller is unable to significantly affect the state of distant nodes, then it can be concluded that increasing the diameter of a network will likely lead to lower levels of control success, as the number of distant nodes should also increase (assuming the number of controllers remains the same). Since the G+ networks were so difficult to control, the analysis here involves the use of a Hellinger threshold value of 0.25, which is significantly higher than any threshold that has been considered elsewhere in this work. However, to ensure success in learning a control policy, this threshold was found to be required. As with the maximum node degree analysis, the first thing that will be analyzed is the convergence rates of the standard deviation state distribution parameter within the different node sets.

Figure 5.12 shows the expected SD of state values within each of these node sets where no control action is taken. Figure 5.13 includes a similar plot, but applies control actions selected by a learned controller. In addition to these figures, Figure 5.14 plots the average absolute difference between the controlled and uncontrolled scenarios for the close and distant node sets. These figures demonstrate that, in a way similar to the large node neighbourhood discussed in the previous section, the set of nodes that are further than three steps from all control nodes converge at an almost identical rate regardless of whether or not a control signal is being applied to the network.

The previous results indicate that having a larger proportion of nodes that are distant from the control nodes should lead to decreased control success. But does increasing the network diameter lead to an increase in distant nodes? To answer this question, correlation values were calculated between the diameter of the network instantiation and the average number of nodes beyond three hops, calculated over each FAR control set. This resulted in a rank-based correlation coefficient of

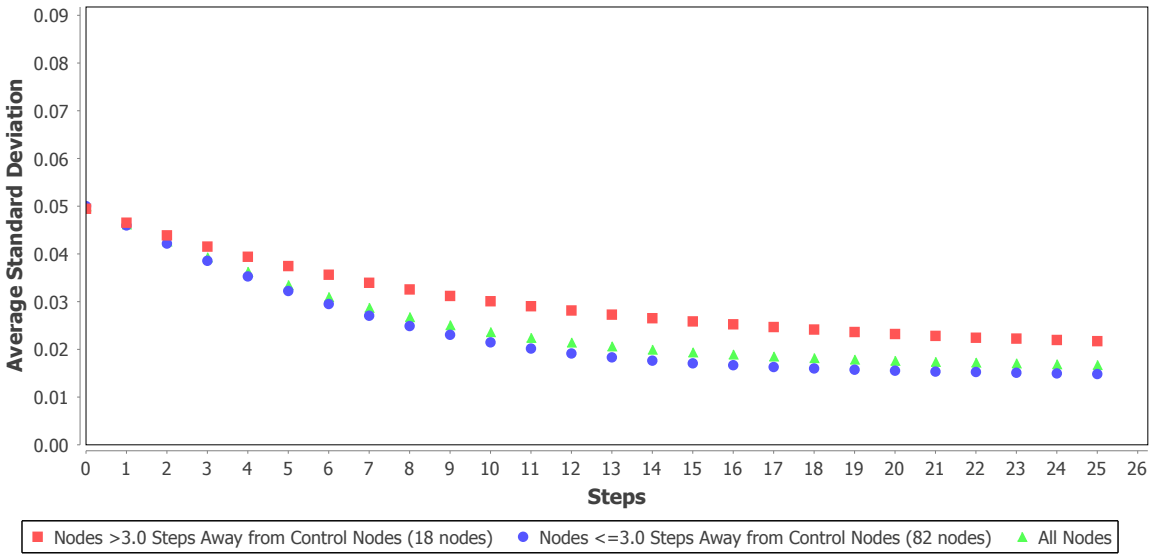


Figure 5.12: Average standard deviation of state values within the close/distant node sets over time without control

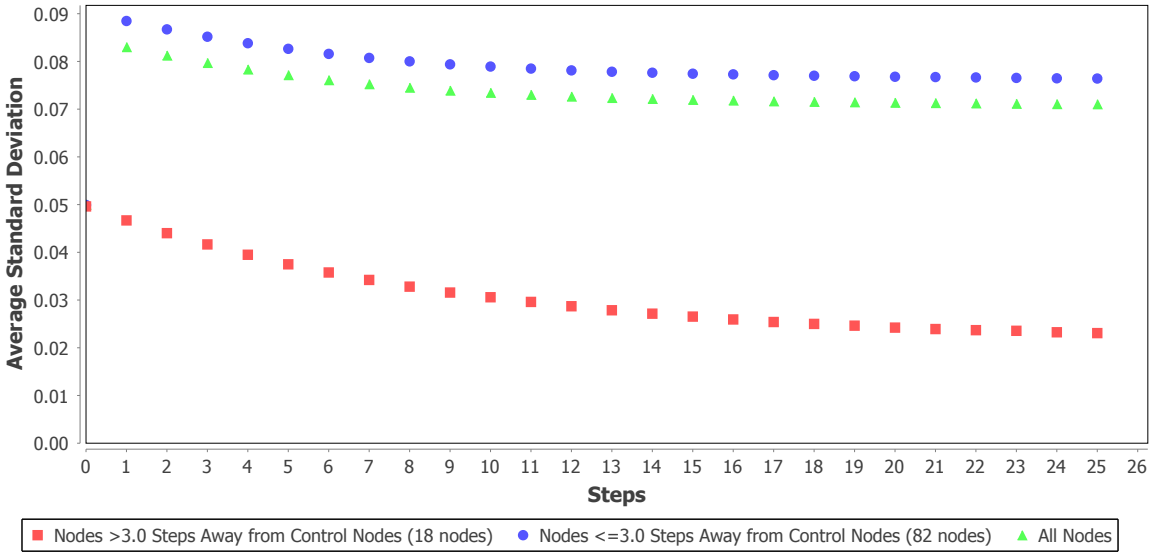


Figure 5.13: Average standard deviation of state values within the close/distant node sets over time with control

0.60, which signifies that, at least in general, the number of nodes that are distant from all control nodes tends to increase as the diameter of the network increases.

An additional question to consider is: does a relationship exist between the proportion of 'distant' nodes within the network and control success? To answer this question, the control success

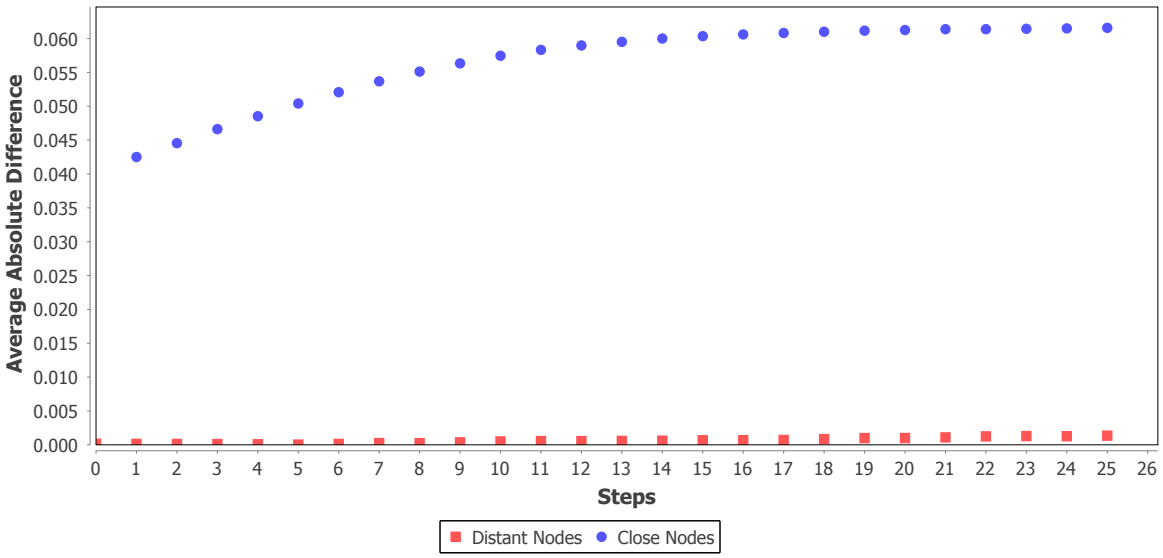


Figure 5.14: Average absolute difference (controlled case vs. uncontrolled case) in state distribution value within the close/distant node sets over time

using each FAR control set across all of the G+Similar network instantiations was plotted against the proportion of distance nodes. It should be noted that, in this case, the results include budgets of 0.5%, 1%, and 2% of the network's edges, and a Hellinger threshold of 0.08. This decision was made to allow for greater variation in both the number of distant nodes and the control success.

The control success relative to the proportion of 'distant' nodes is plotted in Figure 5.15. The most interesting conclusion that can be drawn from this figure is that the only scenarios in which a respectable level of control success can be expected are those in which approximately 50% or less of the nodes are considered distant (more than 3 hops) from the control set. While it is possible that the limit of 3 hops may not be applicable in a general sense, the average path length from a node to all other nodes in the network is often considered an estimate of that node's influence/importance within the network. So it is possible, then, that these results also indicate the importance of maximizing the control node set's influence within the system by minimizing the average path length of other nodes. Based on this information, it would seem that a control set selection algorithm that attempts to minimize the number of distant nodes relative to the control set could produce effective control sets. Contrasting this with the results previously discussed in this section, another way of putting this statement is that an ideal control set selection algorithm will maximize the number of nodes that are not distant, which should also maximize the number of nodes that can be influenced by the control nodes. Once again, since the FAR heuristic does not account for any influence dynamics within the

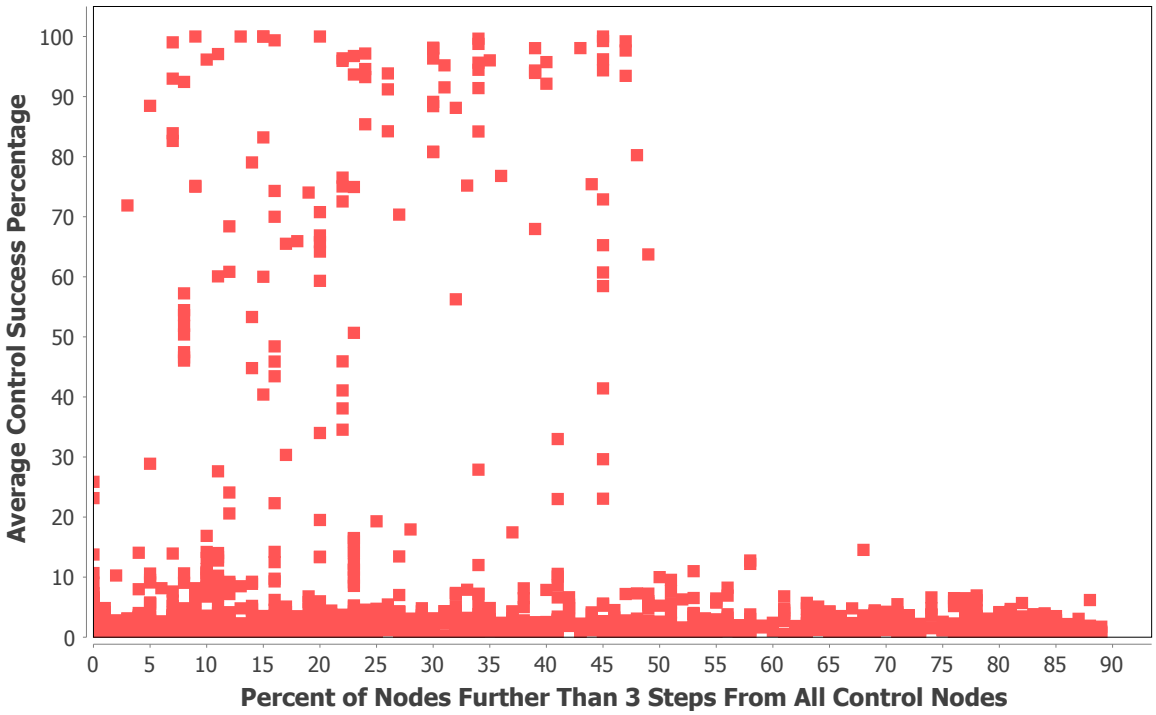


Figure 5.15: Average control success relative to the percent of distant nodes (≥ 3 steps) for G+Similar networks with a Hellinger threshold of 0.08

system, this is an obvious target for improvement, as will be discussed in the following sections.

5.4.5 Criticisms of the FAR Heuristic

Within the previous NCP research, the FAR heuristic was shown to produce higher levels of control success than any of the other control set algorithms considered, but the reason for FAR's success was not fully explored. Some of the analysis within the previous sections, especially that relating to close/distant nodes, has provided some evidence in support of FAR's success. By choosing the next node to be the one with the maximum shortest path to any other control node, the FAR heuristic attempts to space control nodes evenly throughout the network. This is effective because the next selected node should be close to nodes within the network that were previously the most distant from the control set, thereby reducing the number of distant nodes. In addition to this, spreading the nodes throughout the network lowers the chance that control effects from different control nodes overlap and/or compete with each other, which is intuitively inefficient. There are, however, still some issues that can be identified within the existing FAR heuristic.

Lack of Influence Dynamics

The first criticism of the FAR heuristic is that it does not account for influence dynamics within the system. Instead, the shortest paths between nodes are measured using unweighted, homogeneous links. However, when the influence dynamics of the system are accounted for, these shortest paths may be significantly different. For example, consider the network in Figure 5.16. In this case, the shortest hop-based (i.e., unweighted) path from S to D would be S-A-B-D, while the shortest weighted path would be S-1-2-3-D. In addition to this basic example, the betweenness centrality measure of nodes within the networks studied here using weighted or unweighted edges were compared to quantify the difference between the weighted and unweighted cases. It was found that the percentage of total shortest paths passing through a node in the weighted case can change by almost 20% in some cases from the unweighted case. A common simplifying assumption in social network research is that information flows through the shortest/easiest pathway in a network (e.g., several centrality measures, such as Freeman, 1978). The fact that weighting the network using the influence dynamics can change these shortest paths significantly indicates that this information should be included within any algorithms used for control node set selection.

Heuristic Nature

In addition to this, while the heuristic nature of FAR allows control sets to be computed quickly, it can limit the overall quality of the control set. Based on the discussion related to network diameter from the previous section, one goal of a control node selection algorithm may be to limit the number of nodes that are considered distant (e.g., greater than 3 steps away) from a control node in the network. Intuitively, minimizing the total shortest path distance from the control nodes to all other nodes in the network could produce a control set that also minimizes, or comes close to minimizing, the number of distant nodes. Further analysis confirmed this intuitive hypothesis, finding Pearson and Spearman correlation values of 0.94 and 0.98 respectively between the total shortest distance and the number of distant nodes. Minimizing the total minimum distance from all nodes to a control node, then, should produce a more effective control node set, as it also minimizes the number of distant nodes. One problem, however, is that the FAR heuristic does not achieve this minimization outright. Instead, the overall result depends on the seed value selected. While it is possible to consider

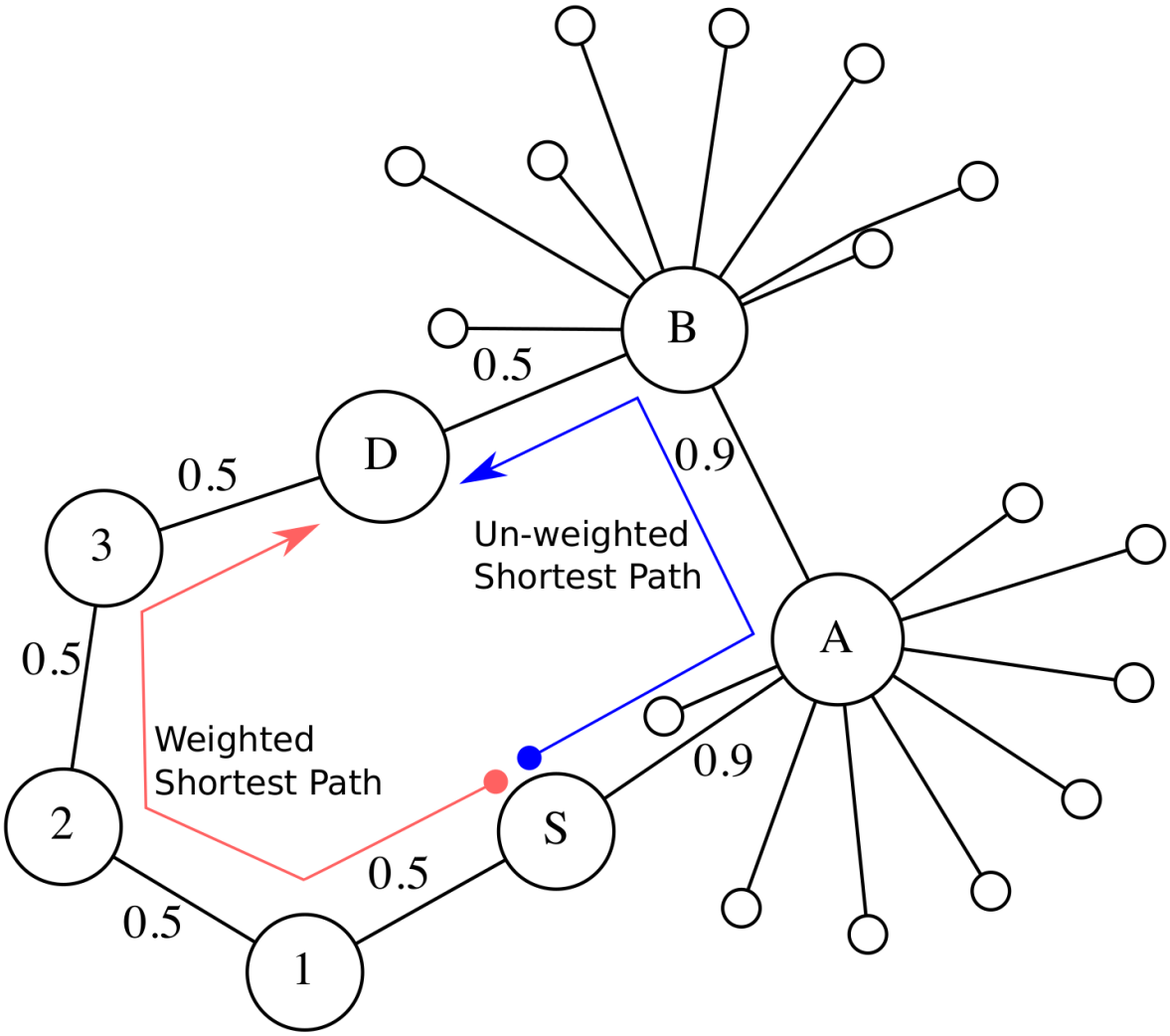


Figure 5.16: Difference in weighted and unweighted shortest path from node S to node D

the overall distance metric across each possible seed and choose the minimum, this still does not necessarily produce the true minimum, as the analysis in Section 5.5 will demonstrate. While minimizing these metrics over all possible control node sets is significantly more computationally expensive, the results presented in Section 5.5 indicate that this may result in higher levels of control success.

5.4.6 The Ideal Control Node Set

Based on the criticisms of the FAR heuristic, as well as the other analysis presented in the preceding sections, there are several qualities that should be exhibited by an ideal control node set selection algorithm:

- Maximizing Breadth of Influence: As indicated by the discussion presented in Sections 5.4.3 and 5.4.4, scenarios in which a large proportion of nodes are not being influenced by the control node set typically have a low level of control success. An ideal control node selection algorithm, then, should select a set of nodes that is capable of influencing a large proportion of nodes within the system. It is important to note that this type of influence maximization problem for control purposes may be different from the influence maximization investigated in much of the existing literature (e.g., those considering cascade problems similar to that of Kempe et al., 2005). As an example, some researchers have now started to analyze the 'timeliness' of influence within systems (Han et al., 2016), which is likely more important within the domain of control than a measure of a node's ability to spread an idea/cascade over an indefinite time period. As the system under control may naturally move over time toward failure, identifying control nodes that can influence the proper nodes in a timely manner is an important problem to consider. The FAR heuristic is capable of increasing the influence of a control node set to a degree, by selecting control nodes that are 'far away'. In general, this has the effect of minimizing any overlap between the influence of the control nodes, but the lack of any real influence information can lead to misleading results. The most significant example within the model considered here may be found in nodes with large degree, as they may be seen as presenting a short path between nodes within the network, while the influence dynamics actually make this path difficult to travel.

- Minimizing Per-Node Influence: While the ideal control node set should be capable of influencing a large proportion of the network, it may not be ideal for any one specific node to be contributing a significant portion of this influence. This can be seen in the last paragraph of discussion presented in Section 5.4.3, where controlling the largest degree node within scale-free networks, which are arguably the most influential nodes, resulted in extremely low control success.

- Non-heuristic: While the FAR algorithm allows control node sets to be selected with a relatively small amount of computational effort, its greedy heuristic nature can lead to sub-optimal solutions. Ideally, the selected control node set should represent the absolute best out of all of the possible combinations of control node sets. This, of course, requires significantly more computation, but should increase the likelihood of control success. It is possible that a heuristic search method could be applied to this optimization to more

efficiently find a near-optimal solution that still improves upon the purely heuristic nature of FAR, but this is not investigated in detail here.

5.5 Improving the FAR Heuristic

Based on the analysis of FAR presented previously, it is suspected that control set selection algorithms which account for influence dynamics and perform a true minimization may produce higher levels of control success. This section will present three modifications to the original FAR algorithm that either perform a minimization, account for influence dynamics, or both. The control success of these algorithms will then be compared across the various control scenarios available to determine which algorithm produce the most successful control sets overall. These results will provide evidence in support of the claims that minimization and the inclusion of influence dynamics within a control set selection algorithm lead to higher levels of control success overall.

5.5.1 Proposed FAR Variants

To evaluate the effect of minimization and influence weighting on the quality of control node sets selected, three variants to FAR are proposed: one which minimizes the hop-based distance, one which greedily selects using influence-weighted distance, and one that minimizes the influence-weighted distance. These three algorithms are described below and a summary of the defining characteristics of the four FAR variants is included in Table 5.7. The following sections will then compare the control success of these variants across the various networks, thresholds, and budgets to determine which of the variants produces the most successful controllers overall.

FAR_{min}: The FAR_{min} algorithm views distance from the same perspective as the original FAR heuristic (hop-based), but performs a minimization over the entire set of nodes. Algorithm 14 outlines this minimization process that is accomplished via depth-first search. For FAR_{min} control node selections, the distance function used on line 3 of this algorithm calculates the sum of the minimum distances from each node in the network to any one of the control set nodes, using hop-based distance. This solution produces the set of controllers within the budget that has the minimum total distance between the network nodes and the control nodes.

$InFAR$: $InFAR$ relies on a greedy approach, similar to that of the original FAR heuristic. The only change in the $InFar$ algorithm is how distance between nodes is calculated. In the case

Table 5.7: Breakdown of the defining characteristics for the four FAR variants that are investigated here

	Network Hops	**Influence-Weighted**
Greedy	FAR	$InFar$
Minimized	FAR_{min}	$InFAR_{min}$

of $InFAR$, the influence dynamics are accounted for within the distance metric by assigning a weight to an edge from node i to node j equal to $1-\frac{1}{deg(j)}$. This is the same weighting, based on the specification of the diffusion and learning equations of the real-valued voter model, that has been used previously in this work to represent the likelihood of influencing a neighbour's state in any given time step. While different weightings could be used to represent the influence dynamics (e.g., using transfer entropy analysis), these possibilities will not be investigated within this thesis.

$InFAR_{min}$: $InFAR_{min}$ uses the same minimization approach used by the FAR_{min} algorithm, but includes influence-weighted edges in the calculation. $InFar_{min}$, then, produces the controller set that meets the budget constraint and minimizes the total influence-weighted distance between nodes in the network and the control node set. The main difference between $InFar_{min}$ and FAR_{min}, then, is that $InFar_{min}$ uses a distance function (line 3 of Algorithm 14) that computes the minimal influence-weighted distance, instead of using a hop-based distance calculation.

5.5.2 Controller Distance - FAR vs. FAR_{min}

One of the main criticisms of the FAR heuristic from the previous section is that it does not perform a true minimization of the distance from all nodes to the nearest control node within the network, which in turn can result in more nodes being considered 'distant' from the control nodes than necessary. Table 5.8 presents the percentage difference in total distance between nodes and control nodes, as well as the number of distant nodes (using a distances threshold of 3 hops), using the FAR algorithm instead of the FAR_{min} algorithm. For each measure, the results include the average of all FAR control sets (across all seeds), as well as the control set seed that produced the minimal value for that measure. In addition to that, the average and maximum values realized across all network instantiations are included. From this table, it can be seen that the average case FAR seed

Algorithm 14 Backtracking algorithm for selection of control nodes using minimized FAR distance

Input: D - matrix of network's all-pairs shortest paths

Input: V_U - copy of set of network's nodes

Input: BGT - budget

Output: V_C - current set of control nodes, initially empty

Output: V_CMIN - set of minimal control nodes found to this point, initially empty

```
1: function FindMinFar(D,V_U,V_C, V_CMIN, BGT)
2:     if cost(V_C)=BGT then
3:         if dist(V_C)<dist(V_CMIN) then
4:             V_CMIN ← V_C
5:     else
6:         for v∈V_U do
7:             if cost(V_C∪v)≤BGT then
8:                 V_C ← V_C∪v
9:                 V_U ← V_U\v
10:                FindMinFar(D,V_U,V_C, V_CMIN, BGT)
11:                V_C ← V_C\v
12:                V_U ← V_U∪v
```

generates control sets that have higher total distance and number of distant nodes when compared to the control sets generated using the FAR_{min} algorithm, with average increases of 14.1% and 18% respectively. In addition to this, these values are also considered statistically significant when applying a two-sided T-test with $\alpha = 0.01$. In the case of the largest difference realized across all network instantiations, the percentage difference increases to 37.3% for distance and 45.3% for the number of distant nodes. As mentioned in the previous section, it is possible to incur additional computation cost relative to the number of nodes in the network and calculate the seed node which produces the minimal distance value using the FAR heuristic, which is an approximation of the true minimum computed by the FAR_{min} algorithm. The Minimal Far Seed columns of Table 5.8 compare the control sets generated using this method to those created with the FAR_{min} algorithm. While the difference was not found to be statistically significant across all network instantiations, there exists instantiations in which this approach results in a percentage increase of 12.7% for total distance and 17% for the number of distant nodes. So even when using the more computationally

Table 5.8: Percentage difference in overall distance and number of distant nodes between FAR and FAR_{min} using either the average of all FAR control sets or the seed which produced minimal values for each metric

	Average FAR		**Minimal FAR Seed**	
Metric	**Average**	**Max**	**Average**	**Max**
Distance	14.1%	37.3%	2.2%	12.7%
Distant Nodes	18.0%	45.3%	2.7%	17.0%

expensive technique of selecting the 'best' seed node for the FAR algorithm, there could certainly be a significant affect on control success when the results of Figure 5.15 are considered (see page 114).

5.5.3 FAR Variant Comparison

To compare the performance of the four FAR variants, the first thing that will be considered is the percentage of times each variant was assigned each possible rank across the many scenarios simulated (different network instantiation, threshold, and budget values) using the ranking algorithm described in Algorithm 12. Table 5.9 shows the percentage of scenarios each FAR variant was assigned each rank when considering scenarios in which at least one algorithm achieved an overall success percentage of 25% with the specified network/threshold/budget combination. This limitation was used to avoid scenarios in which control was nearly impossible for all algorithms (i.e., the lowest threshold and budget combinations) from skewing the results. It should be noted that the rows of this table sum to 100%, while the columns do not necessarily, as the case of ties allow multiple algorithms to be ranked the same. The bolded values within the table signify the most frequent rankings associated with each variant, showing that the two minimized variants are most frequently ranked 1st, while the greedy variants are most frequently ranked 3rd. In addition to this, these values show that in a majority of the scenarios considered, the minimized variants are ranked in the top two, while the non-minimized variants are ranked in the bottom two. Finally, this table shows that the $InFAR_{min}$ variant is ranked first in 5% more scenarios than the FAR_{min} variant.

As the ultimate goal of control is to avoid any system failure, another interesting view of the FAR variant rankings is presented in Table 5.10, which uses a minimum success threshold of 100%. In this case, the only variants that are ever ranked first are those that perform a minimization. That is to say, there are no cases in which a non-minimized variant achieved 100% control success, while the minimized variants were responsible for achieving 100% control success in 10.9% of

Table 5.9: Percentage of scenarios each variant achieved the specified rank when a minimum overall success threshold of 25% was used (548 scenarios, 46.1% of the 1188 scenarios in total)

	Percent of Scenarios Rank Was Achieved			
Variant	**First**	**Second**	**Third**	**Fourth**
FAR	13.7	22.1	**36.7**	27.6
FAR_{min}	**40.3**	20.8	16.1	22.8
$InFAR$	9.9	29.4	**37.6**	23.2
$InFAR_{min}$	**45.4**	19.2	17.2	18.2

Table 5.10: Percentage of scenarios each variant achieved the specified rank when a minimum overall success threshold of 100% was used (130 scenarios, 10.9% of the 1188 scenarios in total)

	Percent of Scenarios Rank Was Achieved			
Variant	**First**	**Second**	**Third**	**Fourth**
FAR	0.0	13.1	48.5	38.5
FAR_{min}	60.8	13.1	1.5	24.6
$InFAR$	0.0	23.8	46.2	30.0
$InFAR_{min}$	73.8	15.4	3.8	6.9

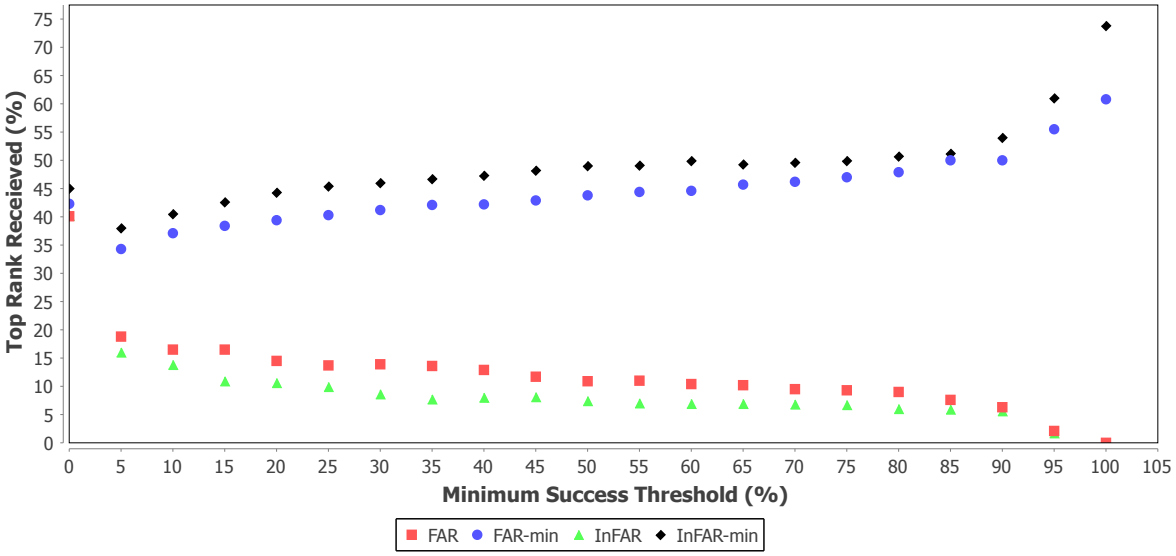


Figure 5.17: Percent of top ranks received for each FAR variant over different minimum thresholds of success

the scenarios considered. Figure 5.17 further demonstrates the divergence of the percentage of top ranks between the minimizing and non-minimizing FAR variants, with a generally consistent ranking of $InFAR_{min} > FAR_{min} > FAR > InfFAR$.

Table 5.11: Statistically significant difference in control rank and success percentage using a paired T-test ($\alpha = 0.05$) and limited to scenarios in which one variant achieved at least 25% control success – an X represents that the variant in the row performs better than the variant in the column

	Rank Significantly Different				**Success Significantly Different**			
	FAR	FAR_{min}	$InFAR$	$InFAR_{min}$	FAR	FAR_{min}	$InFAR$	$InFAR_{min}$
FAR	—				—		X	
FAR_{min}	X	—	X			—		
$InFAR$			—				—	
$InFAR_{min}$	X		X	—				—

While the previous results demonstrated that the minimizing variants were most frequently ranked as the top control set selection algorithm across all scenarios, is there a statistically significant difference between the different variants? To answer this question, the rank and control success percentage for each FAR variant was compared to the other variants using a two-tailed paired (same network/budget/threshold) T-test with an alpha value of 0.05. Table 5.11 shows the comparisons that resulted in statistically significant differences when a minimum success threshold of 25% was used, where an X represents the variant listed in that row being significantly better than the variant in the associated column. This data confirms the previous rank observations by finding that the two minimizing variants are ranked higher than the non-minimizing ones. When the average control success percentage was compared, the only statistically significant difference was found in comparing the original FAR variant to the influence-weighted greedy variant $InFAR$. The percentage difference between the two overall (1.23%), however, was still rather small. So while the ranks of the minimized variants are consistently better than those of the non-minimized variants, it is not possible to draw a strong conclusion regarding the overall control success percentage. The failure to disprove the null hypothesis of the T-tests, though, could be due to the large variance seen across the different network types and instantiations.

Once again, it may be of interest to consider only those scenarios in which at least one of the variants achieved 100% control success. The same significance tests were performed on these scenarios and the results are included in Table 5.12. In this case, there is a clear, statistically significant, rank-based ordering of $InFAR_{min} > FAR_{min} > InFAR > FAR$. In addition to this, the $InFAR_{min}$ algorithm also produces a statistically significant increase in the overall control success percentage across these scenarios, while the others do not distinguish themselves in a statistically significant way. Based on these results, then, it seems that variants which perform an

Table 5.12: Statistically significant difference in control rank and success percentage using a paired T-test ($\alpha = 0.05$) and limited to scenarios in which one variant achieved 100% control success – an X represents the variant in the row performs better than the variant in the column

	Rank Significantly Different				**Success Significantly Different**			
	FAR	FAR_{min}	$InFAR$	$InFAR_{min}$	FAR	FAR_{min}	$InFAR$	$InFAR_{min}$
FAR	—				—			
FAR_{min}	X	—	X			—		
$InFAR$	X		—				—	
$InFAR_{min}$	X	X	X	—	X	X	X	—

absolute minimization are a better choice than those that do not, and also variants that account for influence dynamics within the system outperform the variants that do not.

5.6 Summary

This chapter introduced a new view of network control, called distribution-based control, which uses distributions as targets for control instead of state vectors. The chapter also formalized a failure avoidance problem within the distribution-based control framework and investigated control success within the domain of this problem across a wide range of theoretical and real-world networks. The analysis presented within the chapter identified the maximum node degree and network diameter as being the strongest predictors of network control difficulty and evidence was presented to determine why this was the case. This analysis demonstrated the importance of including influence properties when considering network control success by demonstrating that the most difficult scenarios to control were those in which the control node set was unable to influence a large proportion of the nodes within the network. Based on these findings, several criticisms of the FAR heuristic, which was found to be the best of a number of investigated algorithms by Runka (2016), were presented. Several variants on the original FAR heuristic, including those which include influence dynamics in the control node selection process, were proposed and the control success of each algorithm was compared. The final conclusion was that the control node set selection algorithm that included influence-weighted edges and performed an absolute minimization resulted in significantly improved control success when compared to the original FAR heuristic and the other variants considered.

While this chapter made several interesting knowledge contributions to the area of network control, there are several important questions that should be considered in the future. For example, the rudimentary controllability analysis that was presented in Section 5.2.3 could be expanded

upon to produce a more accurate estimation of the likelihood of control success. Additionally, further influence-based control node set selection algorithm development should be carried out. The following chapter will highlight the key results of this thesis and propose future research directions related to distribution-based control, network control in general, and influence measurement.

Chapter 6

Conclusion

This dissertation had two main objectives. The first of these was to investigate the use of transfer entropy as a tool for both measuring influence and inferring the existence of influential links within a social network. The second of these objectives was to gain a better understanding as to why certain networks are harder to control than others. Influence measures based on transfer entropy measurements were formulated and found to be highly correlated with theoretical influence measures derived from the simulated model under consideration. Additionally, through analysis of state change over time within specific node subsets, evidence was generated to support claims that certain network properties have a negative affect on network control success. The remainder of this chapter will summarize the contributions of this work (Section 6.1) and provides insights into future research directions (Section 6.2).

6.1 Summary of Results

This thesis made several important contributions to the areas of influence measurement, network inference, and network control. These contributions were outlined in Section 1.4. The remainder of this section will discuss these contributions in more detail, highlighting the most significant results.

6.1.1 Anti-Majority Game

To analyze the use of transfer entropy for influence measurement and network inference, it was desirable to have a simulated model with known properties. To this end, the first contribution of this thesis was the formalization of the anti-majority game. This game requires agents to make decisions based on information shared through network connections. While the anti-majority game and agent models used within this work were purposefully implemented in a straightforward manner, these models can be easily extended to incorporate more complex agent behaviour and game rules. This game, then, is not only applicable to the work presented within this thesis, but could also be used to further study influence, social networks, and network control.

6.1.2 Transfer Entropy Influence Measurement

One deficiency within the existing influence measurement literature that was discussed in Section 3.2, was a lack of behaviour-based influence measures. This deficiency was addressed to some degree through the work of Ver Steeg and Galstyan (2013), who used transfer entropy measurements calculated on a Twitter dataset to measure influence. This existing work, however, did not provide any in-depth analysis regarding the relationship between transfer entropy values and other influence measures. Additionally, while the existing work identified several key links that exist within the network through transfer entropy values, it did not consider the problem of inferring all existing links using this approach. This thesis addressed both of these issues by computing transfer entropy values on time series data generated through simulation of the anti-majority game. As this game has known properties, it was possible to formulate theoretical influence measures for the global (system-wide) and local (node specific) influence of any node, based on the underlying properties of the simulated model. This work proposed methods for computing global and local influence measures for a node using transfer entropy values and compared those values to these theoretical influence measures. In comparing the theoretical and transfer entropy measures across a range of network types and instantiations, a high degree of linear correlation was observed, with average correlations over all networks being 0.856 in the local influence case and 0.727 in the global influence case. These correlation results are strong enough to motivate future research in the use of transfer entropy for influence measurement, as discussed later in this chapter.

6.1.3 Transfer Entropy Influence Network Inference

The second contribution of the transfer entropy work presented within this thesis was the investigation of methods for predicting which links exist within a social network based on agent behaviour over time. Knowledge relating to the connections within a network and their underlying properties are useful, and in some cases a prerequisite, when applying different types of network control. In practice, however, connection information may be unavailable or misleading due to the lack of relationship information available. The prediction methods investigated as part of this thesis, then, could be used to generate a prediction of an unknown network or to gain improved information regarding the connections of a known network (e.g., by pruning edges that are not actually influential), as is discussed in Section 6.2.1.

The original transfer entropy influence work of Ver Steeg and Galstyan (2013) identified a small

number of links that were the most obvious outliers within the set of all transfer entropy values. As a known model and network was used within this thesis, prediction of links can be investigated in a much more detailed way, with exact prediction accuracy being easily calculated. Here, a thresholding approach was used, in which a transfer entropy threshold value was algorithmically selected and any pair of nodes with a transfer entropy value higher than this threshold was predicted as being connected within the network. Analysis of the best case prediction scenarios across a wide variety of networks demonstrated that a high F-score was attainable across all theoretical network types and several real-world network samples. Some real-world networks produced a low best case F-score, but verification of the cause of these low scores and proposal of improvements are left as future work. Several threshold selection algorithms were proposed, which require different degrees of system knowledge. It was found that the assumed degree and sampled edge prediction methods produced the most accurate predictions, with an average of 98.8% and 99.0% of the best case F-score being realized respectively. The assumed degree solution, however, required the exact average degree of the network to be known. The sampled edges approach, on the other hand, only required a sample of 5% of the network nodes to achieve this accuracy, and was even shown to produce highly accurate prediction with a sample of only 1% of nodes. Finally, a kernel density estimate (KDE) algorithm was investigated, which required absolutely no network knowledge. This method did not perform as well as others, producing a prediction accuracy that was only 79% of the best case F-score on average. However, the most significant benefit of the KDE approach is that is requires no assumed or sampled knowledge of the network. Improving upon this initial KDE approach and the introduction of additional zero-knowledge solutions are left as future work.

6.1.4 Distribution-Based Control

A significant focus of existing network control research has centered on the idea of 'full state controllability', with the goal of moving the system between two specific state vectors. This thesis proposed distribution-based control, which eschews the commonly used vector-space model of system state, and instead, uses parameterized distributions as targets for control. Arguably, using distributions allows for a more expressive way of defining control targets and system states. Additionally, defining a single distribution control target can encapsulate many different acceptable state vectors, which is a more logical choice for many problem types that do not require a specific state vector to be reached. Another possible benefit to a distribution-based network control

approach is that the parameters of a distribution can be estimated using a sample of the network states, instead of requiring the state of all nodes to be observable. This thesis has demonstrated that it is possible to control a system using this method and has also briefly discussed how existing NCP problems could be adapted to fit into the distribution-based control view of network control.

6.1.5 Network Controllability Analysis

The existing NCP work compared the success rate of a number of different control node selection heuristics, but did not thoroughly investigate what types of networks were more/less controllable. This thesis addressed this problem by considering the success rates of learned controllers over a wide range of networks and identifying possible network properties that lead to decreased control success. Using decision tree algorithms and correlation values, two network properties were identified as having a negative effect on control success: the presence of large degree nodes and a large network diameter. This thesis provided in-depth analysis to determine why these two network properties led to such a decrease in control success. Through analysis of subsets of system state over time within a simulation, it was shown that in cases with large degree nodes or high network diameters, the control nodes were incapable of influencing the state distribution of a large portion of nodes. As the number of nodes that cannot be influenced grows larger, the difficulty of the control problem also increases, as there are less nodes that can be controlled to offset the lack of control in other parts of the network. This information will be useful in future research related to improved control node selection algorithms, control success prediction methods, and additional control strategies, such as network modification.

6.1.6 Improved Control Node Selection Solutions

Within the original NCP research, the best network control performance was produced using the FAR heuristic for control node selection. The final contribution made by this thesis was the proposal and comparison of three new FAR variants for control node selection. Based on the previously discussed results, it was determined that two of the ideal characteristics of a control node selection algorithm were the incorporation of influence measures and the use of an absolute optimization, as opposed to the greedy, heuristic nature of the original FAR algorithm. When the control success of these different variants was compared, it was found that those variants that factor influence dynamics within the selection of control nodes outperformed those that did not include

influence dynamics. Additionally, it was found that variants that perform a complete minimization outperformed those that used a greedy approach. When considering only those control scenarios where at least 1 control set selection algorithm produced successful control on all test cases, it was found that the influence-based minimization variant provided a statistically significant improvement to control success over all other variants. These results should motivate future research to incorporate influence dynamics and optimization within control node set selection algorithms.

6.2 Future Research

Research related to the work presented in this thesis should continue in several directions. This section will discuss several of these possible research directions. While the possible directions included here are certainly not exhaustive, this section represents some of the most important research directions that have been identified during the production of this thesis.

6.2.1 Transfer Entropy and Network Prediction

The transfer entropy network analysis conducted as part of this thesis was carried out using a rather simple model. As has been discussed previously, this was advantageous for the current work, as it allowed for the most straightforward calculations to be used. Further work within this area should consider more complex theoretical problems and also investigate the use of these methods on more real-world problems (e.g., as was done with the Twitter dataset by Ver Steeg and Galstyan, 2013). In doing so, the accuracy of the already proposed methods can be verified, while also producing more information related to the convergence rates and the distributions of transfer entropy values. Through analysis of this additional information, it may be possible to develop more advanced prediction mechanisms. For example, if the distribution of neighbour and non-neighbour transfer entropy values are similar across many problems, it may be possible to represent the entire population of transfer entropy values as a mixture model. In that case, it could be possible to label each value as belonging to one particular class (influential or not) within the mixture model (Kalai et al., 2012).

Another possible extension to the transfer entropy work presented here would be the consideration of prediction methods that are more advanced than the threshold approach. It may be possible to combine transfer entropy values with other information useful in determining the existence of connections between participants. For example, the idea of triadic closure (Easley and Kleinberg, 2010), which states that the existence of strong links between A/B and A/C within

a social system increase the likelihood that exists a link between B/C, could be used as additional evidence within a prediction algorithm. Furthermore, by computing transfer entropy measurements using different values of the time window parameter, o, it may be possible to gain additional information regarding the possible relationships within a network. For example, A may influence B when using a time window of 1 step, while A influences C when using a time window of 2 steps. This could indicate that A influences C over 2 steps or that B influences C over 1 step (further transfer entropy values, such as the 1-step value from B to C could provide more evidence). It may be possible to combine all of this information using a learning or optimization algorithm to produce more accurate predictions than the threshold approach.

6.2.2 Controllability Analysis

An important problem posed previously within this thesis was: how can the likelihood of control success be determined in NCP-like network control problems? By producing a better understanding of how different network properties affect control success likelihood, this thesis has made progress toward answering this question. The use of parameterized distributions within the distribution-based control problem could also be beneficial when considering this problem, as the distance between distributions can be computed. In doing so, it is possible to produce measurements for the rate of change of a system by considering the distance between distributions at different points in time. This idea was discussed briefly in Section 5.2.3, but was not considered in detail. Future work, however, could apply probabilistic analysis using this type of information in order to estimate the likelihood of control success. Future work could consider framing the control success problem as a Gambler's Ruin problem, which has been used in other domains such as the analysis of genetic algorithms (Harik et al., 1999) and business growth/failure (Coad et al., 2013). This type of analysis could combine measurements of the system's natural velocity and a controller's ability to affect that velocity to produce an estimate of control success probability over some time interval. One significant step that would be required to maximize the utility of this type of analysis is the development of a method for measuring the effect of a controller on a system. The velocity of the system itself can be estimated through simulation (as is done in this thesis), observation, or analysis of available system information, while the effect of a controller may be more difficult to determine. Still, this type of analysis may be useful in providing a measure of the difficulty of a control problem, regardless of any specific controller. As different control methods are investigated and compared,

this information could then be used in selecting an effective controller from a set of possible options.

6.2.3 Network Modification

While the focus of this work, as well as much of the previous NCP work, has been on the selection of control node sets and the injection of control signals into the network, an additional direction for control research may lie in network modification. This thesis has presented significant evidence indicating that certain network properties negatively impact control success. Based on these results, an additional method of improving control beyond selecting better control nodes/signals, is to select and introduce network modifications that decrease control difficulty. Using the results presented within this thesis as a starting point, it may be possible to improve control success by decreasing the network diameter or by introducing additional information flow paths within the neighbourhood of very high degree nodes. Future work should consider the effect that these changes may have on information flow and control success, which could lead to network modification algorithms to improve controllability.

6.2.4 Scaling

The network control experiments conducted as part of this thesis and within the original NCP work have used relatively small network sizes. Further network and information flow analysis should aim to improve network control solutions to a point where they can be applied to extremely large networks, as are commonly found in practice (e.g., social networks, financial systems). The results presented within Chapter 5 of this thesis demonstrated that a control node has a limited area of effect within the network. Further research should consider how this information can be used to decompose a network into (possibly many) components that present minimal overlap of control influence. This approach could then be used to develop distributed control architectures capable of scaling to meet the requirements of larger networks. This could, in turn, allow for different control approaches to be applied within the system. For example, the single signal control method used here was limited in its ability, but a multi-signal control method presented a computationally infeasible learning problem. By distributing controllers throughout a network, while also minimizing the interactions of the separate controllers' signals, it may be possible to have multiple learners executing the single signal approach within their own area of the network.

6.2.5 Control Problem Diversity

Finally, in moving forward, research should consider both different types of network control problems, as well as different mechanisms to achieve control. The focus of this thesis and the previous NCP work has involved maintaining the system state within some threshold distance of a target state. Other control problems, however, may involve limiting the rate of change of system state, moving between desired system states, or some other control objective. While investigating the feasibility of these types of problems in general, future work could also attempt to determine how well the existing heuristics, algorithms, and analysis apply across the different types of control problems. Intuitively, it would be expected that network and information flow properties would affect different control problems in a similar way, but it is also quite possible that different types of control problems present different challenges in this respect. Furthermore, the existing research has focused on one specific method of control, which involves injecting signals into the network through control nodes. Other forms of control, however, are possible, and may even be more suitable for some network control problems. One type of control that may be of interest involves the limitation of information flow within the network. This could be used to regulate the ability of specific nodes to cause changes within the system. This type of control could be especially important within control problem types in which the goal is to limit the rate of change of the system. There is also the possibility of combining several control methods within a single control system. For example, limiting the information flow from large degree nodes, which were recognized as problematic within this work, could reduce the negative effect of those nodes within the system. At the same time, control signals could be injected into the network to achieve the necessary state modification.

6.3 Summary

In summary, this thesis has achieved the objectives outlined in its problem statement (Section 1.2). This thesis has demonstrated that it is possible to measure influence and infer social network connections from activity traces produced by the network participants. Additionally, this thesis has shown that it is possible to control the distribution of state values within a social network system. Finally, this thesis proposed algorithms for control node selection that demonstrably improved upon the existing state of the art FAR heuristic.

Appendix A

Network parameters for all network instantiations

The tables within this appendix detail the network parameter values for all network instantiations considered within this thesis. Different tables are included for the 400 node theoretical networks used in the transfer entropy work of Chapter 4 (Table A.1) and the 100 node theoretical networks used in the network control work of Chapter 5 (Table A.2). The final table includes the network parameters for the 100 node real-world networks that were used consistently through all the presented research.

Table A.1: Network properties for theoretical network instantiations used in transfer entropy work (Chapter 4)

Network	**Diameter**	**Average Degree**	**Max Degree**	**Average Betweenness**	**Max Betweenness**	**CC**
Random-8 #0	5.00	7.97	15.00	421.89	1301.32	0.02
Random-8 #1	5.00	7.96	17.00	421.90	1590.31	0.02
Random-8 #2	5.00	7.96	16.00	421.53	1517.66	0.02
Random-8 #3	5.00	7.99	18.00	421.74	1805.36	0.03
Random-8 #4	6.00	7.97	16.00	421.67	1495.53	0.02
Random-8 #5	5.00	7.97	21.00	422.23	2194.96	0.02
Random-8 #6	6.00	7.95	20.00	422.09	2261.27	0.02
Random-8 #7	5.00	7.96	16.00	420.64	1389.13	0.01
Random-8 #8	6.00	7.96	16.00	422.99	1414.64	0.02
Random-8 #9	5.00	7.96	17.00	421.81	1675.00	0.02
Random-16 #0	4.00	15.85	28.00	294.01	881.94	0.04
Random-16 #1	4.00	15.85	29.00	293.82	932.19	0.04
Random-16 #2	4.00	15.81	28.00	294.74	889.28	0.04
Random-16 #3	4.00	15.85	26.00	294.15	742.18	0.04
Random-16 #4	4.00	15.85	28.00	293.93	855.02	0.04

Network	Diameter	Average Degree	Max Degree	Average Betweenness	Max Betweenness	CC
Random-16 #5	4.00	15.82	28.00	294.08	912.51	0.04
Random-16 #6	4.00	15.81	28.00	293.88	833.39	0.04
Random-16 #7	4.00	15.88	28.00	293.80	831.20	0.04
Random-16 #8	4.00	15.86	30.00	293.83	1068.61	0.04
Random-16 #9	4.00	15.88	27.00	293.70	823.57	0.04
Scale Free #0	9.00	2.95	50.00	633.71	33943.55	0.03
Scale Free #1	9.00	3.17	47.00	597.73	26099.21	0.03
Scale Free #2	9.00	3.15	38.00	631.55	18202.68	0.02
Scale Free #3	9.00	3.08	55.00	633.41	35401.13	0.03
Scale Free #4	9.00	3.14	47.00	599.78	25124.09	0.02
Scale Free #5	10.00	3.13	37.00	676.48	25932.87	0.01
Scale Free #6	9.00	3.06	48.00	643.17	29159.91	0.02
Scale Free #7	9.00	3.06	32.00	678.73	14626.32	0.02
Scale Free #8	9.00	3.27	47.00	618.43	32363.92	0.02
Scale Free #9	8.00	3.22	37.00	590.41	18542.25	0.03
Small-0.1 #0	7.00	8.00	11.00	658.50	4025.77	0.48
Small-0.1 #1	7.00	8.00	11.00	651.40	3173.51	0.48
Small-0.1 #2	8.00	8.00	11.00	649.90	2834.39	0.47
Small-0.1 #3	7.00	8.00	12.00	648.43	4341.36	0.48
Small-0.1 #4	8.00	8.00	10.00	682.87	3205.95	0.50
Small-0.1 #5	8.00	8.00	12.00	656.16	4225.80	0.48
Small-0.1 #6	8.00	8.00	11.00	664.39	3431.01	0.48
Small-0.1 #7	7.00	8.00	11.00	639.56	3329.10	0.47
Small-0.1 #8	7.00	8.00	10.00	645.64	3339.17	0.47
Small-0.1 #9	7.00	8.00	11.00	643.51	2397.00	0.47
Small-0.2 #0	6.00	8.00	12.00	536.38	2043.37	0.33
Small-0.2 #1	7.00	8.00	11.00	541.80	1991.50	0.35

Network	Diameter	Average Degree	Max Degree	Average Betweenness	Max Betweenness	CC
Small-0.2 #2	6.00	8.00	12.00	542.38	2096.89	0.35
Small-0.2 #3	6.00	8.00	12.00	540.27	1829.51	0.34
Small-0.2 #4	6.00	8.00	14.00	540.36	2628.89	0.34
Small-0.2 #5	6.00	8.00	13.00	528.25	2143.99	0.32
Small-0.2 #6	6.00	8.00	13.00	534.05	2186.91	0.33
Small-0.2 #7	6.00	8.00	12.00	531.40	2374.95	0.34
Small-0.2 #8	6.00	8.00	12.00	542.14	2155.16	0.36
Small-0.2 #9	6.00	8.00	13.00	540.40	2208.40	0.35

Table A.2: Network properties for theoretical network instantiations used in network control work (Chapter 5)

Network	Diameter	Average Degree	Max Degree	Average Betweenness	Max Betweenness	CC
Random-3.1 #0	10.00	3.38	8.00	150.16	753.00	0.02
Random-3.1 #1	11.00	3.10	9.00	166.88	880.77	0.02
Random-3.1 #2	9.00	3.12	9.00	159.75	707.51	0.06
Random-3.1 #3	12.00	2.70	7.00	196.97	898.87	0.01
Random-3.1 #4	12.00	3.00	7.00	181.02	777.20	0.05
Random-3.1 #5	10.00	2.94	6.00	174.10	767.59	0.03
Random-3.1 #6	9.00	3.18	10.00	151.16	1010.27	0.02
Random-3.1 #7	10.00	2.76	8.00	184.23	875.92	0.02
Random-3.1 #8	8.00	3.38	9.00	140.42	836.88	0.04
Random-3.1 #9	9.00	3.34	9.00	148.58	671.14	0.02
Scale Free #0	6.00	3.10	38.00	106.07	2992.09	0.10
Scale Free #1	7.00	3.48	20.00	112.17	1577.25	0.05

Network	Diameter	Average Degree	Max Degree	Average Betweenness	Max Betweenness	CC
Scale Free #2	7.00	3.26	22.00	113.81	2057.15	0.06
Scale Free #3	6.00	3.02	23.00	111.27	1978.88	0.04
Scale Free #4	7.00	3.08	23.00	116.55	1916.15	0.06
Scale Free #5	7.00	3.40	20.00	116.54	1353.18	0.05
Scale Free #6	6.00	3.16	22.00	115.05	1753.51	0.09
Scale Free #7	8.00	3.08	29.00	115.89	2770.09	0.06
Scale Free #8	7.00	2.94	29.00	113.76	2647.29	0.11
Scale Free #9	7.00	3.00	34.00	113.39	3264.77	0.10
Small-0.1 #0	10.00	4.00	5.00	209.71	999.03	0.36
Small-0.1 #1	9.00	4.00	7.00	168.26	721.45	0.33
Small-0.1 #2	10.00	4.00	6.00	186.85	873.12	0.35
Small-0.1 #3	9.00	4.00	6.00	183.92	745.73	0.33
Small-0.1 #4	10.00	4.00	6.00	189.82	555.65	0.35
Small-0.1 #5	10.00	4.00	5.00	198.82	622.58	0.36
Small-0.1 #6	11.00	4.00	6.00	223.72	1142.73	0.43
Small-0.1 #7	12.00	4.00	6.00	218.29	960.79	0.39
Small-0.1 #8	9.00	4.00	6.00	178.28	1001.24	0.35
Small-0.1 #9	10.00	4.00	6.00	200.38	1089.76	0.37

Table A.3: Network properties for real-world network instantiations used in both the transfer entropy work (Chapter 4) and network control work (Chapter 5)

Network	Diameter	Average Degree	Max Degree	Average Betweenness	Max Betweenness	CC
Facebook #0	6.00	17.16	50.00	61.52	508.79	0.55
Facebook #1	4.00	41.00	86.00	31.06	185.81	0.65

Network	**Diameter**	**Average Degree**	**Max Degree**	**Average Betweenness**	**Max Betweenness**	**CC**
Facebook #2	4.00	34.54	67.00	35.66	195.28	0.61
Facebook #3	5.00	23.52	54.00	57.91	399.69	0.68
Facebook #4	4.00	21.70	62.00	46.90	430.76	0.60
Facebook #5	5.00	15.60	45.00	64.03	422.00	0.51
Facebook #6	7.00	11.38	42.00	87.43	1247.59	0.53
Facebook #7	5.00	20.32	51.00	57.57	569.21	0.60
G+ #0	14.00	4.20	18.00	209.13	1418.01	0.27
G+ #1	8.00	23.32	65.00	69.01	812.66	0.54
G+ #2	18.00	2.66	18.00	273.64	4019.89	0.15
G+ #3	22.00	25.02	60.00	205.88	1829.11	0.61
G+ #4	11.00	13.12	58.00	88.21	1325.22	0.51
G+ #5	14.00	3.74	43.00	195.13	4105.00	0.32
G+ #6	10.00	6.50	22.00	167.99	1951.42	0.36
G+ #7	16.00	6.78	30.00	207.13	1582.76	0.42
G+ #8	17.00	2.96	10.00	273.21	2315.57	0.18
G+ #9	12.00	15.46	66.00	107.26	1190.23	0.61
G+Similar #0	20.00	8.74	36.00	276.64	2832.67	0.16
G+Similar #1	17.00	11.70	28.00	172.39	1010.24	0.15
G+Similar #2	18.00	8.24	33.00	190.22	1708.30	0.13
G+Similar #3	11.00	10.08	34.00	111.31	1643.94	0.14
G+Similar #4	16.00	12.06	31.00	161.08	2090.78	0.16
G+Similar #5	17.00	10.66	46.00	184.56	1385.73	0.12
G+Similar #6	12.00	10.38	42.00	130.26	2792.46	0.16
G+Similar #7	23.00	8.30	30.00	253.32	2566.47	0.17
G+Similar #8	13.00	11.30	37.00	168.64	2545.70	0.15
G+Similar #9	16.00	10.74	34.00	167.28	1453.25	0.17
Twitter #0	3.00	33.52	73.00	34.31	222.90	0.61

Network	Diameter	Average Degree	Max Degree	Average Betweenness	Max Betweenness	CC
Twitter #1	4.00	13.22	72.00	53.56	1441.93	0.49
Twitter #2	5.00	7.28	46.00	72.45	1567.90	0.40
Twitter #3	3.00	28.86	77.00	38.66	235.23	0.61
Twitter #4	4.00	22.04	58.00	43.60	476.54	0.53
Twitter #5	4.00	20.28	61.00	43.89	454.53	0.53
Twitter #6	9.00	4.82	22.00	126.24	1219.79	0.35
Twitter #7	3.00	18.96	91.00	41.30	947.32	0.75
Twitter #8	5.00	16.78	35.00	55.88	267.08	0.47
Twitter #9	5.00	10.30	47.00	63.09	854.79	0.38

Appendix B

Complete Network Control Success Rankings

Section 5.4.1 presented a summary of the network controllability rankings for each network type. Table B.1 provides the complete ranking results of each network instantiation when the FAR heuristic was used for control node set selection. These rankings were computed over all of the budget and threshold combinations considered. Additionally, Table B.1 includes summary information relating to the overall control success achieved when controlling that specific network instantiation.

Table B.1: Ranking and control success information for all network instantiations

Overall Rank	**Network**	**Average Control Success (%)**	**SD Control Success (%)**	**Median Control Success (%)**
1	Small-0.1 #0	70.6	39.9	95.8
2	Small-0.1 #4	69.8	38.8	92.0
3	Small-0.1 #2	70.1	39.1	93.6
4	Small-0.1 #5	69.8	39.2	92.1
5	Small-0.1 #7	69.1	39.1	91.3
6	Small-0.1 #3	63.6	40.1	87.0
7	Small-0.1 #8	67.3	39.4	89.9
8	Small-0.1 #1	63.2	39.5	87.8
9	Small-0.1 #9	63.3	40.2	84.6
10	Random-3.1 #8	54.8	37.6	68.2
11	Twitter #0	49.8	38.4	58.1
12	Small-0.1 #6	62.4	38.8	83.8
13	Random-3.1 #2	56.3	36.3	72.3
14	Random-3.1 #9	54.3	38.0	65.1
15	Twitter #4	44.7	39.5	43.8
16	Facebook #2	45.1	40.5	46.4

Overall Rank	Network	Average Control Success (%)	SD Control Success (%)	Median Control Success (%)
17	Twitter #3	43.8	39.9	41.8
18	Random-3.1 #0	49.3	36.1	58.0
19	Random-3.1 #5	44.4	40.6	33.9
20	Random-3.1 #6	49.5	35.6	56.7
21	Facebook #1	48.1	38.9	52.3
22	Twitter #5	43.0	38.7	37.7
23	Random-3.1 #1	47.3	35.1	52.7
24	Facebook #5	43.1	38.2	41.1
25	Twitter #8	43.5	38.2	40.1
26	Facebook #3	43.7	38.4	44.4
27	Facebook #4	41.0	39.7	33.6
28	Random-3.1 #3	41.6	37.2	37.9
29	Random-3.1 #7	39.7	37.3	32.4
30	Twitter #1	37.6	39.4	22.8
31	Random-3.1 #4	39.7	32.6	40.7
32	Scale Free #5	36.9	35.9	31.9
33	Scale Free #1	35.9	36.4	29.0
34	Scale Free #4	35.0	35.3	26.7
35	Scale Free #6	34.9	35.8	23.8
36	Facebook #6	31.0	31.4	24.1
37	Facebook #7	35.0	34.9	25.8
38	Twitter #7	33.7	39.1	8.9
39	Scale Free #3	32.7	36.1	18.5
40	Scale Free #2	34.0	36.7	21.2
41	Facebook #0	32.0	35.6	16.3
42	Scale Free #9	30.8	34.3	16.5
43	Twitter #2	31.3	36.5	14.4

Overall Rank	Network	Average Control Success (%)	SD Control Success (%)	Median Control Success (%)
44	Scale Free #7	30.1	34.2	17.7
45	Twitter #6	26.8	31.8	9.0
46	G+ #0	19.4	22.6	12.5
47	Twitter #9	26.8	32.7	11.8
48	Scale Free #8	26.2	34.8	1.9
49	Scale Free #0	19.8	30.0	2.1
50	G+ #8	9.9	12.8	5.1
51	G+Similar #3	9.0	14.9	1.5
52	G+ #6	5.1	5.1	4.3
53	G+ #4	10.7	16.1	3.0
54	G+ #1	9.4	12.6	3.5
55	G+Similar #9	3.3	4.1	1.8
56	G+Similar #2	4.8	7.2	1.3
57	G+Similar #4	6.4	12.1	0.0
58	G+Similar #8	3.2	4.8	0.8
59	G+ #9	1.6	2.0	0.5
60	G+ #2	2.1	3.2	0.3
61	G+Similar #7	1.7	2.5	0.4
62	G+ #5	1.1	1.6	0.0
63	G+Similar #0	1.8	3.9	0.0
64	G+Similar #6	2.1	5.5	0.0
65	G+Similar #1	1.9	4.4	0.0
66	G+ #3	0.5	0.6	0.0
67	G+ #7	0.2	0.4	0.0
68	G+Similar #5	0.2	0.6	0.0

Bibliography

B Abrahao, F Chierichetti, R Kleinberg, and A Panconesi. Trace complexity of network inference. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 491–499. ACM, 2013.

CG Akcora, B Carminati, and E Ferrari. User similarities on social networks. *Social Network Analysis and Mining*, 3(3):475–495, 2013.

M Alarcón-del Amo, M Gómez-Borja, and C Lorenzo-Romero. Are the users of social networking sites homogeneous? a cross-cultural study. *Frontiers in Psychology*, 6:1–15, 2015.

JS Albus. A new approach to manipulator control: the cerebellar model articulation controller (CMAC). *Journal of Dynamic Systems, Measurement, and Control*, 97(3):220–227, 1975.

A Anderson, D Huttenlocher, J Kleinberg, and J Leskovec. Effects of user similarity in social media. In *Proceedings of the 5th ACM International Conference on Web Search and Data Mining*, pages 703–712. ACM, 2012.

A Arenas, A Díaz-Guilera, J Kurths, Y Moreno, and C Zhou. Synchronization in complex networks. *Physics Reports*, 469(3):93–153, 2008.

WB Arthur. Inductive reasoning and bounded rationality. *The American Economic Review*, 84 (2):406–411, 1994.

AL Barabási and R Albert. Emergence of scaling in random networks. *Science*, 286(5439): 509–512, 1999.

N Barbieri, F Bonchi, and G Manco. Topic-aware social influence propagation models. In *IEEE 12th International Conference on Data Mining*, pages 81–90. IEEE, 2012.

N Barbieri, F Bonchi, and G Manco. Topic-aware social influence propagation models. *Knowledge and Information Systems*, 37(3):555–584, 2013.

B Barzel, Y Liu, and AL Barabási. Constructing minimal models for complex system dynamics. *Nature Communications*, 6:7186, May 2015.

F Bauer and JT Lizier. Identifying influential spreaders and efficiently estimating infection numbers in epidemic models: a walk counting approach. *EPL (Europhysics Letters)*, 99(6):68007, 2012.

S Boccaletti, V Latora, Y Moreno, M Chavez, and DU Hwang. Complex networks: structure and dynamics. *Physics Reports*, 424(4-5):175–308, 2006.

N Bof, G Baggio, and S Zampieri. On the role of network centrality in the controllability of complex networks. *IEEE Transactions on Control of Network Systems*, 4(3):1–10, 2016.

EF Bompard and B Han. Market-based control in emerging distribution system operation. *IEEE Transactions on Power Delivery*, 28(4):2373–2382, 2013.

SP Borgatti. Centrality and network flow. *Social Networks*, 27(1):55–71, 2005.

SP Borgatti and MG Everett. A graph-theoretic perspective on centrality. *Social Networks*, 28 (4):466–484, 2006.

B Bringmann, M Berlingerio, F Bonchi, and A Gionis. Learning and predicting the evolution of social networks. *IEEE Intelligent Systems*, 25(4):26–35, 2010.

E Brookner. *Tracking and Kalman Filtering Made Easy*. Wiley-Interscience, 1998.

M Buchanan. A bar may be the place to understand markets. `http://www.bloomberg.com/news/articles/2012-02-07/a-bar-may-be-best-place-to-understand-markets-commentary-by-mark-buchanan`, March 2012.

C Campbell, J Ruths, D Ruths, K Shea, and R Albert. Topological constraints on network control profiles. *Scientific Reports*, 5:18693, 2015.

M Cataldi and MA Aufaure. The 10 million follower fallacy: audience size does not prove domain-influence on Twitter. *Knowledge and Information Systems*, 44:559–580, 2014.

M Cataldi, N Mittal, and MA Aufaure. Estimating domain-based user influence in social networks. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, pages 1957–1962, 2013.

M Cha, H Haddai, F Benevenuto, and KP Gummadi. Measuring user influence in Twitter: the million follower fallacy. In *Proceedings of the 4th International AAAI Conference on Weblogs and Social Media*, pages 10–17, 2010.

D Challet and YC Zhang. On the minority game: Analytical and numerical studies. *Physica A: Statistical Mechanics and its Applications*, 256(3-4):514 – 532, 1998.

G Chen and Z Duan. Network synchronizability analysis: a graph-theoretic approach. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 18(3):037102, 2008.

SH Clearwater, editor. *Market-based Control: A Paradigm for Distributed Resource Allocation*. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 1996.

A Coad, J Frankish, RG Roberts, and DJ Storey. Growth paths and survival chances: An application of gambler's ruin theory. *Journal of Business Venturing*, 28(5):615 – 632, 2013.

R Cohen, S Havlin, and D Ben-Avraham. Efficient immunization strategies for computer networks and populations. *Physical Review Letters*, 91(24):247901, 2003.

SB Cools, C Gershenson, and B D'Hooghe. *Self-Organizing Traffic Lights: A Realistic Simulation*, pages 45–55. Springer London, 2013.

NJ Cowan, EJ Chastain, DA Vilhena, JS Freudenberg, and CT Bergstrom. Nodal dynamics, not degree distributions, determine the structural controllability of complex networks. *PLoS ONE*, 7(6), 2012.

RAP da Silva, MP Viana, and L da Fontoura Costa. Predicting epidemic outbreak from individual features of the spreaders. *Journal of Statistical Mechanics: Theory and Experiment*, 2012(07): P07005, 2012.

H Daneshmand, M Gomez-Rodriguez, L Song, and B Schoelkopf. Estimating diffusion network structures: Recovery conditions, sample complexity & soft-thresholding algorithm. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32, pages 793–801, 2014.

M De Choudhury, WA Mason, JM Hofman, and DJ Watts. Inferring relevant social networks from interpersonal communication. In *Proceedings of the 19th International Conference on World Wide Web*, pages 301–310. ACM, 2010.

P DeLellis, M di Bernardo, and LFR Turci. Fully adaptive pinning control of complex networks. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, pages 685–688, 2010.

T Dimpfl and FJ Peter. Using transfer entropy to measure information flows between financial markets. *Studies in Nonlinear Dynamics and Econometrics*, 17(1):85–102, 2013.

P Domingos and M Richardson. Mining the network value of customers. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 57–66. ACM, 2001.

N Eagle, AS Pentland, and D Lazer. Inferring friendship network structure by using mobile phone data. *Proceedings of the National Academy of Sciences*, 106(36):15274–15278, 2009.

D Easley and J Kleinberg. *Networks, crowds, and markets: reasoning about a highly connected world*. Cambridge University Press, 2010.

P Erdös and A Rényi. On random graphs, I. *Publicationes Mathematicae (Debrecen)*, 6:290–297, 1959.

E Even-Dar and A Shapira. A note on maximizing the spread of influence in social networks. *Proceedings of the 3rd International Conference on Internet and Network Economics*, pages 281–286, 2007.

F Fouss, A Pirotte, JM Renders, and M Saerens. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 19(3):355–369, 2007.

JH Fowler and NA Christakis. Dynamic spread of happiness in a large social network: longitudinal analysis over 20 years in the Framingham heart study. *BMJ*, 337:a2338, 2008.

LC Freeman. Centrality in social networks - conceptual clarification. *Social Networks*, 1(3): 215–239, 1978.

T Funato, D Kurabayashi, and M Nara. *Synchronization Control by Structural Modification of Nonlinear Oscillator Network*, pages 41–50. Springer Japan, Tokyo, 2006.

LK Gallos, F Liljeros, P Argyrakis, A Bunde, and S Havlin. Improving immunization strategies. *Phys. Rev. E*, 75:045104, 2007.

ER Gansner and SC North. An open graph visualization system and its applications to software engineering. *Software: Practice and Experience*, 30(11):1203–1233, 2000.

J Gao, Y Liu, RM D'Souza, and AL Barabási. Target control of complex networks. *Nature Communications*, 5:5415, 2014.

M Gomez Rodriguez, J Leskovec, and A Krause. Inferring networks of diffusion and influence. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1019–1028. ACM, 2010.

M Gomez Rodriguez, D Balduzzi, and B Schölkopf. Uncovering the temporal dynamics of diffusion networks. In *Proceedings of the 28th International Conference on Machine Learning*, pages 561–568, 2011.

A Goyal, F Bonchi, and LVS Lakshmanan. Learning influence probabilities in social networks. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, pages 241–250. ACM, 2010.

B Hajian. On Measuring Influence and its Properties in Social Networks. Master's thesis, Carleton University, 2011.

B Hajian and T White. Modelling influence in a social network: metrics and evaluation. In *IEEE Third International Conference on Privacy, Security, Risk and Trust (PASSAT) and IEEE Third International Conference on Social Computing (SocialCom)*, pages 497–500, 2011.

M Han, M Yan, Z Cai, and Y Li. An exploration of broader influence maximization in timeliness networks with opportunistic selection. *Journal of Network and Computer Applications*, 63 (Supplement C):39–49, 2016.

G Harik, E Cant-Paz, DE Goldberg, and BL Miller. The gambler's ruin problem, genetic algorithms, and the sizing of populations. *Evolutionary Computation*, 7(3):231–253, 1999.

G Hartvigsen, JM Dresch, AL Zielinski, AJ Macula, and CC Leary. Network structure, and vaccination strategy and effort interact to affect the dynamics of influenza epidemics. *Journal of Theoretical Biology*, 246(2):205 – 213, 2007.

RA Holley and TM Liggett. Ergodic theorems for weakly interacting infinite systems and the voter model. *The Annals of Probability*, 3:643–663, 1975.

MO Jackson. *Social and economic networks*. Princeton University Press, 2010.

JGraphT. `http://jgrapht.org/`, March 2015.

T Jia and AL Barabási. Control capacity and a random sampling method in exploring controllability of complex networks. *Scientific Reports*, 3:2354, 2013.

T Jia, Y Liu, E Csóka, M Pósfai, JJ Slotine, and AL Barabási. Emergence of bimodality in controlling complex networks. *Nature Communications*, 4:1–6, 2013.

A Kaiser and T Schreiber. Information transfer in continuous processes. *Physica D: Nonlinear Phenomena*, 166(1-2):43–62, 2002.

AT Kalai, A Moitra, and G Valiant. Disentangling gaussians. *Communications of the ACM*, 55(2):113–120, 2012.

D Kempe, J Kleinberg, and É Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 137–146. ACM, 2003.

D Kempe, J Kleinberg, and É Tardos. Influential nodes in a diffusion model for social networks. In *International Colloquium on Automata, Languages, and Programming*, pages 1127–1138. Springer, 2005.

C Kiss and M Bichler. Identification of influencers - measuring influence in customer networks. *Decision Support Systems*, 46(1):233–253, 2008.

K Klemm, MA Serrano, VM Eguíluz, and MS Miguel. A measure of individual role in collective dynamics. *Scientific Reports*, 2:292, 2012.

K Kuwabara, T Ishida, Y Nishibe, and T Suda. An equilibratory market-based approach for distributed resource allocation and its applications to communication network control. In SH Clearwater, editor, *Market-based control: A paradigm for distributed resource allocation*, pages 53–73. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 1996.

H Kwak, C Lee, H Park, and S Moon. What is Twitter, a social network or a news media? In *Proceedings of the 19th International Conference on World Wide Web*, pages 591–600. ACM, 2010.

A Landherr, B Friedl, and J Heidemann. A critical review of centrality measures in social networks. *Business & Information Systems Engineering*, 2(6):371–385, 2010.

G Lawyer. Understanding the influence of all nodes in a network. *Scientific Reports*, 5:8665, 2015.

J Leskovec and C Faloutsos. Sampling from large graphs. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 631–636. ACM, 2006.

J Leskovec and A Krevl. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, June 2014.

G Li, W Hu, G Xiao, L Deng, P Tang, J Pei, and L Shi. Minimum-cost control of complex networks. *New Journal of Physics*, 18:013012, 2015.

D Liben-Nowell and J Kleinberg. The link-prediction problem for social networks. *Journal of the Association for Information Science and Technology*, 58(7):1019–1031, 2007.

CT Lin. Structural controllability. *IEEE Transactions on Automatic Control*, 19(3):201–208, 1974.

T Liu, DJ Hill, and J Zhao. Synchronization of dynamical networks by network control. *IEEE Transactions on Automatic Control*, 57(6):1574–1580, 2011a.

Y Liu and AL Barabási. Control principles of complex systems. *Reviews of Modern Physics*, 88(3):035006, 2016.

Y Liu, JJ Slotine, and AL Barabási. Controllability of complex networks. *Nature*, 473(7346): 167–173, 2011b.

Y Liu, JJ Slotine, and AL Barabási. Control centrality and hierarchical structure in complex networks. *PloS One*, 7(9):e44459, 2012.

L Lü, CH Jin, and T Zhou. Similarity index based on local paths for link prediction of complex networks. *Physical Review E*, 80(4):046122, 2009.

R Marschinski and H Kantz. Analysing the information flow between financial time series. *The European Physical Journal B - Condensed Matter and Complex Systems*, 30(2):275–281, 2002.

D McKenney. Distributed and adaptive traffic signal control within a realistic traffic simulation. Master's thesis, Carleton University, 2011.

D McKenney and T White. Observations on the role of influence in the difficulty of social network control. In *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 1294–1301. IEEE, 2016.

MS Miller, D Krieger, N Hardy, C Hibbert, and ED Tribble. An automated auction in ATM network bandwidth. In SH. Clearwater, editor, *Market-based Control: A paradigm for distributed resource allocation*, pages 96–125. World Scientific Publishing Co., Inc., River Edge, NJ, USA, 1996.

A Molderink, V Bakker, MGC. Bosman, JL Hurink, and GJM Smit. Management and control of domestic smart grid technology. *IEEE Transactions on Smart Grid*, 1(2):109–119, 2010.

S Myers and J Leskovec. On the convexity of latent social network inference. In *Advances in Neural Information Processing Systems 23*, pages 1741–1749, 2010.

G Neff and P Nagy. Automation, algorithms, and politics - talking to bots: symbiotic agency and the case of Tay. *International Journal of Communication*, 10:4915–4931, 2016.

P Netrapalli and S Sanghavi. Learning the graph of epidemic cascades. In *ACM SIGMETRICS Performance Evaluation Review*, volume 40, pages 211–222. ACM, 2012.

T Nishikawa, AE Motter, YC Lai, and FC Hoppensteadt. Heterogeneity in oscillator networks: are smaller worlds easier to synchronize? *Phys. Rev. Lett.*, 91:014101, 2003.

L Page, S Brin, R Motwani, and T Winograd. The PageRank citation ranking: bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999.

W Pan, Y Altshuler, and A Pentland. Decoding social influence and the wisdom of the crowd in financial trading network. In *2012 International Conference on Privacy, Security, Risk and Trust (PASSAT) and 2012 International Confernece on Social Computing (SocialCom)*, pages 203–209. IEEE, 2012.

PN Paraskevopoulos. *Modern control engineering*. CRC Press, 2001.

R Pastor-Satorras and A Vespignani. Immunization of complex networks. *Physical Review E*, 65(3):036104, 2002.

M Piraveenan, M Prokopenko, and L Hossain. Percolation centrality: Quantifying graph-theoretic impact of nodes during percolation in networks. *PloS one*, 8(1):e53095, 2013.

M Pósfai, Y Liu, J Slotine, and AL Barabási. Effect of correlations on network controllability. *Scientific Reports*, 3:1067, 2013.

I Psorakis, SJ Roberts, I Rezek, and BC Sheldon. Inferring social network structure in ecological systems from spatio-temporal data streams. *Journal of the Royal Society Interface*, page rsif20120223, 2012.

R Quax, D Kandhai, and P Sloot. Information dissipation as an early-warning signal for the Lehman Brothers collapse in financial time series. *Scientific Reports*, 3:1898, 2013.

JR Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

JR Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers, Inc., 1993.

CJ Quinn. *Identification and Approximation of the Structure of Networks of Stochastic Processes*. PhD thesis, University of Illinois, 2014.

BH Raven. *Social Influence and Power*. Defense Technical Information Center, 1964.

Y Rubner and C Tomasi. *The earth mover's distance*, pages 13–28. Springer US, Boston, MA, 2001.

GA Rummery and M Niranjan. On-line Q-learning using connectionist systems. Technical report, University of Cambridge, Department of Engineering, 1994.

A Runka. *On the Control of Opinion in Social Networks*. PhD thesis, Carleton University, 2016.

A Runka and T White. Towards intelligent control of influence diffusion in social networks. *Social Network Analysis and Mining*, 5(1):9, 2015a.

A Runka and T White. Evolving neurocontrollers for the control of information diffusion in social networks. In *Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation*, GECCO Companion '15, pages 1471–1472, 2015b.

J Ruths and D Ruths. Control profiles of complex networks. *Science*, 343(6177):1373–1376, 2014.

T Schreiber. Measuring information transfer. *Phys. Rev. Lett.*, 85:461–464, 2000.

AA Sherstov and P Stone. Function approximation via tile coding: Automating parameter choice. In *SARA*, volume 3607, pages 194–205. Springer, 2005.

B Siginam. Adaptive tile coding methods for the generalization of value functions in the RL state space. Master's thesis, University of Minnesota, 2012.

M Šikić, A Lančić, N Antulov-Fantulin, and H Štefančić. Epidemic centrality: is there an underestimated epidemic impact of network peripheral nodes? *The European Physical Journal B*, 86(10):440, 2013.

Q Song and J Cao. On pinning synchronization of directed and undirected complex dynamical networks. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 57(3):672–680, 2010.

AR Sorkin. *Too Big to Fail: The Inside Story of How Wall Street and Washington Fought to Save the Financial System–and Themselves*. Penguin Publishing Group, 2010.

H Su, Z Rong, MZQ Chen, X Wang, G Chen, and H Wang. Decentralized adaptive pinning control for cluster synchronization of complex dynamical networks. *IEEE Transactions on Cybernetics*, 43(1):394–399, 2013.

K Subbian, C Aggarwal, and J Srivastava. Content-centric flow mining for influence analysis in social streams. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*, pages 841–846. ACM, 2013.

J Sung, S Moon, and JG Lee. The influence in Twitter: are they really influenced? In *Behavior and Social Computing*, pages 95–105. Springer International Publishing, 2013.

R Sutton and A Barto. *Reinforcement learning: An introduction*. MIT press, 1998.

RS Sutton. *Temporal Credit Assignment in Reinforcement Learning*. PhD thesis, University of Massachusetts Amherst, 1984.

RS Sutton. Generalization in reinforcement learning: Successful examples using sparse coarse coding. In *Advances in Neural Information Processing Systems*, pages 1038–1044, 1996.

J Tang, J Sun, C Wang, and Z Yang. Social influence analysis in large-scale networks. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 807–816. ACM, 2009.

B Travençolo and L Costa. Accessibility in complex networks. *Physics Letters A*, 373(1):89–95, 2008.

MF Tsai, CW Tzeng, ZL Lin, and ALP Chen. Discovering leaders from social network by action cascade. *Social Network Analysis and Mining*, 4(1):165, 2014.

J Valverde-Rebaza and A de Andrade Lopes. Exploiting behaviors of communities of Twitter users for link prediction. *Social Network Analysis and Mining*, 3(4):1063–1074, 2013.

M Vasirani and S Ossowski. A market-inspired approach to reservation-based urban road traffic management. In *Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems - Volume 1*, pages 617–624, 2009.

G Ver Steeg and A Galstyan. Information-theoretic measures of influence based on content dynamics. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, pages 3–12. ACM, 2013.

M Viana, J Batista, and L Costa. Effective number of accessed nodes in complex networks. *Physical Review E*, 85(3):036105, 2012.

P Wang, BW Xu, YR Wu, and XY Zhou. Link prediction in social networks: the state-of-the-art. *Science China Information Sciences*, 58(1):1–38, 2015.

XF Wang and G Chen. Pinning control of scale-free dynamical networks. *Physica A: Statistical Mechanics and its Applications*, 310(3-4):521 – 531, 2002.

DJ Watts and SH Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684): 440–442, 1998.

J Weng, E Lim, J Jiang, and Q He. TwitterRank: finding topic-sensitive influential Twitterers. In *WSDM '10 Proceedings of the Third ACM International Conference on Web Search and Data Mining*, pages 261–270, 2010.

G Yan, G Tsekenis, B Barzel, JJ Slotine, Y Liu, and AL Barabási. Spectrum of controlling and observing complex networks. *Nature Physics*, 11(9):779–786, 2015.

W Yu, G Chen, and J L. On pinning synchronization of complex dynamical networks. *Automatica*, 45(2):429 – 435, 2009.

W Yu, G Chen, J Lu, and J Kurths. Synchronization via pinning control on general complex networks. *SIAM Journal on Control and Optimization*, 51(2):1395–1416, 2013.

Z Yuan, C Zhao, Z Di, W Wang, and Y Lai. Exact controllability of complex networks. *Nature Communications*, 4:2447, 2013.

P Zarchan and H Musoff. *Fundamentals of Kalman Filtering: A Practical Approach*. AIAA, 2005.

C Zhao, W Wang, Y Liu, and JJ Slotine. Intrinsic dynamics induce global symmetry in network controllability. *Scientific Reports*, 5:8422, 2015.

M Zhao, T Zhou, BH Wang, G Yan, HJ Yang, and WJ Bai. Relations between average distance, heterogeneity and network synchronizability. *Physica A: Statistical Mechanics and its Applications*, 371(2):773 – 780, 2006.

CC Zou, DT, and W Gong. Email worm modeling and defense. In *Proceedings of the 13th International Conference on Computer Communications and Networks*, pages 409–414, 2004.