Interacting with users in social networks: The Follow-back Problem

by

Krishnan Rajagopalan

B.S. Operations Research, United States Naval Academy (2014)

Submitted to the Sloan School of Management
in partial fulfillment of the requirements for the degree of
Master of Science in Operations Research

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2016

© Massachusetts Institute of Technology 2016. All rights reserved.

Author ............................................................... Sloan School of Management
May 13, 2016

Certified by .......................................................... Dr. Danelle Shah
Intelligence and Decision Technologies Group
MIT Lincoln Laboratory
Thesis Supervisor

Certified by .......................................................... Prof. Tauhid R. Zaman
KDD Career Development Professor in Communications and Technology
Assistant Professor of Operations Management
Thesis Supervisor

Accepted by .......................................................... Prof. Dimitris Bertsimas
Boeing Professor of Operations Research
CoDirector, Operations Research Center
Interacting with users in social networks: The Follow-back Problem

by

Krishnan Rajagopalan

Submitted to the Sloan School of Management on May 13, 2016, in partial fulfillment of the requirements for the degree of Master of Science in Operations Research

Abstract

An agent wants to form a connection with a predetermined set of target users over social media. Because forming a connection is known as “following” in social networks such as Twitter, we refer to this as the follow-back problem. The targets and their friends form a directed graph which we refer to as the “friends graph.” The agent’s goal is to get the targets to follow it, and it is allowed to interact with the targets and their friends. To understand what features impact the probability of an interaction resulting in a follow-back, we conduct an empirical analysis of several thousand interactions in Twitter. We build a model of the follow-back probabilities based upon this analysis which incorporates features such as the friend and follower count of the target and the neighborhood overlap of the target with the agent. We find optimal policies for simple network topologies such as directed acyclic graphs. For arbitrary directed graphs we develop integer programming heuristics that employ network centrality measures and a graph score we define as the follow-back score. We show that these heuristic policies perform well in simulation on a real Twitter network.

Thesis Supervisor: Dr. Danelle Shah
Title: Intelligence and Decision Technologies Group
MIT Lincoln Laboratory

Thesis Supervisor: Prof. Tauhid R. Zaman
Title: KDD Career Development Professor in Communications and Technology
Assistant Professor of Operations Management
Acknowledgments

I would first like to thank three sponsoring institutions. I am grateful that the US Navy has given me the privilege of pursuing a Master’s degree immediately upon my commissioning. I am grateful that it has continued investing in my development, and I look forward to repaying that debt in service during the coming years. I also owe thanks to MIT Lincoln Laboratory, and specifically, Group 104, Intelligence & Decision Technologies, which has sponsored me on a military fellowship. I’d also like to thank the MIT Operations Research Center, whose unparalleled students, faculty, and staff have helped make my experience at MIT so enriching.

Next, I’d like to thank two research mentors that have contributed to this thesis as well as my intellectual growth while at MIT. Dr. Tauhid Zaman of the MIT Sloan School of Management’s Operations Management Group served as a dedicated and patient research adviser over the past two years. The energy and time he devoted to our research as well as his fun demeanor made learning from him a pleasure. Dr. Danelle Shah of MIT Lincoln Laboratory gave me tremendous support in my research and was always willing to offer a guiding hand. Her attention to detail as well as her emphasis on precision and clarity of presentation are qualities I will strive to emulate in the future.

Lastly, I would not have had such a great experience at MIT without the love and care of my family and friends. I’d like to thank my wife, Caitlin, whom I got to spend the last two years growing with in the Boston area. I would like to also thank my parents, Sri and Lakshmy, and siblings, Kartik, Keshav, Andrea, and Chelsea, for their love and support as well as their continued confidence in me. Finally, I am grateful to my fellow ORC Master’s buddies, Zach and Carter. I enjoyed getting to know them, and doing problem sets and projects without their company would not have been the same.

The views expressed in this thesis are those of the author and do not reflect the official policy or position of MIT Lincoln Laboratory, the United States Navy, Department of Defense, or the U.S. Government.
Contents

1 Introduction .......................................................... 13
   1.1 Business Motivation .............................................. 13
   1.2 National Defense Motivation .................................... 14
   1.3 Our Contribution .................................................. 16
   1.4 Literature Review ................................................ 17
      1.4.1 Empirical Measurement of Influence in Social Networks ... 17
      1.4.2 Triadic Closure .............................................. 18
      1.4.3 Influence Maximization in Social Networks ................ 18
      1.4.4 Network Centrality ...................................... 19

2 Empirical Analysis of Interactions in Social Networks .......... 21
   2.1 Empirical Setup .................................................. 21
   2.2 Interaction Type .................................................. 23
   2.3 Interaction Type, Follower Counts, Friend Counts ............... 24
   2.4 Friend and Follower Counts .................................... 25
   2.5 Overlap ........................................................... 27

3 The Follow-back Problem ............................................. 31
   3.1 Problem Definition ............................................... 31
   3.2 Assumptions on Follow Probability Model ....................... 33
      3.2.1 Optimal Policy on a Directed Acyclic Graph (DAG) ....... 34
   3.3 Expected Follows on a DAG: Follow-back Score ................. 37
   3.4 Low Order Approximation of $J(G)$ ............................ 39
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.5</td>
<td>Optimal Policy on a Directed Graph</td>
<td>41</td>
</tr>
<tr>
<td>3.5.1</td>
<td>Proof of Lemma 3.2.2</td>
<td>42</td>
</tr>
<tr>
<td>3.5.2</td>
<td>Proof of Lemma 3.3.2</td>
<td>43</td>
</tr>
<tr>
<td>3.5.3</td>
<td>Proof of Lemma 3.5.2</td>
<td>45</td>
</tr>
<tr>
<td>4</td>
<td>Integer Programming Formulation of the Follow-back Problem</td>
<td>49</td>
</tr>
<tr>
<td>4.1</td>
<td>Set-up and Motivation</td>
<td>49</td>
</tr>
<tr>
<td>4.2</td>
<td>Integer Program</td>
<td>50</td>
</tr>
<tr>
<td>4.3</td>
<td>Tractability Issues</td>
<td>51</td>
</tr>
<tr>
<td>4.3.1</td>
<td>Lazy Constraints</td>
<td>51</td>
</tr>
<tr>
<td>4.3.2</td>
<td>Low Order Approximation of Expected Follows</td>
<td>52</td>
</tr>
<tr>
<td>4.4</td>
<td>Integer Program Based Heuristic</td>
<td>54</td>
</tr>
<tr>
<td>4.5</td>
<td>Simulation</td>
<td>57</td>
</tr>
<tr>
<td>4.5.1</td>
<td>Network</td>
<td>57</td>
</tr>
<tr>
<td>4.5.2</td>
<td>Policies</td>
<td>57</td>
</tr>
<tr>
<td>4.5.3</td>
<td>Simulation Methodology</td>
<td>59</td>
</tr>
<tr>
<td>5</td>
<td>Future Work and Conclusions</td>
<td>65</td>
</tr>
<tr>
<td>5.1</td>
<td>Future Work</td>
<td>65</td>
</tr>
<tr>
<td>5.1.1</td>
<td>Optimal Timing of Interactions</td>
<td>65</td>
</tr>
<tr>
<td>5.1.2</td>
<td>Integer Program Scaling Parameter $\lambda$</td>
<td>67</td>
</tr>
<tr>
<td>5.1.3</td>
<td>Maintaining Connections</td>
<td>67</td>
</tr>
<tr>
<td>5.2</td>
<td>Conclusion</td>
<td>67</td>
</tr>
<tr>
<td>5.3</td>
<td>Application</td>
<td>68</td>
</tr>
</tbody>
</table>
List of Figures

2-1 Conversion rate for different agent interaction and target reaction combinations (RT stands for retweet). ................................................. 24
2-2 Plot of the follow conversion rate versus agent follower count. The horizontal axis shows the follower counts of four agents. The vertical axis shows the percentage of targets that followed each agent. ........... 27
2-3 Plot of the follower count versus the friend count for each target. The markers indicate which targets followed the agent and which did not. 28
2-4 Illustration of the overlap feature between the agent and target vertex. Overlap is defined as the number of users the target follows who also follow the agent. The overlap in the figure is three. ...................... 29
2-5 Plot of the follow conversion rate versus overlap. The red circles are the median and the error bars are 95% confidence intervals. .......... 29
2-6 Plot of the normalized conversion rate for our data on follows in Twitter and Facebook signups from [21] versus overlap. ....................... 30
3-1 Illustration of the initial setting of the follow-back problem at time $n = 0$. The agent $a$ wants to maximize its number of followers in the friends graph by interacting with them. ....................................... 32
3-2 Illustration of two optimal linear extension policies on a DAG. Any interaction sequence that respects the partial order of the graph is optimal. ................................................................. 35
4-1 Contribution to objective value by order for two accounts. $J^l$ is the $lth$ order contribution ................................................................. 54
4-2 Simulation with different network centrality regularization and different order .................................................. 60
4-3 Best performing policy (Closeness 1st Order) compared with random policies and non-regularized policies ....................... 63
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Regression over different types of Twitter interactions</td>
<td>24</td>
</tr>
<tr>
<td>2.2</td>
<td>Regression over different types of Twitter interactions as well as user</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>degree features</td>
<td></td>
</tr>
<tr>
<td>2.3</td>
<td>Regression over user degree features</td>
<td>26</td>
</tr>
<tr>
<td>2.4</td>
<td>Regression over overlap and user degree features</td>
<td>30</td>
</tr>
<tr>
<td>4.1</td>
<td>Twitter Network Properties</td>
<td>53</td>
</tr>
<tr>
<td>4.2</td>
<td>Computer Scientist Friends Graph</td>
<td>57</td>
</tr>
<tr>
<td>4.3</td>
<td>Computer Scientist Friends Graph Results</td>
<td>61</td>
</tr>
<tr>
<td>5.1</td>
<td>Follow-back Delay</td>
<td>66</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Business Motivation

Online social networks have rapidly risen in size over the past decade and today represent one of the main platforms through which social interactions occur. Current estimates are that 74% of adult Internet users are on social media, and this penetration rate is even higher in some emerging markets [28]. Today people use social networks as the primary mechanism to interact with friends, make business connections, and find like-minded individuals. With the ever-increasing volume of content online, if one wishes to have influence over social media, it is important to understand how to effectively engage with other users in the network.

Consider an individual or company trying to establish its presence in a new market. One way to build its brand recognition is to advertise over social media. Presumably, there is a group of influential individuals in this market that already have well-established social media presences. The company that is new to the market may want to connect with these influential users to tap into their well-established networks. What is the best way to do this over social media? In the off-line setting, humans understand and observe the basics of social interaction. For instance, some people are harder to connect with than others, either due to a lack of commonality and/or because of some other intrinsic qualities. Also, it is often easier to connect with someone with whom you share mutual contacts. We wish to understand if these social intuitions translate...
into the online setting and if so, how does one implement these intuitions? Such an understanding would allow for more effective interactions in social networks. People could more easily connect with users they deem interesting. Companies and organizations could engage with more customers and grow their online social networks more quickly. Individuals could systematically gain influence and followers in social networks. In short, if we understand what drives online interactions, we can develop policies to make us more effective users of social networks.

1.2 National Defense Motivation

In summer 2014, the Islamic State in Iraq and the Levant (ISIL) launched an application on the Google Play Store called “The Dawn of Glad Tidings.” If a Twitter user downloaded this application, his or her account could be used by ISIL to post content [5]. The idea behind the application was simple: use existing Twitter accounts of jihadist-sympathizers to amplify the message of the terror group. However, the consequences were far greater. By hijacking Twitter accounts to broadcast its message, ISIL built a quasi-active community of supporters on the social media site. An interested onlooker would perceive a network of Twitter users engaging one another and posting content sympathetic to the jihadist group’s cause, even while much of this activity was automated. ISIL social media recruiters also deliberately attempt to isolate their targets from the rest of the world by encouraging them to terminate their existing social media connections [8]. ISIL attempted to restrict the flow of alternative information to its target audience: it denied its prospective recruits free access to information. ISIL recruits by framing a perverse, inflammatory narrative about the tenets of Islam and the West’s subjugation of the Muslim world [5]. It can recruit more easily where opposing voices are absent. It relies, in part, upon an Anti-Access/Area Denial (A2/AD) capability to block counter-propaganda efforts.

A2/AD is a defense paradigm used to describe a strategy of preventing one’s adversary from operating military forces within a specific region of interest. United States defense strategists consider the A2/AD capability posed by American adver-
saries to be one of the greatest threats of the twenty-first century [29]. An adversary can also pose an A2/AD threat over the information space, which encompasses both physical and non-physical domains where information is spread [29], such as social media platforms. Among other things, A2/AD over social media is a threat that the United States has seen from ISIL.

With its rapid growth, social media has gained the interest of national defense policy makers and strategists. The United States Department of State launched a handle on Twitter, @ThinkAgain_DOS, as a means to insert counter-propaganda content into networks of jihadist-sympathizers. The profile engaged many prominent jihadist Twitter accounts in the fall of 2014 but was widely criticized as being unsuccessful [23]. American leadership is still trying to understand how to combat extremist recruitment over social media. In January of 2016, The White House held a summit with executives from some of the leading American technology firms including Facebook and Twitter to help generate strategies to counter extremist recruitment online [34].

Military leaders have been tasked with similar challenges before, but social media is a relatively new operating environment. The United States Department of Defense uses Military Information Support (MIS) units to penetrate contested information environments. Military Information Support Operations (MISO) are the conveying of specific information to target foreign audiences in order to sway their perceptions and ultimately influence their actions in support of national objectives [1]. As social media platforms have made the spread of information more nuanced, MIS units require tools that will allow them to penetrate this new, increasingly important space, in the event that an adversary employs an A2/AD capability. When an adversary is attempting to use propaganda to influence a target audience over social media, MIS units may be tasked with inserting counter-propaganda information in the network.

One problem confronting MIS units is ensuring that their counter-propaganda efforts can penetrate an information blockade over social media. On Twitter, for instance, a user will generally only see the content posted by users who he “follows” or is connected with. MIS units can improve their ability to penetrate contested social
media platforms by learning how to form desired connections. We need enhanced capabilities to penetrate social networks.

We study strategies that will enable actors to wield greater influence over social media; such tools should not be taken lightly. They must be guarded closely. Controlling the flow of information to target audiences may have tremendous impact over the audience’s understanding of world events and their subsequent actions. Employing these tools should be done sparingly and only when the situation and the targeted individuals really warrant their use, such as when innocent lives hang in the balance. We leave the use cases of such tools to high level policymakers, who have the experience and legal resources to understand when situations are sufficiently dire.

1.3 Our Contribution

We study the the follow-back problem, where the goal is to find the most effective way to interact with users in a social network in order to maximize the number of social connections. The name is based on terminology from the social network Twitter, where when a user forms such a connection he is said to become a “follower.” We will see in Chapter 2 and Chapter 3 how sequencing users to interact with in a specific way is crucial in maximizing the number of follows.

The problem assumes some model for user behavior in social networks, in particular, what features affect the probability of being followed. To better estimate this model, we conduct an online field experiment in Twitter where online agents interact with users in order to gain followers. Our experimental findings show that the follow probability is affected by the number of friends and followers of the targeted user and also the number of friends of the target user that follow the agent, a feature we refer to as the overlap. In particular, we find that the follow probability is larger if the targeted user has many friends, few followers, and a large overlap with the agent. These observations match our social intuition which suggests that it is easier to connect with people who are less popular (many friends, few followers) and who have mutual contacts (large overlap).
We propose a simple model for the follow probability based on this empirical analysis and use this model in our analysis of the follow-back problem. We are able to provide the optimal interaction policy for directed acyclic graphs (DAGs) and provide an approximation for the expected number of follows this policy gives on a DAG, which we define as the DAG’s follow-back score. We use follow-back score in an integer programming-based heuristic.

The thesis is outlined as follows. We review related literature in Section 1.4. We then present our empirical analysis of follow probabilities in the Chapter 2. We formally define the follow-back problem, analyze optimal policies for it, and define follow-back score in Chapter 3. Then, we formulate the follow-back problem with constrained optimization, show how integer programming can be used to find a policy on a graph with multiple targets, and show the performance of our heuristic in simulation in Chapter 4. We explain some avenues of future work and conclude in Chapter 5.

1.4 Literature Review

There are several lines of research related to our work. First, there is work on empirical studies of influence in social networks. Second, there is the body of work on the sociological notion of triadic closure and its manifestation in online social networks. Third, there is a sizable body of work on theoretical analysis of influence maximization in social networks. Finally, there is the broad area of network centrality measures.

1.4.1 Empirical Measurement of Influence in Social Networks

Many researchers have studied what factors affect influence in social networks. Influence can be roughly defined as the ability to cause an individual to adopt a behavior, such as wearing a new line of fashion or joining an exercise gym. The ability of social influence to affect health behavior was demonstrated in [14] and [15]. The effect of the local structure of a social network on health behavior was demonstrated in [10]. The use of randomized online experiments to discern causal social influence was
done in [3]. In [4] a massive randomized online experiment in Facebook was carried out to infer what types of users are influential and susceptible. It was found that influence and susceptibility depended on demographic characteristics as well as network structure. [27] analyzes the Twitter network structure as well as the temporal aspects and network structure of retweets. [11] analyzes the influence of users in a network based upon number of retweets and number of followers. [21] showed that data from Facebook could be used to understand structural features of networks to better comprehend social contagion.

1.4.2 Triadic Closure

We discuss in further detail a social science theory known as “triadic closure” in Section 2.5 but present some related literature here. This principle is used in [7] to design a social botnet on Facebook. The algorithm in [7] strives to infiltrate a Facebook network by leveraging the principle of triadic closure. They initially acquire friends from a set of vulnerable users and then snowball their engagements by identifying users with whom they have mutual connections. The effect of social overlap on influence was shown in [21], in which a massive Facebook dataset was studied and it was found that adoption of Facebook by a new user depended strongly on how many people invited that user and their local network structure. A similar result is found in [18] where it is shown that the likelihood of accepting a connection request in an online social network is about three times higher given the existence of some number of mutual connections.

1.4.3 Influence Maximization in Social Networks

Related to the follow-back problem is the influence maximization problem where the goal is to maximize the spread of a piece of information through a social network by seeding a select set of users. One of the first theoretical studies of the influence maximization problem was in [24] where it was shown that the problem was submodular, so a greedy algorithm would have good performance. This work led to
subsequent variations of the problem and different algorithmic solutions [25], [12], [13]. Each of these works exploited the sub-modular structure of the problem to obtain efficient algorithms.

1.4.4 Network Centrality

Many problems in graphs can be solved through the development of functions which map vertices to real numbers. These functions are known as network centralities. They quantify how central a vertex is to the problem at hand, with the definition of centrality being set by the application. In our work we use network centralities in the solution to the follow-back problem. One of the simplest is distance centrality which measures how close a vertex is to all other vertices [32]. A related measure is betweenness centrality, which measures how many paths pass through a vertex [19], [20]. This measure gives importance to vertices that are critical to the flow through a graph. Eigenvector centrality measures how important a vertex is by how important its neighbors are [6]. This centrality measure is closely related to the PageRank [30] and HITS [26] centrality measures. Rumor centrality was developed to find the source of a rumor that spreads in a graph [33]. It was shown that the vertex with the highest rumor centrality corresponded to the maximum likelihood estimate of the rumor source in some specific graph topologies and that rumor centrality was related to the number of linear extensions of a partially ordered set.
Chapter 2

Empirical Analysis of Interactions in Social Networks

To determine the structure of the follow probabilities we require data on interactions in social networks. To obtain such data we conducted an online experiment involving user accounts to which we were given access. The accounts belonged to six Moroccan artists who wished to connect with Twitter users interested in Moroccan art. The artist accounts would interact with Twitter users using different types of interactions, and we recorded the results of the interactions after one week. We were able to obtain data for over 6,000 Twitter interactions. We now discuss our empirical findings in detail.

2.1 Empirical Setup

We conducted a series of experiments to understand different aspects of interactions in social networks. As mentioned above, we utilized six different Twitter accounts from Moroccan artists. These accounts acted as our agents. We conducted three different experiments. The first experiment was designed to understand the effect of the different types of interactions in Twitter on the follow probability. The second experiment aimed to understand how vertex features such as friend count and follower count affect the follow probabilities. The third experiment looked at more complex
graph features, such as the number of the agent’s followers who are also friends of the target, which we refer to as the overlap.

The empirical procedure was as follows. For the first two experiments we had the agents use the Twitter search API to find target users who have posted tweets about Morocco or art. We acknowledge that people assess the profile content of other users when deciding to follow or not to follow them. Our research focuses on the types of interaction between the users as well as network structure. We attempt to control for profile content by ensuring that all of the users that the artist accounts interact with have posted at least one piece of content about Morocco or art.

The agents then interacted with all of these targets by some form of Twitter interaction: either a retweet, reply, or a follow. We had the agents perform this search and interact procedure in randomly spaced intervals that were on average two hours apart. We spaced out the intervals when the artist accounts would interact to make the artist accounts appear more authentic. Rapid account activity is often the sign of Twitter bots, or automated accounts. The experiments were each run for one week. We waited one week from when the interaction occurred and checked if there was any sort of reaction from the targets, such as a decision to follow, a reply, or a retweet. For the third experiment the procedure was the same, except to find target users, in addition to using the search API, we also searched for followers of users who followed-back the agents during the first two experiments, which we refer to as overlap and define in Section 2.5.

We made sure that each target user only interacted with a single Moroccan artist account to avoid the risk of a target’s reaction being confounded by multiple interactions. Also, each target user was only interacted with one time. The six different Twitter agent accounts we used have different names and profile images, but the content they posted on their Twitter timelines is similar. This content consisted of images of their artwork, which was similar across the different accounts.

We modeled the follow probabilities using a logistic regression model. If we denote the follow probability of a target vertex $v$ as $P(X_v = 1|G, x, F) = p(F_v)$ where $F_v$ is a vector of features describing the interaction such as user properties of the agent and
target, type of interaction (retweet, reply, follow), local graph properties, etc. Under our logistic regression model the follow probability for a vertex \( v \) is

\[
\log \left( \frac{p(F_v)}{1 - p(F_v)} \right) = \sum_{k=1}^{d} \beta_k F_{vk},
\]  

(2.1)

where we have assumed that for a vertex \( v \) the set \( S_v \) has \( d \) elements. Furthermore, we assume each interaction is an independent event.

### 2.2 Interaction Type

There are three main ways an agent can interact with a target in Twitter. The first is following. When the agent follows a target any content the target posts is shown to the agent in his Twitter timeline. In addition, when the agent follows the target, the target is notified that the agent is now following him. The second interaction is retweeting, which is reposting one of the target’s tweets. The agent’s timeline displays the original tweet, and the target is notified of the retweet. The third interaction is replying, which involves the agent creating a tweet that mentions the target. The tweet appears on the agent’s timeline and the target is notified.

We had one agent account post original content and not interact with other users. This agent acted as a control. The remaining five agents interacted with 150 to 220 different targets over a one week period. Each agent was assigned a specific type of interaction. These interactions were following, retweeting, replying, retweeting and following, and replying and following. Agents who replied used a fixed set of messages such as “I like that” or “Nice”.

The target can similarly react to the agent’s interaction by mentioning, retweeting, or following. If the agent’s interaction produces a reaction from the target, we refer to this as a conversion. We show the conversion rate for each combination of agent interaction and target reaction in Figure 2-1. The control agent that did not interact with anyone is referred to as “None” in the figure. This agent did not receive any form of target reaction. One observation from this figure is that to gain followers, the
Table 2.1: Regression over different types of Twitter interactions

<table>
<thead>
<tr>
<th>Feature</th>
<th>( \beta )</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>-4.18***</td>
<td>( 3.25 \times 10^{-15} )</td>
</tr>
<tr>
<td>Follow</td>
<td>2.14***</td>
<td>( 1.25 \times 10^{-5} )</td>
</tr>
<tr>
<td>Retweet</td>
<td>-0.14</td>
<td>0.68</td>
</tr>
<tr>
<td>Reply</td>
<td>-0.22</td>
<td>0.48</td>
</tr>
</tbody>
</table>

The significance codes are ***: 0.001, **: 0.01, and *: 0.05.

We run our regression model for the target reaction of following using three indicator variables for following, retweeting, or replying. Table 2.1 shows that the only interaction type that was significant (at the 5% level) in predicting the follow reaction was following, which increased the follow probability.

2.3 Interaction Type, Follower Counts, Friend Counts

We next run a regression using the same data set from above but include the agent and target friend and follower counts. We see that in Table 2.2, the relative significance of the interaction types decrease, as the target friend and follower counts are the features with highest significance at a 0.1% level. Replying and following are significant at 5% level, with the act of following increasing the follow probability. But in general, we postulate that social media interactions may be too complicated...
Table 2.2: Regression over different types of Twitter interactions as well as user degree features

<table>
<thead>
<tr>
<th>Feature</th>
<th>$\beta$</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>-10.6994***</td>
<td>1.31 x 10^{-08}</td>
</tr>
<tr>
<td>log10(target friend count)</td>
<td>3.4066***</td>
<td>6.28 x 10^{-10}</td>
</tr>
<tr>
<td>log10(target follower count)</td>
<td>-1.9575***</td>
<td>1.03 x 10^{-05}</td>
</tr>
<tr>
<td>log10(agent friend count)</td>
<td>0.2954</td>
<td>0.7634</td>
</tr>
<tr>
<td>log10(agent follower count)</td>
<td>1.4605</td>
<td>0.2883</td>
</tr>
<tr>
<td>Follow</td>
<td>1.6365*</td>
<td>0.0211</td>
</tr>
<tr>
<td>Retweet</td>
<td>-0.7689</td>
<td>0.1096</td>
</tr>
<tr>
<td>Reply</td>
<td>-0.9330*</td>
<td>0.0238</td>
</tr>
</tbody>
</table>

The significance codes are ***: 0.001, **: 0.01, and *: 0.05.

of phenomena to adequately predict responses with simple indicator variables. People respond very differently to the varying forms of interaction available over social media—both verbal and non-verbal. However, the fact that the agent who did not initiate interaction with anyone gained no following, supports the intuitive idea that interacting in some manner is best if one wants to gain connections. Being an isolated user that avoids interaction does not lend well to establishing a network. In practice, to most effectively gain followers, a social media operator would want to tailor each interaction so that he has the greatest chance of winning the favor of the target. We further investigate the significance of follower and friend counts in Section 2.4.

### 2.4 Friend and Follower Counts

Our next experiment aimed to understand how the friend and follower counts of both the target and agent impacted the follow probability. Based on our study of different Twitter interactions, we decided to have the agents only follow and retweet the targets. Prior to this experiment, the agents obtained followers from an online service. This allowed for sufficient variance in their follower counts, with one agent receiving as many as 10,000 followers. These follower accounts appeared to be automated bot accounts. However, our interest was in changing the follower count displayed on the agent’s profile, not in the quality of the followers themselves. We wanted to see if having a higher follower count would make the agent seem more important and
increase the follow probability. We were able to measure over 2,000 interactions for the agents.

We first examine the impact of the agent follower count on the follow probability. In Figure 2-2 we plot the conversion rate for follows versus the follower count of the agent. The results indicate that the follower count of the agent does not have a significant impact on the conversion rate for follower counts up to 10,000. This is somewhat counter-intuitive as we would expect users to be more likely to follow someone with a higher follower count, as this is a symbol of status on Twitter. Our data suggest that it is not clear if the agent’s follower count has any effect on the follow probability.

The follower and friend counts of the target had a clearer relationship to the follow probability. We plot in Figure 2-3 the friend and follower count of each target. We use different markers for those that followed the agent and those that did not. A clear pattern is seen here. Target’s with lower follower counts and higher friend counts tended to follow the agents more frequently. This makes sense intuitively as we would assume that user’s with higher friend-to-follower ratios have a propensity to follow others on Twitter.

Our regression model used the logarithm (base 10) of the friend and follower counts of the agent and targets as features. The results of our model fitting is shown in Table 2.3 and support what was seen in the figures. The follower and friend counts for the target were significant while the friend and follower counts of the agents were not. Also, targets with many friends and few followers were more likely to follow the agent.
Figure 2-2: Plot of the follow conversion rate versus agent follower count. The horizontal axis shows the follower counts of four agents. The vertical axis shows the percentage of targets that followed each agent.

2.5 Overlap

Our final experiment looked at the overlap feature, which is related to the concept of triadic closure from social network theory. Triadic closure is the property that if two individuals $A$ and $B$ both have strong ties to $C$, then it is likely that some form of tie exists between $A$ and $B$. Strong triadic closure states that, given this mutual connection with $C$, a weak tie between $A$ and $B$ must exist [17]. Triadic closure is a term usually applied to undirected graphs. [16] posits the existence of an analog in a directed network, or “directed closure.” We define the overlap for two vertices. Formally, the overlap of a vertex $v$ with another vertex $u$ is equal to the number of directed two-hop paths from $v$ to $u$. In Twitter terms, user $v$ has an overlap with user $u$ equal to the number of friends of $u$ that follow $v$. We illustrate this overlap feature in Figure 2-4. The intuition behind this definition is based on the flow of information. In the case of Twitter, content posted by a user can be seen by his followers. Therefore, the target could see some of the agent’s content if it is retweeted by friends of the target that follow the agent. The target could also see any replies these users give to the agent. Using our terminology, we call these users members of the agent’s overlap with the target. Seeing this content may make the target more interested in the agent and result in an increased follow probability.

We had the agents select targets by using the Twitter search API and also by
choosing the followers of users who had previously followed-back the agent. This way we were able to obtain targets with zero and positive overlap. The agents retweeted and followed each target. We plot the follow conversion rate versus the overlap in Figure 2-5. As can be seen, the conversion rate increases as the overlap increases. The increase is not monotonic, but this is likely due to not having sufficient data points for different overlap values, as seen by the 95% confidence intervals in the figure. Our regression model uses overlap as a feature along with the log transformed friend and follower counts of the target and the results are shown in Table 2.4. As can be seen, the overlap is a significant feature that increases the follow probability.

Another interesting result is found if we compare our measured conversion rates with those from [21]. In that work, the conversion rate was for users who were invited to Facebook that actually joined, and the overlap was the number of email invitations the user received from distinct people. We plot conversion rate versus overlap from
Figure 2-4: Illustration of the overlap feature between the agent and target vertex. Overlap is defined as the number of users the target follows who also follow the agent. The overlap in the figure is three.

Figure 2-5: Plot of the follow conversion rate versus overlap. The red circles are the median and the error bars are 95% confidence intervals.

our Twitter data and the Facebook data from [21] in Figure 2-6. We normalize our results so that an overlap of one has conversion rate one, just as in [21]. Remarkably, we see that the two curves are almost exactly equal, with slight differences for an overlap of four. The normalized conversion rate is the same for gaining a follow in Twitter and joining Facebook. These are very different actions in terms of effort required, and the overlap has a different meaning in each situation. This suggests that the overlap feature may be an important feature in a wide variety of social networks and its impact may be very similar.
Table 2.4: Regression over overlap and user degree features

<table>
<thead>
<tr>
<th>Feature</th>
<th>$\beta$</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>-2.49***</td>
<td>$4.10 \times 10^{-8}$</td>
</tr>
<tr>
<td>Overlap</td>
<td>0.28***</td>
<td>$6.57 \times 10^{-12}$</td>
</tr>
<tr>
<td>$\log_{10}$(Target Friend Count)</td>
<td>0.45*</td>
<td>0.05</td>
</tr>
<tr>
<td>$\log_{10}$(Target Follower Count)</td>
<td>-0.63***</td>
<td>$1.77 \times 10^{-4}$</td>
</tr>
</tbody>
</table>

The significance codes are *** : 0.001, ** : 0.01, and * : 0.05.

Figure 2-6: Plot of the normalized conversion rate for our data on follows in Twitter and Facebook signups from [21] versus overlap.
Chapter 3

The Follow-back Problem

3.1 Problem Definition

Here we formalize and introduce notation used throughout this chapter. Initially at time $n = 0$ we have a set of users and an agent vertex $a$. The users and $a$ form a network which we refer to as the friends graph $G_0 = (V, E_0)$ where $E_0$ is the initial edge set. A directed edge $(u, v)$ pointing from $u$ to $v$ for some $\{u, v \in V | u \neq v\}$ in the friends graph means that $u$ is followed by $v$. In a social network such as Twitter this means that any content $u$ posts will be visible to $v$. Each vertex $v$ has a set of features $F_v$ associated with it and we define the set of all vertex features as $F = \bigcup_{v \in V} F_v$. We will assume that $a$ initially has no edge with any other vertex in $V$. The goal in the follow-back problem is for the agent $a$ to maximize the number of follows he gets from the vertices in $V$ by interacting with them. We illustrate the setting of the follow-back problem in Figure 3-1.

At each time $n$, $a$ can interact with a single vertex in $V$. The vertex chosen by $a$ at time $n$ is denoted $u_n$. The interaction of the agent with a target vertex is some form of social network interaction. In Twitter these interactions include following, retweeting, and mentioning. We require that $u_n \neq u_i$ for any $i < n$, meaning that $a$ can only interact with each $v \in V$ once. We define the interaction policy of $a$ as the vertex sequence $\pi = \{u_1, u_2, \ldots\}$ along with the sequence of interactions $x = \{x_{u_1}, x_{u_2}, \ldots\}$. Here $x_{u_i}$ specifies the type of Twitter interaction (following, retweeting, or mentioning).
Figure 3-1: Illustration of the initial setting of the follow-back problem at time $n = 0$. The agent $a$ wants to maximize its number of followers in the friends graph by interacting with them.

with $u_i$. When at time $n$ the agent engages with $u_n$, either $u_n$ will follow $a$ or $u_n$ will not follow $a$. We let the random variable $X_{u_n}$ be one if the engagement of $a$ results in $u_n$ following $a$, otherwise $X_{u_n}$ is zero. The vertex set does not change, but the actions of $a$ can change the edge set $E_n$. If the agent’s action $x_{u_n}$ gains a follow from vertex $u_n$, then we update the graph to $G_n = (V, E_n)$ where $E_n = E_{n-1} \cup (a, u_n)$. We assume the graph is otherwise static.

The state at time $n$ is the graph $G_n$ and the vertex feature set $F$. Denote the follow probability of a vertex $u_n$ in response to the agent interaction $x_{u_n}$ as $P(X_{u_n} = 1|G_n, x_{u_n}, F)$. We are assuming here that the follow probability of $u_n$ only depends upon the current state of the graph, the type of interaction, and some fixed vertex features.

In the follow-back problem, the agent selects the policy $(\pi, x)$ to maximize the number of users $v \in V$ that will follow-back. That is, the agent chooses a policy $(\pi, x)$ to solve the following problem:

$$\max_{(\pi, x)} \sum_{v \in V} E[X_v|\pi, x, G_0, F]. \quad (3.1)$$

We will use our empirical analysis from Chapter 2 to guide our understanding of the follow probabilities, which we clarify in Section 3.2.
3.2 Assumptions on Follow Probability Model

To solve the follow-back problem we must make some general assumptions on the follow probabilities. In our general model, we assumed the follow probability depended upon vertex features, graph state, and interaction type. Recall that we saw in Table 2.2 that the type of interaction, i.e. follow, retweet, or reply, was relatively less significant than user vertex features in our follow probability regression, so we will consider only one type of generic agent interaction. We also further legitimate this simplification by recognizing that, in practice, a social media operator would tailor his interaction style on an interaction-by-interaction basis. Since we are assuming one type of generic interaction, we do not need to consider this in the functional form of the follow probability. Based on our empirical analysis, the only vertex features we need to consider are the friend and follower count of the target. For a target \( v \) we will call this set of features \( F_v \). However, our proceeding analysis applies to any general vertex features \( F_v \). Also, as our empirical analysis showed, the graph feature we need to include is the overlap between the agent and target. We define the overlap between the agent and target \( v \) as \( \phi_v \).

With these assumptions we can write the follow probability of a target \( v \) in response to an interaction by the agent as \( p(\phi_v, F_v) \). We further assume that the follow probability has the product form \( p(\phi_v, F_v) = f(\phi_v)g(F_v) \). The product form assumption can be viewed as an approximation to the logistic form of the follow probability given by equation 2.1 when the probability value is small. Assume the coefficient of the overlap is \( \beta_\phi \) and the coefficient vector of the vertex feature set is \( \beta_F \). From our logistic regression model, the follow probability is given by

\[
p(\phi_v, F_v) = \frac{e^{\beta_0 + \beta_\phi \phi_v + \beta_F^T F_v}}{1 + e^{\beta_0 + \beta_\phi \phi_v + \beta_F^T F_v}} \\
\approx e^{\beta_0 + \beta_\phi \phi_v + \beta_F^T F_v} \\
\approx f(\phi_v)g(F_v).
\]

The above expression is a good approximation when the term in the exponent is much
less than zero, which is a valid assumption given our empirical observations. We use this approximation because it allows us to obtain simple analytic solutions for the follow-back problem on different graph topologies.

We assume that the follow probabilities are monotonically increasing in the value of the overlap, which we saw in Figure 2-5. Furthermore, recall that in Section 3.1, we assumed that the agent’s interaction(s) with one user must conclude before he begins interacting with another user. In other words, the agent cannot begin interacting with one user, start communicating with another user, and switch back to interacting with the former. Therefore, the follow-back problem reduces to choosing a sequence of vertices $\pi$ in the friends graph to follow which maximizes the expected number of follows for the agent.

3.2.1 Optimal Policy on a Directed Acyclic Graph (DAG)

We begin by analyzing the follow-back problem on a basic graph topology, a directed acyclic graph (DAG). The analysis is simple here because the graph contains no cycles. Because the follow probabilities are increasing in overlap, the intuitive solution here would be to maximize the overlap for each interaction by following a vertex after following its parents. To formalize this, recall that a DAG can also be viewed as a partially ordered set (poset) on the vertices, with the partial order requiring that a vertex must come after its parents. Any sequence of vertices which respects this partial order is known as a linear extension. For DAGs, we have the following result, which matches the intuition.

**Theorem 3.2.1.** Let $\pi^*$ be the optimal policy for the follow-back problem with initial graph $G$ which is a directed acyclic graph. Then $\pi^*$ is a linear extension of $G$.

What is interesting about this result is that any linear extension is an optimal policy. This is because all linear extensions result in the same number of expected follows. Intuitively, this is true because the follow probabilities depend only upon features of the target and the overlap. Under our assumptions on the follow probabilities, interchanging the order of vertices in the policy in a way that respects the
Figure 3-2: Illustration of two optimal linear extension policies on a DAG. Any interaction sequence that respects the partial order of the graph is optimal.

Partial order does not affect any of these features. We illustrate the linear extension policy in Figure 3-2. Here there are multiple optimal policies corresponding to each linear extension of the graph. It does not matter which one is selected because they all respect the order relationships of the graph.

Proof. Let the friends graph be $G = (V, E)$ which is a DAG. Consider two vertices $u, v \in V$ such that there is a directed path from $u$ to $v$ in $G$ and consequently no path from $v$ to $u$ because $G$ is a DAG. We define two policies $\pi = \{u_1, \ldots, u, v, \ldots, u_n\}$ and $\pi' = \{u_1, \ldots, v, u, \ldots, u_n\}$. The only difference in the two policies is that $u$ and $v$ are swapped. In $\pi$ vertices $u$ and $v$ respect the partial order of $G$, whereas in $\pi'$ they do not. We will now show that the expected follows of $\pi'$ is less than $\pi$.

The expected follows of a policy $\pi$ can be written as $\sum_{w \in V} E[X_w|\pi]$, where $X_w$ is one if the agent interaction with $w$ results in a follow by $w$ and zero otherwise. We want to show that the expected follows will be larger if $u$ is followed before $v$. For any vertex $w$ we define $\delta_w = E[X_w|\pi] - E[X_w|\pi']$ and we define the difference in the expected follows of the two policies as $\delta = \sum_{w \in V} \delta_w$. For any vertex $w$ which occurs before $v$ and $u$ in either policy, we have that $\delta_w = 0$. This is because the state of the graph when the agent interacts with these vertices is the same under each policy. For vertex $u$ we have $\delta_u = 0$ because $u$ does not follow $v$ and there is no path from $v$ to $u$. Therefore, there is no way for $v$ to affect the overlap of $u$ with the agent, and thus no way to affect the subsequent follow probability. For $v$ and all vertices after $u$ and $v$ we use the following lemma.
Lemma 3.2.2. Let \((u_1, u_2, \ldots, u_n)\) be a path on a directed graph \(G\). Then \(E[X_{u_j}|X_{u_i} = 1] > E[X_{u_j}|X_{u_i} = 0]\) for \(1 \leq i < j \leq n\).

This lemma shows that if the agent can get a vertex \(w\) to follow it, it will receive a boost in the expected follows for any vertex on a path from \(w\). Now consider vertex \(v\).

\[
\delta_v = E[X_v|X_u = 1, \pi] E[X_u|\pi] + E[X_v|X_u = 0, \pi] (1 - E[X_u|\pi]) - E[X_v|X_u = 0, \pi'] = E[X_v|\pi] (E[X_v|X_u = 1, \pi] - E[X_v|X_u = 0, \pi]) > 0.
\]

Above we have used Lemma 3.2.2 along with the fact that \(X_i\) are Bernoulli random variables, so \(E[X_i] = P(X_i = 1)\). We also made use of the fact that \(E[X_v|X_u = 0, \pi] = E[X_v|X_u = 0, \pi']\) because if \(u\) does not follow the agent, the follow probability is the same under both policies when the agent interacts with \(v\).
For any vertex \( w \) that occurs after \( u \) and \( v \) we have two situations. Either there is no path from \( v \) to \( w \), in which case \( \delta_w = 0 \) because \( v \) cannot affect the follow probability of \( w \), or there is such a path. In the case where a path exists, we have

\[
\delta_w = \mathbb{E}[X_w|X_v = 1, \pi] \mathbb{E}[X_v|\pi] \\
- \mathbb{E}[X_w|X_v = 1, \pi'] \mathbb{E}[X_v|\pi'] \\
- \mathbb{E}[X_w|X_v = 0, \pi'] (1 - \mathbb{E}[X_v|\pi']) \\
= \mathbb{E}[X_w|X_v = 1] (\mathbb{E}[X_v|\pi] - \mathbb{E}[X_v|\pi']) \\
+ \mathbb{E}[X_w|X_v = 0] (\mathbb{E}[X_v|\pi'] - \mathbb{E}[X_v|\pi]) \\
= (\mathbb{E}[X_v|\pi] - \mathbb{E}[X_v|\pi']) (\mathbb{E}[X_w|X_v = 1] - \mathbb{E}[X_w|X_v = 0]) \\
= \delta_v (\mathbb{E}[X_w|X_v = 1] - \mathbb{E}[X_w|X_v = 0]) \\
> 0.
\]

Here, in addition to our result for \( \delta_v \), we also used the fact that the expected value of \( X_w \) conditioned on \( X_v \) is the same for both policies because the relevant graph state is the same. This result shows that \( \delta > 0 \), and \( \pi \) has more expected follows than \( \pi' \). Therefore, any policy can increase its expected follows by swapping any pair of adjacent vertices so they respect the partial order of the underlying DAG. This process can be continued until the policy is a linear extension of the DAG and no further increase in the expected follows is possible.

\[ \square \]

### 3.3 Expected Follows on a DAG: Follow-back Score

We know that the optimal policy on a DAG is a linear extension. A natural question to ask is what is the expected number of follows the optimal policy achieves on a DAG? Here we provide a closed form expression for the expected follows of the optimal policy. We define this expression as the follow-back score. Recall that the follow probability for a target vertex \( v \) with features set \( F_v \) and overlap \( \phi_v \) is given
by \( p(\phi_v, F_v) = f(\phi_v)g(F_v) \) as outline in Section 3.2. To obtain a simple closed form expression, we assume that \( f(\phi_v) \) is monotonically increasing and linear, i.e. \( f(\phi_v) = \alpha + \beta \phi_v \) for some \( \alpha, \beta > 0 \). For each vertex \( v \), we define its susceptibility as \( g(F_v) \).

Also, let \( \mathcal{P}_l(v, G) \) be the set of cycle free directed paths of length \( l \) in a graph \( G \) which terminate on vertex \( v \). For instance, if \( v \) is a root vertex of a DAG \( G \), then \( \mathcal{P}_0(v, G) = \{v\} \), and if \( u \) is a child of \( v \), then \( \mathcal{P}_0(u, G) = \{u\} \) and \( \mathcal{P}_1(u, G) = \{(v, u)\} \).

We have the following result for the expected follows of a linear extension policy (which is optimal) on a DAG.

**Theorem 3.3.1.** Let \( G = (V, E) \) be a DAG with \( N \) vertices. Let the follow probability for a target vertex \( v \) with features set \( F_v \) and overlap \( \phi_v \) be given by \( p(\phi_v, F_v) = f(\phi_v)g(F_v) \) and let \( f(\phi_v) = \alpha + \beta \phi_v \) for some \( \alpha, \beta > 0 \). Let \( J(G) \) be the expected follows of a linear extension policy on \( G \). Then

\[
J(G) = \alpha N - 1 \sum_{l=0}^{N-1} \beta^l \sum_{v \in V} \sum_{T \in \mathcal{P}_l(v, G)} w_T
\]

where for a path \( T \) we define

\[
w_T = \prod_{u \in T} g(F_u).
\]

**Proof.** We make use of the following lemma.

**Lemma 3.3.2.** Let \( G = (V, E) \) be a DAG with \( N \) vertices. Let the follow probability for a target vertex \( v \) with features set \( F_v \) and overlap \( \phi_v \) be given by \( p(\phi_v, F_v) = f(\phi_v)g(F_v) \) and let \( f(\phi_v) = \alpha + \beta \phi_v \) for some \( \alpha, \beta > 0 \). Assume the agent interacts with the parents of a vertex \( v \) before it interacts with \( v \). Let \( X_v \) be the random variable which is one if \( v \) follows as a result of the agent interaction sequence and zero otherwise. Also, let \( P(v) \) denote the set of parent vertices of \( v \). Then

\[
E[X_v] = \alpha \sum_{l=0}^{N-1} \beta^l \sum_{p \in \mathcal{P}_l(u, G)} \prod_{u \in p} g(F_u).
\]
With this lemma we can easily calculate the expected follows of a linear extension policy on $G$. By directly applying Lemma 3.3.2 to each vertex and then summing over all vertices. Doing so we obtain

$$J(G) = \sum_{v \in V} E[X_v]$$

$$= \alpha \sum_{l=0}^{n-1} \beta^l \sum_{v \in V} \sum_{\mathcal{P}(v,G)} \prod_{u \in \mathcal{P}} g(F_u).$$

\[\square\]

### 3.4 Low Order Approximation of $J(G)$

This expression weighs each possible follow path $T$ of length $l$ by $\beta^l w_T$. The expected number of follows is obtained by summing over all possible paths in the DAG. If we look at each term corresponding to a path length $l$ in the expression for $J(G)$ we can gain a better understanding of how the overlap plays a roll in the expected follows. For each value $l$ we define

$$J^l(G) = \alpha \beta^l \sum_{v \in V} \sum_{T \in \mathcal{P}(v,G)} w_T$$

as the contribution to the total expected follows $J(G)$ coming from a specific value of $l$ in the sum in equation (3.2). With this notation, we can rewrite equation (3.2) as

$$J(G) = \sum_{l=0}^{N-1} J^l(G).$$

For $l = 0$ we have

$$J^0(G) = \alpha \sum_{v \in V} g(F_v).$$

That is, the zeroth order contribution to $J(G)$ is the sum of the vertex susceptibilities. This makes sense, as a vertex is a one hop path. The value $J^0(G)$ is the expected follows that would be obtained if the overlap had no impact of the follow probability.
The first order term is

\[ J^1(G) = \alpha \beta \sum_{(u,v) \in E} g(F_u)g(F_v), \]

which is simply the sum over all the edges in the graph, with edge weights given by the product of the susceptibility of their end vertices. Edges can also be thought of as the one-hop paths in the graph. Continuing this way we obtain contributions from higher order paths, up to paths of length \( N - 1 \) (the longest path possible in a graph with \( N \) vertices). A useful computational question is how many terms to keep in this sum when actually calculating \( J(G) \). This is entirely dependent upon the density of the graph and the value of the susceptibilities and overlap function. To understand this question, let us assume that all susceptibilities are equal to some value \( \gamma \). Let us also define \( N_l \) as the number of paths in a graph of length \( l \). With this notation, \( N_0 \) is the number of vertices and \( N_1 \) is the number of edges. For ease of notation, we define \( N_l = N_0 \). Under this scenario, all paths of equal length have the same weight. Therefore, we can write \( J^l(G) \) as

\[ J^l(G) = \alpha \gamma (\beta \gamma)^l N_l. \]

The first term \( \alpha \gamma \) is the same for all \( l \) and can be ignored. The term \( \beta \gamma < 1 \) because the follow probability must be less than or equal to one and we assume \( \alpha > 0 \). The term \( N_l \) depends upon the density of the graph. For a sparse graph \( N_l \) will scale like \( N \), but for a dense graph we can have \( N_l \) scaling like \( N^l \). Therefore, for sparse graphs \( J^l(G) \) will decay exponentially fast in \( \beta \gamma \), while for denser graphs it will scale like \( (\beta \gamma N)^l \) which can decay or increase depending upon the value of \( \beta \gamma \) relative to \( n \). If the overlap is weak (small \( \beta \)) or the graph is small, then \( \beta \gamma n \) will be less than one and the higher order terms will be much smaller than the lower order terms. If the overlap is strong (large \( \beta \)) or the graph is large, then higher order terms may not be negligible.
3.5 Optimal Policy on a Directed Graph

The linear extension policy for a DAG does not apply to a general directed graph because there is more than one possible induced DAG for the graph due to the presence of cycles. However, the DAG analysis suggests that whatever the optimal policy is, it must not violate the order relations imposed by the graph. In fact, a simple application of Lemma 3.2.2 gives us the following result, which we state here without proof.

**Lemma 3.5.1.** Let $\pi^*$ be the optimal policy for an arbitrary directed graph $G = (V, E)$ and let $u, v \in V$. If there is a directed path in $G$ from $u$ to $v$, but not from $v$ to $u$, then $u$ must come before $v$ in $\pi^*$.

This lemma is a generalization of Theorem 3.2.1 to arbitrary graphs. It says that if there is a path between two vertices in one direction, but not in the reverse, then the vertex at the start of the path should always be followed before the other. Furthermore, because the optimal policy must respect the order imposed by the graph, then the optimal policy should be a linear extension of a DAG of the underlying graph. Because all linear extensions of a DAG are optimal, finding the optimal policy reduces to finding the optimal DAG. Formally, let $\mathcal{D}(G)$ be the set of all possible DAGs of $G$. The optimal DAG is the one that maximizes the expected follows. That is, the optimal DAG is the solution to the following optimization problem.

$$\max_{D \in \mathcal{D}(G)} J(D).$$  \hfill (3.7)

There are some features the optimal DAG should have. Consider the expression for $J(G)$ for a DAG $G$ from equation (3.6). The term $J^0(G)$ is independent of the DAG chosen as long as all vertices are included. However, $J^1(G)$ will be larger if there are more edges with heavier weights (product of end vertex susceptibilities). The same goes for larger values of $l$. This suggests that when searching for the optimal DAG, one would like to have as many “heavy” edges as possible. In fact, optimizing over the first order objective, i.e. $\max_{D \in \mathcal{D}(G)} J^1(D)$ is known as the minimum feedback arcset.
problem and is well known to be NP-hard to solve [22]. Therefore, finding an exact solution to (3.7) may not be easily done. However, an exactly optimal solution may not be necessary, as the higher order terms in $J(G)$ may not be significant, depending upon the graph structure and follow probabilities. Instead, heuristic solutions may be sufficient for practice. We will look at such heuristic policies in Chapter 4.

Our analysis thus far has shown that the optimal policy on a graph should correspond to a DAG of the underlying graph. We have also seen how to calculate expected follows on a DAG from Theorem 3.3.1. In addition, by studying the structure of the expression for the expected follows we have seen that more edges in the DAG can lead to increased expected follows.

3.5.1 Proof of Lemma 3.2.2

Proof. We will prove the result by induction. For any $1 < i < n$ we have that $E[X_{u_{i+1}}|X_{u_i} = 1] > E[X_{u_{i+1}}|X_{u_i} = 0]$ because the follow probabilities are monotonically increasing in the overlap and $u_{i+1}$ follows $u_i$. This gives our base case of $E[X_{u_2}|X_{u_1} = 1] > E[X_{u_2}|X_{u_1} = 0]$. We then assume that $E[X_{u_i}|X_{u_1} = 1] > E[X_{u_i}|X_{u_1} = 0]$ for $2 < i < n$. For the case of $i + 1$ we then have that for $k \in \{0, 1\}$

\[
E[X_{u_{i+1}}|X_{u_i} = k] = E[X_{u_{i+1}}|X_{u_i} = 1]E[X_{u_i}|X_{u_1} = k] + \\
E[X_{u_{i+1}}|X_{u_i} = 0](1 - E[X_{u_i}|X_{u_1} = k])
\]

\[
= E[X_{u_{i+1}}|X_{u_i} = 0] + \\
E[X_{u_i}|X_{u_1} = k](E[X_{u_{i+1}}|X_{u_i} = 1] - E[X_{u_{i+1}}|X_{u_i} = 0]).
\]

Now we define $\delta = E[X_{u_{i+1}}|X_{u_1} = 1] - E[X_{u_{i+1}}|X_{u_1} = 0]$. Using the above result we have

\[
\delta = (E[X_{u_{i+1}}|X_{u_i} = 1] - E[X_{u_{i+1}}|X_{u_i} = 0])(E[X_{u_i}|X_{u_1} = 1] - E[X_{u_i}|X_{u_1} = 0])
\]

\[
> 0.
\]
Above we used the induction hypothesis and the monotonicity of the follow probabilities in the overlap. 

3.5.2 Proof of Lemma 3.3.2

Proof. We require the following result.

Lemma 3.5.2. Consider a vertex \( v \) in a DAG \( G = (V, E) \) with parents given by the set \( P(v) \subset V, |P(v)| = n \). Let the susceptibility and overlap functions be \( g \) and \( f \), respectively. Assume the vertices are interacted with according to a linear extension policy \( \pi \). Let \( q_v = E[X_v|\pi] \). Then we have that

\[
q_v = g(F_v) \sum_{k=0}^{n} \Delta_k \sum_{S \subseteq P(v):|S|=k} \prod_{u \in S} q_u
\]

where we define \( \Delta_k = \sum_{i=0}^{k} a_i^k f(i) \) where \( a_i^k = a_{i-1}^k - a_i^{k-1} \), \( a_{k+1}^k = 0 \), \( a_k^k = 1 \), \( a_0^k = (-1)^k \), and \( a_{k-1}^k = 0 \).

If we assume that \( f(\phi) = \alpha + \beta \phi \), then equation (3.8) takes a simpler form. We use the following result regarding the terms \( a_i^k \), which we state without proof.

Lemma 3.5.3. Let the terms \( a_i^k \) be given in Lemma 3.5.2. Then we have that

\[
\sum_{i=0}^{k} a_i^k = 0 \text{ for } k \geq 0, \quad \sum_{i=0}^{1} a_i^1 i = 1, \quad \text{and } \sum_{i=0}^{k} a_i^k i = 0 \text{ for } k > 1.
\]

For linear \( f \), the terms \( \Delta_k \) take the following form.

\[
\Delta_k = \alpha \sum_{i=0}^{k} a_i^k + \beta \sum_{i=0}^{k} a_i^k i
\]

Using Lemma 3.5.3 we find that

\[
\Delta_0 = \alpha \\
\Delta_1 = \beta \\
\Delta_k = 0, \quad k > 1.
\]

43
Substituting this into equation (3.8) we obtain

\[ q_v = g(F_v) \left( \alpha + \beta \sum_{u \in P(v)} q_u \right) \quad (3.9) \]

This expression is a recursion in the depth of the DAG. If we expand it we obtain

\[
q_v = g(F_v) \left( \alpha + \beta \sum_{u \in P(v)} g(F_u) \left( \alpha + \beta \sum_{w \in P(u)} q_w \right) \right)
= \alpha g(F_v) + \alpha \beta \sum_{u \in P(v)} g(F_u) g(F_v) + \beta^2 \sum_{u \in P(v)} \sum_{w \in P(u)} q_u g(F_u) g(F_v)
= \alpha g(F_v) + \alpha \beta \sum_{u \in P(v)} g(F_u) g(F_v) + \beta^2 \sum_{v \in P(v)} \sum_{t \in P(u)} g(F_t) g(F_u) g(F_v) \left( \alpha + \beta \sum_{s \in P(t)} q_s \right)
= \alpha g(F_v) + \alpha \beta \sum_{u \in P(v)} g(F_u) g(F_v) + \alpha \beta^2 \sum_{u \in P(v)} \sum_{t \in P(u)} g(F_t) g(F_u) g(F_v)
+ \beta^3 \sum_{s \in P(t)} q_s g(F_t) g(F_u) g(F_v)
= \alpha \left( \beta \sum_{(u) \in P_0(v,G)} g(F_u) + \beta^2 \sum_{(u,v) \in P_1(v,G)} g(F_u) g(F_v) + \beta^3 \sum_{(u,v,w) \in P_3(v,G)} g(F_u) g(F_v) g(F_w) \right)
+ \beta^3 \sum_{(u,v,w,x) \in P_4(v,G)} q_u g(F_v) g(F_w) g(F_x).\]
If we continue this expansion higher up the DAG, then this pattern continues until we hit the roots, at which point there are no more parents and the expansion terminates, since for a root $r$, we have trivially that $q_r = \alpha g(F_r)$. What is happening is then evident. The expression of $q_v$ is summing over all paths in the DAG which terminate on $v$. Each such path $p$ containing $l$ vertices is weighted by $\alpha \beta^{l-1} \prod_{u \in p} g(F_u)$. Using this, we obtain

$$q_v = \alpha \sum_{l=0}^{N} \beta^l \sum_{p \in P_l(v,G)} \prod_{u \in p} g(F_u).$$

(3.10)

The total follows on the DAG is given by

$$J(G) = \sum_{v \in V} q_v$$

$$= \alpha \sum_{l=0}^{N} \sum_{v \in V} \beta^l \sum_{p \in P_l(v,G)} \prod_{u \in p} g(F_u).$$

3.5.3 Proof of Lemma 3.5.2

Proof. We prove the result by induction. The induction hypothesis is given by equation (3.8). We assume that $v$ has $n$ parents. For $n = 0$, we have $q_v = f(0)g(F_v)$, which satisfies the induction hypothesis. For $n = 1$, let the parent be $u$. From the definition in Lemma 3.5.2 we have that $\Delta_0 = f(0)$ and $\Delta_1 = f(1) - f(0)$. Then we have

$$q_v = g(F_v) (f(0)(1 - q_u) + f(1)q_u)$$

$$= g(F_v) (f(0) + q_u(f(1) - f(0)))$$

$$= g(F_v)(\Delta_0 + q_u\Delta_1)$$

which also satisfies the induction hypothesis. For the $n + 1$ case, let the additional vertex added be denoted $n + 1$. Conditional on this vertex not following, $q_v$ is not
changed. Conditional on this vertex following, the overlap increases by one. The value of $q_v$, conditioned on this event, is given by equation (3.8), except that the argument of $f$ is increased by one in the definition of $\Delta_k$. To ease our notation, let us define $\Delta'_k = \sum_{i=0}^{k} a_i^k f(i+1)$. We let $P(v)$ be the parents of $v$ not including $n + 1$. Then we have

$$q_v = \mathbb{E}[X_v|X_{n+1} = 0](1 - q_{n+1}) + \mathbb{E}[X_v|X_{n+1} = 1]q_{n+1}$$

$$= (1 - q_{n+1})g(F_v) \sum_{k=0}^{n} \Delta_k \sum_{S \subseteq P(v):|S|=k} \prod u \in S q_u$$

$$+ q_{n+1}g(F_v) \sum_{k=0}^{n} \Delta'_k \sum_{S \subseteq P(v):|S|=k} \prod u \in S q_u$$

$$= g(F_v) \sum_{k=0}^{n} \Delta_k \sum_{S \subseteq P(v):|S|=k} \prod u \in S q_u$$

$$+ q_{n+1}g(F_v) \sum_{k=0}^{n} (\Delta'_k - \Delta_k) \sum_{S \subseteq P(v):|S|=k} \prod u \in S q_u \quad (3.11)$$

The difference $\Delta'_k - \Delta_k$ is given by

$$\Delta'_k - \Delta_k = \sum_{i=0}^{k} a_i^k (f(i+1) - f(i))$$

$$= a_k^k f(k+1) - a_k^k f(k) + a_{k-1}^k f(k) - a_k^{k-1} f(k) + \ldots + a_0^k f(1) - a_0^0 f(0)$$

$$= f(k+1)(a_k^k - a_{k+1}^k) + f(k)(a_k^{k-1} - a_k^k) + \ldots + f(0)(a_0^k - a_{-1}^k)$$

$$= \sum_{i=0}^{k+1} a_i^{k+1} f(i)$$

$$= \Delta_{k+1}.$$
Above we used the definition of $a^k_i$ given in Lemma 3.5.2. Substituting this into equation (3.11) and letting $P'(v)$ denote the parents of $v$ including $n + 1$, we obtain

$$q_v = g(F_v) \sum_{k=0}^{n} \Delta_k \sum_{S \subseteq P(v), |S| = k} \prod_{u \in S} q_u + q_{n+1} g(F_v) \sum_{k=0}^{n} \Delta_{k+1} \sum_{S \subseteq P'(v), |S| = k} \prod_{u \in S} q_u$$

$$= g(F_v) \sum_{k=0}^{n} \Delta_k \sum_{S \subseteq P(v), |S| = k} \prod_{u \in S} q_u + \sum_{k=1}^{n+1} \Delta_{k+1} \sum_{S \subseteq P'(v), |S| = k} \prod_{u \in S} q_u$$

$$= g(F_v) \sum_{k=0}^{n+1} \Delta_k \sum_{S \subseteq P'(v), |S| = k} \prod_{u \in S} q_u.$$

The above expression matches the induction hypothesis, completing the proof.
Chapter 4

Integer Programming Formulation of the Follow-back Problem

4.1 Set-up and Motivation

Our empirical analysis led us to assume a follow probability model that was monotonically increasing in the agent’s overlap with the target, as was explained in Section 3.2. This implies that $a$ could further maximize his expected follows among a set of desired users by interacting with these users’ friends first, even if $a$ does not primarily wish to get follows from them. Gaining “extra” follows among these other users will further maximize the number of follows $a$ gets from his true set of targets. For social media operators who have the resources to interact with more users than the ones they want to connect with, this strategy could allow them to further maximize their chance of making the desired connections. We now see how our theoretical result from Chapter 3 can be used in an integer programming formulation to help derive an interaction policy. Here, the friends graph is a union of multiple targets’ friends graphs. There might be connections between the different targets’ friends and the targets themselves. We exploit all of this information when constructing an interaction policy that maximizes expected follows the agent gets from the targets.
4.2 Integer Program

We use constrained integer programming to formulate the problem. We add a constraint that limits the number of interactions that the agent is allowed and makes the problem more practically applicable. Despite the increased complexity of the problem, we use the intuition gained from earlier analysis to formulate our objective. We seek to find a directed acyclic subgraph with the maximum number of paths of maximally weighted vertices that lead into the targets, subject to the number of vertices in the subgraph being less than the maximum number of interactions allowed. In the basic problem where the agent wanted to gain follows from all users in the friends graph, we found that deriving an optimal policy involved selecting a directed acyclic subgraph of maximal followback score. Recall that followback score was calculated by taking a sum of all the vertex susceptibility weighted products of the paths in the graph as shown in equation (3.2). The objective function in our integer program (4.3) is doing the same thing; however, since we are only concerned with the expected follow probabilities of the targets, we restrict the vertex sum in equation (3.2) to be taken over the set of targets instead of the entire graph. Since the acyclic subgraph constitutes a valid partial order, we can, as before, derive an optimally sequenced interaction policy by taking a linear extension of the graph.

We assume we are given an arbitrary directed social network $G = (V, E)$. $V$ is the set of vertices in the graph and $|V| = N$. $E$ is the adjacency matrix of the graph, such that $E_{i,j} = 1$ if the directed edge $(i, j)$ is in $G$. We are given a subset of targets $T \subseteq V$ from whom the agent seeks to gain followers. We give each vertex $v \in V$ a weight $g_v$, its a priori susceptibility. We let $C_G$ be the set of cycles of $G$. Then form a DAG $G^* = (V_{G^*}, E_{G^*})$ from $G$ using the integer program (4.3). $V_{G^*}$ is the set of vertices in the DAG $G^*$, and $E_{G^*}$ is its adjacency matrix. The objective function corresponds to equation (3.2) and the constraints limit the maximum number of interactions that the agent is allowed to $m$, ensure that the graph is acyclic, and guarantee that selected edges have incident vertices that are also selected. This DAG is optimal for the problem, and Section 3.5 showed us that any linear extension of
the DAG is an optimal interaction sequence. We define vertex variables $x_i$ as

$$x_i = \begin{cases} 
1, & \text{if } v_i \in G^* \\
0, & \text{otherwise}
\end{cases} \quad (4.1)$$

and edge variables $y_{i,j}$ as

$$y_{i,j} = \begin{cases} 
1, & \text{if } (i, j) \in G^* \\
0, & \text{otherwise}
\end{cases} \quad (4.2)$$

$$\max_{x,y} \sum_{t \in T} g_t x_t + \sum_{t \in T} \sum_{l=1}^{N-1} \sum_{P \in \mathcal{P}(t,G)} \beta^l \prod_{u \in P} g_u \prod_{(i,j) \in P} y_{i,j}$$

Subject to

$$\sum_{i \in V} x_i \leq m$$

$$y_{i,j} \leq E_{i,j} \quad \forall (i, j) \in E$$

$$\sum_{(i,j) \in C} y_{i,j} \leq |C| - 1 \quad \forall C \in \mathcal{C}$$

$$x_i - y_{i,j} \geq 0 \quad \forall i, j$$

$$x_{j} - y_{i,j} \geq 0 \quad \forall i, j$$

$$x_i \in \{0, 1\}$$

$$y_{i,j} \in \{0, 1\} \quad (4.3)$$

### 4.3 Tractability Issues

#### 4.3.1 Lazy Constraints

First, our acyclic constraint in (4.3) creates scalability issues. The number of cycles in the graph may be exponential in the number of vertices. Enumerating these cycles is not generally possible and since each cycle must have a corresponding constraint
to eliminate it from the set of feasible solutions (recall that the solution must be a DAG), this could lead to a constraint set that is exponential in the number of vertices of the graph as well.

We rely upon a commonly used practice in optimization known as lazy constraint generation. The premise is that many of the acyclic constraints in the formulation (4.3) are redundant. Lazy constraints enable us to introduce a new constraint only when it is necessary [31]. On our first iteration, we solve the integer program in (4.3) without acyclic constraints. We check the optimal solution to the relaxed problem. If we find a cycle in the relaxed solution, we add an acyclic constraint to the relaxed formulation which makes the existence of this cycle infeasible. We continue this process, iteratively adding acyclic constraints to our relaxed formulation until we arrive at a solution that is feasible to the complete formulation (4.3). Since it is optimal for the relaxed problem, it must be optimal to the complete formulation (4.3). Our hope is that we arrive at a feasible solution to (4.3) before we have to enumerate all of the cycles in the graph.

4.3.2 Low Order Approximation of Expected Follows

Secondly, the integer program (4.3) is highly nonlinear. The nonlinear terms come about from the inclusion of higher order paths in the objective function. The authors are not aware of any solvers that would be able to handle such a problem in reasonable time. We will not analyze the complexity of the general integer program, but rather, choose to focus on viable integer-programming based heuristics that exhibit good performance.

Recall that the objective function of (4.3) sums over all of the paths in the DAG and weights them by the vertex susceptibilities along the path as well as a constant term that decreases exponentially in path length. We consider all weighted paths of length \( l \) included in the objective function to be the \( l \)th order contribution to the objective value. Note that the objective function of (4.3) includes paths of all lengths, which, for an \( N \) node graph, could be as large as the \( N - 1 \)st order. The higher order terms in the function give rise to the high degree of non-linearity in it.
Table 4.1: Twitter Network Properties

<table>
<thead>
<tr>
<th>Friends graph</th>
<th>@ORC</th>
<th>@zlisto</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of vertices</td>
<td>127</td>
<td>319</td>
</tr>
<tr>
<td>Number of edges</td>
<td>792</td>
<td>537</td>
</tr>
<tr>
<td>Density</td>
<td>0.05</td>
<td>0.005</td>
</tr>
<tr>
<td>Mean degree</td>
<td>6.24</td>
<td>1.68</td>
</tr>
<tr>
<td>Max out degree</td>
<td>41</td>
<td>25</td>
</tr>
<tr>
<td>Min out degree</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Max in degree</td>
<td>66</td>
<td>42</td>
</tr>
<tr>
<td>Min in degree</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Number weak components</td>
<td>22</td>
<td>135</td>
</tr>
<tr>
<td>Largest component diameter</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>Sum of vertex susceptibilities</td>
<td>50.67</td>
<td>85.34</td>
</tr>
</tbody>
</table>

since the objective must include products of the binary edge variables $y_{ij} \forall (i, j)$ in each path.

We explore the contribution to the objective value that each order term provides for two Twitter graphs that were small enough to allow us to calculate the expected follows completely on their induced DAGs, and we show the contribution by order in Figure 4-1. We rely upon notation from section 3.5.3, where $J_l$ represents the $lth$ order contribution to the objective equation (3.6).

Approximately 90% of the objective value for the @zlisto graph is captured by the zero and first order contributions of the objective value and more than 99% of the objective value is captured by the second order. On the other hand, the first order contribution only captures approximately 25% of the objective value for the @ORC graph and 99% of the value is not captured until the seventh order.

A summary of the two graphs can be found in Table 4.1. We did not find these results surprising since the @ORC friends graph is 100x more dense than the @zlisto friends graph. In general, we expect more dense graphs to have a greater proportion of their objective value captured by higher order paths, since a greater number of longer paths exist in these graphs.
4.4 Integer Program Based Heuristic

Despite the impact that higher order terms have in the objective value on dense graphs, we decide to explore optimizing over the first $k$ lower order terms, $\sum_{l=0}^{k} J^l(G)$, for $k = 0, 1, 2$. We argue that optimizing over lower order terms exclusively is acceptable firstly because of tractability. As discussed earlier, solving our objective to optimality in the second order, $k = 2$ is closely related to the minimum feedback arc set problem and was shown to be NP-hard to solve in [22]. There is an abundance of research on the problem and methods to solve it using integer programming exist in [2]. We also legitimize focusing on lower order terms in our optimization because we saw that for low density graphs, lower order terms carry a high proportion of the objective value in Section 4.3, and a majority of the Twitter network is of density less than 1% [9].

By neglecting the higher order terms in the objective, we potentially lose value that could have been gained from higher order paths into the targets. To compensate, we introduce a centrality regularization function into the objective. Let $c$ be a vector of a common network centrality measure applied to the vertices of $G$, such as out-degree centrality, eigenvector centrality, closeness centrality, etc. $c_i$ corresponds to the
value of the specific centrality score for vertex \(i\). We scale each centrality score by a tuning constant \(\lambda_i\) so that the centrality measures are given appropriate weight in the optimization. We will experiment with different centrality measures for regularization and see their relative performance.

To be more concise, we present the three separate formulations, corresponding to values of \(k = 0, 1, 2\), as one in the formulation (4.4). Note that the formulations are identical except for the sum, which designates the length of paths that are included in the objective function. For instance, when \(k = 1\), we only optimize over paths of length zero and length one. When \(k = 2\), we optimize over paths of lengths zero, one, and two.

\[
\max_{x,y} \sum_{t \in T} g_t x_t + \sum_{i \in V} \lambda_i c_i x_i + \sum_{t \in T} \sum_{l=1}^{k} \sum_{P \in \mathcal{P}(t, G)} \beta_l \prod_{u \in P} g_u \prod_{(i,j) \in P} y_{i,j}
\]

Subject to

\[
\sum_{i \in V} x_i \leq m
\]
\[
y_{i,j} \leq E_{i,j} \quad \forall (i, j) \in E
\]
\[
\sum_{(i,j) \in C} y_{i,j} \leq |C| - 1 \quad \forall C \in \mathcal{C}
\]
\[
x_i - y_{i,j} \geq 0 \quad \forall i, j
\]
\[
x_j - y_{i,j} \geq 0 \quad \forall i, j
\]
\[
x_i \in \{0, 1\} \quad \forall i, j
\]
\[
y_{i,j} \in \{0, 1\} \quad (4.4)
\]

The matrix associated with the second order optimization problem, when \(k = 2\), is not generally positive semi-definite. To ensure that optimization solvers can handle the formulation, we seek to make the objective linear. For \(k = 2\), we add additional binary variables \(z_{i,j,k}\) to be associated with two hop paths \((i, j, k)\) in the formulation (4.5) below. This avoids the non-linearity in the math program 4.4 when we multiply \(y_{i,j}y_{j,k}\) to represent the this path. At the same time, it increases the number of binary

55
variables in our formulation exponentially. For an $N$ vertex graph, this would yield $N^3$ binary variables. We define $z_{i,j,k}$ as

$$z_{i,j,k} = \begin{cases} 
1, & \text{if } (i, j, k) \in G^* \\
0, & \text{otherwise}
\end{cases}$$

and the resulting integer programming formulation is

$$\max_{x,y,z} \sum_{t \in T} \sum_{i \in V} g_i x_i + \lambda_i c_i x_i + \beta g_i g_t y_{i,t} + \beta^2 \sum_{j \in V} g_i g_j g_t z_{i,j,t}$$

Subject to

$$\sum_{i \in V} x_i \leq m$$

$$y_{i,j} \leq E_{i,j} \forall (i,j) \in E$$

$$\sum_{(i,j) \in C} y_{i,j} \leq |C| - 1 \quad \forall C \in \mathcal{C}$$

$$x_i - y_{i,j} \geq 0 \quad \forall i, j$$

$$x_j - y_{i,j} \geq 0 \quad \forall i, j$$

$$z_{i,j,k} - y_{i,j} \leq 0 \quad \forall i, j, k$$

$$z_{i,j,k} - y_{j,k} \leq 0 \quad \forall i, j, k$$

$$z_{i,j,k} - y_{i,j} - y_{j,k} + 1 \geq 0 \quad \forall i, j, k$$

$$x_i \in \{0, 1\}$$

$$y_{i,j} \in \{0, 1\}$$

$$z_{i,j,k} \in \{0, 1\}$$

(4.6)
Table 4.2: Computer Scientist Friends Graph

<table>
<thead>
<tr>
<th>Friends graph</th>
<th>@ylecun, @SebastianThrun, @AndrewYNg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of targets</td>
<td>3</td>
</tr>
<tr>
<td>Number of vertices</td>
<td>218</td>
</tr>
<tr>
<td>Number of edges</td>
<td>3140</td>
</tr>
<tr>
<td>Density</td>
<td>0.07</td>
</tr>
<tr>
<td>Mean degree</td>
<td>14.40</td>
</tr>
<tr>
<td>Max out degree</td>
<td>66</td>
</tr>
<tr>
<td>Min out degree</td>
<td>1</td>
</tr>
<tr>
<td>Max in degree</td>
<td>113</td>
</tr>
<tr>
<td>Min in degree</td>
<td>0</td>
</tr>
<tr>
<td>Number weak components</td>
<td>1</td>
</tr>
<tr>
<td>Largest component diameter</td>
<td>6</td>
</tr>
<tr>
<td>Sum of vertex susceptibilities</td>
<td>46.77</td>
</tr>
</tbody>
</table>

4.5 Simulation

4.5.1 Network

We simulate our integer programming heuristic for the general follow-back problem on a Twitter friends network of three famous computer scientists, Yann LeCun, Andrew Ng, and Sebastian Thrun, who are our targets. We use the Twitter advanced programming interface to compile the collective “friends graph” which includes the three different users, the union of their friends, and the friend/follower connections between all. The properties of the graph are in Table 4.2.

4.5.2 Policies

We use the integer program (4.4) for $k = 0, 1$ and the its linear version (4.6) for $k = 2$ to solve for the optimal policies. For this simulation, we set $m = 20$, implying that the agent is allowed to interact with at most twenty users in the friends’ graph. We derive one policy for $k = 2$ ($2^{nd}$ Order) that optimizes over zero, first, and second order paths in the objective but does not consider vertex or edge centralities, so $c_i = 0 \forall i \in V$. We derive another policy for $k = 1$ ($1^{st}$ Order) that optimizes over the first order paths exclusively and another for $k = 0$ that only consider paths of length zero ($0^{th}$ Order).
We then experiment with different centrality regularization vectors $\mathbf{c}$ such as indegree centrality, out-degree centrality, eigenvector centrality, edge-betweenness centrality (in which case we modify our objective, replacing the term $c_i x_i$ with $c_{i,j} y_{i,j}$, where $c_{i,j}$ is the edge betweenness score between $i$ and $j$ and $y_{i,j}$ is still the binary edge variable associated with $(i, j)$). We normalize each centrality score so that they sum to one, i.e. $\lambda_i = \frac{c_i}{\sum_{i \in V} c_i}$.

For each centrality measure and for each value of $k = 0, 1, 2$, we derive policies using the integer program (4.4). For example, the policy “Eigenvector 1st Order” sets $\mathbf{c}$ to be the vector of eigenvector centralities of the vertices of $G$ and sets $k = 1$ in the math program (4.4). In other words it selects a DAG by optimizing over the vertex weights of the targets, the eigenvector centrality regularization function and the paths of length one terminating on one of the targets. The policy “Eigenvector 2nd Order” does the same but also includes paths of length two which terminate on a target.

We compare the optimization-derived policies with two random policies: 1) a policy “Random Append” that selects a random permutation of seventeen members of the friends graph and then appends a random permutation of the three target users and 2) another policy “Random” that realizes a random permutation of twenty users in the graph that must include the three targets. Note that the difference between the two random policies is that the first one constrains the three targets to be at the end of the sequence, whereas the second version allows the three targets to fall anywhere in the interaction sequence.

In addition, for each centrality measure, we compare the integer programming policies with policies that are based solely on network centrality measures. For example the policy “Eigenvector Centrality” takes the input “friends graph”, removes the three targets, orders the seventeen vertices with the greatest eigenvector centrality in decreasing order of their scores. Then, calculates the eigenvector centralities on the three-vertex graph with just the targets, orders the targets by decreasing eigenvector centrality score, and appends this sequence to the already ordered list of seventeen friends to make a sequence of twenty vertices. The other vertex centrality policies are
To derive a policy based exclusively on edge-betweenness scores, we divided the original graph into two subgraphs as before: one of the targets and one of all the non-targets. We weighted each edge by its edge-betweenness score. Then, we took a maximum branching of each graph. Recall that a maximum branching forest for a directed graph is a subgraph such that each vertex has at most one parent and the edges included in the subgraph are of maximal weight. Since the two branchings were DAGs, we could take topological sorts of each. The final policy was the first seventeen users in the non-target topological sort, followed by the ordering of the target graph’s maximum branching topological sort.

4.5.3 Simulation Methodology

We evaluate the performance of a policy on the computer scientist Twitter graph as follows. We simulate an interaction with the the first vertex in the policy. The result of this interaction is given by a Bernoulli random variable that is one with probability given by the follow probability model outlined in Section 3.2. We then update the overlap of any other vertex in the graph as a result of this interaction. In particular, if this vertex follows back, then all of its children in the graph increase their overlap by one. We note that all edges of the graph are used to simulate the follows. We repeat this process for each vertex in the policy until we have reached the maximum allowed number of interactions \( m \). When we reach the end of the policy, we evaluate if any and which of the primary targets (@ylecun, @SebastianThrun, and @AndrewYNg) followed-back. For our follow probability model, we use the coefficients learned in our regression from Table 2.4. However, we decided to remove the intercept term, i.e. set \( B_0 = 0 \), in the logistic model outlined in Section 3.2 for the simulation. The reason we did this was to inflate the prior follow probabilities of each target. Since the agents that we used to learn the follow probability model used the Twitter programming interface to automatically post content, we assumed that their ability to gain followers would be significantly less than a human operator who could control the account and interact with greater skill.
We simulate each policy 10,000 times and show the percentage (in decimal form) of times that each user followed-back as well as the sum of the three values. For each user, the maximum possible value is 1 and the maximum possible value of the sum is 3. The complete results are listed in Table 4.3 and Figure 4-2 shows the relative performance of the different network centrality regularizers with different orders included in the objective.

As we would expect, the random policy that constrains the targets to fall at the end of the sequence outperforms the random policy that allows the targets to appear anywhere. Since the follow probabilities are increasing in overlap, we maximize the follow probability of a user by being followed by his friends first. Most of the policies that are based solely on centrality measures perform similarly to the random policies. Of note, the policies based on in-degree centrality and eigenvector centrality perform better than the other centrality-based policies. Eigenvector centrality performs 20% better than the next best centrality-based policy. Since eigenvector centrality weights the importance of its nodes by the importance of its neighbors [6], we intuitively

Figure 4-2: Simulation with different network centrality regularization and different order
<table>
<thead>
<tr>
<th>Policy</th>
<th>@ylecun</th>
<th>@SebastianThrun</th>
<th>@AndrewYNg</th>
<th>Sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>0&lt;sup&gt;th&lt;/sup&gt; Order</td>
<td>0.143</td>
<td>0.083</td>
<td>0.110</td>
<td>0.336</td>
</tr>
<tr>
<td>1&lt;sup&gt;st&lt;/sup&gt; Order</td>
<td>0.447</td>
<td>0.083</td>
<td>0.140</td>
<td>0.670</td>
</tr>
<tr>
<td>2&lt;sup&gt;nd&lt;/sup&gt; Order</td>
<td>0.442</td>
<td>0.086</td>
<td>0.143</td>
<td>0.670</td>
</tr>
<tr>
<td>Random</td>
<td>0.177</td>
<td>0.090</td>
<td>0.144</td>
<td>0.410</td>
</tr>
<tr>
<td>Random Append</td>
<td>0.260</td>
<td>0.087</td>
<td>0.156</td>
<td>0.503</td>
</tr>
<tr>
<td>Eigenvector Centrality</td>
<td>0.378</td>
<td>0.084</td>
<td>0.169</td>
<td>0.630</td>
</tr>
<tr>
<td>Eigenvector 0&lt;sup&gt;th&lt;/sup&gt; Order</td>
<td>0.143</td>
<td>0.080</td>
<td>0.111</td>
<td>0.334</td>
</tr>
<tr>
<td>Eigenvector 1&lt;sup&gt;st&lt;/sup&gt; Order</td>
<td>0.443</td>
<td>0.087</td>
<td>0.151</td>
<td>0.682</td>
</tr>
<tr>
<td>Eigenvector 2&lt;sup&gt;nd&lt;/sup&gt; Order</td>
<td>0.407</td>
<td>0.082</td>
<td>0.155</td>
<td>0.643</td>
</tr>
<tr>
<td>In-Degree Centrality</td>
<td>0.263</td>
<td>0.086</td>
<td>0.186</td>
<td>0.535</td>
</tr>
<tr>
<td>In-Degree 0&lt;sup&gt;th&lt;/sup&gt; Order</td>
<td>0.141</td>
<td>0.086</td>
<td>0.110</td>
<td>0.337</td>
</tr>
<tr>
<td>In-Degree 1&lt;sup&gt;st&lt;/sup&gt; Order</td>
<td>0.421</td>
<td>0.084</td>
<td>0.165</td>
<td>0.670</td>
</tr>
<tr>
<td>In-Degree 2&lt;sup&gt;nd&lt;/sup&gt; Order</td>
<td>0.422</td>
<td>0.088</td>
<td>0.138</td>
<td>0.647</td>
</tr>
<tr>
<td>Closeness Centrality</td>
<td>0.164</td>
<td>0.086</td>
<td>0.139</td>
<td>0.389</td>
</tr>
<tr>
<td>Closeness 0&lt;sup&gt;th&lt;/sup&gt; Order</td>
<td>0.145</td>
<td>0.082</td>
<td>0.111</td>
<td>0.338</td>
</tr>
<tr>
<td>Closeness 1&lt;sup&gt;st&lt;/sup&gt; Order</td>
<td>0.460</td>
<td>0.082</td>
<td>0.156</td>
<td>0.692*</td>
</tr>
<tr>
<td>Closeness 2&lt;sup&gt;nd&lt;/sup&gt; Order</td>
<td>0.452</td>
<td>0.081</td>
<td>0.147</td>
<td>0.680</td>
</tr>
<tr>
<td>Out-Degree Centrality</td>
<td>0.177</td>
<td>0.090</td>
<td>0.144</td>
<td>0.410</td>
</tr>
<tr>
<td>Out-Degree 0&lt;sup&gt;th&lt;/sup&gt; Order</td>
<td>0.146</td>
<td>0.086</td>
<td>0.104</td>
<td>0.336</td>
</tr>
<tr>
<td>Out-Degree 1&lt;sup&gt;st&lt;/sup&gt; Order</td>
<td>0.470</td>
<td>0.084</td>
<td>0.132</td>
<td>0.687</td>
</tr>
<tr>
<td>Out-Degree 2&lt;sup&gt;nd&lt;/sup&gt; Order</td>
<td>0.425</td>
<td>0.083</td>
<td>0.148</td>
<td>0.647</td>
</tr>
<tr>
<td>Load Centrality</td>
<td>0.205</td>
<td>0.078</td>
<td>0.156</td>
<td>0.439</td>
</tr>
<tr>
<td>Load 0&lt;sup&gt;th&lt;/sup&gt; Order</td>
<td>0.144</td>
<td>0.082</td>
<td>0.109</td>
<td>0.335</td>
</tr>
<tr>
<td>Load 1&lt;sup&gt;st&lt;/sup&gt; Order</td>
<td>0.370</td>
<td>0.087</td>
<td>0.156</td>
<td>0.614</td>
</tr>
<tr>
<td>Load 2&lt;sup&gt;nd&lt;/sup&gt; Order</td>
<td>0.373</td>
<td>0.079</td>
<td>0.155</td>
<td>0.607</td>
</tr>
<tr>
<td>Edge-Betweenness Centrality</td>
<td>0.246</td>
<td>0.084</td>
<td>0.161</td>
<td>0.492</td>
</tr>
<tr>
<td>Edge-Betweenness 0&lt;sup&gt;th&lt;/sup&gt; Order</td>
<td>0.182</td>
<td>0.084</td>
<td>0.164</td>
<td>0.430</td>
</tr>
<tr>
<td>Edge-Betweenness 1&lt;sup&gt;st&lt;/sup&gt; Order</td>
<td>0.431</td>
<td>0.080</td>
<td>0.138</td>
<td>0.648</td>
</tr>
<tr>
<td>Edge-Betweenness 2&lt;sup&gt;nd&lt;/sup&gt; Order</td>
<td>0.4417</td>
<td>0.085</td>
<td>0.149</td>
<td>0.675</td>
</tr>
</tbody>
</table>
understand that ordering interactions based upon this measure might help to capture valuable paths in the graph that helps increase the overlap phenomenon.

For each centrality measure, both first-order and second-order integer programming policies outperform the purely centrality-based policy. This is not surprising since the integer programming policies take into account the a priori vertex susceptibilities whereas the centrality policies do not. The integer programming policies bias towards selecting vertices that have high prior follow probabilities. Some of the integer programming policies that optimize over the centrality regularizer outperform the integer program that optimizes over first and second order paths exclusively while others do not. Specifically the eigenvector, out-degree, in-degree, and edge betweenness regularizers perform better than the path-only policies. Centrality regularizers can only add value to the integer program if the correct centrality measure is selected.

There is little difference between the performance of the first order integer programming policies and the second order integer programming policies. The first order policy with no centrality regularization performed virtually the same as its second order counterpart. In fact, for all the centrality regularizers except edge-betweenness, the first order policies outperform the integer programs that include second order terms, as shown in Figure 4-2. We believe this might have taken place because our approximation for calculating expected follows assumes that the follow probability is linear in overlap, which may not be the case. Thus, the expression might be over-counting the impact that second order terms have on the objective value.

We found the first order objective, i.e. $k = 1$, along with the closeness centrality regularization function performed the best. We show its performance relative to the integer programming policies without regularization as well as the random policies in figure 4-3. The $0^{th}$ Order policy is simply sum of the a priori susceptibilities of the target: it is equivalent to the agent interacting with the targets before gaining overlap with the target’s friends, without leveraging the target’s social connections. The best policy outperformed this most naive policy by nearly 110%. The best policy outperforms policies of random interaction by as much as 38%. Adding the centrality regularization saw an improvement of 2% over policies that just optimized
over first and second order paths. In general, we posit that different network centrality regularization functions may perform better on other topologies.

These findings also imply that including second order terms in the optimization may not be essential, and using an appropriately selected network centrality regularizer could serve as a valuable surrogate for the higher order terms of the problem. This result is particularly valuable when using integer programming to solve for follow-back policies on large scale graphs. By removing the second-order term from the objective, we can reduce the size of our problem from $N^3$ binary variables to $N^2$, significantly increasing tractability.
Chapter 5

Future Work and Conclusions

5.1 Future Work

We now explore some avenues of future work.

5.1.1 Optimal Timing of Interactions

After determining an interaction sequence, an agent would want to know how to time his interactions. For instance, how long after interacting with the first user in the sequence should the agent wait until he interacts with the second user, and so on? Should the agent wait until the first user decides to follow him? What if this never occurs? The agent should not be expected to wait indefinitely. What if there is a cost per unit time that is incurred while the targets are not following the agent? If the agent does not gain the targets as followers after some period of time, perhaps the targets decide to follow a competitor. The longer the agent takes to acquire the targets as followers, the greater the chance that they are acquired by a competitor. If the agent is aware that a user will take too long to follow-back, he might decide not to waste time interacting with that user.

To incorporate timing into our optimization, we must first understand if and what features of a Twitter user’s behavior is indicative of how quickly he will follow-back. We conducted one final experiment with the same agents we used in Chapter 2.
Table 5.1: Follow-back Delay

<table>
<thead>
<tr>
<th>Feature</th>
<th>$\beta$</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>2.38</td>
<td>0.07</td>
</tr>
<tr>
<td>Number of tweets</td>
<td>-4.97</td>
<td>0.67</td>
</tr>
<tr>
<td>Percentage retweets</td>
<td>8.26</td>
<td>0.62</td>
</tr>
<tr>
<td>Percentage hashtag</td>
<td>6.01</td>
<td>0.73</td>
</tr>
<tr>
<td>Percentage media</td>
<td>6.39</td>
<td>0.70</td>
</tr>
</tbody>
</table>

The significance codes are ****: 0.001, ***: 0.01, and *: 0.05.

We had each agent interact with users that had tweeted about “Morocco” or “arts and crafts” within the past twenty-four hours. Then, we had one of our six agents interact with these users, where an interaction was a retweet and a follow. After interacting with each user, we had the agent check its followers list each minute to see if and when the user it interacted with followed the agent.

Over a 1.5 week period, the agents interacted with 500 Twitter users. Of the 500 interactions, 64 users followed the agents back. The 13% follow-back rate is consistent with what we observed in 2. For each user that followed-back, we gathered some of the features of that user’s Twitter behavior such as the number of tweets they have posted in total, the percentage of these tweets that are retweets, the percentage that contain pieces of media such as linked photos are videos, and the percentage that contain hashtags. We then used linear regression to predict that amount of time it took the users to follow-back, regressing on these features. The results of the regression are shown in Table 5.1.

None of the features we checked were significant in predicting the delay in how long it takes a user to follow-back. However, we believe that given enough data and the proper experimental set up, some user features may be significant in predicting the follow-back delay. We conjecture that people who use Twitter more frequently are more inclined to follow-back faster, as they will likely see the agent’s interaction sooner. A future line of research could involve finding viable features.

Modeling time delays between interactions would also increase the complexity of our general follow-back dynamic programming problem presented in Section 3.1. Future research could use the framework of continuous-time dynamic programming to
analyze the problem and gain insight. Incorporating timing decisions into the integer program outlined in Chapter 4 would greatly increase its complexity. This is another area for future research.

5.1.2 Integer Program Scaling Parameter $\lambda$

We introduce an integer program (4.4) in Chapter 4. Recall that the program (4.4) truncates the higher order terms in the objective function of (4.3). To compensate for these terms, we used a network centrality regularization function in (4.4). We use a scaling factor $\lambda$ as a way to weight the relative contribution of the centrality regularizer versus the lower order terms of the follow-back score, $\sum_{k=0}^{k} J^k(G)$ for $k = 0, 1, 2$. In our simulation, we arbitrarily, selected $\lambda$ so that the centrality scores of the vertices would sum to one, i.e. $\sum_{i \in V} \lambda_i c_i = 1$. Another area of future research may involve selecting $\lambda$ in a more clever way.

5.1.3 Maintaining Connections

The follow-back problem focused on getting users to follow the agent. However, once a connection is established, it can be strengthened through other types of interactions such as replying or retweeting. A natural next step in this work is to develop optimal policies for maintaining or strengthening connections in a social network. Further empirical analysis will be needed to understand the dynamics of repeated interactions. However, once a model for the effect of such interactions on a social connection is developed, the follow-back problem can be extended to find optimal policies for maintaining or strengthening connections through repeated interactions in a social network.

5.2 Conclusion

We have proposed the follow-back problem for gaining specific followers on Twitter. We conducted empirical studies in Twitter to find what user features and interaction
types cause users to follow. Specifically, we identified the importance of an agent’s local “overlap” with a target user in gaining the target’s follow. Using this analysis we construct a simple model for the follow probability. We then use this model to develop optimal policies for the follow-back problem. Through this we develop the notion of a follow-back score on a directed acyclic graph.

We use follow-back score in a constrained integer program to select an optimal directed acyclic subgraph of an arbitrary directed graph. We truncated some of the higher order terms to make the problem tractable on large scale Twitter networks. We used network centrality regularizer functions to replace some granularity lost in the objective by truncating those higher order terms. We show in simulation that this approach performs well. Our work supports the power of network centrality measures. When combinatorial optimization in graphs becomes intractable, network centrality measures may serve as useful functions that are cheap to compute. Researchers may, at times, find it viable to discard intractable portions of their math programs and use network centrality regularizers as helpful surrogates.

5.3 Application

Learning how to network with others and spread information has uses from sectors of society ranging from defense to business. People have long understood the skills needed to network and communicate effectively in the offline setting and have recognized the importance of these skills. As the adoption of social media has made networking and spreading information easier, organizations that want to remain competitive must begin using more strategic ways of connecting with others online. If they do not, they risk being drowned out in the ever increasing volume of content that travels around cyberspace. Though the follow-back problem was framed over Twitter, the intuition and modeling can be used to more effectively connect with users in other types of social media networks. Our model is merely preliminary work in a rapidly growing field.
Bibliography


[34] Dustin Volz and Mark Hosenball. White house, silicon valley to hold summit on militants’ social media use. *Reuters*, 2016.