Information Extraction with Network Centralities: Finding Rumor Sources, Measuring Influence, and Learning Community Structure

by

Tauhid R. Zaman

M.Eng., Massachusetts Institute of Technology (2005)
B.S., Massachusetts Institute of Technology (2004)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2011

Author .....
Department of Electrical Engineering and Computer Science
August 17, 2011

Certified by .....
Devavrat Shah
Associate Professor
Thesis Supervisor

Accepted by.....
Leslie A. Kolodziejski
Chair, Department Committee on Graduate Students

Information Extraction with Network Centralities: Finding Rumor Sources, Measuring Influence, and Learning Community Structure

by

Tauhid R. Zaman

Submitted to the Department of Electrical Engineering and Computer Science
on August 17, 2011, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Network centrality is a function that takes a network graph as input and assigns a score to each node. In this thesis, we investigate the potential of network centralities for addressing inference questions arising in the context of large-scale networked data. These questions are particularly challenging because they require algorithms which are extremely fast and simple so as to be scalable, while at the same time they must perform well. It is this tension between scalability and performance that this thesis aims to resolve by using appropriate network centralities. Specifically, we solve three important network inference problems using network centrality: finding rumor sources, measuring influence, and learning community structure.

We develop a new network centrality called rumor centrality to find rumor sources in networks. We give a linear time algorithm for calculating rumor centrality, demonstrating its practicality for large networks. Rumor centrality is proven to be an exact maximum likelihood rumor source estimator for random regular graphs (under an appropriate probabilistic rumor spreading model). For a wide class of networks and rumor spreading models, we prove that it is an accurate estimator. To establish the universality of rumor centrality as a source estimator, we utilize techniques from the classical theory of generalized Polya's urns and branching processes.

Next we use rumor centrality to measure influence in Twitter. We develop an influence score based on rumor centrality which can be calculated in linear time. To justify the use of rumor centrality as the influence score, we use it to develop a new network growth model called topological network growth. We find that this model accurately reproduces two important features observed empirically in Twitter retweet networks: a power-law degree distribution and a superstar node with very high degree. Using these results, we argue that rumor centrality is correctly quantifying the influence of users on Twitter. These scores form the basis of a dynamic influence tracking engine called Trumor which allows one to measure the influence of users in Twitter or more generally in any networked data.

Finally we investigate learning the community structure of a network. Using arguments based on social interactions, we determine that the network centrality known as degree centrality can be used to detect communities. We use this to develop the leader-follower

algorithm (LFA) which can learn the overlapping community structure in networks. The LFA runtime is linear in the network size. It is also non-parametric, in the sense that it can learn both the number and size of communities naturally from the network structure without requiring any input parameters. We prove that it is very robust and learns accurate community structure for a broad class of networks. We find that the LFA does a better job of learning community structure on real social and biological networks than more common algorithms such as spectral clustering.

Thesis Supervisor: Devavrat Shah
Title: Associate Professor

Acknowledgments

I would like to begin by thanking Professor Devavrat Shah for being the best advisor a student could ask for. Working with you has been amazing. I don't think I have ever had so much fun doing research. Your curiosity, intelligence, and energy have been a potent mix that has helped spark this research. I have also learned some very useful skills from you. You have been my theory guru, teaching me the art of formalizing my ideas in concrete mathematical language. I still remember the day when I showed you the proof of the rumor centrality equation and you responded by giving me a high-five. In addition to all the technical skills you have given me, you have also been a very caring advisor who is always concerned for the well being of his students. It is one of your greatest traits and I am lucky to have had your help with everything along the way. I will miss having you as an advisor, but I am very happy that I get to have you as a mentor and friend from now on.

I would not have met Devavrat had it not been for the help of my former advisor Professor Rajeev Ram. Had it not been for your advice and understanding, I am not sure if I would have had the courage to make the transition to a new group and new research field. You provided the guidance I needed to help me navigate a period of great uncertainty in my graduate studies, and I am forever grateful for that.

I would like to thank the members of my thesis committee, Professor Asu Ozdaglar and Professor Alan Willsky, for their insight and ideas for this research. You provided valuable new perspectives and influenced the direction my thesis has taken.

I would also like to thank the NSF EMT, AFOSR Complex Networks and Shell Graduate Fellowship programs for sponsoring this research.

I feel there will be a great void in my life now that I am leaving my fellow group members behind. Srikanth, I will miss our Naruto viewing sessions. Ammar, I will miss our discussions about faith, philosophy, and other random items. Shreeshankar, I will miss our P90X workouts and all of your mathematical jokes and Hindi wisdom. Kyomin, I have missed your positive energy and all the fun we have had these past few years. Jinwoo, I have missed our discussion of the UFC and debate about who is the greatest fighter of all time. Thank you all for the great memories.

I must thank my favorite administrative assistant, the one and only Lynne Dell. You have been a great friend over the years and I will miss our afternoon chats.

Thanks to all of the friends I made in Boston over the years. Loux, Shaheer, Najwa, and Dan, you have all made graduate school a much more fun experience and I am glad I befriended all of you.

I will miss my visits to Nashua to see Tutul Auntie and Mefta Uncle. Thanks to both of you for all of the dinner parties you invited me to over the years. I have grown very attached to you both during my time at MIT and I hope I can still make the trip to Nashua in the future.

I would like to thank my family for their love and support. Thank you to my parents for always believing in me and showering me with your love. Also, thank you to my brother Robbie for being a great roommate and friend these past few years. Living with you was a true joy, especially when you would DVR television shows for me when I came home late from work. Graduate school would not have been as pleasant without you around.

I want to especially thank my grandparents who have been behind me since day one. You have given me more love than anyone could ever ask for and you have been my role models throughout my life. They really don't make people like you anymore. I'm not sure if I can ever measure up to you, but I will try my best. Without you raising me, I would not be where I am today. So Grandma and Grandpa, this thesis is dedicated to you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Today the world has become inextricably intertwined by a host of different networks. In particular, there has been an explosion in online social networks such as Facebook and Twitter which now record what in the past had been informal social interactions. These seemingly innocent network data contain a wealth of information and for various reasons it is desired to extract information from them. Different types of problems emerge in the presence of this vast network data and the extreme size and complexity of these networks makes it difficult to solve them. However, because the data possesses a natural network structure, it is possible to design methods to solve them efficiently. The key to solving these network problems is a class of functions known as *network centralities*. These functions assign scores to nodes in a network based upon their structural properties. For many network problems, use of the proper network centrality can very often lead to an efficient solution. The main focus of this thesis is to show how to use the following network centrality philosophy for solving network problems.

Given the problem and network data, determine the appropriate network centrality and use it to efficiently solve the problem. Ideally, efficient solutions should scale linearly or sub-linearly in order to cope with the scale of the data.

There are many different types of network problems that can be solved with this centrality philosophy and in this thesis we will focus on three specific ones.

1. **Finding the source of a rumor in a network.** Assume a rumor has spread in a

network. Then, with only knowledge about who has the rumor and the network topology, how does one find the source of the rumor?

2. **Measuring influence in a network.** In a social network where people discuss different topics, how can one measure the influence of people specified by topic and time using the structure of the network?

3. **Learning the community structure of a network.** Many times networks possess some underlying community structure, such as different social groups in a social network. How can one use the network to learn this community structure?

An important contribution of this thesis is the systematic study of network centrality in the context of these three important problems as well as empirical demonstration of this philosophy. This is in sharp contrast to the popular literature on network science where the network centralities are proposed based on "common sense". We instead view the problems as an inference problems and find effective answers in the form of network centralities.

We will find that the unifying theme of these problems is the use of network centralities. Before going into further detail about these network problems and network centralities, we will first present a very important example of a network problem that was solved using this network centrality philosophy. The problem is that of web search, specifically, how to return relevant webpages in the Internet that match a user's query, and the solution that emerged is the Google search engine.

1.0.1 Network Centrality Philosophy and Google

A web search engine is typically provided a user query and then returns all webpages that match the query. The challenge is to determine which pages are most relevant to the user. One naive approach to this is to find all pages which match the query and then look through the content of the pages to determine which are most relevant. Many early search engines took this approach, for example simply using the number of times a query appeared in a webpage as an indicator of relevance. However, this approach was prone to manipulation and often produced poor search results. The founders of Google realized that webpages

in the Internet did not exist in isolation, but rather formed a network through their link structure. For example, a site with a large amount of importance may have many other webpages which link to it. The structure of this network could be used to determine the relevance of webpages, but an appropriate function was needed to convert this network structure to a relevance score. In their original paper, the founders of Google created a network centrality called *PageRank* which they used to calculate scores for webpages [53]. Their search engine would then return the webpages matching the user's query, ranked by their PageRank scores. An illustration of the original Google approach to search is shown in Figure 1-1. Similar network structure based approaches for web search were proposed in [40] and [42], with the method in [42] leading to the creation of Baidu, the top search engine in China. We will describe PageRank in greater detail in Section 1.2.3.

Google's approach to search has been incredibly successful, with the site achieving 1 billion unique visitors in May 2011 [27]. The success of Google is a testament to the power of the network centrality philosophy for solving large scale network problems. For Google, using the network structure of webpages allowed for a very efficient way to ascertain the relevance of webpages, as compared with approaches based on webpage content. Another way to view this is that with network centrality, Google was able to extract the relevant knowledge from the network it needed for ranking webpages. The use of network centrality simplified greatly the search problem and led to a very effective, scalable solution. The centrality based philosophy is very powerful for this reason, and we will use it in this thesis to solve other very important network problems, which we describe next.

1.1 Network Problems

We will now provide a brief overview of the different network problems that will be studied in this thesis. This cursory exposition of the problems is meant to provide more motivation for the network centrality philosophy presented in this chapter. Each of these problems will be studied in greater detail in later chapters.

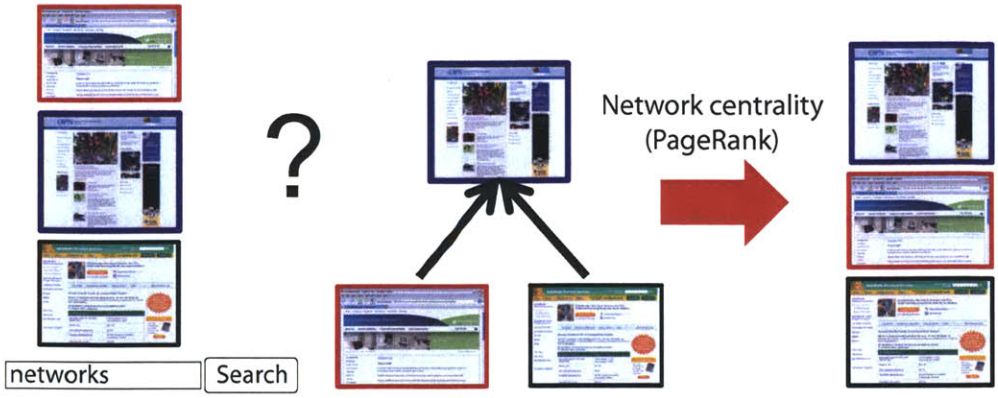

Figure 1-1: The network centrality approach to web search taken by Google. (right) A set of pages which match the user's query are returned, but it is not clear which ones are most relevant for the user. (middle) The webpages link to each other, forming a network structure. (right) Google uses the PageRank network centrality to score the webpages and then ranks the webpages for the user by their PageRank score.

1.1.1 Finding Rumor Sources

Finding the source of a rumor or anything which spreads in a network in an uncontrolled manner is a very important problem. Imagine that a virus spreads through a human contact network and causes an epidemic. Then one would like to find the source of this virus in order to understand what caused the epidemic and possibly quarantine the virus. Similarly, imagine a computer worm spreads through the Internet. Then we would naturally want to find the origin of this cyber attack, or as the attacker find out when it is possible to hide in the network. Finally, imagine that a rumor or a piece of information spreads through a social network. In this case, we would like to find out who is the source of the rumor or information.

In essence, all of these situations can be modeled abstractly as a rumor spreading through a network. The problem of interest is the scenario where the rumor has been spreading for an unknown amount of time in the network and we wish to find the source. All we know is who is infected with the rumor and the underlying network structure. This is depicted in Figure 1-2. With only knowledge of the structure of this "rumor graph", the question is when is it possible to find the rumor source and when is it impossible?

We will see in Chapters 2 and 3 that with just this rumor graph we can indeed find the rumor source with very high accuracy. The solution to this problem involves identifying

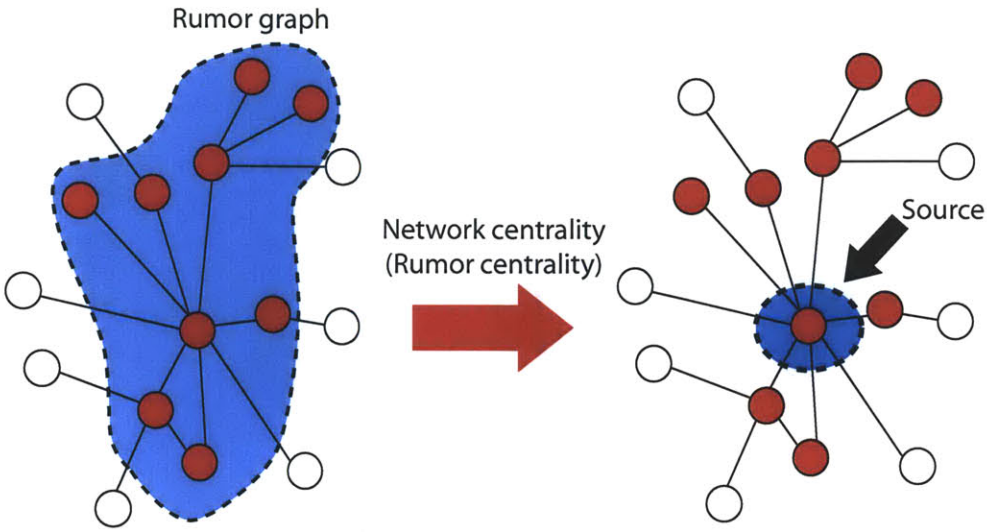

Figure 1-2: (left) A rumor spreads in a network and at some unknown time in the future, we observe the set of rumor infected nodes, or *rumor graph*. From this observation we wish to find the rumor source. (right) The network centrality known as *rumor centrality* will find the most likely source of the rumor in the rumor graph

the right type of structural information for rumor spreading. To capture this structural information, we create a new network centrality called *rumor centrality*. We will see that rumor centrality is the correct centrality for rumor source detection when the problem is viewed as an inference problem. In the network centrality philosophy, we calculate the rumor centrality of each node in the rumor graph and use this centrality score to identify the most likely source, as shown in Figure 1-2.

1.1.2 Measuring Influence

Massive online social networks such as Twitter and Facebook provide the means to possibly measure the influence of different people based on historic communications. In particular, it would be very useful to be able to quantify the influence of people with regard to specific topics. For example, a company which sells chocolate bars may wish to determine in a social network who is influential with regard to the topics of chocolate bars, candy, or sweets. In addition to topic based influence, one may also wish to track influence over different time periods in order to determine who is gaining and losing influence. Therefore, the ultimate goal is to be able to calculate influence with respect to time and topic within a

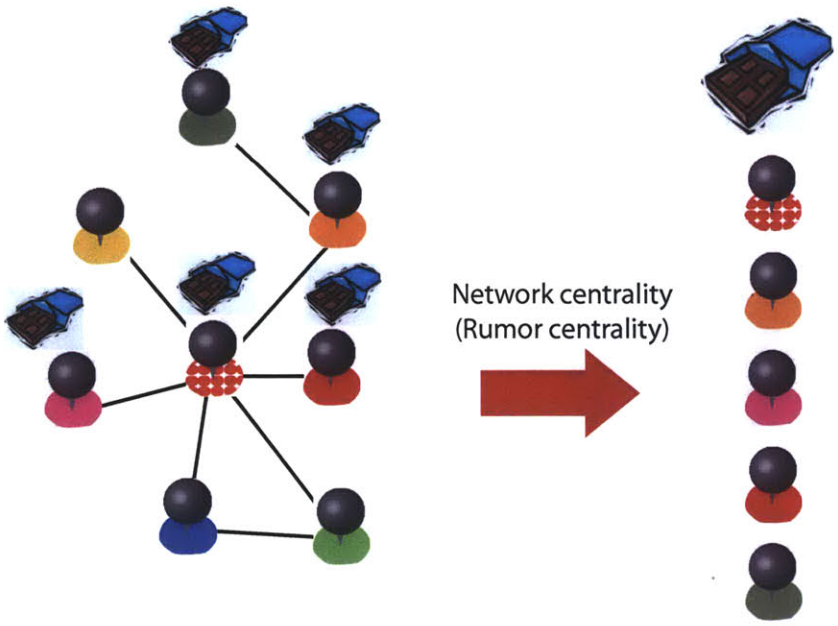

Figure 1-3: An illustration of the Trumor dynamic influence tracking engine for Twitter. (left) The topic of chocolate bar is actively discussed on a social network in a certain time window. (right) An influence score based on *rumor centrality* is used to quantify the influence of each user on the chocolate bar topic in the time window.

social network.

The complexity and scale of online social networks makes it difficult to obtain this type of fine grained influence information. For instance, the micro-blogging site Twitter contains over 200 million users [58] which post messages known as tweets at an average rate of 750 per second, with extreme rates reaching 3,000 per second during certain events [37]. Despite these obstacles, it should be possible to extract useful information due to the network structure inherent in the data. Network centralities are ideal to help make sense of this complex network data. We will find in Chapter 4 that indeed network centralities, in particular *rumor centrality*, can be used to measure influence in social networks such as Twitter. We will use this rumor centrality as the foundation for a dynamic influence tracking engine for Twitter which we call *Trumor*. In the network centrality philosophy, Trumor calculates an influence score based on rumor centrality for each user in the portion of the social network that is actively discussing the user specified topic in the user specified time window and uses this score to quantify the influence of the users. An illustration of this is shown in Figure 1-3.

1.1.3 Learning Community Structure

The structure of many networks is governed by latent communities or clusters. For example, in a social network, people which are part of the same latent community are more likely to be friends and therefore be connected in the network. Very often it is useful to learn these latent communities in order to better understand the structural composition of a network. The challenge is then to figure out how to use the available network data to find these latent communities. Social networks have the added complexity that very often the users belong to multiple communities, so there is considerable overlap in the communities. For example, in Figure 1-4, we show a social network where there are three overlapping communities: people from work, family, and college. This type of overlapping community structure is very common in social networks, so finding the community structure is more complex than simply partitioning the network into disjoint communities. In addition, we assume that we do not know the size of the communities nor the number of communities. This combination of the lack of prior knowledge of the number and size of communities plus the overlapping community structure makes for an extremely challenging problem.

To solve this problem we will use the very simple network centrality measure known as *degree centrality* which is described in Section 1.2.1. Using degree centrality coupled with arguments based on social structure, we develop a very simple algorithm called the *leader-follower algorithm* which can find the overlapping community structure of a network efficiently and accurately. In the network centrality philosophy, roughly what the leader-follower algorithm does is use degree centrality to identify leaders of communities and then assigns followers to leaders to form communities. An illustration of the algorithm applied to an example social network is shown in Figure 1-4.

1.2 Network Centrality

We have seen that the network centrality philosophy is very powerful for solving problems involving large complex networks. The key to this philosophy is a class of functions known as network centralities. In this section we will provide greater details on the different types of network centralities in order to understand the type of information they can provide.

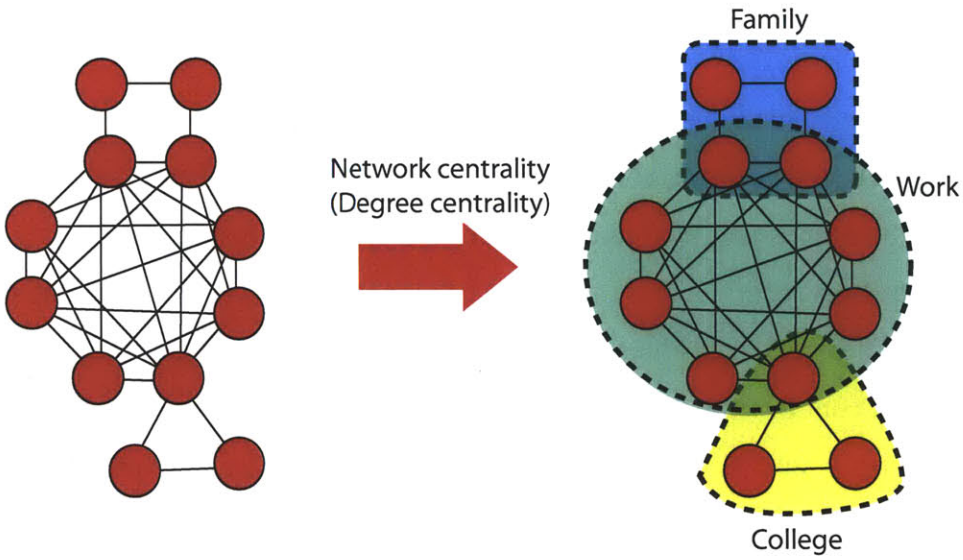

Figure 1-4: (left) A social network with overlapping communities. (right) The leader-follower algorithm uses the network centrality known as *degree centrality* to find this overlapping community structure.

Given a network $G = (V, E)$ with node set V and edgeset $E \subseteq V \times V$, it is natural to assign scores to the nodes in V based upon their structural properties. Depending upon the structural properties considered, nodes may have different scores. Network centralities are functions which measure this score based upon different structural properties. Formally, a network centrality $NC : (V, G) \to \mathbb{R}$ is a function that maps a node v in the graph G to a real number. Putting it another way, network centrality is an embedding of a network in a one-dimensional Euclidean space and the hope is that the embedding brings out the structural properties of interest.

There are two important pieces of information provided by network centrality. The first is the *ranking* of the nodes based upon their centrality score. This can be used to do things such as determine the most important nodes in the network. The second piece of information network centrality provides is a *quantification* of this importance. That is, the actual numeric score of the nodes. It can very often happen that different centralities identify the same set of important nodes in a network, but the amount of importance assigned to each node depends heavily on the choice of centrality. To better understand how different network centralities evaluate the importance of nodes, we will use a simple example network shown in Figure 1-5. In what follows, we will compare how different network centralities

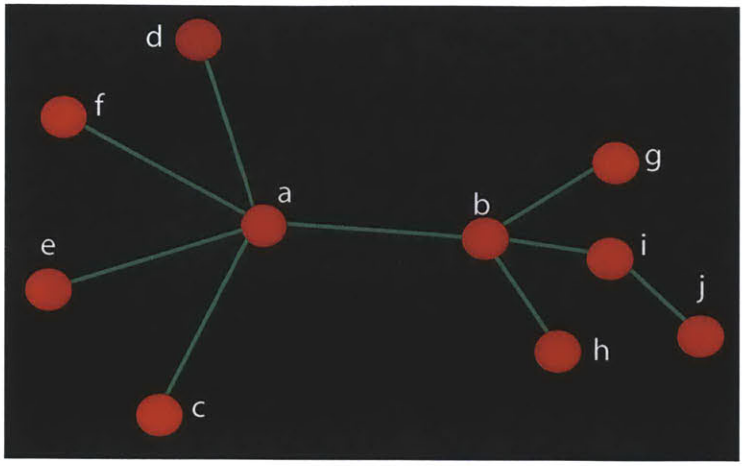

Figure 1-5: Example network for comparing network centralities.

measure the importance of nodes in this network.

1.2.1 Degree Centrality

The simplest structural property of a node in a network is how many neighbors it has. Degree centrality captures this structural property. Formally, for an undirected network $G = (V, E)$, the degree centrality or just degree $\deg(v, G)$ of a node $v \in V$ is given by

$$\deg(v, G) = |\{u \in V : (u, v) \in E\}| \tag{1.1}$$

For simplicity of notation, we will drop the argument referring to the graph G from here on for degree centrality and its variants.

If the network is directed then edges have an associated direction. For instance, in a directed network, for any two nodes $u, v \in V$, the edges (u, v) and (v, u) are distinct edges. In contrast, if the network were undirected, then each edge would just be an unordered pair. Now for directed networks, we can define two different types of degree centrality or degree. First there is in-degree $\deg_{in}(v)$ which is simply the number of edges pointing into a node. Formally,

$$\deg_{in}(v) = |\{u \in V : (u, v) \in E\}| \tag{1.2}$$

Second, we can define the out-degree of a node $\deg_{out}(v)$ as the number of edges pointing out from a node. Formally,

$$\deg_{out}(v) = |\{u \in V : (v, u) \in E\}| \tag{1.3}$$

Our example network is undirected, so we can simply use degree centrality. The values of the degrees of the nodes are shown in Table 1.1. To visualize these values, we redraw the example network in Figure 1-6 with the size of each node proportional to its degree. Notice that nodes d and j have the same degree, but are in distinct structural positions in the network. Node d is connected to a high degree node a, whereas j is connected to node i which is connected to a high degree node b. Therefore, degree centrality only captures the local structural properties of a node and does not provide any information regarding the global structure of the network.

1.2.2 Distance Centrality

Often we may wish to score nodes based upon more global structural information. To do this we can use what is known as distance centrality or closeness centrality [56]. This centrality measure captures how close a node is to all other nodes in the network. For nodes which are very close to most nodes in the network, the distance centrality should be large. If we define the shortest path distance between two nodes $u, v \in V$ on an undirected network G as $d(u, v)$, then the distance centrality of a node v is defined as

$$D(v, G) = \left(\sum_{u \in V} d(v, u) \right)^{-1} \tag{1.4}$$

For simplicity we will assume that the network is connected so that each node can reach all other nodes in the network and all distances are well-defined.

To understand how network structure is translated to scores by distance centrality, we present the values for the example network in Table 1.1. Also, we have redrawn the network in Figure 1-6 with node sizes proportional to their distance centrality. As can be seen, in contrast to degree centrality, the node sizes are much more homogeneous for distance cen-

trality. Thus, we see that even though the central nodes a and b are still the most important, distance centrality assigns them less absolute importance than does degree centrality.

To understand why distance centrality produces scores that are more homogeneous, let us consider nodes a and c in the tree network in Figure 1-6. Because a is the only neighbor of c, a path from c to any other node in the network must pass through a. Therefore, the distance from any node to c will be one hop longer than that node's distance to a, with the exception of node c itself. Let N be the total number of nodes in the network. Then if we look at the ratio of the distance centralities of a and c we obtain

$$\begin{aligned}
\frac{D(a,G)}{D(c,G)} &= \frac{\sum_{u \in V} d(c,u)}{\sum_{u \in V} d(a,u)} \\
&= \frac{\sum_{u \neq c \in V} (d(a,u)+1) + d(a,c) - 1}{\sum_{u \in V} d(a,u)} \\
&= \frac{\sum_{u \in V} d(a,u) + N - 2}{\sum_{u \in V} d(a,u)} \\
&= 1 + \frac{N-2}{\sum_{u \in V} d(a,u)} \\
&\leq 2
\end{aligned}$$

As can be seen, despite the difference in structural position of two neighboring nodes in a tree network, their distance centralities can only differ by at most a factor of two, no matter how many nodes are connected to node a. This is in contrast to the ratio of the degree centralities of these nodes, which will grow as more nodes connect to a.

1.2.3 Eigenvector Centrality

The number of neighbors is what determines a node's degree centrality. However, one can imagine scenarios where the identity of these neighbors also affects a node's importance. For example, a node connected to many unimportant nodes may be less important than a node with a few neighbors of high importance. To capture this type of importance, one can use the network centrality measure known as eigenvector centrality [21]. First, for a node

v we define its set of neighbors $\mathcal{N}(v)$ as

$$\mathcal{N}(v) = \{u \in V \mid (u, v) \in E\} \tag{1.5}$$

For an undirected graph, this is simply the set of neighbors, while for a directed graph this is the set of neighbors which point to v. With this notation, we can define a node's eigenvector centrality $\mathrm{E}(v, G)$ as

$$\mathrm{E}(v, G) = \sum_{u \in \mathcal{N}(v)} \mathrm{E}(u, G) \tag{1.6}$$

If we define $\mathbf{A}$ as the adjacency matrix of the graph where

$$A_{uv} = \left\{ \begin{array}{ll} 1 & \text{if } (u, v) \in E \\ 0 & \text{else} \end{array} \right\} \tag{1.7}$$

then the set of equations for the eigenvector centralities of all nodes in the network can be written in matrix form as

$$\lambda \mathbf{E} = \mathbf{A}^{\mathbf{T}} \mathbf{E} \tag{1.8}$$

where λ is a constant which is needed to ensure the equations have a nontrivial solution. The origin of the term eigenvector centrality is now evident. The solution to this set of equations is an eigenvector of $\mathbf{A}^{\mathbf{T}}$ with λ as the corresponding eigenvalue. For eigenvector centrality one generally uses the eigenvector with the largest eigenvalue.

There are several variations of this simple eigenvector centrality. For example, PageRank, the network centrality originally used in the Google search engine, weighs each neighbor's PageRank by the inverse of its out-degree [53]. The idea is that if a node has high out-degree, it is pointing to many other nodes and hence is less important. The PageRank of a node is expressed as

$$\mathrm{PR}(v, G) = \sum_{u \in \mathcal{N}(v)} \frac{\mathrm{PR}(u, G)}{\deg_{out}(u)} \tag{1.9}$$

Another variation of eigenvector centrality is Bonacich centrality, a parametric centrality measure which has two parameters that control the impact of global and local factors [22]. The Bonacich centrality $\mathrm{BC}(v, G, \alpha, \beta)$ is parameterized by two terms α and β. It can be written as

$$\mathrm{BC}(v, G, \alpha, \beta) = \sum_{u \in \mathcal{N}(v)} (\alpha + \beta \mathrm{BC}(u, G, \alpha, \beta)) \tag{1.10}$$

$$= \alpha \deg_{in}(v) + \beta \sum_{u \in \mathcal{N}(v)} \mathrm{BC}(u, G, \alpha, \beta) \tag{1.11}$$

The α parameter weighs the importance of the node's degree, while the β parameter weighs the importance of the scores of the node's neighbors. If $\beta = 0$ then Bonacich centrality reduces to in-degree. If instead $\alpha = 0$ then it reduces to eigenvector centrality. Thus one can view α as controlling the importance of local structure (degree) while β controls the importance of global structure.

Bonacich centrality emerges naturally as the Nash equilibrium of non-cooperative games with linear-quadratic utilities [12]. The game is played by a set of agents who wish to maximize a utility function by putting forth the appropriate amount of "effort". It turns out that the amount of effort each agent will provide in the Nash equilibrium of the game is proportional to their Bonacich centrality in the network of local complementarities. This game theoretic example is important because it is an instance where a network centrality measure is given a solid theoretical justification.

The basic eigenvector centrality scores for the example network are shown in Table 1.1. The example network is also redrawn in Figure 1-6 with node sizes proportional to their eigenvector centrality. Eigenvector centrality places more importance to higher degree nodes than does distance centrality. In this example network, it assigns importances in a manner similar to degree centrality, with some subtle differences. For example, the

eigenvector centrality of node b is slightly greater than node a, whereas the degree of node b is less than that of a. This is because node b is connected to two non-leaf nodes a and i, whereas a is connected to only one non-leaf node b. Therefore, b has more "important" neighbors than a, and so achieves a slightly larger eigenvector centrality, despite having fewer total neighbors.

1.2.4 Betweenness Centrality

An important structural property of nodes in a network is how many paths between nodes pass through them. That is, how much would the removal of that node affect the connectivity of the network. To capture this structural property, one can use betweenness centrality [32]. This centrality measure counts how many distinct shortest paths pass through a node. Let σ_{st} be the number of shortest paths in G between nodes s and t and let $\sigma_{st}(v)$ be the number of shortest paths in G between nodes s and t that pass through node v. Then the betweenness centrality of v is given by

$$B(v, G) = \sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}} \tag{1.12}$$

A node v which has many shortest paths passing through it will have large values for $\sigma_{st}(v)$ and hence a large betweenness centrality.

There are variations of betweenness centrality which capture different notions of betweenness. If one has a network where edges have weights which indicate a maximum flow that can pass through them, then there exists the notion of flow betweenness centrality [33]. Let f_{st} be the maximum flow between nodes s and t and $f_{st}(v)$ be the amount of the maximum flow between these nodes which also passes through node v. Then the flow betweenness of v is given by

$$FB(v, G) = \sum_{s \neq v \neq t \in V} \frac{f_{st}(v)}{f_{st}} \tag{1.13}$$

It is very similar to betweenness centrality except with the number of paths σ_{st} replaced

Node	Degree cent.	Distance cent.	Eigenvector cent.	Betweenness cent.
a	5	0.071	0.185	26
b	4	0.071	0.197	25
c	1	0.046	0.094	0
d	1	0.046	0.094	0
e	1	0.046	0.094	0
f	1	0.046	0.094	0
g	1	0.046	0.064	0
h	1	0.046	0.064	0
i	2	0.050	0.076	8
j	1	0.036	0.039	0

Table 1.1: Network centralities for example network.

by the maximum flow f_{st}. Flow betweenness centrality is appropriate when the relevant structural information is flows passing through nodes rather than just the number of paths.

The betweenness centrality scores for the example network are shown in Table 1.1. The example network is also redrawn in Figure 1-6 with node sizes proportional to their betweenness centrality. As can be seen, the leaf nodes have betweenness centrality of zero because no paths pass through them. Thus, betweenness centrality gives no importance to leaf nodes at all, in contrast to the other centrality measures examined thus far.

1.2.5 Comparing Network Centralities

The results in Table 1.1 and Figure 1-5 indicate that network centralities will generally identify the same types of nodes as important, but can differ greatly on the absolute importance given each node. For example, the leaf nodes have a substantial importance under distance centrality, but no importance at all under betweenness centrality. For network problems where one wishes to identify nodes which are close to the rest of the network, distance centrality tells us that the leaf nodes are not much worse than the central nodes in this network. However, for network problems where nodes in between many nodes are important, betweenness centrality tells us that that the leaf nodes have no importance at all. Therefore, it is very important for a given network problem to understand the relevant structure. This way, one can select the correct network centrality to obtain a useful solution.

It is important to note that the centrality measures shown here are common sense based

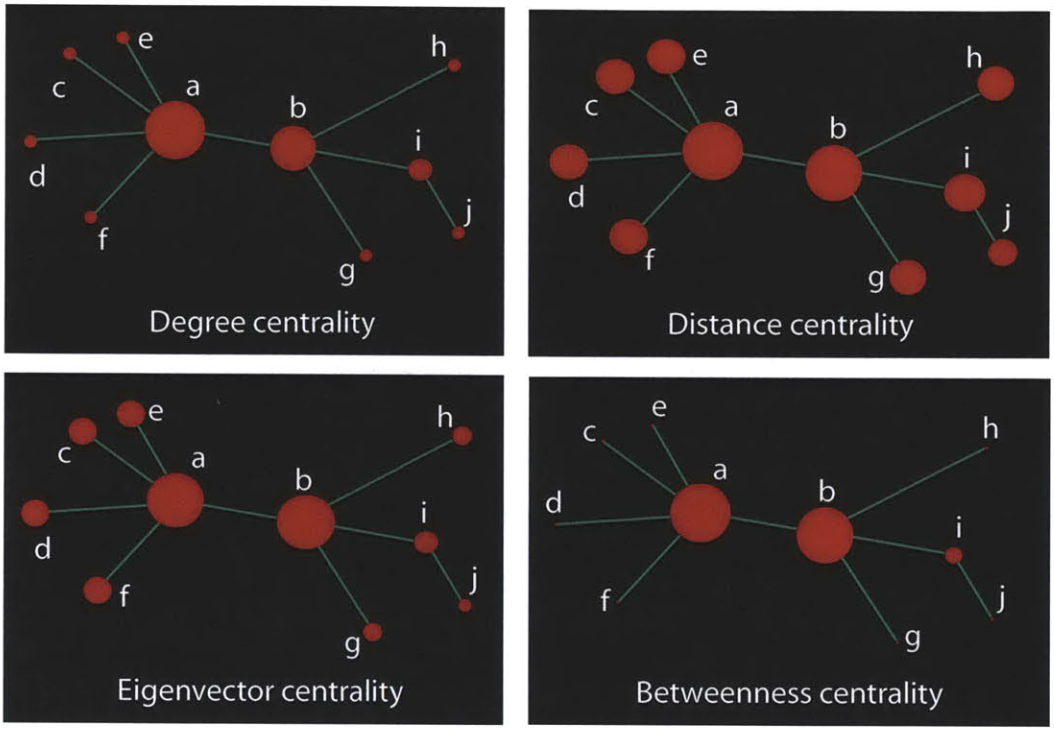

Figure 1-6: Example network with node sizes scaled by corresponding network centrality. (top left) Degree centrality, (top right) distance centrality, (bottom left) eigenvector centrality, and (bottom right) betweenness centrality.

or heuristics. It is not provably established what they are doing, with the exception being Bonacich centrality in certain non-cooperative games. However, network centralities are useful in practice because they are simple enough to lend themselves to efficient, iterative calculations. A theoretical justification for the use of a network centrality is as important as efficient algorithms for calculating it. We provide both of these elements in this thesis. We give theoretical justification for the use of new network centralities in certain problems and also provide efficient algorithms for calculating them.

1.3 Thesis Outline

This thesis provides a systematic study of the use of network centralities for solving network problems by viewing them as inference problems. We give efficient algorithms and also provide provable guarantees and strong empirical validation of our network centrality based solutions. We use network centrality to solve three different network problems: finding rumor sources, measuring influence, and learning community structure.

The network centralities used for solving these problems are determined in different ways. For rumor source detection, the network centrality used is *rumor centrality* and it is obtained from theoretical analysis. For measuring influence, we use *rumor centrality* based upon empirical analysis with real social network data. For learning community structure, the proper network centrality measure is *degree centrality* which is determined using arguments based on social considerations.

1.3.1 Finding Rumor Sources

We begin with finding rumor sources in networks in Chapter 2. This problem has its roots in the field of epidemiology, which has led to many interesting areas of research. We provide an overview for these related problems in Section 2.1. However, the problem of finding the source of a rumor in a network has not been formally investigated. To study the problem formally, we need to define a rumor spreading model. We utilize the popular susceptible infected (SI) model from the literature which was originally used in the context of epidemiology. Here, we use the SI model for rumor spreading. In the model, once a node

has the rumor, it spreads it to each of its neighbors after some random time. These spreading times are arbitrary, but we will assume throughout this thesis that they are independent and identically distributed. We formalize rumor source detection as an inference problem, where the goal is to find the maximum likelihood (ML) estimate of the source under the SI model, given only the rumor infected graph. To evaluate this likelihood, we create a new network centrality called *rumor centrality*. What rumor centrality does is count the number of ways a rumor can spread in a network from a given source and produce the measured rumor graph. The intuition, which we make formal, is that the more ways a node can spread a rumor (the higher its rumor centrality) the more likely it is to be the source. Despite the fact that rumor centrality counts a potentially exponential number of ways to spread a rumor, we provide a very efficient message-passing algorithm for evaluating it. With this algorithm, we can calculate the rumor centrality for every node in a network with a runtime linear in the network size.

Chapter 3 focuses on analyzing the performance of the rumor centrality estimator. We find that for a class of networks known as random regular graphs where every node has the same degree d, rumor centrality is an exact ML estimator under the SI model when the rumor spreading times have an exponential distribution. For more general networks it is not an exact ML estimator, but nonetheless remains a very effective rumor source estimator (which we establish theoretically). We obtain a full theoretical characterization of the error distribution for rumor centrality for random regular graphs. Let us define $C^k_{n(t)}$ as the event that rumor centrality identifies the k^{th} infected node as the most likely source after the rumor has spread for time t and infected $n(t)$ nodes. For $k = 1$, $C^1_{n(t)}$ means the first infected node is identified as the source, which is equivalent to correct source detection. For a random regular graph of degree $d = 2$, which is also known as a linear graph, we obtain the following result for the detection probability.

$$\lim_{t\to\infty} \mathbf{P}\left(C^1_{n(t)}\right) = 0 \tag{1.14}$$

For this simple network structure, rumor centrality, and in fact any estimator, can never find the source as the rumor spreads to more and more nodes. This is due to the fact that the

linear graph's structure is too simple to provide any knowledge that can be used to detect the source. However, if we increase the complexity of the underlying network, a much different result is obtained. In particular, for these random-regular graphs, if the degree is greater than 2, we can always find the source with high accuracy. Specifically, for a random regular graph of degree $d \geq 3$, we obtain the following exact expression for the error distribution of the rumor centrality estimator.

$$\begin{aligned}\lim_{t\to\infty} \mathbf{P}\left(C^k_{n(t)}\right) =& I_{1/2}\left(k-1+\frac{1}{d-2}, 1+\frac{1}{d-2}\right) \\ &+ (d-1)\left(I_{1/2}\left(\frac{1}{d-2}, k+\frac{1}{d-2}\right)-1\right). \end{aligned} \tag{1.15}$$

Here $I_x(a,b)$ is the regularized incomplete Beta function given by the following expression.

$$I_x(a,b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} \int_0^x t^{a-1}(1-t)^{b-1} dt,$$

This result says that the error probability decays exponentially in the distance from the true source, indicating that rumor centrality is a very precise estimator for random regular graphs. For example, for degree $d = 3$, the error distribution simplifies to

$$\lim_{t\to\infty} \mathbf{P}\left(C^k_{n(t)}\right) = k\left(\frac{1}{2}\right)^{(k+1)} \sim \exp\Big(-\Theta(k)\Big).$$

We also obtain interesting results from this analysis for the correct detection probability of rumor centrality on random regular graphs. For a random regular graph of degree 3, we have the following result.

$$\lim_{t\to\infty} \mathbf{P}\left(C^1_{n(t)}\right) = \frac{1}{4} \tag{1.16}$$

This says that the probability that rumor centrality correctly finds the source converges to $1/4$ as the size of the rumor graph goes to infinity. For a random regular graph where the

degree goes to infinity, we have that

$$\lim_{d\to\infty}\lim_{t\to\infty}\mathbf{P}\left(C^1_{n(t)}\right)=1-\ln(2) \tag{1.17}$$

Here, as the rumor graph size and degree grow, the limiting detection probability is $1-\ln(2)$. Our analysis also shows that for any random regular graph with degree $d\geq 3$, the correct detection probability is between $1/4$ and $1-\ln(2)$.

The analysis upon which these results are based utilizes a novel connection between the SI rumor spreading model, Polya urn models, and the theory of branching processes. We are able to use these analysis techniques to provide a theoretical characterization of the performance of the rumor centrality estimator for other networks that are locally tree-like, such as classical Erdos-Renyi graphs. These results hold for the SI model with arbitrary spreading times. Specifically, we find that for a random tree and also for Erdos-Renyi graphs,

$$\liminf_{t\to\infty}\mathbf{P}\left(C^1_{n(t)}\right)>0. \tag{1.18}$$

This result says that for any spreading time distribution, under the SI model, no matter how many nodes are infected in the rumor graph, rumor centrality can always find the source in these networks with non-trivial probability. Essentially, some minimum, strictly positive fixed amount of detection probability always resides at the true source, independent of the number of infected nodes. This is in contrast to linear graphs, where the detection probability goes to zero as more nodes are infected.

Another class of networks which we study are geometric trees. These networks are characterized by polynomial growth. That is, the number of nodes a distance l from a node scales as l^α for some $\alpha\geq 0$. If $\alpha=0$ then this gives a network with a structure very similar to a linear graph. We have already seen that for these networks, detection of the rumor source is not possible. If instead, α is strictly positive, we find that

$$\lim_{t\to\infty}\mathbf{P}\left(C^1_{n(t)}\right)=1. \tag{1.19}$$

That is, for geometric trees which grow faster than a linear graph, we can always find the source with rumor centrality. The proof technique for this result is different than that for random regular graphs. This is because geometric trees have polynomial growth, whereas random regular graphs have exponential growth. Rumor graphs on networks with polynomial growth have much lower variance than on networks with exponential growth. The lower variance of geometric trees allows us to use concentration based estimations to establish our results.

For a more striking visual demonstration of the precision of rumor centrality for source detection, we present an example rumor graph on a grid network in Figure 1-7. The colored circles represent the rumor infected nodes, with the rumor spread via the SI model with exponential spreading times of rate 1. The color bar indicates the normalized rumor centralities of the nodes in the rumor graph, with red nodes having a very high value. As can be seen, rumor centrality singles out a very small set of nodes from this large rumor graph as likely sources. In this instance, it happens that the most likely source as identified by rumor centrality turns out to be the actual source. This example shows just how effective rumor centrality can be at finding sources of rumors in large networks.

In summary, on a linear graph it is not possible to detect the rumor source, but for anything growing faster than a line detection is possible and this detection is accomplished by means of the efficient rumor centrality. Therefore, if one wished to spread information without being caught, one must do so on a "line".

1.3.2 Measuring Influence

In Chapter 4 we study measuring influence in networks. In particular we look at measuring influence in the micro-blogging site Twitter. There have been many heuristic based approaches used to measure influence in Twitter, which we discuss in Section 4.1.2. However, there is no strong empirical or theoretical justification for these approaches. We approach this problem with the hypothesis that influence can be measured using the appropriate network centrality measure. The challenge is then to determine which centrality measure to use. We take a data driven approach here by looking the the structure of networks in

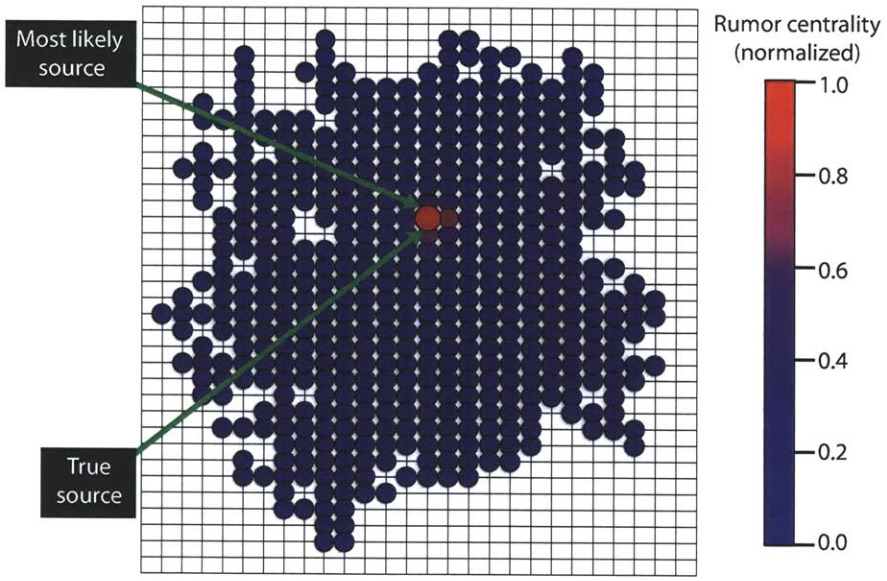

Figure 1-7: A rumor graph on a grid network. The color bar indicates the normalized rumor centralities of the nodes in the rumor graph, with red nodes having a very high value. The most likely source as identified by rumor centrality, and the true source coincide and are indicated on the figure.

Twitter. We then develop a model based on network centrality to describe these networks. Rumor centrality is identified as the proper centrality measure because it models the data most accurately under the proposed model. We then use rumor centrality as the basis for a dynamic influence tracking engine for Twitter we call Trumor.

We begin by studying the empirical properties of dynamic topic networks on Twitter. These are networks of user interaction regarding a specific topic during a specific time window. An example of one of these networks for the topic "BET Awards" is shown in Figure 1-8. We observe two unique properties for all of these networks. First, their degree distribution shows a power-law behavior. This means that the degree distribution decays polynomially fast in the degree. Second, they each have one very high degree node which we refer to as a *superstar*. For this BET Awards network, the superstar is the pop singer known as Lady Gaga, whose image is shown in the network in Figure 1-8, along with other notable high degree nodes. Both of these structural properties are labeled in the degree histogram for the BET Awards network in Figure 1-8. The superstar is particularly interesting because it is a non-local effect and cannot be explained using conventional local models of network growth such as Erdos-Renyi graphs or the so called

preferential attachment model [14]. Rather, a model is needed which can incorporate the global structure of the network.

We propose a new network growth model known as *topological network growth*. This model allows one to grow a random network using an arbitrary *influence measure*. Roughly speaking, this model has new nodes connect to nodes already in the network randomly, with a preference for nodes with a higher influence. We study the effect of different network centralities as influence measures on the resulting network structure. The different ways in which network centralities quantify influence are manifested in the structure of these networks. For example, we find that distance centrality produces networks whose degree distribution decays exponentially fast. That is, these networks do not have many high degree nodes. When degree centrality is used as the importance measure, the model reduces to the aforementioned preferential attachment model and produces networks with a degree distribution which follows a power-law. Here it is more likely to find some nodes of relatively high degree. We find that networks grown using rumor centrality have both of the properties measured in the Twitter networks: power-law degree distribution and superstars. The networks formed using rumor centrality have degree distributions which are very similar to those measured in the Twitter networks for many different topics. An example is shown in Figure 1-8, where the rumor centrality network degree distribution matches that of the Twitter data very closely. Because rumor centrality matches the data so well, we provide a more detailed analysis of the network structure. We are able to characterize the network structure that emerges when rumor centrality is used as an influence measure. In particular, we are able to show how the properties of rumor centrality lead naturally to the observed power-law and superstar node.

We utilize these results to develop an application that measures influence in networks. The topological network growth model with rumor centrality does a very effective job of describing the Twitter data, so we use it as the basis for a dynamic influence tracking engine for Twitter called Trumor. Trumor constructs a Twitter network specific to a user specified topic and time window. It then calculates an influence score based on rumor centrality for the people in this network. In this way it is able to quantify the topic specific influence of people on Twitter. This dynamic, topic dependent influence score can then be used by other

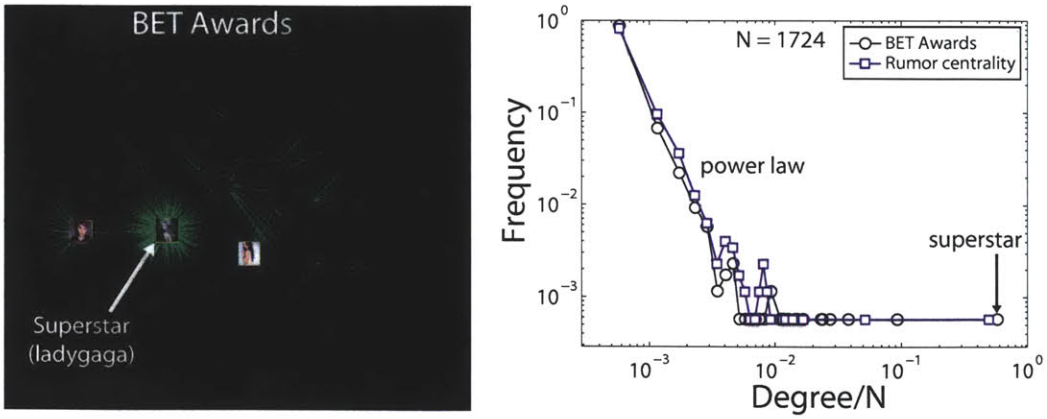

Figure 1-8: (left) The BET Awards network from Twitter. The superstar node (ladygaga) is marked by an arrow. (right) The degree distribution of the BET Awards network which has $N = 1724$ nodes. Also shown is the degree distribution of a synthetic network the same size as the BET Awards network which was created using the topological network growth model with rumor centrality as an influence measure.

applications which wish to determine the influence of users in Twitter on certain topics.

1.3.3 Learning Community Structure

In Chapter 5 we look at learning community structure in networks. The scenario we envision is the following. We are given a network and must determine all communities within it with no knowledge of the number or size of these communities. In addition, we add the complication that the communities are allowed to overlap. We have seen in Section 1.1.3 that in social networks very often people belong to multiple communities, so any community detection method must be able to handle overlapping communities.

Community detection is a very well-known problem with a rich history. We review many of the existing techniques for community detection in Section 5.1. Our aim here is to learn overlapping communities without any knowledge of the size and number of communities and also provide theoretical guarantees for the performance of our algorithm. The existing techniques for community detection do not achieve all of these goals. In our algorithm, we take a non-parametric, data-driven approach to community detection. That is, we seek a solution which can learn all the information regarding the communities from the network structure alone, without assuming any sort of parametric model. Through argu-

ments based on social interactions, we find that the relevant network centrality measure for this problem is *degree centrality*. The argument we use to justify degree centrality is the following. In a community there are nodes which connect the community to the rest of the network, which we call *leaders*, and there are nodes with no neighbors outside of the community, which we call *followers*. The followers provide the community with a distinguishing identity, and leaders keep the network connected. If we assume that everyone in a community knows each other, then the community will be a clique. Under this scenario, within any community leaders will necessarily have a higher degree than followers, and so degree centrality can be used to distinguish them.

With this argument, we use degree centrality in a novel way to develop the *leader-follower algorithm* (LFA) for learning community structure. This algorithm is completely non-parametric, requiring no input parameters. It can find overlapping communities with no constraints on the number or size of communities found. The LFA is able to learn the number of communities naturally from the network structure. Also, the LFA is very fast, having a runtime linear in the network size. We find that it can find the communities in a network of 150,000 nodes in a few minutes on a standard PC with a Pentium Core Duo 1.8 GHz processor.

We also provide theoretical guarantees for the performance of the LFA. We prove that it can accurately find communities in a social setting where the following properties are likely to hold: communities are cliques because everyone knows each other and each community has at least one follower. This result shows that the LFA is quite effective for a broad class of networks. Also, we find that this structural property of communities is found in real network data. Using data from social and biological networks, we show that the LFA is very effective at learning community structure in real networks.

To demonstrate the effectiveness of the LFA, we show in Figure 1-9 an example of its application to a real social network. This network consists of performers in a college culture show. Each node is a performer and there is an edge between any performers which are in the same performance. The communities in this network are the performances. Because many performers were in multiple performances, there is considerable overlap in the community structure. Despite this, the LFA is able to find all of the communities in

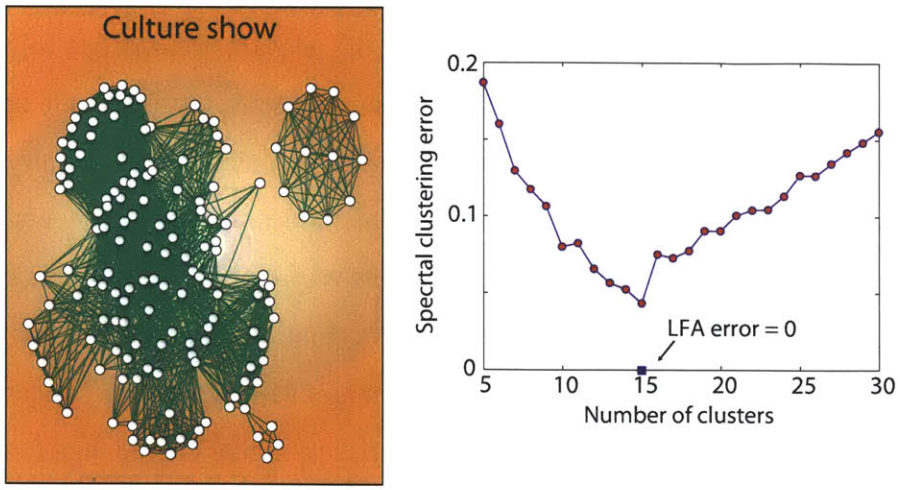

Figure 1-9: (left) Network of performers in a college culture show. (right) Plot of the error of the communities found with spectral clustering for this culture show network as a function of the number of input clusters. Also shown is the error for the LFA, which is zero for this network.

this network exactly. For comparison, also shown is the clustering error that arises when one tries to find the communities using a more conventional technique known as spectral clustering, which requires the number of communities as an input parameter. The clustering error measures the average fraction of misclassified nodes per community. As can be seen, the LFA outperforms this parametric algorithm on this real social network.

Chapter 2

Finding Rumor Sources

In this chapter we will study in depth the problem of finding the source of a rumor in a network. In addition to being a very practical and important problem, the challenge and novelty of finding the rumor source in a network also makes it a very interesting theoretical pursuit. Essentially, one needs to determine the origin of a spreading process based upon the final structure of the rumor infected network. There are two important questions to ask here. First, how does one actually construct the rumor source estimator? The estimator would naturally need to incorporate the topology of the underlying network, but it is not obvious in what manner. Second, what are the fundamental limits to this rumor source detection problem? In particular, how well can one find the rumor source, what is the distribution of the error, and how does the network structure affect one's ability to find the rumor source?

Rumor source detection has remained a relatively unexplored problem mainly because of its inherent difficulty. The challenge arise because one must search for the rumor source within the entire network using only the network structure. For very complex networks, it is akin to finding a needle in a stack of needles. Despite the challenging nature of the problem, in this chapter we will show that it is indeed possible to find rumor sources in networks. We will create a network centrality measure for this problem which we call *rumor centrality*. This centrality measure is an exact maximum likelihood (ML) estimator for certain classes of networks. We naturally extend rumor centrality to make it an estimator for rumor sources in general networks. Also, we provide a precise theoretical characterization

of the performance of the rumor centrality based estimator for networks which are "locally tree-like".

2.1 Previous Work

2.1.1 1854 London Cholera Epidemic

One of the earliest and most well known examples of source detection is the 1854 cholera epidemic in London. Over 616 people were killed by cholera during the epidemic, which ravaged London's Soho district that year [39]. The prevailing theory at the time was that cholera was spread by contaminated air or pollution. A physician named Dr. John Snow was skeptical of this theory took it upon himself to investigate the cause of the epidemic. Dr. Snow's investigations led him to suspect that it was contaminated water which was the cause of the cholera epidemic. His interviews with local residents led him to suspect a pump located at Broad Street (now Broadwick Street) as the source of the cholera. His examination of the water at the pump did not conclusively show that it was contaminated. However, his findings on the disease spreading pattern were strong enough to convince the local council to shut down the pump. One very important thing Dr. Snow did was gather data on the location of deaths that were due to cholera and plot these on a map of London, which is shown in Figure 2-1 [59].

Later it was found that this Broad St. pump was indeed the source of the cholera. What had happened was that a cesspool had been dug within close proximity of the pump. The diaper of a baby which had contracted cholera from some external source was washed in this cesspool, with the resulting contaminated waste leaking into the water pump, thus starting the cholera epidemic in London. On Figure 2-1 the location of the Broad St. pump is labeled as the cholera source.

Now, without any formal epidemiological techniques, we can use Dr. Snow's data on cholera deaths to accurately estimate the cholera source. We treat the death data points as particles of unit mass, and then calculate their center of mass. It turns out that this center of mass is located at the same Broad St. water pump that was the cholera source. Therefore,

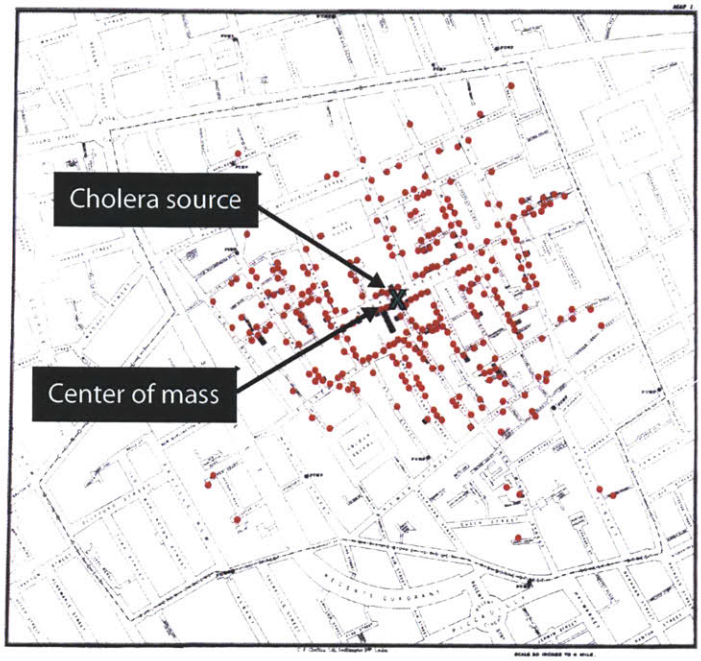

Figure 2-1: Map of London with locations of deaths during the 1854 cholera epidemic as found by Dr. John Snow marked with red dots. The cholera source and center of mass are indicated on the figure. It so happens that they are the same point on the map.

we see here that the center of this cholera map is the cholera source. Now this map is a very simple geometric space (a plane) with no complex structure. However, the networks which we are interested in do not have such a simple topology, so concepts such as center of mass will not directly translate. Despite this, the intuition of the center being the source is powerful, and we will use it in this chapter to find rumor sources. In particular, we will show that the correct notion of a center for rumor source detection is the aforementioned rumor centrality.

2.1.2 Epidemic Models

The classical model for the spread of epidemics on a network is known as the susceptible-infected-recovered (SIR) model [9]. In this model nodes are either susceptible to infection, infected, or recovered from the infection. Much of the work on this model has focused on understanding under what conditions an epidemic can infect the entire network. This work studies the relation between the structural properties of the network and the rate of spread of the disease. One major contribution of this area of work is to establish what

the epidemic threshold is, that is, what is the minimum rate of infection that will give a global epidemic. Epidemic thresholds have been established for many well-known network structures, including small-world networks [43] and scale-free networks [55]. In [34] the authors were able to express the epidemic threshold in terms algebraic properties of the network such as the spectral radius. A similar algebraic approach was used in [44] to bound the time it takes a rumor to spread to a fixed number of nodes in a network.

While there has been a substantial amount of work on understanding epidemic thresholds and rates of rumor spreading, there has been little previous work on finding the source of a rumor in a network. It seems that this question is rather new in general. However, we will see next that similar types of problems have been investigated in different areas. Indeed, the question of what sort of information is preserved under network dynamics is an important intellectual pursuit in the context of a few specific problems such as the reconstruction of the source value [30], [45], [35] or network structure motivated by phylogeny [29], [28].

2.1.3 Reconstruction of Source Value

There has been considerable work in a problem related to rumor source detection known as the source value reconstruction problem. Here a known source node transmits a piece of information under the presence of noise in a network. The goal is then to use the noisy value of the information at the nodes in the network to reconstruct the original value at the source. The connection between the network structure and the maximum amount of noise under which reconstruction is possible is well understood on trees and tree-like graphs [30],[45],[35].

Rumor source detection can be viewed as a much more challenging problem than reconstruction. In the reconstruction problem one knows the source and also the values of the noisy information at all node. In the rumor source detection problem, one only knows which nodes are infected with no knowledge of the source. For reconstruction, one only needs to determine the value at the known source using noisy measurements in the network. In contrast, for rumor source detection, one must determine the identity of the source from

a set of nodes with no obvious distinguishing characteristics. It is akin to trying find a needle in a stack of needles.

2.1.4 Reconstruction of Network Structure

Another line of research has studied using rumor spreading data to reconstruct network structure. The basic idea is that one measures the sequence in which nodes are infected with the rumor for many different rumor sources in the network. These rumor spreading events are referred to as cascades. Then, using the causal information embedded in these cascades, one attempts to reconstruct the edges of the network. This approach has been pursued in [36] and [47]. In [47] the number of cascades needed is on the order of the number of nodes in the network. With this large number of cascades, the methods show reasonable accuracy at recovering the network structure.

Another area of network reconstruction is motivated by phylogeny. In this scenario, one is given the DNA sequence of several species. The goal is then to reconstruct the phylogeny of these species. That is, reconstruct a tree where the observed species are leaves and any branching nodes represent speciation events. There has been much theoretical development is this field. In [29] a reconstruction algorithm is proposed known as the Short Quartet Method (SQM) which is proven to return the correct phylogeny for polynomial length sequences in polynomial time with high probability under certain assumptions on the phylogeny structure. These assumptions included bounds on the tree depth or branch lengths, but allowed for complete reconstruction. Later in [28] an algorithm was provided which gave efficient and accurate reconstruction of the partial phylogeny without any of these restricting assumptions. In the case where the assumptions held, the algorithm of [28] was able to reconstruct the full phylogeny.

In contrast to this type of inference problem, the rumor source detection problem we consider here involves only a single shot measurement. We only have one data point which is the rumor graph. From this we must determine who the source is. Rumor source detection can be considered as the complement of the network inference problem. For network inference, we know the infection sequence (and hence the rumor source) for a large num-

ber of cascades, but do not know the network structure. For rumor source detection we do not know the infection sequence and only have one cascade, but we do know the network structure.

2.2 Rumor Source Estimator

We will now construct estimators for rumor sources. We begin by presenting the rumor spreading model we will assume throughout this chapter in Section 2.2.1. Then we will formulate rumor source detection in precise probabilistic language and construct estimators in in the remainder of this section.

2.2.1 Rumor Spreading Model

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a possibly infinite connected graph. Let $v \in \mathcal{V}$ be a rumor source from which a rumor starts spreading at some time as per the classical Susceptible Infected (SI) model: for each edge $e = (u_1, u_2)$, if either u_1 or u_2 become infected, then the remaining uninfected node on the edge will be infected after a random time with a well-defined distribution. Let $F(t)$ denote the cumulative distribution function (CDF) of this random time which by definition has support on R_+: i.e. $F(0) = 0$ and $F(t) \to 1$ as $t \to \infty$. The spreading times for each edge are independent and identically distributed (i.i.d.). We will assume that $F(t)$ is an exponential distribution with rate 1 throughout this section (i.e. $F(t) = 1 - e^{-t}$) as we construct the rumor source estimator. We will refer to this as the SI model with exponential distribution. Later, in Section 3.1 we will study the effectiveness of the rumor source estimator developed here (for exponential spreading times) when the spreading times have an arbitrary distribution.

2.2.2 Rumor Source Estimator: Maximum Likelihood (ML)

Assume we observe the set of infected nodes, say $V \subset \mathcal{V}$, with $|V| = N$ (N nodes infected), after some time (which we do not know). Given this N node rumor infected graph (or simply rumor graph) $G = (V, E)$ with $E = \mathcal{E} \cap V \times V$, we wish to infer the source

and ideally identify v as the source. A priori, we do not have any information and hence all nodes in V are equally likely to be the source (uniform prior).

With respect to this setup, the maximum likelihood (ML) estimator of the rumor source with respect to the SI model given G minimizes the error probability, i.e. maximizes the correct detection probability. By definition, the ML estimator is

$$\widehat{v} \in \arg\max_{v \in G} \mathbf{P}(G|v), \tag{2.1}$$

where $\mathbf{P}(G|v)$ is the probability of observing G under the SI model assuming v is the source. Thus, ideally we would like to evaluate $\mathbf{P}(G|v)$ for all $v \in \mathcal{V}$ and then select the one with the maximal value (ties broken uniformly at random).

2.2.3 Rumor Source Estimator: ML for Regular Trees

In general, evaluation of $\mathbf{P}(G|v)$ may not be computationally tractable. Here we shall show that for regular trees, $\mathbf{P}(G|v)$ becomes proportional to a quantity $R(v, G)$ which we define later and call rumor centrality. The $R(v, G)$ is a topological quantity and is intimately related to the structure of G.

Now to evaluate $\mathbf{P}(G|v)$ when the underlying graph is a tree, essentially we wish to find the probability of all possible events that result in G after N nodes are infected starting with v as the source under the SI model with exponential distribution. To understand such events, let us consider a simple example as shown in Figure 2-2 with $N = 4$. Now, suppose node 1 was the source, i.e. we wish to calculate $\mathbf{P}(G|1)$. Then there are two disjoint events or node orders in which the rumor spreads that will lead to G with 1 as the source: $\{1, 2, 3, 4\}$ and $\{1, 2, 4, 3\}$. However, due to the structure of the network, infection order $\{1, 3, 2, 4\}$ is not possible because 2 cannot be infected before 3 when the source is 1. Therefore, in general to evaluate $\mathbf{P}(G|v)$, we need to find all such *permitted permutations* and their corresponding probabilities. Because permitted permutations are an important concept in this section, we now define this notion precisely.

Definition 1 (Permitted permutation)**.** *Given a connected tree $G = (V, E)$ and a source node $v \in V$, consider any permutation $\sigma : V \to \{1, 2, ..., |V|\}$ of its nodes where $\sigma(u)$*

denotes the position of node $u \in V$ in the permutation σ. We call σ a permitted permutation for tree G with source node v if

1. $\sigma(v) = 1$.
2. *For any $(u, u') \in E$, if $d(v, u) < d(v, u')$, then $\sigma(u) < \sigma(u')$. Here $d(v, u)$ denotes the shortest path distance from v to u.*

Let $\Omega(v, G)$ be the set of all permitted permutations starting with node v and resulting in rumor graph G. We wish to determine the probability $\mathbf{P}(\sigma|v)$ for each $\sigma \in \Omega(v, G)$. This is the probability of seeing σ as the node order given starting node v. To that end, let $\sigma = \{v_1 = v, v_2, \ldots, v_N\}$. Let us define, $G_k(\sigma)$ as the subgraph (of G) containing nodes $\{v_1 = v, v_2, \ldots, v_k\}$ for $1 \leq k \leq N$. Then,

$$\mathbf{P}\big(\sigma \mid v\big) = \prod_{k=2}^{N} \mathbf{P}\big(k^{\text{th}} \text{ infected node} = v_k \mid G_{k-1}(\sigma), v\big). \tag{2.2}$$

Each term in the product on the right hand side in (2.2), can be evaluated as follows. Given $G_{k-1}(\sigma)$ (and source v), the next infected node could be any of the neighbors of nodes in $G_{k-1}(\sigma)$ which are not yet infected. For example, in Figure 2-2 G_2 is $\{1, 2\}$ when the source is assumed to be 1. In that case, the next infected node could be any one of the 4 nodes: 3, 4, 5 and 6. Now due to the memoryless property of exponential random variables and since the infection times on all edges are independent and identically distributed (i.i.d.), it follows that each of these nodes is equally likely to be the next infected node. Therefore, each one of them has probability $1/4$. More generally, if $G_{k-1}(\sigma)$ has $n_{k-1}(\sigma)$ uninfected neighboring nodes, then each one of them is equally likely to be the next infected node with probability $1/n_{k-1}(\sigma)$. Therefore, (2.2) reduces to

$$\mathbf{P}\big(\sigma \mid v\big) = \prod_{k=2}^{N} \frac{1}{n_{k-1}(\sigma)}. \tag{2.3}$$

Given (2.3), now the problem of computing $\mathbf{P}\big(\sigma \mid v\big)$ boils down to evaluating the size of the *rumor boundary* $n_{k-1}(\sigma)$ for $2 \leq k \leq N$. To that end, suppose the k^{th} added node to $G_{k-1}(\sigma)$ is $v_k(\sigma)$ with degree $d_k(\sigma)$. Then it contributes $d_k(\sigma) - 2$ new edges (and hence

nodes in the tree) to the rumor boundary. This is because, $d_k(\sigma)$ new edges are added but we must remove the edge along which the recent infection happened, which is counted twice. That is, $n_k(\sigma) = n_{k-1}(\sigma) + d_k(\sigma) - 2$. Subsequently,

$$n_k(\sigma) = d_1(\sigma) + \sum_{i=2}^{k}(d_i(\sigma) - 2). \tag{2.4}$$

Therefore,

$$\mathbf{P}(\sigma \mid v) = \prod_{k=2}^{N} \frac{1}{d_1(\sigma) + \sum_{i=2}^{k}(d_i(\sigma) - 2)}. \tag{2.5}$$

For a d regular tree, since all nodes have the same degree d, it follows from (2.5) that every permitted permutation σ has the same probability, independent of the source. Specifically, for any source v and permitted permutation σ

$$\begin{aligned}\mathbf{P}(\sigma \mid v) &= \prod_{k=1}^{N-1} \frac{1}{dk - 2(k-1)} \\ &\equiv p(d, N).\end{aligned}$$

From above, it follows immediately that for a d regular tree, for any G and candidate source v, $\mathbf{P}(G \mid v)$ is proportional to $|\Omega(v, G)|$. Formally, we shall denote the number of distinct permitted permutations $|\Omega(v, G)|$ by $R(v, G)$, which we call the *rumor centrality* of v.

Definition 2 (Rumor Centrality, Rumor Center). *Given a graph G and vertex v of G, we define the **rumor centrality** of v, $R(v, G)$ as the total number of distinct permitted permutations of nodes of G that begin with node $v \in G$ and respect the graph structure of G. The node v with the maximal value for $R(v, G)$ is called the **rumor center**.*

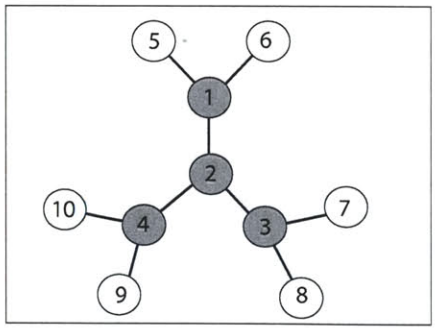

Figure 2-2: Example network where the rumor graph has four nodes.

In summary, the ML estimator for a regular tree becomes

$$\begin{aligned}
\widehat{v} &\in \arg\max_{v \in G} \mathbf{P}\big(G \mid v\big) \\
&= \arg\max_{v \in G} \sum_{\sigma \in \Omega(v, G_N)} \mathbf{P}\big(\sigma \mid v\big) \\
&= \arg\max_{v \in G} R(v, G) p(d, N) \\
&= \arg\max_{v \in G} R(v, G)
\end{aligned} \tag{2.6}$$

with ties broken uniformly at random.

2.2.4 Rumor Source Estimator: General Trees

As (2.6) suggests, the ML estimator for a regular tree can be obtained by simply evaluating $R(v, G)$ for all v. However, as indicated by (2.5), such is not the case for a general tree with heterogeneous degree. This is because in the regular tree, all permitted permutations were equally likely, whereas in a general tree, different permitted permutations have different probabilities. To form an ML estimator for a general tree we would need to keep track of the probability of every permitted permutation. This could be computationally quite expensive because of the exponential number of terms involved. Therefore, we construct a simple heuristic to take into account the degree heterogeneity.

Our heuristic is based upon the following simple idea. The likelihood of a node is a sum of the probability of every permitted permutation for which it is the source. In general, these will have different values, but it may be that a majority of them have a common value. We

then need to determine this value of the probability of the common permitted permutations. To do this, we assume the nodes receive the rumor in a breadth-first search (BFS) fashion. Roughly speaking, this corresponds to the fastest or most probable spreading of the rumor. To calculate the BFS permitted permutation probability, we construct a sequence of nodes in a BFS fashion, with the source node fixed. For example, consider the network in Figure 2-3. If we let node 2 be the source, then a BFS sequence of nodes would be $2, 1, 3, 4, 5$ and the probability of this permitted permutation is given by (2.5). If we define the BFS permitted permutation with node v as the source as σ_v^*, then the rumor source estimator becomes (ties broken uniformly at random)

$$\widehat{v} \in \arg\max_{v \in G} \mathbf{P}\left(\sigma_v^* \mid v\right) R(v, G). \tag{2.7}$$

We now consider an example to show the effect of the BFS heuristic. For the network in Figure 2-3, the corresponding estimator value for node 1 is

$$\begin{aligned} \mathbf{P}\left(\sigma_1^* \mid v\right) R(1, G) &= \left(\frac{1}{4}\right)^4 4! \\ &= 6\left(\frac{1}{4}\right)^3 \end{aligned}$$

and for node 2 it is

$$\begin{aligned} \mathbf{P}\left(\sigma_2^* \mid v\right) R(2, G) &= \frac{1}{2}\left(\frac{1}{4}\right)^3 3! \\ &= 3\left(\frac{1}{4}\right)^3. \end{aligned}$$

For comparison, the exact likelihood of node 1 is

$$\begin{aligned}\mathbf{P}(G|1) =& \mathbf{P}(\{1,2,3,4,5\}|1) + \mathbf{P}(\{1,2,3,5,4\}|1) \\ &+ \mathbf{P}(\{1,2,4,3,5\}|1) + \cdots + \mathbf{P}(\{1,5,4,3,2\}|1) \\ =& 24\left(\frac{1}{4}\right)^4 \\ =& 6\left(\frac{1}{4}\right)^3\end{aligned}$$

and for node 2 it is

$$\begin{aligned}\mathbf{P}(G|2) =& \mathbf{P}(\{2,1,3,4,5\}|2) + \mathbf{P}(\{2,1,3,5,4\}|2) \\ &+ \mathbf{P}(\{2,1,4,3,5\}|2) + \mathbf{P}(\{2,1,4,5,3\}|2) \\ &+ \mathbf{P}(\{2,1,5,3,4\}|2) + \mathbf{P}(\{2,1,5,4,3\}|2) \\ =& 6\frac{1}{2}\left(\frac{1}{4}\right)^3 \\ =& 3\left(\frac{1}{4}\right)^3.\end{aligned}$$

In this case we find that the BFS heuristic equals the true likelihood for both nodes. Second, node 1 is only twice as likely as node 2 to be the source. However, if we look at the ratio of the number of permitted permutations of the nodes we find

$$\begin{aligned}\frac{R(1,G)}{R(2,G)} &= \frac{4!}{3!} \\ &= 4.\end{aligned}$$

Thus, the number of permitted permutations of node 1 is four times as large as that of node 2. What is happening is that without the BFS heuristic, we are being fooled to always select higher degree nodes because we assumes all nodes have the same degree. Therefore, if a node only has a few infected neighbors (such as node 2), rumor centrality assumes that the node was not immediately infected and consequently did not have time to infect its neighbors. However, the BFS heuristic tries to compensate for the tendency of rumor

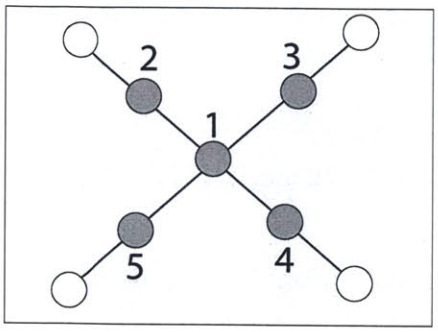

Figure 2-3: Example network where rumor centrality with the BFS heuristic equals the likelihood $\mathbf{P}(G|v)$. The rumor infected nodes are in gray and labeled with numbers.

centrality to favor higher degree nodes.

Indeed, as we shall see in Section 3.2, this heuristic is an improvement over the naive extension of the estimator (2.6) for networks with very heterogeneous degree distributions. That is, biasing as per $\mathbf{P}\left(\sigma_v^* \mid v\right)$ in (2.7) is better than the unbiased version of it.

2.2.5 Rumor Source Estimator: General Graphs

The ML estimator for a general graph, in principle can be computed by following a similar approach as that for general trees. Specifically, it corresponds to computing the summation of the likelihoods of all possible *permitted permutations* given the network structure. This could be computationally prohibitive. Therefore, we propose a simple heuristic.

To that end, note that even in a general graph the rumor spreads along a spanning tree of the observed graph corresponding to the *first time* each node receives the rumor. Therefore, a reasonable approximation for computing the likelihood $\mathbf{P}(G|v)$ is as follows. First, suppose we know which spanning tree was involved in the rumor spreading. Then, using this spanning tree, we could apply the previously developed tree estimator. However, it is the lack of knowledge of the spanning tree that makes the rumor source estimation problem complicated.

We circumvent the issue of not knowing the underlying spanning tree as follows. We assume that if node $v \in G$ was the source, then the rumor spreads along a breadth first search (BFS) tree rooted at v, $T_{\text{bfs}}(v)$. The intuition is that if v was the source, then the BFS tree would correspond to the fastest (intuitively, most likely) spread of the rumor.

Therefore, effectively we obtain the following rumor source estimator for a general rumor infected graph G:

$$\widehat{v} \in \arg\max_{v \in G} \mathbf{P}\big(\sigma_v^* \mid v\big) R(v, T_{\text{bfs}}(v)). \tag{2.8}$$

In the above ties are broken uniformly at random as before. Also, like in (2.7), the σ_v^* represents the BFS ordering of nodes in the tree $T_{\text{bfs}}(v)$.

For example, consider the network in Figure 2-4. The BFS trees for each node are shown. Using the expression for $R(v, G)$ from Section 2.3.1, the general graph estimator values for the nodes are

$$\begin{aligned}
\mathbf{P}\big(\sigma_1^* \mid 1\big) R(1, T_{\text{bfs}}(1)) &= \frac{1}{4*6*8*10} \frac{5!}{20} \\
\mathbf{P}\big(\sigma_2^* \mid 2\big) R(2, T_{\text{bfs}}(2)) &= \frac{1}{4*6*8*10} \frac{5!}{30} \\
\mathbf{P}\big(\sigma_3^* \mid 3\big) R(3, T_{\text{bfs}}(3)) &= \frac{1}{4*6*8*10} \frac{5!}{20} \\
\mathbf{P}\big(\sigma_4^* \mid 4\big) R(4, T_{\text{bfs}}(4)) &= \frac{1}{4*6*8*10} \frac{5!}{10} \\
\mathbf{P}\big(\sigma_5^* \mid 5\big) R(5, T_{\text{bfs}}(5)) &= \frac{1}{4*6*8*10} \frac{5!}{40}.
\end{aligned}$$

Node 4 maximizes this value and would be the estimate of the rumor source for this network. We will show with simulations that this general graph estimator performs well on different network topologies.

2.3 Rumor Centrality: Properties & Algorithm

The quantity $R(v, G)$ plays an important role in each of the rumor source estimators (2.6), (2.7) and (2.8). Recall that $R(v, G)$ counts the number of distinct ways a rumor can spread in the network G starting from source v. Thus, it assigns each node of G a non-negative number or score. We shall call this number, $R(v, G)$, the *rumor centrality* of the node v with respect to G. The node with maximal rumor centrality will be called the *rumor center* of the network. Indeed, the rumor center is the ML estimation of the rumor source for

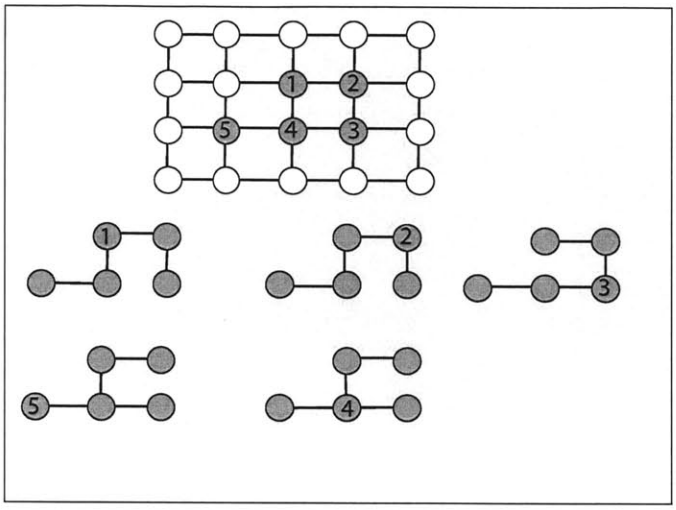

Figure 2-4: Example network with a BFS tree for each node shown. The rumor infected nodes are shown in gray.

regular trees.

This section describes ways to evaluate $R(v, G)$ efficiently when G is a tree. It also describes an important property of rumor centrality that will be useful in establishing theoretical results later in Chapter 3. Further, we discuss a surprising relation between the rumor center and the so called distance center of a tree. We remark on the relation between rumor centrality and the number of linear extensions of a partially ordered set described by a tree graph. Finally, we show how rumor centrality emerges as the stationary distribution of an appropriately defined Markov chain on a network.

2.3.1 Rumor Centrality: A Succinct Representation

Let G be a tree graph. Define T_u^v as the number of nodes in the subtree rooted at node u, with node v as the source. To illustrate this notation, a simple example is shown in Figure 2-5. Here $T_2^1 = 3$ because there are 3 nodes in the subtree with node 2 as the root and node 1 as the source. Similarly, $T_7^1 = 1$ because there is only 1 node in the subtree with node 7 as the root and node 1 as the source. We now can count the permitted permutations of G with v as the source. In the following analysis, we will abuse notation and use T_u^v to refer to both the subtrees and the number of nodes in the subtrees. Recall that we are looking for permitted permutations of the N nodes of G. That is, we have N slots in a

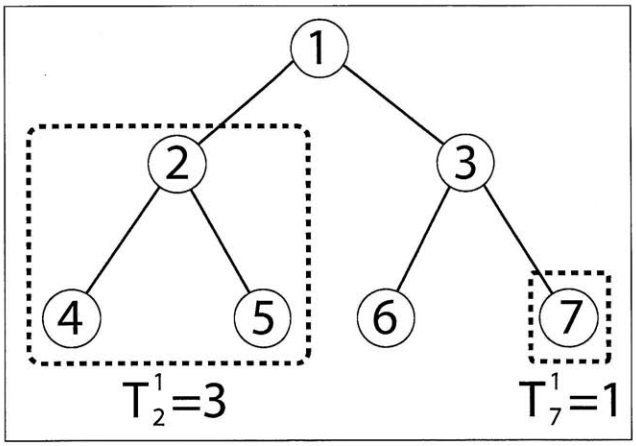

Figure 2-5: Illustration of subtree variable T_u^v.

given permitted permutation, the first of which must be the source node v. The question is, how many distinct ways can we fill the remaining $N-1$ slots. The basic constraint is due to the causality induced by the tree graph: a node u must come before all the nodes in its subtree T_u^v. Given a slot assignment for all nodes in T_u^v subject to this constraint, there are $R(u, T_u^v)$ different ways in which these nodes can be ordered. This suggests a natural recursive relation between the rumor centrality $R(v, G)$ and the rumor centrality of its immediate children's subtrees $R(u, T_u^v)$ with $u \in \text{child}(v)$. Here child(v) represents the set of all children of v in tree G assuming v as its root. Specifically, there is no constraint between the orderings of the nodes of different subtrees T_u^v with $u \in \text{child}(v)$. This leads to the following relation.

$$R(v, G) = (N-1)! \prod_{u \in \text{child}(v)} \frac{R(u, T_u^v)}{T_u^v!}. \tag{2.9}$$

To understand the above expression, note that the number of ways to partition $N-1$ slots for different subtrees is $(N-1)! \prod_{u \in \text{child}(v)} \frac{1}{T_u^v!}$, and the partition corresponding to T_u^v, $u \in \text{child}(v)$ leads to $R(u, T_u^v)$ distinct orderings, thus resulting in (2.9).

If we expand this recursion (2.9) to the next level of depth in G we obtain

$$\begin{aligned} R(v,G) &= (N-1)! \prod_{u \in \text{child}(v)} \frac{R(u,T_u^v)}{T_u^v!} \\ &= (N-1)! \prod_{u \in \text{child}(v)} \frac{(T_u^v-1)!}{T_u^v!} \prod_{w \in \text{child}(u)} \frac{R(w,T_w^v)}{T_w^v!} \\ &= (N-1)! \prod_{u \in \text{child}(v)} \frac{1}{T_u^v} \prod_{w \in \text{child}(u)} \frac{R(w,T_w^v)}{T_w^v!}. \end{aligned}$$

A leaf node l will have have 1 node and 1 permitted permutation, so $R(l,T_l^v) = 1$. If we continue this recursion until we reach the leaves of the tree, then we find that the number of permitted permutations for a given tree G rooted at v is

$$\begin{aligned} R(v,G) &= (N-1)! \prod_{u \in G \setminus v} \frac{1}{T_u^v} \\ &= N! \prod_{u \in G} \frac{1}{T_u^v}. \end{aligned} \tag{2.10}$$

In the last line, we have used the fact that $T_v^v = N$. We thus end up with a simple expression for rumor centrality in terms of the size of the subtrees of all nodes in G.

As an example of the use of rumor centrality, consider the network in Figure 2-6. Using the rumor centrality formula, we find that the rumor centrality of node 1 is

$$R(1,G) = \frac{5!}{5*3} = 8.$$

Indeed, there are 8 permitted permutations of this network with node 1 as the source, which we list below.

$$\{1,3,2,4,5\}, \{1,2,3,4,5\}, \{1,2,4,3,5\}, \{1,2,4,5,3\},$$
$$\{1,3,2,5,4\}, \{1,2,3,5,4\}, \{1,2,5,3,4\}, \{1,2,5,4,3\}.$$

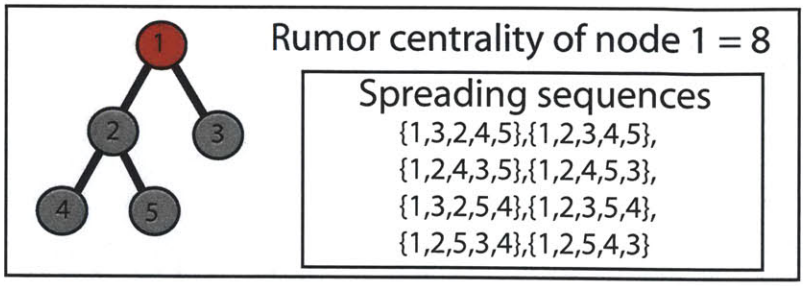

Figure 2-6: Example network for calculating rumor centrality.

2.3.2 Rumor Centrality via Message-Passing

In order to find the rumor center of an N node tree G, we need to first find the rumor centrality of every node in G. To do this we need the size of the subtrees T_u^v for all v and u in G. There are N^2 of these subtrees. Therefore, a naive algorithm can lead to $\Omega(N^2)$ operations. We shall utilize a local relation between the rumor centrality of neighboring nodes in order to calculate it in $O(N)$ computation in a distributed, message-passing manner.

To this end, consider two neighboring nodes u and v in G. All of their subtrees will be the same size except for those rooted at u and v. In fact, there is a special relation between these two subtrees.

$$T_u^v = N - T_v^u. \tag{2.11}$$

For example, in Figure 2-5, for node 1, T_2^1 has 3 nodes, while for node 2, T_1^2 has $N - T_2^1$ or 4 nodes. Because of this relation, we can relate the rumor centralities of any two neighboring nodes.

$$R(u, G) = R(v, G)\frac{T_u^v}{N - T_u^v}. \tag{2.12}$$

This result is the key to our algorithm for calculating the rumor centrality for all nodes in G. We first select any node v as the source node and calculate the size of all of its subtrees T_u^v and its rumor centrality $R(v, G)$. This can be done by having each node u pass two messages up to its parent. The first message is the number of nodes in u's subtree, which we call $t^{up}_{u\to\text{parent}(u)}$. The second message is the cumulative product of the size of the subtrees of all nodes in u's subtree, which we call $p^{up}_{u\to\text{parent}(u)}$. The parent node then adds the $t^{up}_{u\to\text{parent}(u)}$ messages together to obtain the size of its own subtree, and multiplies the $p^{up}_{u\to\text{parent}(u)}$ messages together to obtain its cumulative subtree product. These messages

are then passed upward until the source node receives the messages. By multiplying the cumulative subtree products of its children, the source node will obtain its rumor centrality, $R(v, G)$.

With the rumor centrality of node v, we then evaluate the rumor centrality for the children of v using equation (2.12). Each node passes its rumor centrality to its children in a message we define as $r^{down}_{u\to u'}$ for $u' \in \text{child}(u)$. Each node u can calculate its rumor centrality using its parent's rumor centrality and its own subtree size T^v_u. We recall that the rumor centrality of a node is the number of permitted permutations that result in G. Thus, this message-passing algorithm is able to count the (exponential) number of permitted permutations for every node in G using only $O(N)$ computations. The pseudocode for this message-passing algorithm is included for completeness.

Algorithm 1 Rumor Centrality Message-Passing Algorithm

Choose a root node $v \in G$
for u in G **do**
  if u is a leaf **then**
    $t^{up}_{u\to\text{parent}(u)} = 1$
    $p^{up}_{u\to\text{parent}(u)} = 1$
  else
    if u is root v **then**
      $\forall\, v' \in \text{child}(v)$: $r^{down}_{v\to v'} = \dfrac{N!}{N \prod\limits_{j\in\text{child}(v)} p^{up}_{j\to v}}$
    else
      $t^{up}_{u\to\text{parent}(u)} = \sum\limits_{j\in\text{child}(u)} t^{up}_{j\to u} + 1$
      $p^{up}_{u\to\text{parent}(u)} = t^{up}_{u\to\text{parent}(u)} \prod\limits_{j\in\text{child}(u)} p^{up}_{j\to u}$
      $\forall\, u' \in \text{child}(u)$: $r^{down}_{u\to u'} = r^{down}_{\text{parent}(u)\to u} \frac{t^{up}_{u\to\text{parent}(u)}}{N - t^{up}_{u\to\text{parent}(u)}}$
    end if
  end if
end for

2.3.3 A Property of Rumor Centrality

The following is an important characterization of the rumor center (node with maximal rumor centrality) in terms of the sizes of its local subtrees. As we shall see, this will play

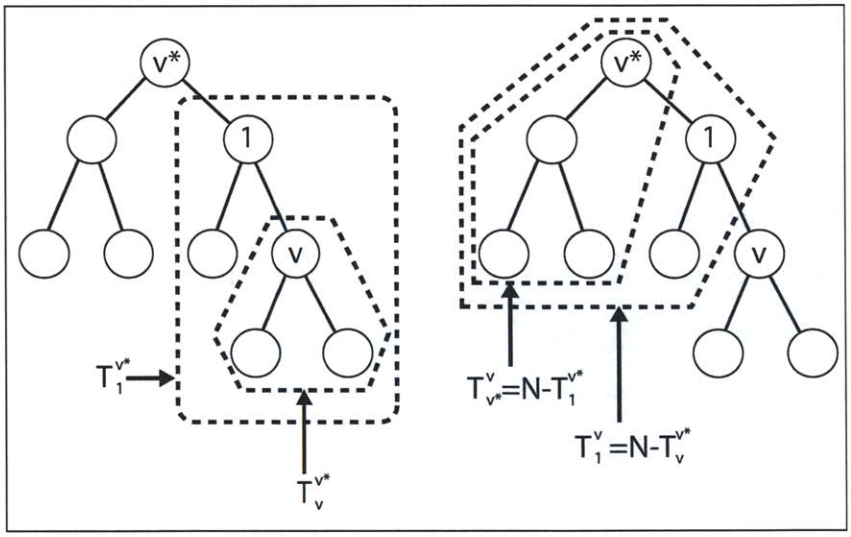

Figure 2-7: T_i^j variables for source nodes 2 hops apart.

a crucial role in establishing our main results about the performance of rumor centrality as an estimator for tree graphs.

Proposition 1. *Given an N node tree, if node v^* is the rumor center, then any subtree with v^* as the source must have the following property:*

$$T_v^{v^*} \leq \frac{N}{2}. \tag{2.13}$$

If there is a node u such that for all $v \neq u$

$$T_v^u \leq \frac{N}{2} \tag{2.14}$$

then u is a rumor center. Furthermore, a tree can have at most 2 rumor centers.

Proof. We showed that for a tree with N total nodes, for any neighboring nodes u and v,

$$T_u^v = N - T_v^u. \tag{2.15}$$

For a node v one hop from v^*, we find

$$\frac{R(v,T)}{R(v^*,T)} = \frac{T_{v^*}^{v^*} T_v^{v^*}}{T_{v^*}^v T_v^v} = \frac{T_v^{v^*}}{(N - T_v^{v^*})}.$$

The notation above assumes that $T_v^v = N$ for any v in the tree, i.e. T_v^v is the entire tree. When v is two hops from v^*, all of the subtrees are the same except for those rooted at v, v^*, and the node in between, which we call node 1. Figure 2-7 shows an example. In this case, we find

$$\frac{R(v,T)}{R(v^*,T)} = \frac{T_v^{v^*} T_1^{v^*}}{(N - T_1^{v^*})(N - T_v^{v^*})}.$$

Continuing this way, we find that in general, for any node v in T,

$$\frac{R(v,T)}{R(v^*,T)} = \prod_{i \in \mathcal{P}(v^*,v)} \frac{T_i^{v^*}}{(N - T_i^{v^*})} \tag{2.16}$$

where $\mathcal{P}(v^*, v)$ is the set of nodes in the path between v^* and v, not including v^*.

Now imagine that v^* is the rumor center. Then we have

$$\frac{R(v,T)}{R(v^*,T)} = \prod_{i \in \mathcal{P}(v^*,v)} \frac{T_i^{v^*}}{(N - T_i^{v^*})} \leq 1 \tag{2.17}$$

For a node v one hop from v^*, this gives us that

$$T_v^{v^*} \leq \frac{N}{2}. \tag{2.18}$$

For any node u in subtree $T_v^{v^*}$, we will have $T_u^{v^*} \leq T_v^{v^*} - 1$. Therefore, (2.18) will hold for any node $u \in T$. This proves the first part of Proposition 1.

Now assume that the node v^* satisfies (2.18) for all $v \neq v^*$. Then the ratios in (2.16) will all be less than or equal to 1. Thus, we find that

$$\frac{R(v,T)}{R(v^*,T)} = \prod_{i \in \mathcal{P}(v^*,v)} \frac{T_i^{v^*}}{(N - T_i^{v^*})} \leq 1. \tag{2.19}$$

Thus, v^* is the rumor center, as claimed in the second part of Proposition 1.

Finally, assume that v^* is a rumor center and that all of its subtrees satisfy $T_v^{v^*} < N/2$. Then, any other node v will have at least one subtree that is larger than $N/2$, so v^* is the unique rumor center. Now assume that v^* has a neighbor v such that $T_v^{v^*} = N/2$. Then, $T_{v^*}^v = N/2$ also, and all other subtrees $T_u^v < N/2$, so v is also a rumor center. There can

be at most 2 nodes in a tree with subtrees of size $N/2$, so a tree can have at most 2 rumor centers. □

2.3.4 Rumor Centrality vs. Distance Centrality

Here we shall compare rumor centrality with distance centrality that has become popular in the literature as a graph based score function for various other applications. To start with, we recall the definition of distance centrality. For a graph G, we will define the distance centrality of node $v \in G$, $D(v, G)$ as

$$D(v, G) = \sum_{j \in G} d(v, j) \tag{2.20}$$

where $d(v, j)$ is the shortest path distance from node v to node j. With this definition, which is slightly different than the one used in Chapter 1, the distance center of a graph is the node with the *smallest* distance centrality. Intuitively, it is the node closest to all other nodes. On a tree, we will show the distance center is equivalent to the rumor center. Therefore, by establishing correctness of rumor centrality for tree graphs, one immediately finds that such is the case for distance centrality.

We will prove the following proposition for the distance center of a tree.

Proposition 2. *On an N node tree, if v_D is the distance center, then, for all $v \neq v_D$*

$$T_v^{v_D} \leq \frac{N}{2}. \tag{2.21}$$

Furthermore, if there is a unique rumor center on the tree, then it is equivalent to the distance center.

Proof. Assume that node v_D is the distance center of a tree T which has N nodes. The distance centrality of v_D is less than any other node. We consider a node v_ℓ which is ℓ hops from v_D, and label a node on the path between v_ℓ and v_D which is h hops from v_D by v_h. Now, because we are dealing with a tree, we have the following important property. For a node j which is in subtree $T_{v_h}^{v_D}$ but not in subtree $T_{v_{h+1}}^{v_D}$, we have $d(v_\ell, j) = d(v_D, j) + l - 2h$.

Using this, we find

$$\begin{aligned}
D(v_D,T) \leq & D(v_\ell,T) \\
\sum_{j\in T} d(v_D,j) \leq & \sum_{j\in T} d(v_\ell,j) \\
\leq & \sum_{j\in T} d(v_D,j) + \ell(N - T_{v_1}^{v_D}) + \\
& (\ell-2)(T_{v_1}^{v_D} - T_{v_2}^{v_D}) + ... + (\ell - 2\ell)(T_{v_\ell}^{v_D}) \\
0 \leq & \ell N - 2T_{v_1}^{v_D} + 2T_{v_2}^{v_D} - 4T_{v_2}^{v_D} + ... \\
& + 2(\ell-1)T_{v_l}^{v_D} - 2\ell T_{v_l}^{v_D} \\
0 \leq & (N - T_{v_1}^{v_D}) - T_{v_1}^{v_D} + (N - T_{v_2}^{v_D}) - T_{v_2}^{v_D} + ... \\
& + (N - T_{v_l}^{v_D}) - T_{v_l}^{v_D} \\
\sum_{h=1}^{\ell} T_{v_h}^{v_D} \leq & \sum_{h=1}^{\ell} (N - T_{v_h}^{v_D}).
\end{aligned} \tag{2.22}$$

If we consider a node v_1 adjacent to v_D, we find the same condition we had for the rumor center. That is,

$$T_{v_1}^{v_D} \leq \frac{N}{2}. \tag{2.23}$$

For any node u in subtree $T_{v_1}^{v_D}$, we will have $T_u^{v_D} \leq T_{v_1}^{v_D} - 1$. Therefore, (2.23) will hold for any node $u \in T$. This proves the first half of Proposition 2.

If v_D is a rumor center, then, it also satisfies (2.23) as previously shown. Thus, when unique, the rumor center is equivalent to the distance center on a tree. This proves the second half of Proposition 2. □

It is interesting to view how rumor centrality quantifies the importance of nodes versus distance centrality. We consider the example network from Chapter 1, reproduced here in Figure 2-8. We draw the network with node sizes proportional to their distance centrality (using the definition in equation (1.4)) and rumor centrality. We also show in Table 2.1 the values for these centralities. As can be seen, rumor centrality gives a great deal of importance to the more central nodes a and b as compared to distance centrality, and almost

Node	Distance centrality	Rumor centrality
a	0.071	36,288
b	0.071	36,288
c	0.046	4,032
d	0.046	4,0321
e	0.046	4,032
f	0.046	4,032
g	0.046	4,032
h	0.046	4,032
i	0.050	9,072
j	0.036	1,008

Table 2.1: Distance and rumor centralities for example network.

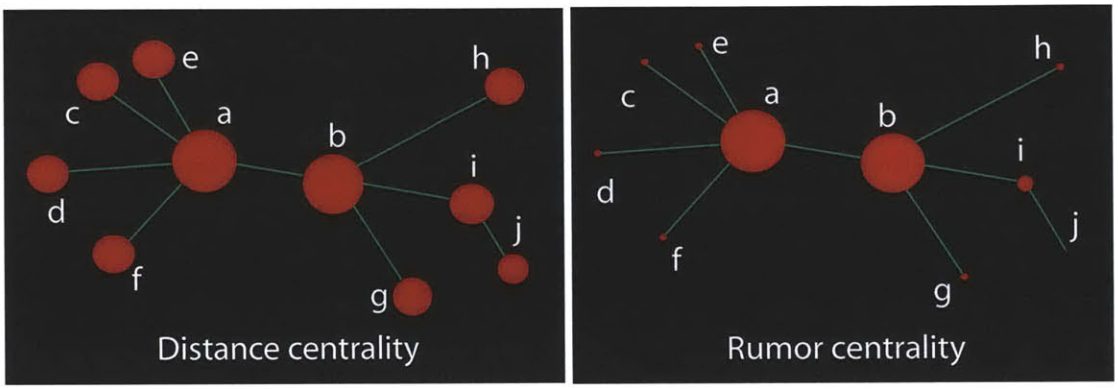

Figure 2-8: Example network with node sizes scaled by corresponding network centrality: (left) distance centrality and (right) rumor centrality.

no importance to node j. Therefore, even though the rumor center and distance center may be equivalent on trees, the amount of importance given to these nodes by rumor and distance centrality is vastly different.

Now in contrast to trees, in a general non-tree network, the rumor center and distance center need not be equivalent. Specifically, we shall define rumor centrality for a general network to be the node with maximal value of rumor centrality on its own BFS tree. Stated more precisely, the rumor center of a general graph is the node $\hat{v}$ with the following property (ties broken uniformly at random):

$$\hat{v} \in \arg\max_{v \in G} R(v, T_{\text{bfs}}(v)). \tag{2.24}$$

In a general network, as can be seen in Figure 2-9, this general network rumor center is

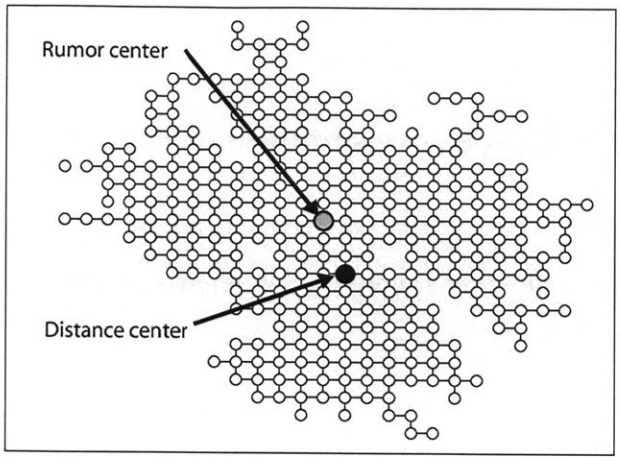

Figure 2-9: A network where the distance center does not equal the general graph rumor center. In this network the rumor center turns out to be the source of the rumor.

not always equivalent to the distance center as it was for trees. This particular network was formed by spreading a rumor on a grid network. It turns out that for this network the rumor center is equal to the true source. We will see later that the general network rumor center will be a better estimator of the rumor source than the distance center. The intuition for this is that the distance center is evaluated using only the shortest paths in the network, whereas the general network rumor centrality utilizes more of the network structure for estimation of the source.

2.3.5 Rumor Centrality and Linear Extensions of Posets

The rumor graph on a network can be viewed as a partially ordered set, or *poset*, of nodes if we fix a source node as the root and consider the network to be directed, with edges pointing from the node that had the rumor to the node it infected. These directed edges impose a partial order on the nodes. We have referred to any permutation of the nodes which satisfies this partial order as a permitted permutation. However, it is also known as a *linear extension* of the poset. It is known that counting the number of linear extensions of a poset is in general a very hard problem (specifically, it falls in the complexity class #P-complete [24]). However, on trees, counting linear extensions becomes computationally tractable. To the best of our knowledge, the fastest known algorithm for counting linear extensions on a tree requires $O(N^2)$ computation [8]. In contrast, the message-passing

algorithm we presented in Section 2.3.2 required only $O(N)$ computation.

2.3.6 Markov Chain Representation of Rumor Centrality

There is an interesting connection between the rumor centrality of nodes in a network and a certain Markov chain or random walk defined on the same network. We consider a tree network $G = (V, E)$, where each node $v \in V$ is a state of the Markov chain and the network has $|V| = N$ nodes. Transitions are allowed along the edges of E between the nodes. We also allow for self-loop transitions to guarantee that the Markov chain is aperiodic. The connection between this Markov chain and rumor centrality occurs if we define the following transition probabilities. Let p_{uv} be the probability of transitioning from node u to node v. Then we define these probabilities as follows.

$$p_{uv} = \begin{cases} \frac{1}{2}\frac{T_v^u}{N-1} & \text{if } u \neq v \text{ and } (u,v) \in E \\ \frac{1}{2} & \text{if } u = v \\ 0 & \text{else.} \end{cases} \tag{2.25}$$

It can be checked that these are proper transition probabilities and that $\sum_{v \in G} p_{uv} = 1$. In each step of this Markov chain, we either stay where we are or transition to a neighbor of the current state with probability proportional to the size of the neighbor's subtree. An example of this Markov chain is shown in Figure 2-10.

We find that the stationary distribution of this Markov chain is proportional to the rumor centrality of the nodes. Let $\mathbf{P} \in \mathbf{R}^{n \times n}$ be the transition matrix for the random walk , with p_{uv} as defined in equation (2.25). The the stationary distribution π for this random walk is a vector that satisfies the following equations

$$\mathbf{P}^T \pi = \pi \tag{2.26}$$

and

$$\sum_{v \in G} \pi_v = 1. \tag{2.27}$$

The first of these equations says that π remains invariant as one transitions in the Markov chain, and the second equation simply constrains π to be a properly normalized probability distribution.

For this Markov chain, we establish the following result.

Theorem 1. *Given a tree $G = (V, E)$ and the associated Markov chain with transition probabilities given by equation* (2.25)*, the stationary distribution π is given by*

$$\pi_v \propto R(v, G) \ \forall v \in G. \tag{2.28}$$

Another way to view this Markov chain is as a random walk is what we refer to as the *trendy gossiper model.* In this model there is one gossiper who randomly walks on the network. He will either stay where he is or move from node to node along the edges of the network, but the gossiper is trendy and prefers to visit nodes which can spread gossip to many other nodes. That is, he prefers to go to nodes with large subtrees. This is exactly the transition probability defined in equation (2.25). What Theorem 1 tells us is that under this trendy gossiper model the probability of the gossiper being found at a node is proportional to that node's rumor centrality. This provides us with a probabilistic perspective for rumor centrality which is distinct from the purely combinatorial perspective based on counting linear extensions of partial orders from section 2.3.5.

Proof. We now provide a simple proof of Theorem 1. We wish to show that the stationary distribution π is indeed proportional to the rumor centrality of the nodes. The Markov chain as defined is aperiodic, irreducible, and positive recurrent so the stationary distribution exists and is unique. We will prove that $\pi_v \propto R(v, G)$ by showing that with this stationary distribution the Markov chain is reversible and satisfies detailed balance. To do this, let us

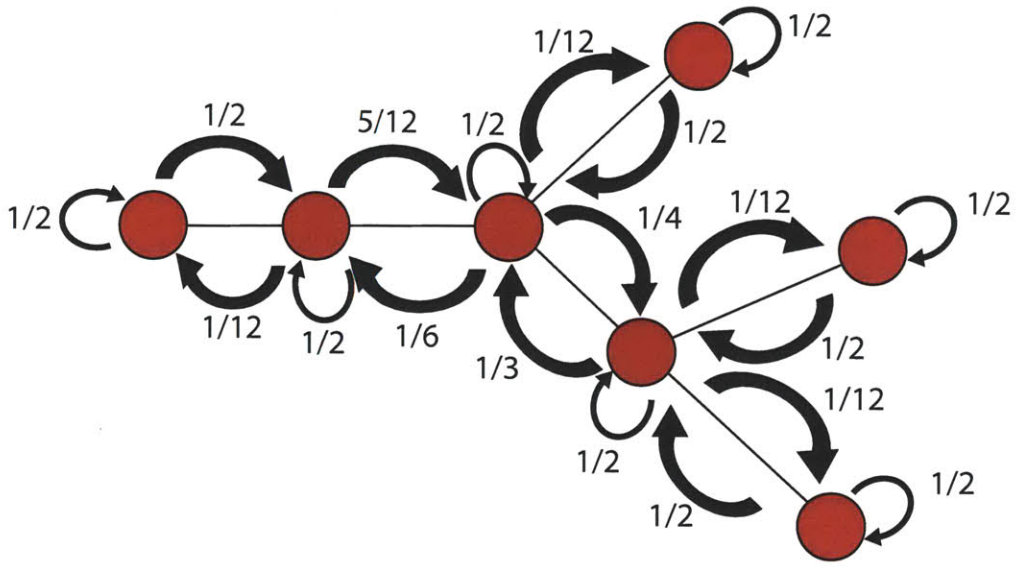

Figure 2-10: An example of a Markov chain on a tree network where the stationary distribution is proportional to rumor centrality. The transition probability between a node and one of its neighbors is proportional to the size of the neighbor's subtree. These transition probabilities are indicated on the edges of the network in the figure.

look at the detailed balance equation for two nodes u and v such that $u \neq v$ and $(u, v) \in E$.

$$\begin{aligned}
\pi_u p_{uv} &= \pi_v p_{vu} \\
R(u,G)\frac{1}{2}\frac{T_v^u}{N-1} &= R(v,G)\frac{1}{2}\frac{T_u^v}{N-1} \\
R(v,G)\frac{T_u^v}{N-T_u^v}T_v^u &= R(v,G)T_u^v \\
R(v,G)\frac{T_v^u}{T_u^v} &= R(v,G) \\
R(v,G) &= R(v,G)
\end{aligned}$$

□

Above, in the third line we used equation (2.12) and in the fourth line we used (2.11).

2.4 Chapter Summary

Finding the source of a rumor in a network is an important and challenging problem. In this chapter we constructed the appropriate network centrality measure for this problem

which we called rumor centrality. This centrality measure captured the relevant structural property for the rumor source detection problem. Specifically, it counted the ways a rumor could spread from a fixed source. We saw that under the SI model with exponentially distributed spreading times, rumor centrality was an exact ML estimator for regular trees. For non-regular trees and for arbitrary graphs we were able to construct heuristic estimators based upon rumor centrality. We presented an efficient message passing algorithm to evaluate rumor centrality. This demonstrated that rumor centrality is a very tractable source estimator for large networks. By comparing rumor centrality with distance centrality, we found that both centralities identify the same types of nodes as important, but rumor centrality gives much more importance to these nodes than does distance centrality. For regular trees, this importance is precisely the likelihood of the node being the source, which is the relevant knowledge we wish to extract from the network for this problem. For a different perspective on rumor centrality, we showed how it emerged as the stationary distribution of an appropriately defined Markov chain on the network.

Chapter 3

Rumor Source Estimator Performance Analysis

In this chapter we will study in depth the problem of finding the source of a rumor in a network. We look at the fundamental limits to this rumor source detection problem. In particular, we study how well one can find the rumor source, the distribution of the error, and how the network structure affect one's ability to find the rumor source. Also, we provide a precise theoretical characterization of the performance of the rumor centrality based estimator for networks which are "locally tree-like".

3.1 Theoretical Results

We will analyze the theoretical limits of the rumor source detection problem for a wide class of networks. We will find that rumor centrality is a very accurate rumor source estimator across many different networks including geometric trees, regular trees, random trees, random regular graphs, and Erdos-Renyi graphs. We will establish our results for rumors which spread as per the SI model with exponential spreading times, and also for arbitrary spreading times. Our theoretical results will draw upon techniques from classical Polya urn models and branching processes.

3.1.1 Notation

We will be interested in analyzing the error of the rumor centrality based estimator in this section. Therefore, we must define precisely what is meant by an error. Assume a rumor spreads on a network $\mathcal{G}$ for time t and that we observe the rumor infected network G which is a subgraph of $\mathcal{G}$ with $n(t)$ nodes and that the nodes were infected in the following sequence: $\{v_1, v_2, ..., v_{n(t)}\}$. In this notation, v_1 is the rumor source. We then define $C^k_{n(t)}$ as the event that the node estimated as the rumor source by rumor centrality (the rumor center) is the k^{th} infected node. $C^1_{n(t)}$ is then the event of correct detection. We will be interested in evaluating $\mathbf{P}\left(C^k_{n(t)}\right)$ for different classes of networks.

3.1.2 SI Model With an Exponential Distribution on Regular Tree Networks

We first consider rumor spreading on regular tree networks as per the SI model with an exponential distribution. Because we specify the node degrees and the distribution of the spreading time we are able to obtain rather precise results. We first look at regular trees of degree 2 (linear graphs) and then regular trees of degree greater than 2.

Regular Trees With Degree $d = 2$: Linear Graphs

First we consider linear graphs, which are regular trees of degree 2. On these networks rumor centrality is an exact ML estimator when the spreading times are exponentially distributed. We assume the rumor spreads according to the SI model with exponential spreading times. For this class of network we establish the following result.

Theorem 2. *Let $\mathcal{G}$ be an infinite linear graph. Assume a rumor spreads on $\mathcal{G}$ as per the SI model with spreading times distributed as per an exponential distribution with parameter $\lambda > 0$. Then we have that*

$$\mathbf{P}(C^1_{n(t)}) = \Theta\left(\frac{1}{\sqrt{t}}\right).$$

As can be seen, the linear graph detection probability scales as $t^{-1/2}$, which goes to 0 as t goes to infinity. The intuition for this result is that the estimator provides very little

information because of the linear graph's trivial structure.

Another way to view this is to think about the properties of the rumor center from Proposition 1. The rumor graph must be balanced around the rumor center. Because there are only two neighbors of the source, this means that their subtrees must be *exactly* the same size. Because the rumor spreading is random, it is very difficult to achieve this perfect balance, especially as t grows large. Therefore, it becomes impossible to find this source in a linear graph.

This balance observation has deeper implications. It implies that if the source has degree 2, then rumor centrality can never detect it because achieving this perfect balance is near impossible when t is large. Therefore, for any non-trivial detection to be possible, the source must have degree of at least 3.

Regular Trees With Degree $d \geq 3$

We now look at rumor source detection on regular trees with degree $d \geq 3$, where rumor centrality is an exact ML estimator when the spreading times are exponentially distributed. Our results will utilize properties of Beta random variables. Therefore, recall that the regularized incomplete Beta function $I_x(a, b)$ is the probability that a Beta random variable with parameters a and b is less than $x \in [0, 1]$,

$$I_x(a,b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)} \int_0^x t^{a-1}(1-t)^{b-1} dt, \tag{3.1}$$

where $\Gamma(\cdot)$ is the standard Gamma function.

For regular trees of degree ≥ 3 we obtain the following result.

Theorem 3. *Let $\mathcal{G}$ be d-regular infinite tree with $d \geq 3$. Assume a rumor spreads on $\mathcal{G}$ as per the SI model with spreading times distributed as per an exponential distribution with rate $\lambda > 0$. Then, for any $k \geq 1$,*

$$\begin{aligned}\lim_{t\to\infty} \mathbf{P}\left(C^k_{n(t)}\right) =& I_{1/2}\left(k-1+\frac{1}{d-2}, 1+\frac{1}{d-2}\right) \\ &+ (d-1)\left(I_{1/2}\left(\frac{1}{d-2}, k+\frac{1}{d-2}\right) - 1\right).\end{aligned} \tag{3.2}$$

For $d = 3$, the above result has a clean form and provides the insight that the probability of error in the estimation decays exponentially with error distance from the true source.

Corollary 1. *When $\mathcal{G}$ is a 3-regular infinite tree, for any $k \geq 1$,*

$$\lim_{t\to\infty} \mathbf{P}\Big(C^k_{n(t)}\Big) = k\Big(\frac{1}{2}\Big)^{(k+1)} \sim \exp\Big(-\Theta(k)\Big).$$

For $k = 1$, Theorem 3 yields that $\mathbf{P}\left(C^1_{n(t)}\right)$ is monotonic in the degree d. More interestingly,

Corollary 2.

$$\lim_{d\to\infty}\lim_{t\to\infty} \mathbf{P}\left(C^1_{n(t)}\right) = 1 - \ln 2 \approx 0.307. \tag{3.3}$$

3.1.3 SI Model With a General Distribution on Tree Networks

The above precise results were obtained using the memoryless property of the exponential distribution and the regularity of the trees. We now look at a more general setting where spreading times are drawn from arbitrary distributions and node degrees are random. In this more general setting, while we cannot obtain precise values for the detection and error probabilities, we are able to make statements about the non-triviality of the detection probability of rumor centrality.

First, we assume that the spreading times are no longer limited to exponential random variables. We let each edge $e \in \mathcal{E}$ in the network have an independent spreading time T_e which has distribution $F(t)$, with all spreading times independent and identically distributed. That is

$$\mathbf{P}\left(T_e \leq t\right) = F(t) \;\; \forall e \in \mathcal{E} \tag{3.4}$$

We will first look at random trees, and then at geometric trees.

Random Trees

We allow the underlying network $\mathcal{G}$ to be a random tree. Specifically, we consider the following generative model of a random tree. Starting with a root node, add a random number of children, say η_0 to the root with $\eta_0 \in \{0, 1 \dots\}$ having some distribution $\mathcal{D}_0$. If $\eta_0 \neq 0$, then to each child of the root, add a random number of children chosen as per distribution $\mathcal{D}$ over $\{0, 1, \dots\}$. Recursively, to each newly added node, add independently a random number of nodes as per distribution $\mathcal{D}$. When 0 children are added to a node, that branch of the tree terminates there.

The generative model described above is precisely the standard Galton-Watson branching process if $\mathcal{D}_0 = \mathcal{D}$. If we take $\mathcal{D}_0$ and $\mathcal{D}$ to be deterministic distributions with support on d and $d-1$ respectively, then it gives the d-regular tree. If we take $\mathcal{D}_0 = \mathcal{D}$ as a Poisson distribution with mean $c > 0$, then it equals the local graph structure of a randomly chosen node in an appropriate Erdos-Renyi graph. Recall that a (sparse) Erdos-Renyi graph on n nodes with parameter c is generated by selecting each of the $\binom{n}{2}$ edges to be present with probability c/n independently.

We state the result for the probability of correct detection, $C^1_{n(t)}$, over the generic random tree as described above. This will, as a consequence, lead to the characterization of the non-trivial detection probability for Erdos-Renyi graphs.

Theorem 4. *Let η_0, distributed as $\mathcal{D}_0$, be such that $\Pr(\eta_0 \geq 3) > 0$ and let η, distributed as per $\mathcal{D}$, be such that $\mathbb{E}[\eta^2] < \infty$, $\mathbb{E}[\eta] > 1$. Suppose the rumor starts from the root of the random tree generated as per distributions $\mathcal{D}_0$ and $\mathcal{D}$ as described above and spreads as per the SI model with spreading times having distribution $F(t)$. Then, there is a strictly positive probability (dependent only on $\mathcal{D}_0$, $\mathcal{D}$, and $F(t)$) of the root being detected as the rumor source by an estimator based on rumor centrality.*

What this result says is that no matter how many nodes are infected by the rumor, there is always a positive probability of detecting the source. This is in contrast to linear graphs (Theorem 2) where the detection probability went to zero as the number of infected nodes approached infinity. Therefore, we see that on these random trees, a non-trivial amount of the detection probability is being trapped at the true source.

Geometric Trees

We now consider the detection probability of rumor centrality in networks which grow polynomially. We shall call them *geometric trees*. These non-regular trees are parameterized by constants α, b, and c, with $0 < b \leq c$. We fix a source node v^* and consider each neighboring subtree of v^*. Let d^* be the degree of v^*. Then there are d^* subtrees of v^*, say $T_1, \ldots, T_{d^*}$. Consider the ith such subtree T_i, $1 \leq i \leq d^*$. Let v be any node in T_i and let $n^i(v, r)$ be the number of nodes in T_i at distance exactly r from the node v. Then we require that for all $1 \leq i \leq d^*$ and $v \in T_i$

$$br^\alpha \leq n^i(v, r) \leq cr^\alpha. \tag{3.5}$$

The condition imposed by (3.5) states that each of the neighboring subtrees of the source should satisfy polynomial growth (with exponent $\alpha > 0$) and regularity properties. The parameter $\alpha > 0$ characterizes the growth of the subtrees and the ratio c/b describes the regularity of the subtrees. If $c/b \approx 1$ then the subtrees are somewhat regular, whereas if the ratio is much greater than 1, there is substantial heterogeneity in the subtrees.

We note that unlike in regular trees, in a geometric tree the rumor centrality is not necessarily the ML estimator due to the heterogeneity. Nevertheless, we can use it as an estimator. Indeed, as stated below we find that the rumor centrality based estimator has an asymptotic detection probability of 1. That is, it is as good as the best possible estimator.

Theorem 5. *Let $\mathcal{G}$ be a geometric tree as described above with parameters $\alpha > 0$, $0 < b \leq c$ that satisfy* (3.5) *for a node v^* with degree $d_{v^*} \geq 3$. Let the following condition also be satisfied*

$$d_{v^*} > \frac{c}{b} + 1.$$

Assume a rumor spreads on $\mathcal{G}$ as per the SI model with spreading times having distribution $F(t)$ and finite moment generating function. Then

$$\lim_t \mathbf{P}(C^1_{n(t)}) = 1.$$

This theorem says that $\alpha = 0$ and $\alpha > 0$ serve as a threshold for non-trivial detection: for $\alpha = 0$, the graph is essentially a linear graph, so we would expect the detection probability to go to 0 as $t \to \infty$ based on Theorem 2, but for $\alpha > 0$ the detection probability converges to 1 as $t \to \infty$.

3.1.4 SI Model With a General Distribution on Locally Tree-Like Networks

We now extend our results for tree networks to general networks which are locally tree-like. We first look at random regular graphs under the SI model with an exponential distribution, and then at Erdos-Renyi graphs under the SI model with a general distribution.

SI Model With an Exponential Distribution on Random Regular Graphs

Using the local tree structure of random regular graphs, Theorem 3 can be extended to random d-regular graphs. To that end, let $G(m(t), d)$ be a random instance of a d-regular graph over $m(t)$ nodes, where we are allowing the underlying graph to grow with time t in order to make sure that the resulting rumor infected graph with $n(t)$ nodes is a tree with high probability.

Theorem 6. *Let the rumor spread for time t to $n(t)$ nodes from a uniformly chosen node in a random regular graph $G(m(t), d)$ for $d \geq 3$ with $n(t) = o(\sqrt{m(t)})$ as per the SI model with spreading times distributed as per an exponential distribution with rate $\lambda > 0$. Then*

$$\begin{aligned}\lim_{t\to\infty} \mathbf{P}\left(C^k_{n(t)}\right) =& I_{1/2}\left(k-1+\frac{1}{d-2}, 1+\frac{1}{d-2}\right) \\ &+ (d-1)\left(I_{1/2}\left(\frac{1}{d-2}, k+\frac{1}{d-2}\right) - 1\right).\end{aligned}$$

Thus we see that there is very similar error probability for random regular graphs and regular trees if the rumor spreads along a locally tree-like subgraph of the underlying random regular graph.

SI Model With a General Distribution on Erdos-Renyi Graphs

Next we look at rumor spreading on Erdos-Renyi graphs. Again, we are allowing the underlying graph to grow with time t in order to make sure that the resulting rumor infected graph with $n(t)$ nodes is a tree with high probability. As a consequence of Theorem 4, we establish the following result for Erdos-Renyi graphs.

Theorem 7. *Consider an Erdos-Renyi graph $G(m(t), c/m(t))$ over $m(t)$ nodes with $c > 1$. Let the rumor spread for time t on this graph starting with a randomly chosen node as per the SI model with spreading times having distribution $F(t)$. Suppose we observe the rumor infected subgraph of size $n(t)$ nodes with $n(t) = o(\sqrt{m(t)})$. Then with strictly positive probability the true source is the rumor center of the infected subgraph.*

Similar to the result for random regular graphs, here we are able to utilize the local tree-like property of Erdos-Renyi graphs to establish that non-trivial detection is possible.

3.2 Experiments

We have seen that non-trivial rumor source detection is possible for a whole host of network topologies. In this section we present results from rumor source detection simulations for different network topologies to demonstrate how effective rumor centrality is as a source estimator. These topologies include synthetic topologies such as the popular scale-free and small-world networks, and also real topologies such as the Internet and the U.S. electric power grid.

We will see with simulations on regular trees, grid networks, and Erdos-Renyi networks that in a typical rumor infected network, only a few nodes will have a non-trivial rumor centrality and that with high probability the true rumor source will be among these nodes. Therefore, rumor centrality has the very nice feature of narrowing down the set of possible sources in a large complex network to a very small number of nodes.

Throughout this section we perform rumor spreading simulations as per the SI model with exponential spreading times with rate 1.

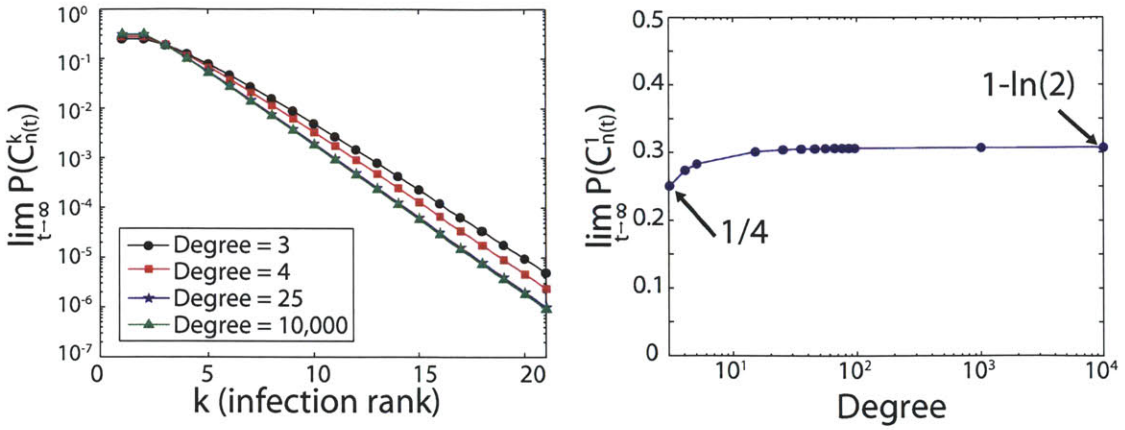

Figure 3-1: (left) $\lim_{t\to\infty} \mathbf{P}\left(C^k_{n(t)}\right)$ versus k for random regular graphs of different degree. (right) $\lim_{t\to\infty} \mathbf{P}\left(C^1_{n(t)}\right)$ versus degree for random regular graphs.

3.2.1 Random Regular Graphs

We first look at random regular graphs of different degree. We plot the asymptotic error distribution $\lim_{t\to\infty} \mathbf{P}\left(C^k_{n(t)}\right)$ from Theorem 6 for different degree random regular graphs in Figure 3-1. As can be seen, for degrees greater than 4, all the error distributions fall on top of each other, and the probability of detecting the k^{th} infection as the source decays exponentially in k. Thus, we see tight concentration of the error for this class of graphs. We also show in Figure 3-1 the asymptotic correct detection probability versus degree for these graphs. It can be seen that the detection probability starts at $1/4$ for degree 3 and rapidly converges to $1 - \ln(2)$ as the degree goes to infinity.

We also look at the speed of convergence to these asymptotic values. We perform rumor spreading simulations on random regular graphs and plot the empirical correct detection probability for different numbers of infected nodes n in Figure 3-2. As can be seen, for $n = 200$ we see that the detection probability seems to have already converged for degree greater than two. We also plot on Figure 3-2 the degree two random regular graph (linear graph). As can be seen, the detection probability decays as $n^{-1/2}$ as predicted by Theorem 2. In contrast, observe that for degree greater than two, the detection probability has a strictly positive asymptotic value as predicted by Theorem 6.

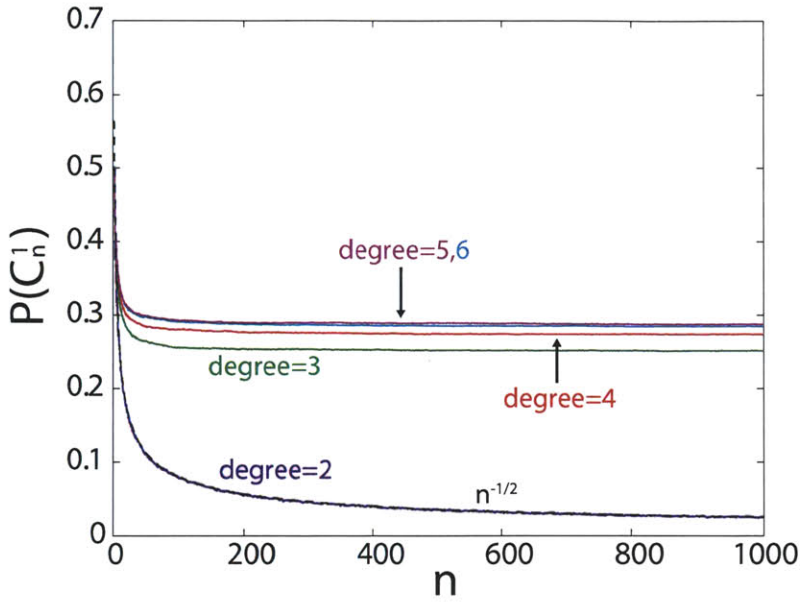

Figure 3-2: $\mathrm{P}\left(C_n^1\right)$ versus n for random regular graphs of different degree. The black dotted line is a plot of $n^{-1/2}$.

3.2.2 Geometric Trees

Simulation results for the empirical detection probability of rumor centrality versus network size for different geometric trees is show in Figure 3-3. We performed rumor spreading simulations for rumor graphs of different size n, with $10,000$ simulations per data point. As can be seen, the detection probability decays as $n^{-1/2}$ as predicted in Theorem 2 for the graphs which grow like lines ($\alpha = 0$). For geometric trees with $\alpha > 0$, we see that the detection probability does not decay to 0 and is very close to 1 as predicted by Theorem 5.

3.2.3 Erdos-Renyi Graphs

We performed simulations to gain insight into the behavior of the error distribution for Erdos-Renyi graphs. We used graphs with with $m = 50,000$ nodes and edge probabilities $p = c/m$ for $c = 10$ and $c = 20$. The rumor graph contained $n = 500$ nodes. We ran $10,000$ rumor spreading simulations to obtain the empirical error distributions, which are plotted in Figure 3-4. As can be seen, the error drops of exponentially, very similar to the random regular graph error distribution. In fact, we also plot the asymptotic distribution for a random regular graph of degree $10,000$ and it can be seen that the error decays at similar

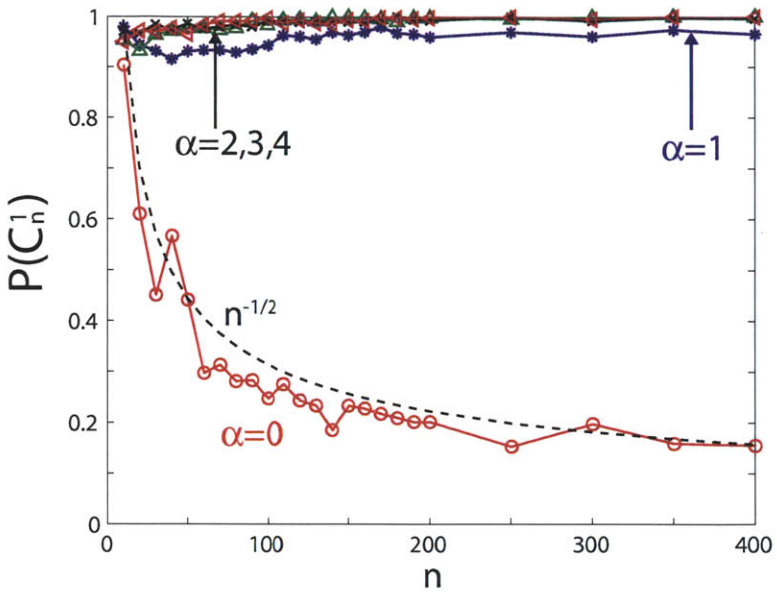

Figure 3-3: $\mathbf{P}\left(C_n^1\right)$ versus n for geometric trees. The black dotted line is a plot of $n^{-1/2}$.

rates. This indicates that even though there is substantial randomness in the graph, the asymptotic rumor source detection error distribution behaves as though it were a random regular graph.

3.2.4 Scale-free and Small-world Networks

We performed simulations on synthetic small-world [62] and scale-free [14] networks. These are two very popular models for networks and so we would like our rumor source estimator to perform well on these topologies. For both topologies, the underlying graph contained 5000 nodes and in the simulations we let the rumor spread to 400 nodes.

Figures 3-5 and 3-6 show an example of rumor spreading in a small-world and a scale-free network. The graphs show the rumor infected nodes in white. Also shown are the histograms of the estimator error for three different estimators. The estimators are distance centrality, rumor centrality on a BFS tree, and rumor centrality on a BFS tree with the BFS heuristic. For comparison, we also show with a dotted line a smooth fit of the histogram for the error from randomly choosing the source from the 400 node rumor network. As can be seen, for both networks, the histogram for the random guessing is shifted to the right of the estimator histograms. Thus, the centrality based estimators are a substantial improvement

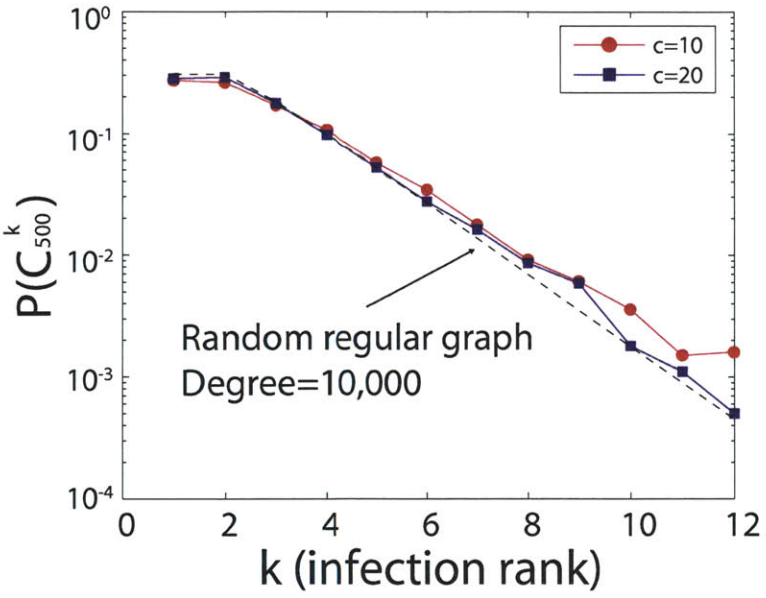

Figure 3-4: $\mathbf{P}\left(C_{500}^{k}\right)$ versus k for an Erdos-Renyi graph with mean degree 10 and 20. Also shown with a black dashed line is $\lim_{t\to\infty}\mathbf{P}\left(C_{n(t)}^{k}\right)$ for a degree $10,000$ random regular graph.

over random guessing for both small-world and scale-free networks.

The distance centrality estimator performs very similarly to the rumor centrality estimator. However, we see that on the small-world network, rumor centrality is better able to correctly find the source (0 error) than distance centrality (16% correct detection versus 2%). For the scale-free network used here, the average ratio of edges to nodes in the 400 node rumor graphs is 1.5 and for the small-world network used here, the average ratio is 2.5. For a tree, the ratio would be 1, so the small-world rumor graphs are less tree-like. This may explain why rumor centrality does better than distance centrality at correctly identifying the source on the small-world network.

The BFS heuristic leads to two visible effects. First, as can be seen for the scale-free network, we have a larger correct detection probability. Scale-free networks have power-law degree distributions, and thus contain many high degree *hubs*. The BFS heuristic works well in these types of networks because it was precisely designed for networks with heterogeneous degree distributions.

The second effect of the BFS heuristic is that larger errors become more likely. For both networks, the histograms spread out to higher errors. We see that for networks with less

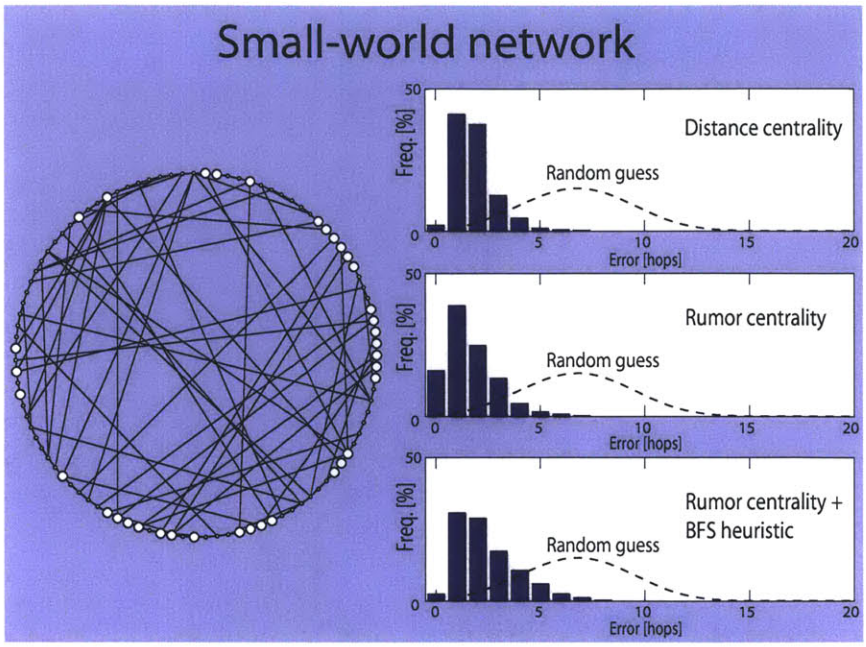

Figure 3-5: Histograms of the error for distance centrality, rumor centrality, and rumor centrality with BFS heuristic estimators on a 400 node rumor network on a small-world network. The dotted line is a smooth fit of the histogram for randomly guessing the source in the rumor network. An example of a rumor graph (infected nodes in white) is shown on the right.

heterogeneous degree distributions, such as the small-world network, the BFS heuristic is actually degrading performance. It may be that for more regular networks, the BFS heuristic amplifies the effect any slight degree heterogeneity and causes detection errors.

3.2.5 Real Networks

We performed simulations on an Internet autonomous system (AS) network [2] and the U.S electric power grid network [14]. These are two important real networks so we would like our rumor source estimator to perform well on these topologies. The AS network contained 32,434 nodes and the power grid network contained 4941 nodes. In the simulations we let the rumor spread to 400 nodes.

Figures 3-7 and 3-8 show an example of rumor spreading in both of these networks. Also shown are the histograms of the estimator error for three different estimators. The estimators are distance centrality, rumor centrality on a BFS tree, and rumor centrality on a BFS tree with the BFS heuristic. For comparison, we also show with a dotted line a smooth

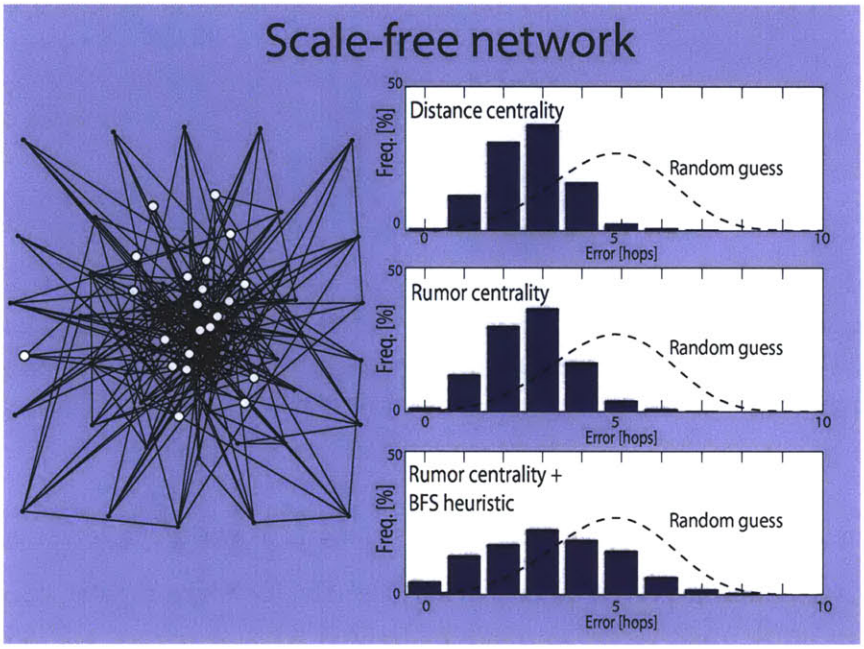

Figure 3-6: Histograms of the error for distance centrality, rumor centrality, and rumor centrality with BFS heuristic estimators on a 400 node rumor network on a scale-free network. The dotted line is a smooth fit of the histogram for randomly guessing the source in the rumor network. An example of a rumor graph (infected nodes in white) is shown on the right.

fit of the histogram for the error from randomly choosing the source from the 400 node rumor network. As with the synthetic networks, the histogram for the random guessing is shifted to the right of the estimator histograms. Thus, on these real networks, the centrality based estimators are a substantial improvement over random guessing.

We see that rumor centrality and distance centrality have similar performance, but for the power grid network, rumor centrality is better able to correctly find the source than distance centrality (3% correct detection versus 0%). For the power grid network, the average ratio of edges to nodes in the 400 node rumor graphs is 4.2, and for the AS network the average ratio is 1.3. Thus, the rumor graphs on the power grid network are less tree-like. Similar to the small-world networks, this may explain why rumor centrality outperforms distance centrality on the power grid network.

The BFS heuristic improves the correct detection probability for the AS network. This is due to the fact that the AS network has many high degree hubs, similar to scale-free networks. However, in the powergrid network, the BFS heuristic spreads out the histogram to higher errors. Again, this may be due to the fact that the powergrid network does not have as much degree heterogeneity as the AS network, and the BFS heuristic is amplifying weak heterogeneities, similar to small-world networks.

3.2.6 Rumor Centrality Precision

For regular trees we have seen that rumor centrality is an exact ML estimator. This means that the rumor centrality of a node is proportional to the likelihood of that node being the source. The actual value of rumor centrality therefore provides us with the confidence that a node is the source. This confidence is best visualized with a heat map plot. Below we plot these types of heat maps for rumor graphs on different network topologies. The rumor centralities are normalized so that the rumor center has value one. Red indicates a high rumor centrality, and blue indicates a low rumor centrality.

The first network we look at is a regular tree of degree three. For this network rumor centrality is an exact ML estimator, so the values here are proportional to the true likelihood. In Figure 3-9 we plot a 300 node rumor graph on this network, with the heat map

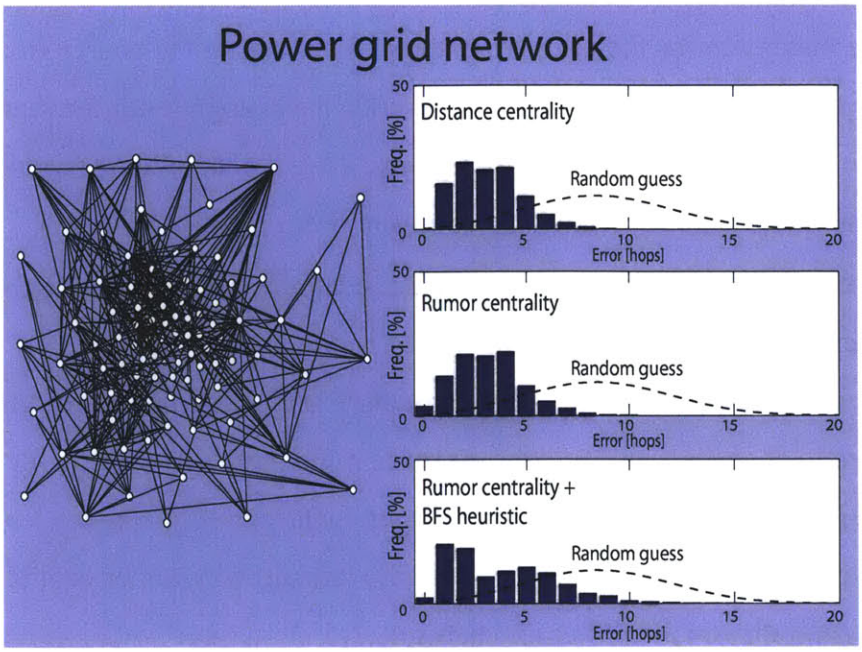

Figure 3-7: Histograms of the error for distance centrality, rumor centrality, and rumor centrality with BFS heuristic estimators on a 400 node rumor network on the U.S. electric power grid network. The dotted line is a smooth fit of the histogram for randomly guessing the source in the rumor network. An example of a rumor graph (infected nodes in white) is shown on the right.

indicating the rumor centrality of the nodes. As can be seen, only a very few nodes have any significant rumor centrality, while the rest of the network is blue. For this particular simulation, the true source happens to equal the rumor center.

Second we look at a 500 node rumor graph on a two-dimensional grid network in Figure 3-9. We have already seen this figure earlier in Chapter 1 in order to demonstrate the precision of rumor centrality. Here we reproduce it to emphasize the fact that rumor centrality does a very effective job at narrowing down the set of likely sources for different network topologies.

3.3 Proofs

Here proofs of the results stated in Section 3.1 are presented. We begin with the proof of Theorem 2 for linear graphs. This proof utilizes a connection of rumor spreading with a random walk. Then we establish results for d-regular trees by connecting rumor spreading

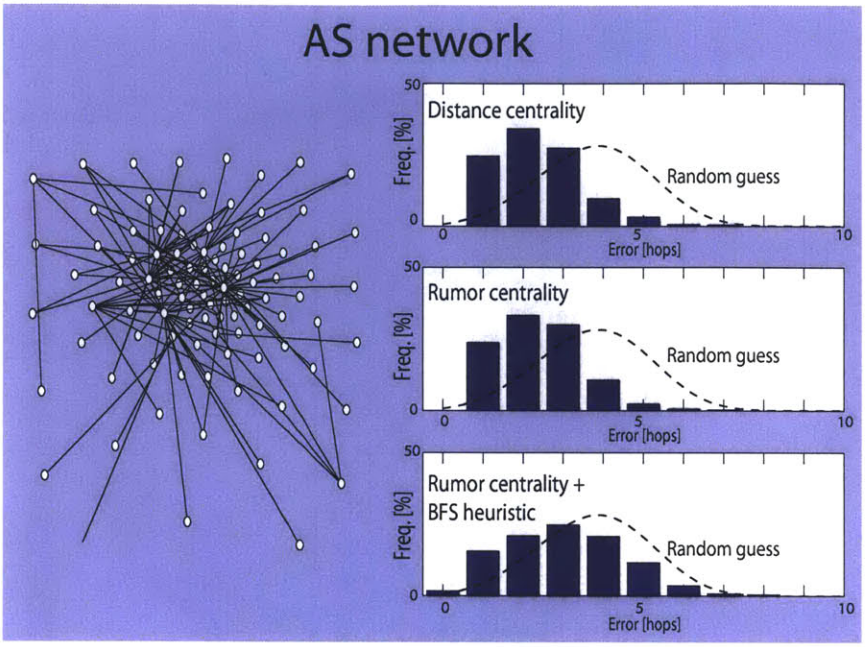

Figure 3-8: Histogram of the error for distance centrality, rumor centrality, and rumor centrality with BFS heuristic estimators on a 400 node rumor network on an Internet autonomous system (AS) network. The dotted line is a smooth fit of the histogram for randomly guessing the source in the rumor network. An example of a rumor graph (infected nodes in white) is shown on the right.

with Polya urn models and branching processes. Later we extend the method to establish results for generic random trees under arbitrary spreading time distributions. Using local-tree like arguments, we prove our results for random regular graphs (Theorem 6) and sparse Erdos-Renyi graphs (Theorem 7). After this, we prove Theorem 5 for geometric trees. For this proof we are able to make use of simpler concentration based techniques due to the polynomial growth of geometric trees.

3.3.1 Proof of Theorem 2: Linear Graphs

We consider the spread of the rumor in a linear graph for a time t starting from a source, say v^*. We shall establish that for a rumor infected graph of $n(t) > 0$ nodes, the probability of the rumor center being equal to v^* decays as $\Theta(1/\sqrt{t})$. Since a line is a regular graph and hence the rumor center is the ML estimator, it follows that the detection probability of any estimator decays as $O(1/\sqrt{t})$. This is because in the absence of any prior information (or uniform prior) the ML estimator minimizes the detection error (cf. see [16]).

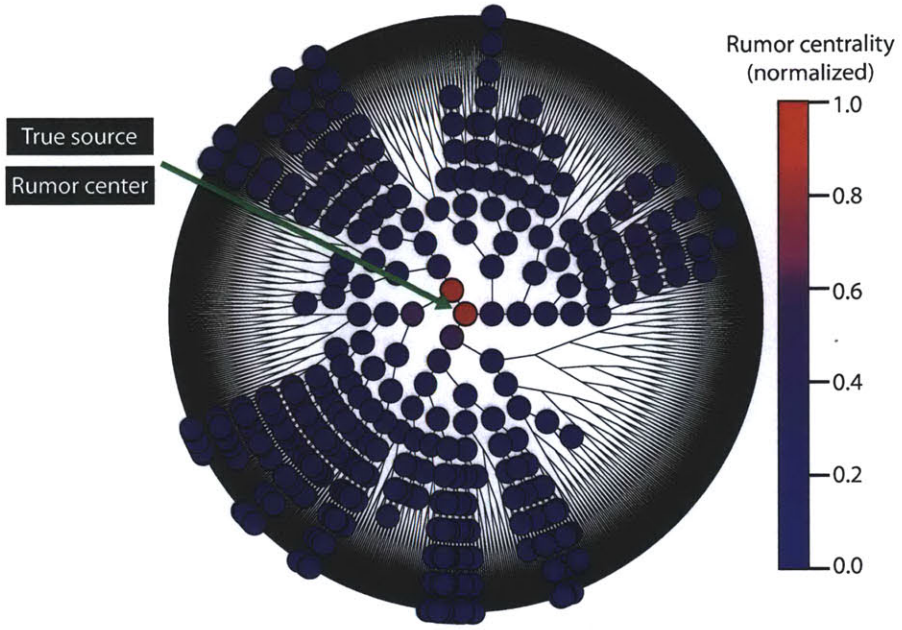

Figure 3-9: Heat map of rumor centrality for a rumor graph on a degree three regular tree.

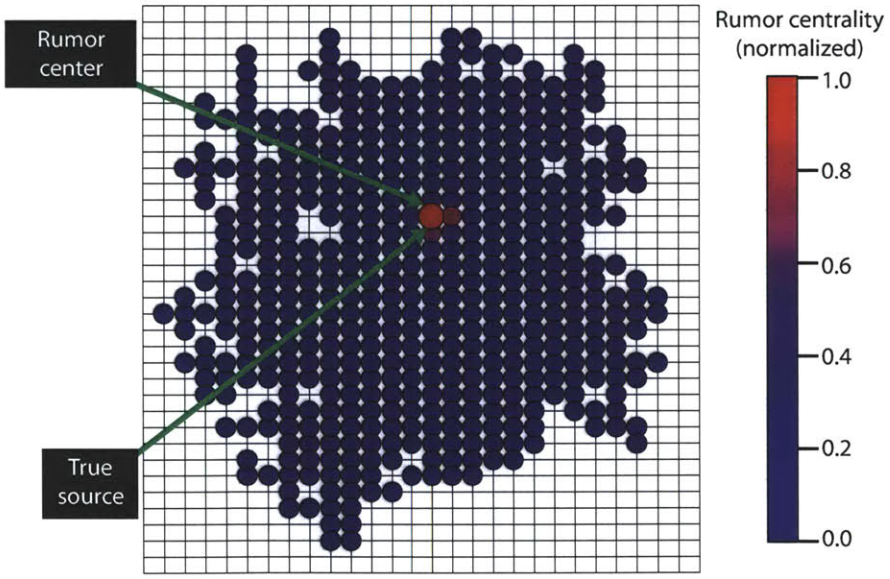

Figure 3-10: Heat map of rumor centrality for a rumor graph on a grid network.

We will first show that with high probability $n(t) = \Theta(t)$. Because of the structure of a linear graph, the rumor spreads in two directions from the source v^*. Let us refer to the number of nodes in these two spreading processes as $n_1(t)$ and $n_2(t)$. In this notation we have $n(t) = n_1(1) + n_2(t) + 1$. These two spreading processes are renewal processes with holding times that are exponentially distributed with parameter $\lambda > 0$. To establish the scaling of $n(t)$ we use Theorem 15, which is proved in Appendix A.2, with $\gamma = t^{-1/3}$ and $\mu = 1/\lambda$. This says that with high probability for large t both $n_1(t)$ and $n_2(t)$ scale as $\lambda t + O\left(t^{2/3}\right) = \Theta(t)$. Therefore, we have that

$$\begin{aligned} n(t) &= n_1(t) + n_2(t) + 1 \\ &= \Theta(t) + \Theta(t) + 1 \\ &= \Theta(t) \end{aligned}$$

Next we will analyze the detection probability in terms of a fixed number of nodes n, and then replace this with $n(t)$ in the end.

Now because of the structure of a linear graph, there are always two nodes which can be infected at any given time: one node to the left of the rumor source and one node to the right of it. Due to the memoryless property of the exponential spreading times, at each infection time one of these nodes is infected with probability $1/2$. Another equivalent way to view this is as a one dimensional random walk on $\mathbb{Z}$ which starts at zero with the first infection (the source) and then moves one step to the left when there is an infection on the left side of the rumor source or one step to the right for a right side infection. Define the value of the step at infection $i \geq 2$ as $X_i \in \{-1, 1\}$. Then X_i is a Bernoulli random variable with parameter $1/2$. The total distance traveled by the random walk after n infections is

$$S_n = \sum_{i=2}^{n} X_i \tag{3.6}$$

which is a binomial random variable with parameters $n - 1$ and $p = 1/2$. We will have correct detection if both subtrees of the rumor source have equal size. This can only happen if there are an odd number of infected nodes. If n is instead even, then there will be two

rumor centers and we will pick the correct one with probability $1/2$. Now, for odd n, we have correct detection if $S_n = 0$ because this means there were an equal number of left and right infections. For even n, we will have correct detection with probability $1/2$ if $S_n = 1$ or $S_n = -1$. To simplify the analysis, we will define $m = n - 1$. For odd n we can write the probability of correct detection as

$$\begin{aligned}\mathbf{P}\left(C_n^1\right) &= \mathbf{P}\left(S_n = 0\right) \\ &= \binom{m}{\frac{m}{2}} \left(\frac{1}{2}\right)^m\end{aligned}$$

This expression can be simplified by using Stirling's approximation for the factorial which is accurate for large m.

$$m! = \Theta\left(\sqrt{2\pi m}\left(\frac{m}{e}\right)^m\right) \tag{3.7}$$

With Stirling's approximation, we obtain

$$\begin{aligned}\mathbf{P}\left(C_n^1\right) &= \frac{m!}{\left(\frac{m}{2}!\right)^2}\left(\frac{1}{2}\right)^m \\ &= \Theta\left(\frac{\sqrt{2\pi m}\left(\frac{m}{e}\right)^m}{\pi m\left(\frac{m}{2e}\right)^m}\left(\frac{1}{2}\right)^m\right) \\ &= \Theta\left(\sqrt{\frac{2}{\pi m}}\right) \\ &= \Theta\left(\frac{1}{\sqrt{n}}\right)\end{aligned}$$

For even n, we obtain the following using a similar analysis.

$$\begin{aligned}
\mathbf{P}\left(C_n^1\right) &= \frac{1}{2}\left(\mathbf{P}\left(S_n = 1\right) + \mathbf{P}\left(S_n = -1\right)\right) \\
&= \frac{1}{2}\binom{m}{\frac{m+1}{2}}\left(\frac{1}{2}\right)^m + \frac{1}{2}\binom{m}{\frac{m-1}{2}}\left(\frac{1}{2}\right)^m \\
&= \Theta\left(\frac{\sqrt{2\pi m}\left(\frac{m}{e}\right)^m}{\pi\sqrt{(m-1)(m+1)}\left(\frac{m+1}{2e}\right)^{m/2}\left(\frac{m-1}{2e}\right)^{m/2}}\left(\frac{1}{2}\right)^m\right) \\
&= \Theta\left(\sqrt{\frac{2m}{\pi\left(m^2-1\right)}}\left(\frac{m}{\sqrt{m^2-1}}\right)^m\right) \\
&= \Theta\left(\frac{1}{\sqrt{n}}\right)
\end{aligned}$$

Now if we replace n with $n(t)$ we obtain

$$\begin{aligned}
\mathbf{P}\left(C_{n(t)}^1\right) &= \Theta\left(\frac{1}{\sqrt{n(t)}}\right) \\
&= \Theta\left(\sqrt{\frac{1}{\Theta\left(t\right)}}\right) \\
&= \Theta\left(\frac{1}{\sqrt{t}}\right)
\end{aligned}$$

Therefore, it follows that the probability of correct detection scales as $\Theta(1/\sqrt{t})$ in a linear graph. This completes the proof of Theorem 2.

3.3.2 Proof of Theorem 3: d-regular trees

Setup, Notations

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an infinite d-regular tree and let the rumor start spreading from a node, say v_1. Without loss of generality, we view it as a randomly generated tree, as described in Section 3.1, with v_1 being the root with d children and all the subsequent nodes with $d-1$ children (hence each node has degree d). We shall be interested in $d \geq 3$. Now suppose the rumor is spread on this tree starting from v_1 as per the SI model with exponential spreading times of rate $\lambda > 0$.

Initially, node v_1 is the only rumor infected node and its d neighbors are potential nodes that can receive the rumor. We will denote the set of nodes that are not yet rumor infected but neighbors of rumor infected nodes as the *rumor boundary*. Thus, initially the rumor boundary consists of the d neighbors of v_1. Under the SI model, each edge has an independent exponential clock of mean $1/\lambda$. The minimum of d independent exponentials of mean $1/\lambda$ is an exponential random variable (of mean $1/\left(d\lambda\right)$) and hence one of the d nodes (chosen uniformly at random) in the rumor boundary gets infected after an exponential amount of time (of mean $1/\left(d\lambda\right)$). Upon this infection, this node gets removed from the boundary but adds it's $d-1$ children to the rumor boundary. That is, each infection effectively adds $d-2$ new nodes to the rumor boundary. In summary, let $Z(t)$ denote the number of nodes in the rumor boundary at time t, then $Z(0)=d$ and $Z(t)$ evolves as follows: each of the $Z(t)$ nodes has an exponential clock of mean $1/\lambda$; when it ticks, it dies and replaces itself with $d-2$ new nodes which in turn start their own independent exponential clocks of mean $1/\lambda$ and so on. The thus described $Z(t)$ is precisely the multi-class Markov branching process (MCMBP): the multi-class comes when we think of the contributions of each of the d sub-tress of the root v_1 to the rumor boundary separately and hence effectively having d branching process each starting at time 0 with initial value equal to 1: let $u_1,\ldots,u_d$ be the children of v_1; let $Z_i(t)$ denote the number of nodes in the rumor boundary that belong to the subtree $T_i(t)$ that is rooted at u_i with $Z_i(0)=1$ for $1\leq i\leq d$; $Z(t)=\sum_{i=1}^{d}Z_i(t)$. With an abuse of notation, let $T_i(t)$ also denote the total number of nodes infected in the subtree rooted at u_i at time t; initially $T_i(0)=0$ for $1\leq i\leq d$. Since each infected node add $d-2$ nodes to the rumor boundary, it can be easily checked that $Z_i(t)=(d-2)T_i(t)+1$ and hence $Z(t)=(d-2)T(t)+d$ with $T(t)$ being the total number of infected nodes at time t (excluding v_1, i.e. $T(0)=0$).

Probability of Correct Detection P $\left(C^1_{n(t)}\right)$

Suppose we observe the rumor infected nodes at some time t which we do not know. That is, we observe the rumor infected graph $G(t)$ which contains the root v_1 and its d infected subtrees $T_i(t)$ for $1\leq i\leq d$. We recall the following result of Proposition 1 that characterizes the rumor center, rephrased using the subtree random variables defined in Section

2.3.1.

$$T_i^v \leq \frac{1}{2}\Big(\sum_{j\in\mathcal{N}(v)} T_j^v\Big), \quad \forall\, i \in \mathcal{N}(v), \tag{3.8}$$

This immediately suggests the characterization of the event that node v_1, the true source, is identified by rumor centrality at time t: v_1 is a rumor center only if $2T_i(t) \leq \sum_{j=1}^{d} T_j(t)$ for all $1 \leq i \leq d$, and if the inequality is strict then it is indeed the rumor center. Now if $n(t)$ is the total number of infected nodes at time t, then as per our earlier notation, $C^1_{n(t)}$ is the event of the correct detection at time t. Let $E_i = \{2T_i(t) < \sum_{j=1}^{d} T_j(t)\}$ and $F_i = \{2T_i(t) \leq \sum_{j=1}^{d} T_j(t)\}$. Then,

$$\mathbf{P}\Big(C^1_{n(t)}\Big) \geq \mathbf{P}\Big(\cap_{i=1}^{d} E_i\Big) = 1 - \mathbf{P}\Big(\cup_{i=1}^{d} E_i^c\Big) \overset{(a)}{\geq} 1 - \sum_{i=1}^{d} \mathbf{P}\Big(E_i^c\Big) \overset{(b)}{=} 1 - d\mathbf{P}\Big(E_1^c\Big). \tag{3.9}$$

Above, (a) follows from the union bound of events and (b) from symmetry. Similarly, we have

$$\mathbf{P}\Big(C^1_{n(t)}\Big) \leq \mathbf{P}\Big(\cap_{i=1}^{d} F_i\Big) = 1 - \mathbf{P}\Big(\cup_{i=1}^{d} F_i^c\Big) \overset{(a)}{=} 1 - \sum_{i=1}^{d} \mathbf{P}\Big(F_i^c\Big) \overset{(b)}{=} 1 - d\mathbf{P}\Big(F_1^c\Big). \tag{3.10}$$

Above, (a) follows because events $F_1^c, \ldots, F_d^c$ are disjoint and (b) from symmetry. Therefore, the probability of correct detection boils down to evaluating $\mathbf{P}(E_1^c)$ and $\mathbf{P}(F_1^c)$ which, as we shall see, will coincide with each other as $t \to \infty$ (equivalently, $n(t) \to \infty$). Therefore, the bounds of (3.9) and (3.10) will provide the exact evaluation of the correct detection probability as $t \to \infty$.

$\mathbf{P}(E_1^c)$, $\mathbf{P}(F_1^c)$ and Polya's Urn

Effectively, the interest is in the ratio $T_1(t)/(\sum_{i=1}^{d} T_i(t))$ especially as $t \to \infty$ (implicitly we are assuming that this ratio is well defined for a given t or else by definition there is only one node infected which will be v_1, the true source). It can be easily verified that as

$t \to \infty$, $T_i(t) \to \infty$ for all i almost surely and hence $Z_i(t) = (d-2)T_i(t)+1$ goes to ∞ as well. Therefore, it is sufficient to study the ratio $Z_1(t)/(\sum_{j=1}^{d} Z_j(t))$ as $t \to \infty$ since we shall find that this ratio converges to a random variable with density on $[0,1]$. In summary, if we establish that the ratio $Z_1(t)/(\sum_{j=1}^{d} Z_j(t))$ converges in distribution on $[0,1]$ with a well defined density, then it immediately follows that $\mathbf{P}(E_1^c) = \mathbf{P}(F_1^c)$ as $t \to \infty$ and we can use $Z_1(t)/(\sum_{j=1}^{d} Z_j(t))$ in place of $T_1(t)/(\sum_{j=1}^{d} T_j(t))$ to evaluate $\mathbf{P}(E_1^c)$ as $t \to \infty$.

With these in mind, let us study the ratio $Z_1(t)/(\sum_{j=1}^{d} Z_j(t))$. For this, it is instructive to view the simultaneous evolution of $(Z_1(t), Z_{\neq 1}(t))$ $(Z_{\neq 1}(t) \stackrel{\Delta}{=} \sum_{j=2}^{d} Z_j(t))$ as that induced by the standard, discrete time, Polya's urn: initially, $\tau_0 = 0$ and there is one ball of type 1 representing $Z_1(\tau_0) = 1$ and $d-1$ balls of type 2 representing $Z_{\neq 1}(\tau_0) = d-1$ in a given *urn*; the n^{th} event happens at time τ_n when one of the $Z_1(\tau_{n-1}) + Z_{\neq 1}(\tau_{n-1})$ $(= d + (n-1)(d-2))$ balls chosen uniformly at random is thrown out of the urn and new $d-2$ balls of its type are added to the urn. If we set $\tau_n - \tau_{n-1}$ equal to an exponential random variable with mean $1/(\lambda(d + (n-1)(d-2)))$, then it is easy to check that the fraction of balls of type 1 is identical in law to that of $Z_1(t)/(\sum_{j=1}^{d} Z_j(t))$ (here we are using the *memoryless* property of exponential random variables crucially). Therefore, for our purposes, it is sufficient to study the limit law of fraction of balls of type 1 under this Polya's urn model.

It is well understood that the fraction of balls of type 1 at time τ_n, which is equal to $Z_1(\tau_n)/(Z_1(\tau_n)+Z_{\neq 1}(\tau_n))$, is a martingale with value in $[0,1]$. By the standard martingale convergence theorem, it converges to a well defined random variable almost surely. Further, the law of this limiting random variable in our particular case turns out to be the Beta distribution with parameters $a = 1/(d-2)$ and $b = (d-1)/(d-2)$. (See chapter on generalized Polya's urn in [7], for example, for proof details of this statement; a more general version of this result will be utilized later in the context of general random trees).

From the above discussion, we conclude that the ratio $Z_1(t)/(\sum_{i=1}^{d} Z_i(t))$ converges to a Beta random variable with parameters $a = 1/(d-2)$ and $b = (d-1)/(d-2)$. Since the Beta distribution has density on $[0,1]$, from the above discussion it follows that as $t \to \infty$

(equivalently, $n(t) \to \infty$), $\mathbf{P}(E_1^c) = \mathbf{P}(F_1^c)$ and hence from (3.9), (3.10)

$$\lim_{t\to\infty} \mathbf{P}\Big(C^1_{n(t)}\Big) = 1 - d\Big(1 - I_{1/2}\Big(\frac{1}{d-2}, 1 + \frac{1}{d-2}\Big)\Big), \tag{3.11}$$

where recall that $I_{1/2}(a, b)$ is the probability that a Beta random variable with parameters a and b takes value in $[0, 1/2]$. Note that this establishes the result of Theorem 3 for $k = 1$ in (3.2).

Probability of $C^k_{n(t)}$

Thus far we have established Theorem 3 for $k = 1$, the probability of the rumor center being the true source. The probability of the event $C^k_{n(t)}$ (the kth infected node being the rumor center) is evaluated in an almost identical manner with a minor difference. For this reason, we present an abridged version of the proof.

Let $v_k, k \geq 2$ be the k^{th} infected node when the rumor starts from v_1. We will evaluate the probability of identifying v_k as the rumor center. As before, let us suppose we observe the infected tree G at time t with $n(t) \geq k$ nodes. Let $w_1, \ldots, w_d$ be the d neighbors of v_k with respect to $\mathcal{G}$. Note that one of the neighbors of v_k is infected before it. Equivalently, viewing the tree being rooted at v_1, this neighbor is the 'parent' of v_k. We shall denote it by w_1 and let $w_2, \ldots, w_d$ be the $d-1$ 'children' of v_k. Let $T^k_i(t)$ be the subtrees of G rooted at w_i if we imagine v_k as the root of G: therefore, $T^k_1(t)$ is rooted at w_1 and includes $v_1, \ldots, v_{k-1}$; $T^k_i(t)$ are rooted at w_i for $2 \leq i \leq d$ and contain nodes in G that are away from v_k; none of the $T^k_i(t)$ for $1 \leq i \leq d$ include v_k. By definition $T^k_1(t)$ is never empty, but $T^k_i(t)$ can be empty if w_i is not infected, for $2 \leq i \leq d$. As before, with abuse of notation, we shall denote $T^k_i(t)$ as the size of the subtree as well. As per Proposition 1, v_k is identified as a rumor center if and only if all of its d subtrees are balanced, i.e.

$$2T^k_i(t) \leq \sum_{j=1}^{d} T^k_j(t), \ \ \forall\, 1 \leq i \leq d. \tag{3.12}$$

Therefore, as before,

$$\mathbf{P}\Big(C^k_{n(t)}\Big) \geq \mathbf{P}\Big(\cap_{i=1}^d E_i\Big) = 1 - \mathbf{P}\Big(\cup_{i=1}^d E_i^c\Big) \geq 1 - \sum_{i=1}^d \mathbf{P}\Big(E_i^c\Big), \tag{3.13}$$

and

$$\mathbf{P}\Big(C^k_{n(t)}\Big) \leq \mathbf{P}\Big(\cap_{i=1}^d F_i\Big) = 1 - \mathbf{P}\Big(\cup_{i=1}^d F_i^c\Big) = 1 - \sum_{i=1}^d \mathbf{P}\Big(F_i^c\Big). \tag{3.14}$$

Above, $E_i = \{2T_i^k(t) < \sum_{j=1}^d T_j^k(t)\}$ and $F_i = \{2T_i^k(t) \leq \sum_{j=1}^d T_j^k(t)\}$.

To evaluate these probabilities, as before, we shall study the evolution of the rumor boundaries in each tree. Unlike the earlier situation where all events had the same probability, we need to be a bit careful for $k \geq 2$. Specifically, note that the time when the kth node, v_k gets infected, the tree $T_1^k(\cdot)$ has size $k-1$ and it's rumor boundary, $Z_1^k(\cdot)$, is of size $(d-2)(k-1)+1$. But for $2 \leq i \leq d$, $T_i^k(\cdot)$ is empty and has its rumor boundary, $Z_i^k(\cdot)$, of size 1. Beyond this difference in initial values, the evolution is the same as before and therefore the limiting laws of the ratios of the sizes of rumor boundaries can be evaluated as before as the limit of the fraction of balls of a given type in a Polya's urn model with different initial number of balls of two types. Specifically, to evaluate E_1^c (and F_1^c), we consider a Polya's urn in which we start with $(d-2)(k-1)+1$ balls of type 1 (corresponding to $Z_1^k(\cdot)$) and $d-1$ balls of type 2 (corresponding to $\sum_{j=2}^d Z_j^k(\cdot)$). With these initial conditions, the limit law of fraction of balls of type 1 turns out to be (see [7] for details) a Beta distribution with parameters $a = ((d-2)(k-1)+1)/(d-2) = (k-1)+1/(d-2)$ and $b = (d-1)/(d-2) = 1 + 1/(d-2)$. In summary, the fraction of balls of type 1 equals the ratio $Z_1^k(t)/(\sum_{j=1}^k Z_j^k(t))$ which is equal to the ratio of $T_1^k(t)/(\sum_{j=1}^d T_j^k(t))$ as $t \to \infty$, since these quantities go to infinity as $t \to \infty$ and $Z_i^k(t)$ is linearly related to $T_i^k(t)$ for $1 \leq i \leq d$. Since the limit law of the fraction of balls of type 1 has density on $[0,1]$, we conclude that as $t \to \infty$

$$\mathbf{P}\Big(E_1^c\Big) = \mathbf{P}\Big(F_1^c\Big) = 1 - I_{1/2}\Big(k-1+\frac{1}{d-2},\, 1+\frac{1}{d-2}\Big). \tag{3.15}$$

To evaluate E_i^c (and F_i^c) for $2 \leq i \leq d$, in the corresponding Polya's urn model, we start with 1 ball of type 1 and $k(d-2)+1$ balls of type 2. Therefore, using an identical sequence of arguments, we obtain that for $2 \leq i \leq d$ as $t \to \infty$,

$$\mathbf{P}\Big(E_i^c\Big) \;=\; \mathbf{P}\Big(F_i^c\Big) \;=\; 1 - I_{1/2}\Big(\frac{1}{d-2},\, k + \frac{1}{d-2}\Big). \tag{3.16}$$

From (3.13)-(3.16), it follows that

$$\begin{aligned}\lim_{t\to\infty} \mathbf{P}\Big(C_{n(t)}^k\Big) =& I_{1/2}\Big(k-1+\frac{1}{d-2},\, 1+\frac{1}{d-2}\Big) \\ &+ (d-1)\Big(I_{1/2}\Big(\frac{1}{d-2},\, k+\frac{1}{d-2}\Big) - 1\Big).\end{aligned} \tag{3.17}$$

This establishes (3.2) for all k and hence completes the proof of Theorem 3.

3.3.3 Proof of Theorem 4: Random Trees

The goal is to establish that there is a strictly positive probability of detecting the source correctly as the rumor center when the rumor starts at the root of a generic randomly generated tree: this is with respect to the joint probability distribution induced by the tree construction and the SI rumor spreading model with independent spreading times with distribution $F(t)$. We establish this result along the lines of the proof for Theorem 2. However, it requires additional details due to the irregularity and randomness in the node degrees in the tree and the arbitrary spreading time distribution $F(t)$.

Notations

We quickly recall some notations. To start with, as before let v_1 be the root node of the tree. It has η_0 children distributed as per $\mathcal{D}_0$. By assumption of Theorem 4, $\mathbf{P}(\eta_0 \geq 3) > 0$. We shall condition on this positive probability event that $\eta_0 \geq 3$ and let $d = \eta_0$ for the remainder of the proof. Let $u_1, \ldots, u_d$ be the d children of v_1. The random tree $\mathcal{G}$ is constructed by adding a random number of children to $u_1, \ldots, u_d$ recursively as per distribution $\mathcal{D}$ as explained in Section 3.1.3. The rumor starts at v_1 at time 0 and spreads as per SI model on $\mathcal{G}$ with spreading times with distribution $F(t)$.

Let G be the sub-tree of $\mathcal{G}$ that is infected at time t, let $n(t)$ be the number of nodes in G at time t, and let $T_i(t)$ denote the subtree of G rooted at node u_i at time t, for $1 \leq i \leq d$. We shall abuse notation as before by using $T_i(t)$ as the subtree size as well. By definition $T_i(0) = 0$ for $1 \leq i \leq d$. Let $Z_i(t)$ denote the size of the rumor boundary of $T_i(t)$; initially $Z_i(0) = 1$.

Since we are interested in evaluating the probability of detection with respect to the joint distribution over the tree generation and SI spreading model, we model the evolution of $Z_i(\cdot)$ and $T_i(\cdot)$ as follows. Each node in the rumor boundary has its own independent clock with distribution $F(t)$. When the clock of a particular node ticks, it dies and it is replaced by η new nodes where η is an independent random variable distributed as per $\mathcal{D}$. If the node that died belonged to $Z_i(\cdot)$ (i.e. tree $T_i(\cdot)$), then the new nodes are added to $Z_i(\cdot)$. Therefore, each $Z_i(\cdot)$ is a general time branching process with $Z_i(0) = 1$ for all $1 \leq i \leq d$. Now we recall some facts about branching processes that will be useful for establishing non-triviality of the probability of correct detection for such randomly generated trees.

First we define what is known as the Malthusian parameter for the branching process.

Definition 3. *[7, (pg. 146)] The Malthusian parameter for a constant γ and a distribution F is the root, provided it exists, of the equation*

$$\gamma \int_0^\infty e^{-\alpha y} dF(y) = 1. \tag{3.18}$$

We denote it by $\alpha = \alpha(\gamma, F)$.

For any $\gamma > 1$ the Malthusian parameter always exists. What the Malthusian parameter does is characterize the growth rate of a general time branching process. Consider for example a Markov branching process with exponential spreading times of rate λ and let $\mathbb{E}[\eta] = m > 1$. The spreading time distribution is $F(t) = 1 - e^{-\lambda t}$ Then the Malthusian

parameter for this process $\alpha(m, F)$ is given by

$$\begin{aligned} m\int_0^\infty e^{-\alpha y}\lambda e^{-\lambda y}dy &= 1 \\ m\frac{\lambda}{\alpha+\lambda} &= 1 \\ \lambda(m-1) &= \alpha. \end{aligned}$$

This is exactly the growth rate for the mean of a Markov branching process [7]. The Malthusian parameter also describes the growth of the mean of general time branching processes, as the following theorem shows.

Theorem 8. *[7, Theorem 3A (pg. 152)] Let $Z(\cdot)$ be a continuous time branching process as described above: $Z(0) = 1$; each node in $Z(t)$ has an independent clock with distribution $F(t)$ and it dies upon the tick of the clock; upon death of a node, it is replaced by η new nodes chosen independently for each node, and so on. If $\mathbb{E}[\eta] = m > 1$ then as $t \to \infty$,*

$$\lim_{t\to\infty} \frac{\mathbb{E}\left[Z(t)\right]}{c'e^{\alpha t}} = 1$$

where α is the Malthusian parameter for (m, F) and

$$c' = \frac{m-1}{\alpha m^2 \int_0^\infty y e^{-\alpha y} dF(y)}$$

This theorem says that the mean of the branching process $Z(t)$ has exponential growth with rate given by the Malthusian parameter $\alpha(m, F)$. To see an application of this theorem, consider again the Markov branching process with exponential spreading times with rate λ. Then we have already seen that the Malthusian parameter $\alpha = \lambda(\mathbb{E}[\eta] - 1)$. The

constant c' evaluates to

$$\begin{aligned} c' &= \frac{m-1}{\alpha m^2 \int_0^\infty y e^{-\alpha y} \lambda e^{-\lambda y} dy} \\ &= \frac{(m-1)(\alpha+\lambda)^2}{\lambda \alpha m^2} \\ &= \frac{(m-1)(\lambda(m-1)+\lambda)^2}{\lambda^2 (m-1) m^2} \\ &= \frac{(m-1)\lambda^2 m^2}{\lambda^2 (m-1) m^2} \\ &= 1 \end{aligned}$$

Therefore, we have that for the Markov branching process, $\mathbb{E}[Z(t)] = e^{\lambda(\mathbb{E}[\eta]-1)}$, a well known result [7].

For the general time branching process, we have the following result.

Theorem 9. *[7, Theorem 2 (pg. 172)] Let $Z(\cdot)$ be a continuous time branching process as described above: $Z(0) = 1$; each node in $Z(t)$ has an independent clock with distribution $F(t)$ and it dies upon the tick of the clock; upon death of a node, it is replaced by η new nodes chosen independently for each node, and so on. Let α be the Malthusian parameter for $(\mathbb{E}[\eta], F)$ and c' be defined as in Theorem 8. If $\mathbb{E}[\eta] > 1$ and $\mathbb{E}[\eta \log \eta] < \infty$,*

$$Z(t)/c'e^{\alpha t} \to W, \qquad \textit{in distribution},$$

where W is such that

$$\mathbf{P}\big(W = 0\big) = q, \tag{3.19}$$

$$\mathbf{P}\big(W \in (x_1, x_2)\big) = \int_{x_1}^{x_2} w(y)dy, \quad \textit{for} \quad 0 < x_1 < x_2 < \infty, \tag{3.20}$$

where $q \in (0,1)$ is the smallest root of the equation $f_\eta(s) = s$ in $[0,1]$ and $w(\cdot)$ is absolutely continuous with respect to the Lebesgue measure so that $\int_0^\infty w(y)dy = 1-q$. Here $f_\eta(s) = \sum_{k=0}^\infty s^k \mathbf{P}(\eta = k)$.

As we will see, this theorem will be key to proving Theorem 4.

Correct Detection

Recall from the proof for regular trees, the probability of the event of correct detection at time t, $C^1_{n(t)}$, is lower bounded as

$$\mathbf{P}\Big(C^1_{n(t)}\Big) \geq \mathbf{P}\Big(\cap_{i=1}^d \Big\{2T_i(t) < \sum_{j=1}^d T_j(t)\Big\}\Big). \tag{3.21}$$

We shall establish a non-trivial lower bound for the right hand side of (3.21) using Theorem 9. This will be done in the two steps: (i) identify an event $\mathcal{E} \subset \cap_{i=1}^d \Big\{2Z_i(t) < \sum_{j=1}^d Z_j(t)\Big\}$ and then establish non-trivial lower bound on $\mathcal{E}$ using Theorem 9; (ii) establish that as $t \to \infty$, $\mathcal{E} \subset \cap_{i=1}^d \Big\{2T_i(t) < \sum_{j=1}^d T_j(t)\Big\}$. This will immediately imply that $\mathbf{P}(\mathcal{E})$ is a non-trivial lower bound on $\mathbf{P}\Big(C^1_{n(t)}\Big)$.

A Non-Trivial Event

The event $2Z_i(t) < \sum_{j=1}^d Z_j(t)$ is equivalent to $\frac{Z_i(t)c'^{-1}e^{-\alpha t}}{\sum_{j=1}^d Z_j(t)c'^{-1}e^{-\alpha t}} < 1/2$, with the Malthusian parameter α and c' defined as in Theorem 8. For any $x > 0$, define event $\mathcal{E}(x)$ as

$$\mathcal{E}(x) = \cap_{i=1}^d \Big\{ Z_i(t)c'^{-1}e^{-\alpha t} \in (x, (d-1)x)\Big\}. \tag{3.22}$$

Under event $\mathcal{E}(x)$, since each $Z_i(t)c'^{-1}e^{-\alpha t}$ is at least x and at most $(d-1)x$, it follows immediately that

$$\mathcal{E}(x) \subset \Big\{2\max_{i=1}^d Z_i(t) < \sum_{j=1}^d Z_j(t)\Big\} \;=\; \cap_{i=1}^d \Big\{2Z_i(t) < \sum_{j=1}^d Z_j(t)\Big\}. \tag{3.23}$$

Now $Z_i(\cdot)$ are independent across $1 \leq i \leq d$. By Theorem 9 it follows that $Z_i(t)c'^{-1}e^{-\alpha t}$ converges to W_i (because $\mathbb{E}[\eta^2] < \infty$ and hence $\mathbb{E}[\eta \log \eta] < \infty$) which are independent across i and W_i are distributed as per (3.19)-(3.20). From this it follows that as $t \to \infty$,

$$\mathbf{P}\Big(\mathcal{E}(x)\Big) = p(x)^d, \qquad \text{where} \quad p(x) \stackrel{\triangle}{=} \int_x^{(d-1)x} w(y)dy \;>\; 0. \tag{3.24}$$

From the above discussion, it follows that as $t \to \infty$

$$\mathbf{P}\Big(\cap_{i=1}^{d} \Big\{ 2Z_i(t) < \sum_{j=1}^{d} Z_j(t) \Big\} \Big) \geq \Big(\sup_{x>0} p(x) \Big)^d > 0. \tag{3.25}$$

Equivalence of Boundary and Tree Processes

For regular trees, the fraction $Z_i(t)/(\sum_{j=1}^{d} Z_j(t))$ and $T_i(t)/(\sum_{j=1}^{d} T_j(t))$ converge to the same limit primarily because each of them converge to ∞ and $Z_i(t)$ and $T_i(t)$ are related linearly. Such a relation does not exist for the random tree. However, with an additional, careful argument we shall establish the same fact for random trees as well under the event $\mathcal{E}(x)$ for $x > 0$.

To that end, for $1 \leq i \leq d$ and $n \geq 0$, define

$$M_{n+1}^i = M_n^i + Q_{n+1}^i(\eta_{n+1} - \mu), \tag{3.26}$$

where Q_{n+1}^i is the probability that $n+1$st rumor infected node is added to the tree $T_i(\cdot)$, η_{n+1} is the number of children of this infected node, and $\mu = \mathbb{E}[\eta]$. We define $M_0^i = 0$. It can be checked that M_n^i is a martingale with respect to the filtration $\mathcal{F}_n$, where $\mathcal{F}_n$ contains all the information about the part of the graph $\mathcal{G}$ to which the rumor has spread including the rumor boundary. Now M_n^i is a martingale with respect to $\mathcal{F}_n$ because, (a) Q_{n+1}^i is a binary random variable with its probability of being 1 determined by $\mathcal{F}_n$, and (b) η_{n+1} is independent of $\mathcal{F}_n$ and distributed as per $\mathcal{D}$. Now $|M_{n+1}^i - M_n^i| \leq \eta_{n+1}$ and η_{n+1} has a well defined mean and finite second moment. By property of the martingale, therefore it follows that

$$\mathbb{E}[(M_n^i)^2] \leq n\mathbb{E}[\eta^2].$$

Therefore, a straightforward application of Chebychev's inequality will lead to the following 'weak law of large numbers':

$$\frac{1}{n} M_n^i \to \mathbb{E}[M_0^i] = 0, \qquad \text{in probability.} \tag{3.27}$$

Now under event $\mathcal{E}(x)$ (for any $x > 0$), $Z_i(t)$ scales like $e^{\alpha t}$ for all $1 \leq i \leq d$. Therefore,

it can be checked that $T_i(t)$ also scales like $e^{\alpha t}$ (since $Z_i(t)$ represents the 'rate' at which $T_i(t)$ is growing). Precisely, we have that

$$\begin{aligned}\mathbb{E}\left[Z_i(t)\right] &= 1 + \mathbb{E}\left[\sum_{l \in T_i(t)} \eta_l\right] \\ &= \left(1 + \mathbb{E}\left[T_i(t)\right]\right)\mu\end{aligned}$$

Now using Theorem 8, we have that

$$\begin{aligned}\lim_{t\to\infty} \frac{\mathbb{E}\left[T_i(t)\right]}{c'e^{\alpha t}} &= \frac{1}{\mu} \lim_{t\to\infty} \frac{\mathbb{E}\left[Z_i(t)\right]}{c'e^{\alpha t}} - \frac{1}{c'e^{\alpha t}} \\ &= \frac{1}{\mu}\end{aligned}$$

Therefore, we see that the means of both $Z_i(t)$ and $T_i(t)$ grow as $e^{\alpha t}$.

Since d is finite, it further follows that $n(t)$, the number of rumor infected nodes till time t, or equivalently $\sum_{i=1}^{d} T_i(t)$ scales like $e^{\alpha t}$. Using these facts under event $\mathcal{E}(x)$ and an application of (3.27) with n replaced by $n(t)$ (and taking $t \to \infty$ or equivalently $n(t) \to \infty$), we obtain that for $1 \leq i \leq d$ as $t \to \infty$,

$$\frac{T_i(t)}{n(t)}\left(\frac{1}{T_i(t)} M^i_{n(t)}\right) \to 0. \tag{3.28}$$

Since under event $\mathcal{E}(x)$, $T_i(t)/n(t)$ remains bounded away from 0 as $t \to \infty$, from (3.28), it follows that

$$\frac{1}{T_i(t)} M^i_{n(t)} \to 0. \tag{3.29}$$

But

$$\begin{aligned}M^i_{n(t)} &= \sum_{j \in T_i(t)} (\eta_j - \mu) \\ &= T_i(t)\Big(\frac{1}{T_i(t)} \sum_{j \in T_i(t)} \eta_j - \mu\Big).\end{aligned} \tag{3.30}$$

From (3.29) and (3.30), it follows that under event $\mathcal{E}(x),\ x > 0$ for $1 \leq i \leq d$, as $t \to \infty$,

$$\frac{1}{T_i(t)} \sum_{j \in T_i(t)} \eta_j \to \mu. \tag{3.31}$$

Now we are ready to conclude the proof of Theorem 4 by establishing that under the event $\mathcal{E}(x),\ x > 0$, $Z_i(t)/(\sum_{j=1}^d Z_j(t))$ and $T_i(t)/(\sum_{j=1}^d T_j(t))$ converge to the same quantity. To that end, observe that for $1 \leq i \leq d$,

$$\begin{aligned} \frac{Z_i(t)}{\sum_{j=1}^d Z_j(t)} &= \frac{1 + \sum_{\ell \in T_i(t)} \eta_\ell}{d + \sum_{\ell'=1}^{n(t)} \eta_{\ell'}} \\ &= \frac{T_i(t)}{n(t)} \frac{\frac{1}{T_i(t)} + \frac{1}{T_i(t)} \left(\sum_{\ell \in T_i(t)} \eta_\ell \right)}{\frac{d}{n(t)} + \frac{1}{n(t)} \left(\sum_{\ell'=1}^{n(t)} \eta_{\ell'} \right)}. \end{aligned} \tag{3.32}$$

Now as $t \to \infty$, under the event $\mathcal{E}(x),\ x > 0$, the right most term in (3.32) goes to 1 since the numerator and denominator both go to μ due to $T_i(t), n(t) \to \infty$, (3.31) and the standard Law of Large Numbers. This concludes that under event $\mathcal{E}(x)$, the ratio $Z_i(t)/(\sum_{j=1}^d Z_j(t))$ and $T_i(t)/(\sum_{j=1}^d T_j(t))$ are asymptotically equal as $t \to \infty$. Therefore, from (3.25) and the fact that the initial conditioned event $\mathbf{P}(\eta_0 \geq 3)$ has strictly positive probability, it follows that

$$\liminf_{t \to \infty} \mathbf{P}\left(C^1_{n(t)}\right) > 0.$$

This concludes the proof of Theorem 4.

3.3.4 Proof of Theorem 6: Random Regular Graph

Here we establish Theorem 6 regarding the effectiveness of rumor centrality on d-regular random graphs. This is done by recalling the known fact that the local (not too large) neighborhood of a randomly chosen node in a random d-regular graph is a d-regular tree. Therefore, Theorem 6 is an immediate application of Theorem 3.

Formally, we shall establish the result as follows: we produce a random d-regular graph $\mathcal{G}$ of m nodes as per a (randomized) constructive procedure. The rumor is spread on this

graph as per the SI model with exponential spreading times of rate $\lambda > 0$ starting from a randomly chosen node till time t (so that $n(t)$ is not too large compared to m). The rumor infected subgraph G of $\mathcal{G}$ will be shown to be a d-regular tree with high probability. Therefore, the rumor centrality results about regular trees from Theorem 2 apply immediately and establish Theorem 3.

We start by describing the randomized construction of the d-regular graph $\mathcal{G}$ known as the configuration model [19]. To construct an m node random d-regular graph, create m sets v_1,v_2,...v_m each of which contains d points. Each set corresponds to a node in the resulting graph and d points in a set correspond to the degree of that node. Construct a perfect matching or pairing of these md points (assume that such is feasible by choosing appropriate, say even, m). If the points in a pairing are from sets v_i and v_j, then there is an edge between nodes v_i and v_j in the graph $\mathcal{G}$. Not every pairing will correspond to a graph because there may be parallel edges between the same nodes or self-loops. Those pairing which do not contain any self-loops or parallel edges are referred to as simple. It has been shown that the probability of a pairing being simple is asymptotically (in m) $exp\left(\frac{1-d^2}{4}\right)$ [15].

One way to generate random pairing of dm elements in the above construction is as follows: order them arbitrarily and start matching them sequentially by choosing, among the unpaired elements, an element uniformly at random. We shall use this 'arbitrary' nature of ordering to our advantage in establishing that the rumor infected graph G is a tree with high probability as long as the number of nodes in G, n is such that $n = o(\sqrt{m})$.

To that end, we start revealing the construction of the random graph along with the rumor spreading (since we are interested in establishing results with respect to the joint probability distribution). Let the rumor start from node v_1. For the random graph construction, we take the d points corresponding to the node v_1 and connect them to d of the remaining $md - d$ points uniformly at random (sequentially). This leads to the determination of the immediate d neighbors of v_1. Next, the rumor spreads to one of these d nodes and subsequently the infected node's d neighbors are chosen by selecting d more nodes, and so on.

Suppose we let the rumor infection process run till n nodes get infected in the above

manner. Simultaneously, we have revealed (or constructed) the neighborhood of these n nodes starting from v_1 in the random d-regular graph. Beyond this, continue the construction of the d-regular graph till all m nodes are matched. As per the result of [15], for all m large enough, the probability of the event that the resulting graph is valid, denoted by S, is at least $\frac{1}{2}\exp\left(\frac{1-d^2}{4}\right)$ for all m large enough. Let P_n denote the event that the rumor graph G of n nodes is a tree. Our interest is in the probability of P_n conditioned on the event S.

$$\mathbf{P}(P_n|S) \geq 1 - \frac{\mathbf{P}(P_n^c)}{\mathbf{P}(S)}.$$

We shall establish that for $n = o(\sqrt{m})$, $\mathbf{P}(P_n^c) \to 0$ as $m \to \infty$. Since $\mathbf{P}(S)$ is at least $\frac{1}{2}\exp\left(\frac{1-d^2}{4}\right)$, it will follow that $\mathbf{P}(P_n|S) \to 1$. This will conclude the proof of Theorem 6.

To evaluate $\mathbf{P}(P_n^c)$, recall the simultaneous rumor spreading and construction of the regular graph. Initially, when $n = 1$, there is exactly one infected node and d of its boundary nodes. When the second node is added, it will effectively add $d-2$ nodes to the boundary. In general, for G to remain a tree when the $k+1$st node gets infected (assuming the G has remained a tree for the first k nodes), it must not select its d neighbors from any of the k nodes of G nor their $d + (k-1)(d-2)$ neighbors that do not belong to G. From this, we conclude that

$$\begin{aligned}\mathbf{P}\left(P_{k+1}|P_k\right) &\geq \left(1 - \frac{((k+1)(d-1)+2)(d-1)}{dm - ((k+1)(d-1)+2)(d-1)}\right)^d \\ &\geq \left(1 - \frac{2d^2(k+1)}{dm}\right)^d = \left(1 - \frac{2d(k+1)}{m}\right)^d\end{aligned}$$

as long as $k = o(m)$. Subsequently,

$$\begin{aligned}\mathbf{P}\left(P_n\right) &= \prod_{k=1}^{n} \mathbf{P}\left(P_k|P_{k-1}\right) \\ &\geq \left(1 - \frac{2dn}{m}\right)^{nd}\end{aligned}$$

Since $n = o(\sqrt{m})$ and the $\mathbf{P}(P_n)$ decreases with n, we have

$$\mathbf{P}\Big(P_n\Big) \geq 1 - o(1).$$

That is, $\mathbf{P}(P_n^c) \to 0$ as $m \to \infty$. This completes the proof of G being a tree with high probability and hence the proof of Theorem 3.

3.3.5 Proof of Theorem 7: Erdos-Renyi Graph

The proof is very similar to that for random regular graphs: establish that in an Erdos-Renyi graph with m nodes with edge probability $p = c/m$, $c > 1$, if the rumor spreads to $n = o(\sqrt{m})$ nodes, then the rumor infected graph is like a randomly generated tree with the degree distribution of nodes in this tree being Poisson with parameter c for m large enough. Therefore, by invoking Theorem 4, it follows that there is a strictly positive probability of detecting the rumor source in such a setup. Now the fact that the rumor infected graph is like a Poisson random tree as long as the number of infected nodes is $n = o(\sqrt{m})$ follows from an argument very similar to that used in establishing the proof of Theorem 6 along with the Binomial-Poisson approximation: a binomial variable with parameters m and c/m converges to a Poisson variable as $m \to \infty$. Next we will show that the rumor graph of size $n = o(\sqrt{m})$ is a tree with high probability.

First, define P_n as the event that the rumor graph G of n nodes on an underlying Erdos-Reny graph is a tree. Our interest is in the probability of P_n. We shall establish that for $n = o(\sqrt{m})$, $\mathbf{P}\left(P_n^c\right) \to 0$ as $m \to \infty$. This result, plus the fact that the degree of each node converges to a Poisson distribution will establish Theorem 7.

The rumor graph remains a tree as long as the next infected node has no neighbors which were already infected. Recall that all edges in the graph exist independently with probability $p = c/m$. Then we have that given that the rumor graph is a tree when n nodes are infected, it will remain a tree when node $n+1$ is infected with probability given by

$$\mathbf{P}\left(P_{n+1}|P_n\right) = \left(1 - \frac{c}{m}\right)^{n-1}$$

Using this we can obtain a lower bound for $\mathbf{P}(P_n)$.

$$\begin{aligned}\mathbf{P}(P_n) &= \prod_{k=1}^{n-1} \mathbf{P}(P_{k+1}|P_k) \\ &= \prod_{k=1}^{n-1} \left(1 - \frac{c}{m}\right)^{k-1} \\ &\geq \left(1 - \frac{c}{m}\right)^{n^2}\end{aligned}$$

Since $n = o(\sqrt{m})$ and $\mathbf{P}(P_n)$ decreases with n, we have

$$\mathbf{P}(P_n) \geq 1 - o(1).$$

That is, $\mathbf{P}(P_n^c) \to 0$ as $m \to \infty$. This completes the proof of G being a tree with high probability and hence the proof of Theorem 7.

3.3.6 Proof of Theorem 5: Geometric Trees

The proof of Theorem 5 uses the characterization of the rumor center provided by Proposition 1. That is, we wish to show that for all n large enough, the probability of the event that the size of the d^* rumor infected sub-trees of the source v^* are essentially 'balanced' with high enough probability. To establish this, we shall use coarse estimations on the size of each of these sub-trees using the standard concentration property of renewal processes along with geometric growth. This will be unlike the proof for regular trees where we had to necessarily delve into very fine detailed probabilistic estimates of the size of the sub-trees to establish the result. This relatively easier proof for geometric trees (despite heterogeneity) brings out the fact that it is fundamentally much more difficult to analyze expanding trees than geometric structures as expanding trees do not yield to generic concentration based estimations as they necessarily have very high variances.

To that end, we shall start by obtaining sharp estimations on the size of each of the rumor infected d^* sub-trees of v^* for any given time t. We are assuming here that the spreading times have distribution $F(t)$ with mean $\mu > 0$. Initially, at time 0 the source

node v^* has the rumor. It starts spreading along its d^* children (neighbors). Let $T_i(t)$ denote the size of the rumor infected subtree, denoted by $G_i(t)$, rooted at the ith child (or neighbor) of node v^*. Initially, $T_i(0) = 0$. The $T_i(\cdot)$ is a renewal process with time-varying rate: the rate at time t depends on the 'boundary' of the tree as discussed earlier. Due to the balanced and geometric growth conditions assumed in Theorem 5, the following will be satisfied: for small enough $\epsilon > 0$ (a) every node within a distance $\frac{t}{\mu}(1-\epsilon)$ of v^* is in one of the $G_i(t)$, and (b) no node beyond distance $\frac{t}{\mu}(1+\epsilon)$ of v^* is in any of the $G_i(t)$. Such a tight characterization of the 'shape' of $G_i(t)$ along with the polynomial growth will provide sharp enough bound on $T_i(t)$ that will result in establishing Theorem 5. This result is summarized below with its proof in the Appendix.

Theorem 10. *Consider a geometric tree with parameters $\alpha > 0$ and $0 < b \leq c$ as assumed in Theorem 5 and let the rumor spread from source v^* starting at time 0 as per the SI model distribution $F(t)$ with a well defined moment generating function and mean $\mu > 0$. Define $\epsilon = t^{-1/2+\delta}$ for any small $0 < \delta < 1/2$. Let $G(t)$ be the set of all rumor infected nodes in the tree at time t. Let $\mathcal{G}_t$ be the set of all sub-trees rooted at v^* (rumor graphs) such that all nodes within distance $\frac{t}{\mu}(1-\epsilon)$ from v^* are in the tree and no node beyond distance $\frac{t}{\mu}(1+\epsilon)$ from v^* is in the tree. Then*

$$\begin{aligned}\mathbf{P}(G_t \in \mathcal{G}_t) &= 1 - O\big(e^{-t^{\delta}}\big) \\ &\stackrel{t\to\infty}{\longrightarrow} 1.\end{aligned}$$

Define $\mathcal{E}_t$ as the event that $G_t \in \mathcal{G}_t$. Under event $\mathcal{E}_t$, consider the sizes of the sub-trees $T_i(t)$ for $1 \leq i \leq d_{v^*}$. Due to the polynomial growth condition and $\mathcal{E}_t$, we obtain the following bounds on each $T_i(t)$ for all $1 \leq i \leq d_{v^*}$:

$$\begin{aligned}\sum_{r=1}^{\frac{t}{\mu}(1-\epsilon)-1} br^{\alpha} &\leq T_i(t) \\ &\leq \sum_{r=1}^{\frac{t}{\mu}(1+\epsilon)-1} cr^{\alpha}.\end{aligned}$$

Now bounding the summations by Reimann's integrals, we have

$$\int_0^{L-1} r^\alpha dr \leq \sum_{r=1}^{L} r^\alpha \leq \int_0^{L+1} r^\alpha dr.$$

Therefore, it follows that under event $\mathcal{E}_t$, for all $1 \leq i \leq d_{v^*}$

$$\frac{b}{1+\alpha}\left(\frac{t}{\mu}(1-\epsilon)-2\right)^{\alpha+1} \leq T_i(t) \leq \frac{c}{1+\alpha}\left(\frac{t}{\mu}(1+\epsilon)\right)^{\alpha+1}.$$

In the most 'unbalanced' situation, $d_{v^*}-1$ of these sub-trees have minimal size $T_{\min}(t)$ and the remaining one sub-tree has size $T_{\max}(t)$ where

$$T_{\min}(t) = \frac{b}{1+\alpha}\left(\frac{t}{\mu}(1-\epsilon)-2\right)^{\alpha+1},$$
$$T_{\max}(t) = \frac{c}{1+\alpha}\left(\frac{t}{\mu}(1+\epsilon)\right)^{\alpha+1}.$$

Since by assumption $c < b(d_{v^*}-1)$, there exists $\gamma > 0$ such that $(1+\gamma)c < b(d_{v^*}-1)$. Therefore, for any choice of $\epsilon = t^{-1/2+\delta}$ for some $\delta \in (0, 1/2)$, we have

$$\begin{aligned}
\frac{(d^*-1)T_{\min}(t)+1}{T_{\max}(t)} =& \frac{b(d_{v^*}-1)}{c}\left(\frac{\frac{t}{\mu}-t^{\frac{1}{2}+\delta}-2}{\frac{t}{\mu}+t^{\frac{1}{2}+\delta}}\right)^{\alpha+1} \\
&+\frac{1+\alpha}{c}\left(\frac{1}{\frac{t}{\mu}+t^{\frac{1}{2}+\delta}}\right)^{\alpha+1} \\
\overset{(i)}{>}& (1+\gamma)\left(\frac{1-t^{-\frac{1}{2}+\delta}\mu-2\mu t^{-1}}{1+t^{-\frac{1}{2}+\delta}\mu}\right)^{\alpha+1} \\
&+\frac{1+\alpha}{c}\left(\frac{1}{\frac{t}{\mu}+t^{\frac{1}{2}+\delta}}\right)^{\alpha+1} \\
>& 1+\gamma \\
>& 1,
\end{aligned}$$

for t large enough since as $t \to \infty$ the first term in inequality (i) goes to 1 and the second

term goes to 0. From this, it immediately follows that under event $\mathcal{E}_t$ for t large enough

$$\max_{1\leq i\leq d_{v^*}} T_i(t) < \frac{1}{2}\left(\sum_{i=1}^{d_{v^*}} T_i(t) + 1\right).$$

Therefore, by Proposition 1 it follows that the rumor center is unique and equals v^*. We also have that $\mathcal{E}_t \subset C^1_{n(t)}$. Thus, from above and Theorem 10 we obtain

$$\begin{aligned}\lim_t \mathbf{P}\left(C^1_{n(t)}\right) &\geq \lim_t \mathbf{P}\left(\mathcal{E}_t\right)\\ &= 1.\end{aligned}$$

This completes the proof of Theorem 5.

3.4 Chapter Summary

We characterized the performance of rumor centrality as a source estimator for many different classes of networks. We saw first an interesting threshold phenomenon. For networks which grow linearly, detection is never possible, whereas for networks which grow faster than a line, we can always find the source with strictly positive probability. In fact, we found the exact error distribution for regular trees. As a consequence of this we also showed that for rumor spreading on a random regular graph, the probability that the estimated source is more than k hops away from the true source decays exponentially in k. For general random trees we showed that there is always a strictly positive probability of detection. We were able to extend this result to sparse Erdos-Renyi graphs. Our analysis for networks with expansion was based upon multi-type continuous time branching processes and generalized Polya's urn models. For networks with polynomial growth such as geometric trees, we were able to use a simpler concentration based analysis to show that rumor centrality has an asymptotic detection probability of 1.

We performed simulations on different network topologies to demonstrate the effectiveness and breadth of application of rumor centrality. Simulations on random regular graphs

and geometric trees agreed with our theoretical results. We found that rumor centrality was a more effective estimator than distance centrality for different network topologies, including real networks such as the power grid and the Internet AS network. Using heat maps of rumor centrality on regular trees and grids, we found that rumor centrality concentrates on a very small number of nodes, thereby greatly narrowing the set of likely sources. This is a very important property of rumor centrality that will make it extremely useful for finding sources.

In summary, we have shown that rumor centrality is an excellent estimator for detecting the source of a rumor on a large class of networks.

Chapter 4

Measuring Influence

Measuring influence of individuals is of great importance in many scenarios. Advertisers look for influential individuals to endorse products. Politicians try to gain favor with influential individuals to win elections. Advocacy groups try and spread their policies by recruiting influential individuals to their causes. In some scenarios determining who is influential can be easy. For example, among media figures, it is obvious that Oprah Winfrey is extremely influential. But measuring the influence of individuals using only network data is quite challenging. First, one must identify these influential individuals in the network with respect to a topic of interest. Some people may be influential or have strong reputations on certain topics, but not others. Second, one must use the network structure to quantify the influence of these individuals. Some may have limited influence, while others have a vast reach. Knowledge of the amount of influence of an individual can then be used for other purposes such as allocating resources when designing effective information dissemination campaigns.

Determining this influence can be difficult and often subjective. However, if the individuals exist in a network then it is natural to assume that the network structure can tell us a great deal about the amount of influence possessed by each individual. In this chapter we will look at the problem of measuring influence in the micro-blogging website known as Twitter. We will see that rumor centrality is the right measure of influence in Twitter. To arrive at this conclusion, we develop a new network growth model known as *topological network growth*. When rumor centrality is used in conjunction with this model, the

networks generated have a remarkable similarity to actual networks measured in Twitter. We use this observation to argue that rumor centrality is correctly capturing the amount of influence of individuals in Twitter. We then use rumor centrality as the basis for a dynamic influence tracking engine for Twitter which we call Trumor. We will find that Trumor does an effective job of finding influential individuals specified by topic and time.

4.1 Background

4.1.1 Twitter, Tweets, and Retweets

Twitter is a micro-blogging site which was created in 2006. Its primary aim is to allow users to disseminate information via short messages known as tweets. There is an incredible volume of content on Twitter. The site has over 200 million users [58] and typically has 750 tweets per second [37]. Twitter is also the method of information dissemination for many highly influential celebrities and politicians. For example, President Barack Obama hosted the first ever Twitter townhall meeting from the Whitehouse on July 2011, taking questions posted by users on Twitter [4]. The ability of users to spread information with Twitter coupled with its massive volume of information and the presence of several high profile users makes it an ideal medium to study in order to measure the influence of individuals.

In addition to having users post messages, Twitter also possesses several network structures. For instance, users on Twitter may follow other users. Any tweets made by a user will show up in their followers' Twitter pages. These follower relationships form a directed *follower network* on Twitter. Another important feature of Twitter is known as *retweeting*. When a user retweets another tweet it is equivalent to forwarding that tweet to all of the user's followers. Retweets are useful because they allow one to track the flow of information on Twitter. Every retweet can be thought of as a directed edge in a network, pointing from the tweeter to the retweeter and indicating some level of influence of the tweeter on the retweeter. If one connects these retweets together, one obtains the *retweet network*.

This retweet network is another network structure in Twitter in addition to the follower network. However, the retweet network is potentially more useful for two important rea-

sons. First, retweets are a direct measure of influence, as the act of retweeting involves some action on the part of the retweeter in response to the tweeter. Second, retweet networks can be specified by time and topic. That is, one may use only the portion of the full retweet network corresponding to a certain topic and time window. These two properties of the retweet network make it more ideal for measuring influence than the follower network, and it will be the network upon which we focus in this chapter.

4.1.2 Influence in Twitter: Previous Work

Many researchers have studied different measures of influence in Twitter. One of the earliest studies of Twitter compared different influence measures on both the retweet and follower networks [41]. The authors calculated the PageRank centrality measure on the follower network and compared this with the in-degree on both the follower network and retweet network. They found a gap in the influence calculated on the follower network versus the retweet network, indicating that these two networks provide very different information. In [25] the authors found that there was very little relation between the in-degree (number of followers) of a user in the follower network and their ability to spread information in the retweet network. This is further evidence that the retweet network should be the focus of any attempt to quantify influence in Twitter.

Both of the above studies looked at influence without taking the topic into consideration. In [63] a new topic based influence measure known as twitterRank was introduced. It was calculated on the follower network for specific topics. Topics were learned using latent Dirichlet allocation on a set of tweets. Once topics were learned, the twitterRank for each user on each topic could be calculated. TwitterRank is very similar to PageRank, in that it is a type of eigenvector centrality and involves a random walk on the follower network. However, the transition probabilities between users of the random walk depend upon how much they have discussed the topic in question and also how similar their topic interests are. If two neighbors on the follower graph have not discussed any common topics, then there will be no transition allowed between them in the random walk.

Another approach to measuring influence in Twitter used an influence measure based

upon the number of people reached by a tweet [10]. The influence of a user was defined as the logarithm of the average number of people a user reached with their tweets. A regression model was fit to this influence measure using features such as number of followers, number of tweets, and influence measure of the user's immediate followers. Predictions of the reach of individuals with this regression model were very poor, showing the difficulty of prediction in Twitter.

As can be seen, there are several different approaches to calculating influence in Twitter, but they are all heuristics with no solid experimental or theoretical justification. What we show in this chapter is that rumor centrality is a logical choice of influence measure and we justify this using retweet networks and a novel network growth model. Next, we look at models for network growth to see their connection with influence.

4.1.3 Network Growth Models

The retweet networks in Twitter are dynamic growing networks. As more people retweet, the network grows. This notion of network growth is important because it provides us another perspective on influence. In a growing network, new nodes must connect with nodes already in the network. The choice of which node to connect with depends on the network growth model considered. However, we will assume that the node connects with more influential nodes, where the appropriate influence measure depends upon the model considered. We now look at one of the most well-known network growth models known as preferential attachment (PA).

A very simple measure of influence is the degree of a node. Intuitively, the higher the degree of a node, the more people to which it is connected and so the more influential it is. If one uses degree as an influence measure, then this is exactly the preferential attachment (PA) model [14]. The model starts with a single node. At each discrete time step a new node with m edges arrives and connects each edge to a random node in the existing network. The probability of an edge connecting to a node is proportional to its degree. That is, as the model name implies, new nodes *prefer* to *attach* to higher degree nodes. This attachment probability of a node can be viewed as a measure of influence.

PA networks have an important property known as a power-law degree distribution. Define $\mathbf{P}(degree = d)$ as the probability that the degree of a randomly selected node in a PA network equals d. Then we have that for large d,

$$\mathbf{P}(degree = d) \propto d^{-\alpha} \tag{4.1}$$

It has been shown that for the PA model the exponent α is three [14].

The importance of this power-law degree distribution is that it is observed in several real networks. In fact, originally the PA model was conceived in order to explain power-laws observed in the Internet, biological networks, and a network of actor collaborations [14]. The PA model is an important conceptual development because it allows one to relate influence in a network to the observed structure of the network. In particular, PA allows one to relate degree centrality as an influence measure to power-law degree distributions. We will come back to this model when we study the structure of retweet networks in Twitter.

4.1.4 Web Search and Influence

The search engine Google was originally based upon the network centrality measure known as PageRank. One justification of PageRank as a measure of influence or importance for webpages is the following. The webpages that exist in the Internet are known to have a power-law degree distribution [14]. We have seen that the PA model which uses degree centrality to grow networks also produces a power-law degree distribution. Degree centrality is very closely related to PageRank, in that webpages with high degree generally have high PageRank. This makes sense intuitively, because if a webpage has many other websites pointing to it, then it is more influential. Therefore, we can view the measured power-law degree distribution in the Internet and the PA model as a justification for PageRank as a measure of influence for webpages.

We will follow this *scientific* approach in our search for an influence measure for Twitter. First we will study the retweet networks and learn what important properties they possess. These properties could for example be things such as power-law degree distributions. Second, we will develop network growth models which explain these properties.

The hope is that these models will provide us with the appropriate influence measure for Twitter.

4.2 Twitter Retweet Networks

4.2.1 Data Collection

We begin our search for an influence measure for Twitter by constructing retweet networks. We collected retweet data from Twitter for several different topics, such as sports events and musical performances [1]. For each topic, we reconstructed the corresponding retweet network in the following manner. We recorded any retweet which contained the topic phrase. For example, for the retweet network on the 2010 World Cup, we recorded any retweet with the phrase "World Cup" during the World Cup opening ceremony. The time range of data collection varied from topic to topic, but was typically on the order of a few hours.

Once we had collected the retweets for a topic, to reconstruct the retweet network we would extract the name of the source of the tweet being retweeted (tweeter) and the user who retweeted (retweeter). These tweeter-retweeter edges were then combined to form the retweet network for the given topic. These retweet networks are not for a single tweet, but for a single topic. Therefore, there are nodes in the network which correspond to multiple tweets. However, because of the short time interval over which we collected retweets and the specificity of the topics, these retweet networks do represent the spread of influence or information regarding a given topic, even if nodes in the network may have tweeted multiple times.

The topic phrases we used were Federer, BET Awards, World Cup, and England. For Federer, the retweet network was gathered during Roger Federer's first round match in Wimbledon 2010. The BET Awards network was gathered during the 2010 Black Entertainment Television (BET) Awards show. The World Cup network was gathered during the opening ceremony of the 2010 World Cup. The England network was gathered during the England versus Slovenia match in the 2010 World Cup. Each of these networks contained

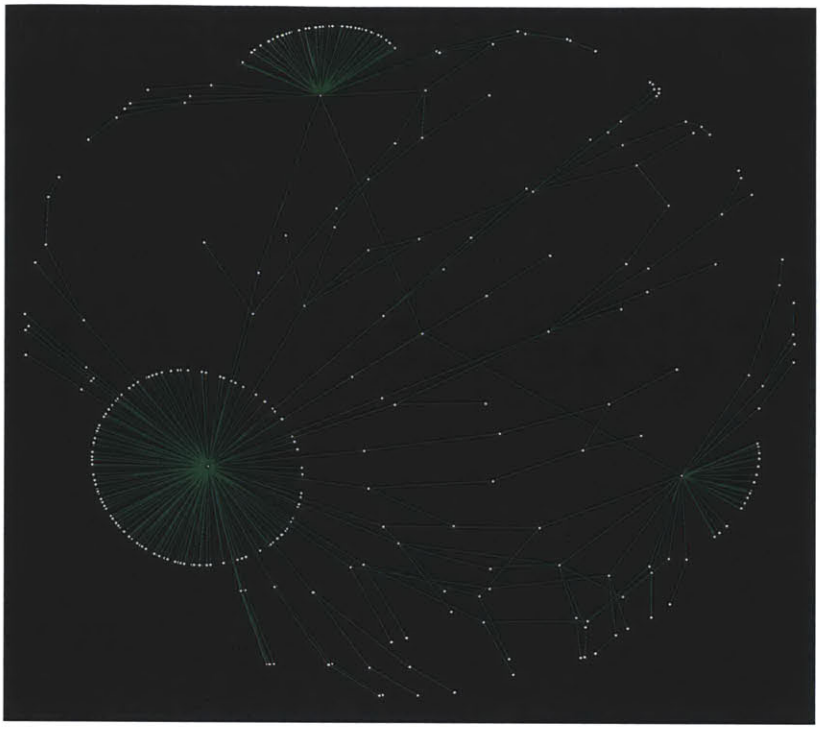

Figure 4-1: Largest connected component of the retweet network for topic "Federer".

many small connected components, and one large connected component. We focus on this largest connected component in this chapter. Pictures of the largest connected component for each of these retweet networks are shown in Figures 4-1, 4-2, 4-3, and 4-4.

4.2.2 Data Analysis

We first look at the degree distribution of the largest connected component of the retweet networks. Because of the success of the PA model in describing networks such as the Internet, we wish to see if it can also describe the degree distribution observed in the retweet networks. Figure 4-5 shows the degree distribution of the retweet networks along with a simulated PA network of equal size. It can be seen that both distributions exhibit a power-law behavior with similar exponent. Therefore, one may be tempted to think that PA is a reasonable model for these retweet networks. However, closer inspection of the distributions reveals another important feature that PA does not capture. The retweet networks always have a node whose degree is much larger than the maximum degree of the corresponding PA network. We refer to this maximum degree node as a *superstar*. The identities

Figure 4-2: Largest connected component of the retweet network for topic "England".

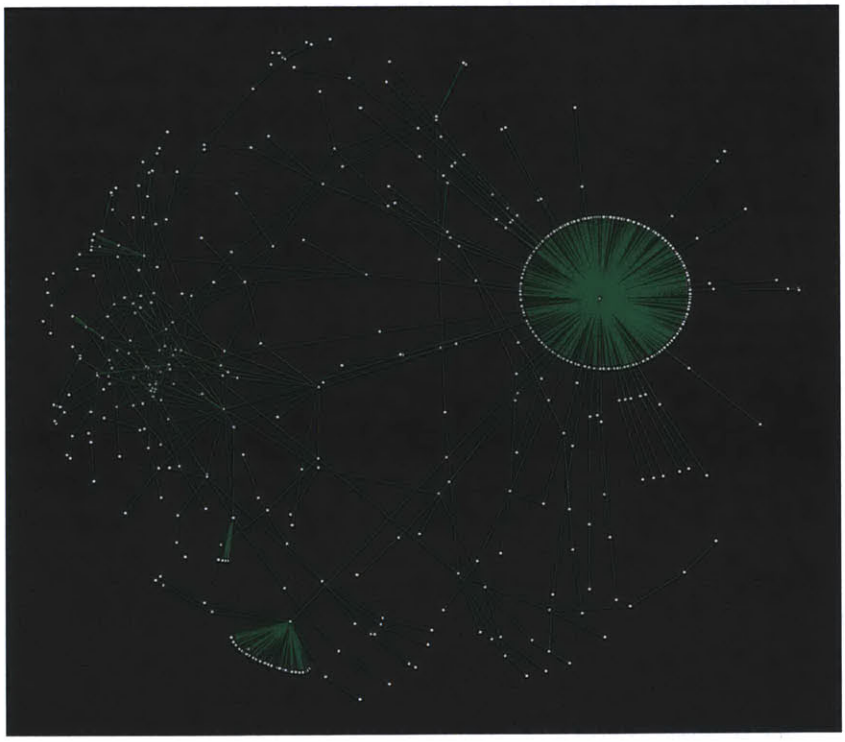

Figure 4-3: Largest connected component of the retweet network for topic "BET Awards".

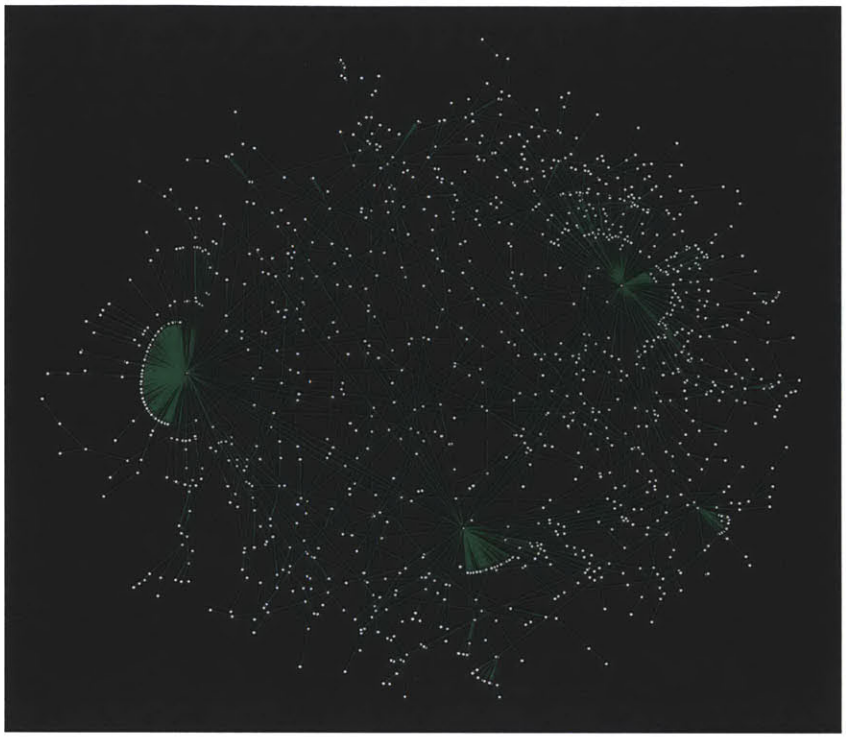

Figure 4-4: Largest connected component of the retweet network for topic "World Cup".

Network Topic	Superstar Twitter screen name
Federer	Wimbledon
England	SkyNewsBreak
BET Awards	ladygaga
World Cup	FIFAWorldCupTM

Table 4.1: Identity of superstars for the retweet networks.

of the superstars for these networks are shown in Table 4.1. These superstar nodes never emerge in the PA model. To make this more evident, in Figure 4-6 we show the maximum degree for the largest connected component of the retweet networks, along with the corresponding PA network of equivalent size. The retweet networks all possess a very high degree superstar node whose degree is on the order of the network size N. No such superstar node emerges in the preferential attachment networks. In fact, for a network with N nodes, the preferential attachment model has maximum degree on the order of $O(N^{1/2})$ [14]. The maximum degree does not scale linearly in N, which is what is seen in the data. Therefore, we require a new network growth model that can keep the power-law degree distribution of PA, while at the same time allow for the emergence of a superstar node.

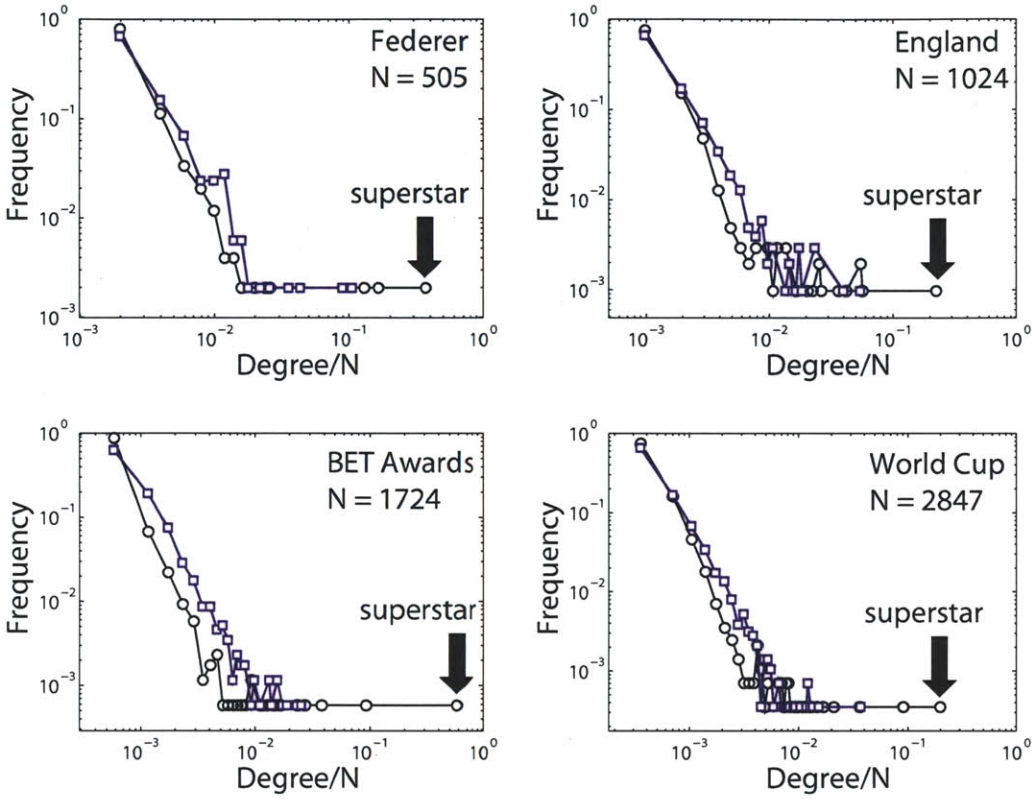

Figure 4-5: Distribution of degree normalized by network size for the largest connected component of the topic retweet networks (black circles) and corresponding PA model of equivalent size (blue squares). The size of the largest connected component N is shown for each retweet network.

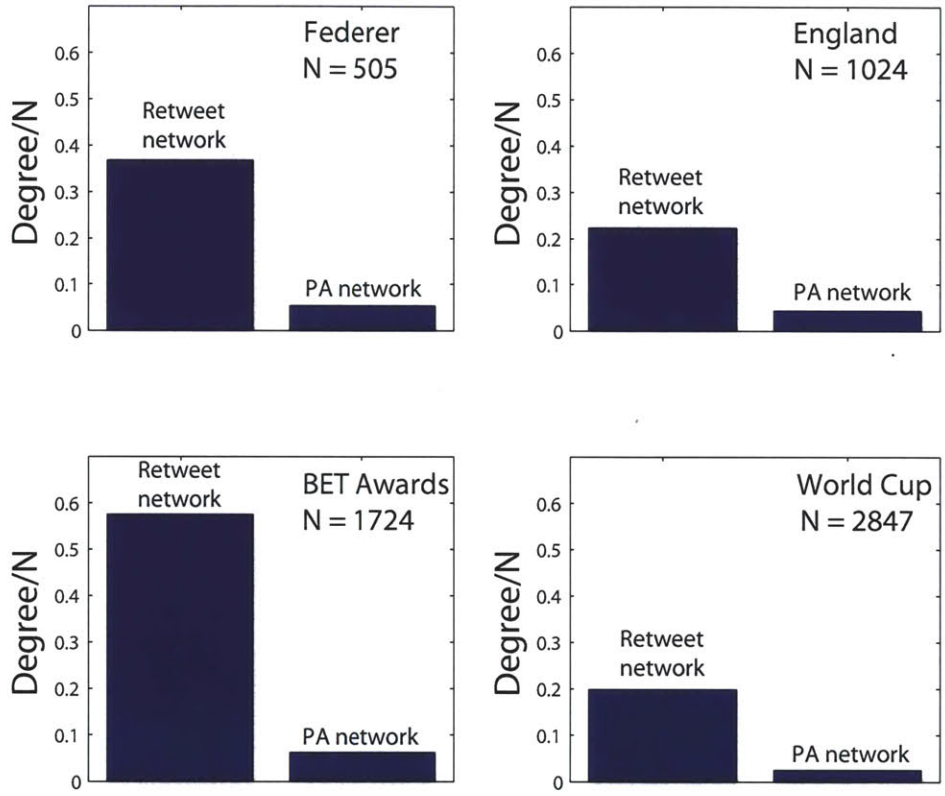

Figure 4-6: Maximum degree normalized by network size for the largest connected component of the topic retweet networks and corresponding PA model of equivalent size. The size of the largest connected component N is shown for each retweet network.

4.3 Topological Network Growth

We have seen that the retweet networks possess power-law degree distributions and superstar nodes. PA was not an adequate model to explain the superstar nodes. These superstars are an inherently global phenomenon, so it is not expected that a model with attachment probabilities based on local properties such as degree can produce them. What we need is a network growth model based on the global topology of the network. A natural class of functions which can capture global structural information are network centralities. Therefore, in our model we have the attachment probabilities proportional to a network centrality. We refer to this very general model as *topological network growth* (TNG) because the growth of the network depends upon its global topology via network centrality.

The model begins with a single node. Then, at each discrete time step a new node with one edge arrives. It connects to a node already in the network with probability proportional to its network centrality. Formally, let the the network at a given time step be G. Initially, G is a single node with no edges. The probability of the node which arrives at the subsequent time connecting to a node $v \in G$ is $\mathbf{P}(v|G)$. Then for a network centrality $NC(v, G)$, we have that

$$\mathbf{P}(v|G) \propto NC(v, G)$$

The networks formed with this model are trees, which are sufficient because the retweet networks are generally trees or tree-like. However, the model can be generalized by allowing new nodes to connect to $m \geq 1$ nodes, as in the PA model.

The attachment probabilities in the TNG model can be viewed as measures of influence. Nodes with higher attachment probabilities are more influential, and so new nodes are more likely to connect to them. The influence measure depends on the network topology and the resulting network topology depends upon the influence measure. Next we will look at different network centralities as influence measures/attachment probabilities and see which ones reproduce the phenomena observed in the retweet networks, i.e. the power-law and the superstar.

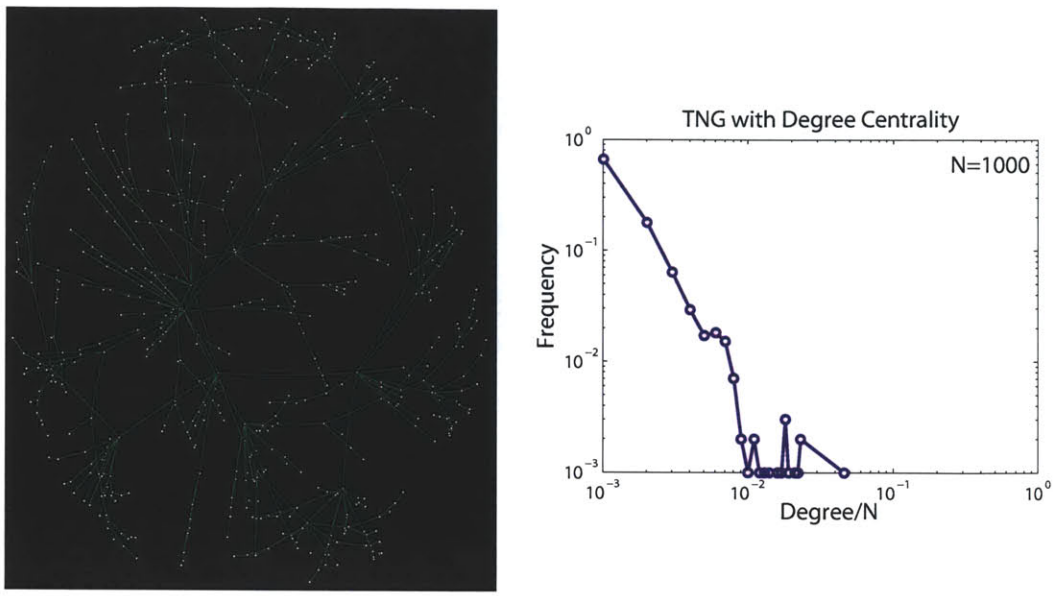

Figure 4-7: (left) An $N = 1000$ node degree centrality TNG network (PA network) and (right) its corresponding degree distribution normalized by network size N.

4.3.1 Degree Centrality: PA Model

If the attachment probabilities are proportional to the degree, then we reproduce the PA model. Thus we can view TNG as a natural generalization of PA. We show in Figure 4-7 a PA network with 1000 nodes and its corresponding degree distribution. As can be seen, there is a power-law behavior in the degree distribution but no superstar node is emerging. This is due to the fact that degree is a local centrality measure and would not be capable of modeling such global phenomenon.

4.3.2 Distance Centrality

A more global centrality measure we can use is distance centrality. Recall the definition of distance centrality:

$$D(v, G) = \left(\sum_{u \in V} d(v, u) \right)^{-1}. \tag{4.2}$$

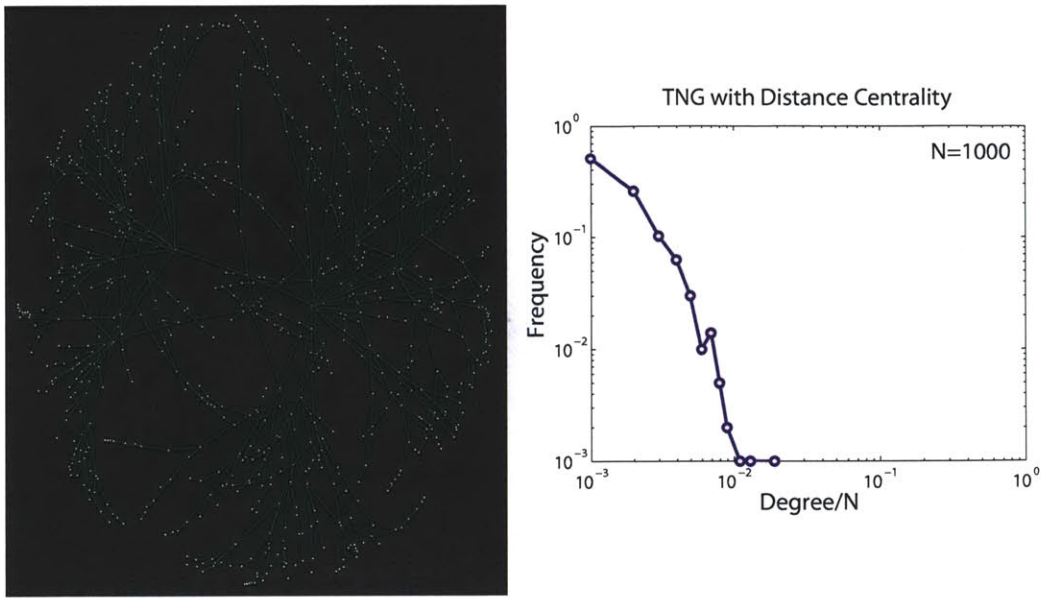

Figure 4-8: (left) An $N = 1000$ node distance centrality TNG network and (right) its corresponding degree distribution normalized by network size N.

Here $d(v, u)$ is the shortest path distance on G between nodes u and v. For our model, we then have

$$\mathbf{P}\left(v|G\right) \propto \left(\sum_{u \in G} d(v, u)\right)^{-1}.$$

An example of a 1000 node TNG network grown with distance centrality is shown in Figure 4-8 along with its degree distribution. Observe that there is no superstar node.

If we recall the analysis of distance centrality in Section 1.2.2, we saw that on a tree the ratio of the distance centrality of two neighbors could not be greater than two. Thus, a node with d neighbors can only ever be twice as influential as a leaf connected to it. High degree nodes do not exert very strong influence, in contrast to TNG with degree centrality where a node with d neighbors would be d times as influential as a leaf connected to it.

Here we see that even global network centralities are sometimes incapable of producing global phenomena such as superstars. TNG with distance centrality shows how the influence value really does matter. For TNG, centralities which produce similar rankings of nodes can lead to very different network structures.

4.3.3 Rumor Centrality

Next we look at rumor centrality as an influence measure. Recall the definition of rumor centrality for an N node network:

$$R(v, G) = \frac{N!}{\prod_{u \in G} T_u^v}$$

For our model, we then have

$$\mathbf{P}\left(v|G_N\right) \propto \frac{N!}{\prod_{u \in G_N} T_u^v}$$

An example of a 1000 node network grown with rumor centrality is shown in Figure 4-9 along with its degree distribution. Observe that the degree distribution follows a power-law. Also, a superstar node appears in the network, as indicated in the figure. The maximum degree scales as $\Theta(N)$ for an N node network. To see the emergence of the superstar more clearly, we plot in Figure 4-10 the degree of the first four nodes in a TNG rumor centrality network versus time. As can be seen, one node has a linearly growing degree, while others grow at a much slower rate. The degree of this node grows like $N/2$, as indicated in the figure. We will see the origin of this $1/2$ factor in Section 4.4. This is the superstar emerging as the network grows. Also observe that once this node becomes a superstar, there is no one else overtaking it. Thus, there is only one superstar in the TNG rumor centrality model.

Rumor centrality seems to have both properties we desire: power-law degree distribution and a superstar. We wish to see if these properties match with the retweet networks. We plot the degree distribution of the retweet networks and corresponding TNG rumor centrality network of equivalent size in Figure 4-11. Surprisingly, there is excellent agreement between the degree distribution of the rumor centrality networks and the retweet networks across all the different topics. Rumor centrality predicts not only the superstar degree very closely, but also the exponent of the power-law distribution. Therefore, it seems that rumor centrality is a good influence measure to describe the growth of these networks. Next, we

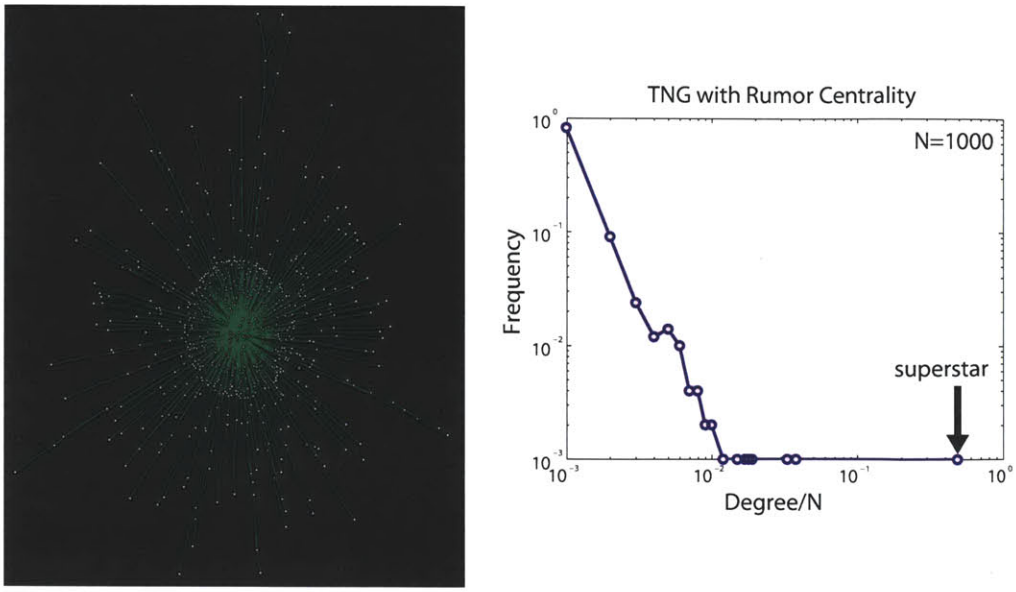

Figure 4-9: (left) An $N = 1000$ node rumor centrality TNG network and (right) its corresponding degree distribution normalized by network size N.

will look more closely at rumor centrality in order to understand why it produces superstars and power-laws.

4.4 Analysis of TNG with Rumor Centrality

The emergence of superstars and power-laws in TNG models with the use of rumor centrality is a curious phenomenon. The other centrality measures we looked at did not produce non-local phenomena such as superstars. In this section we provide an analysis of the TNG model with rumor centrality in order to understand how rumor centrality leads to these phenomena.

There has been much rigorous work studying network growth models based on local centrality measures such as degree. Much of this work can be found in [20] and references therein. Despite the simplicity of degree centrality based models, fairly advanced theoretical machinery is needed to analyze the properties of the networks. Our situation is by comparison even more difficult. We are using rumor centrality, which is a global centrality measure with a rather complex structure. Many of the techniques used for degree centrality networks will not be applicable. Therefore, we take a different approach. We first conjec-

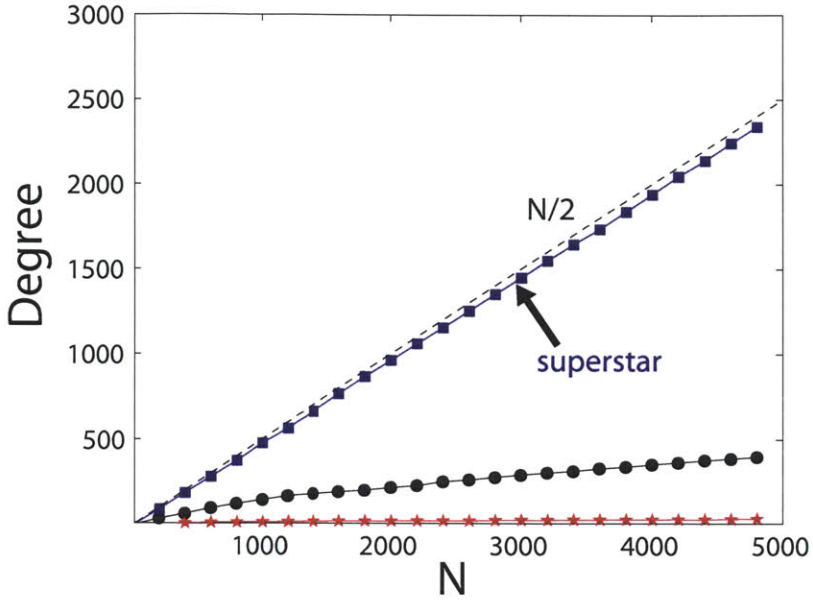

Figure 4-10: Degree versus time N for the first four nodes in a TNG rumor centrality network. The dotted line is a plot of $N/2$.

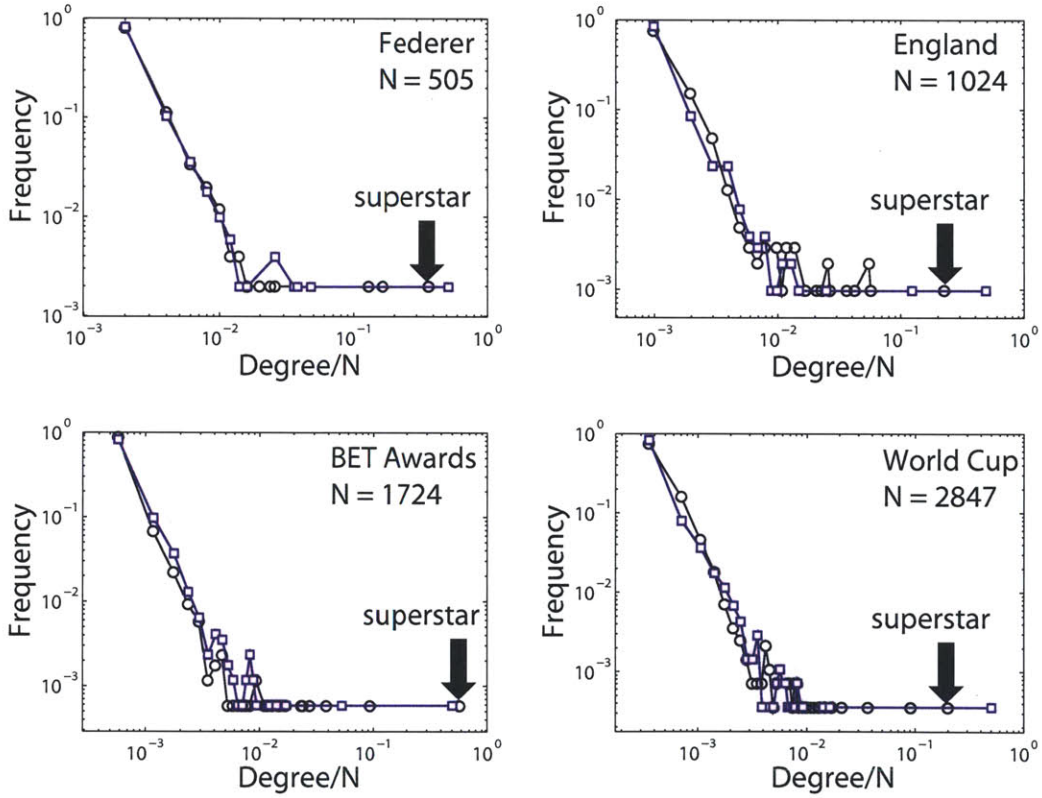

Figure 4-11: Degree distribution for retweet networks (black circles) and corresponding rumor centrality TNG model of equivalent size (blue squares). The network size N is shown for each retweet network.

ture the characteristics of the *fixed point* network under TNG with rumor centrality. This fixed point network is the characteristic network structure that remains invariant as the network grows according to the model. We cannot prove that this is the unique fixed point network structure, but based upon our simulations, we conjecture that there are no other fixed points. Second, we prove that this fixed point structure is invariant. Specifically, we prove if we grow a network as per TNG with rumor centrality starting from the fixed point network, it will retain all of its properties. That is, it is indeed a *fixed point* of TNG with rumor centrality. We will also show how this fixed point structure leads to the observed superstar and power-law degree distribution.

4.4.1 Fixed Point Network

We first describe the *fixed point* network which emerges in the TNG rumor centrality model. This structure is observed repeatedly in simulations, and we conjecture that it is the only structure which can emerge in this model. We will first describe this fixed point network and then we will show that it is invariant. That is, we will assume we start with an N node fixed point network. We will add N nodes to this network as per the TNG with rumor centrality model. In this new network with $N' = 2N$ nodes, we will show that with high probability all fixed point properties remain invariant. In essence, we show that once we reach this fixed point (which we conjecture we always do) we remain there.

Fixed Point Network Structure

We begin by characterizing the fixed point network structure for TNG with rumor centrality. The following theorem describes this network structure.

Theorem 11. *For all N large enough, the following N node network with the properties listed below is a fixed point of the TNG rumor centrality model:*

1. *The network is a tree.*

2. *There is a superstar node with degree $N/2 - o(N)$.*

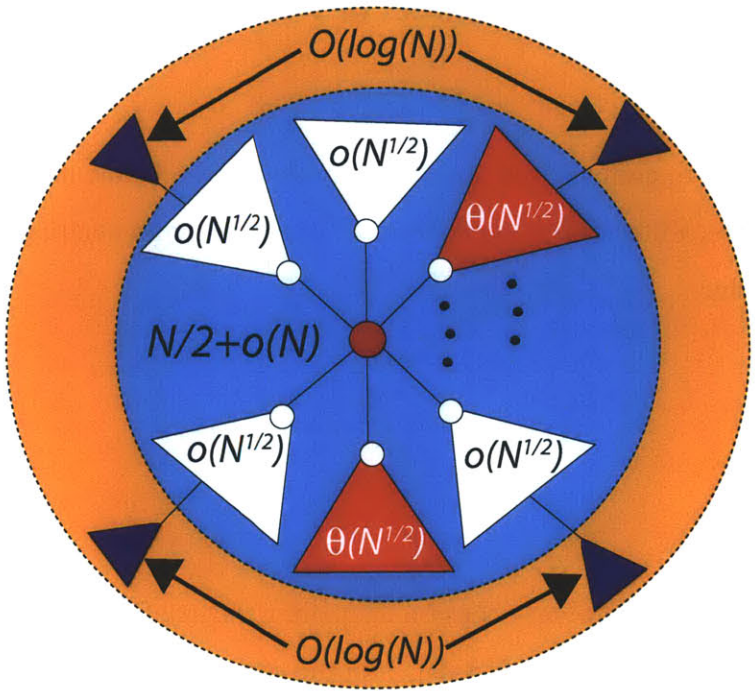

Figure 4-12: Illustration of an N node fixed point network for TNG with rumor centrality

3. *There are finitely many neighbors of the superstar which are are roots of subtrees which do not include the superstar and which contain* $\Theta\left(N^{1/2}\right)$ *nodes.*

4. *The remaining neighbors of the superstar are roots of subtrees which do not contain the superstar and which contain* $o\left(N^{1/2}\right)$ *nodes.*

5. *All nodes more than one hop from the superstar are roots of subtrees which do not contain the superstar and which contain* $O\left(\log(N)\right)$ *nodes.*

We show an illustration of this fixed point network structure in Figure 4-12. The superstar node is the center of the network with approximately $N/2$ neighbors. These nodes in turn are roots of subtrees, most of which are of size $o\left(N^{1/2}\right)$, but a finite number of which have size $\Theta\left(N^{1/2}\right)$. The remaining nodes are roots of subtrees with $O\left(\log(N)\right)$ nodes.

The tree property of the fixed point network is obvious because each new node contributes only one edge. We will next prove the invariance of the remaining properties of the fixed point from Theorem 11.

4.4.2 Attachment Probabilities of Fixed Point Network

Our first task is to compute the attachment probabilities for the nodes in this network G. To begin, we recall the following relation between the rumor centralities of any two nodes v and u in a tree.

$$R(u,G) = R(v,G) \prod_{j \in \mathcal{P}(v,u)} \frac{T_j^v}{N - T_j^v} \tag{4.3}$$

where $\mathcal{P}(v,u)$ is the set of all nodes on the path in the tree from v to u, not including node v. We define the partition function $Z(G)$ for the attachment probabilities on the network as

$$Z(G) = \sum_{v \in G} R(v,G).$$

With this notation, the attachment probabilities are

$$\mathbf{P}\left(v|G\right) = \frac{R(v,G)}{Z(G)}$$

Using equation (4.3) we express the rumor centrality of all nodes in the network in terms of the rumor centrality of the superstar node v^*.

$$\begin{aligned} Z(G) &= R(v^*,G)\left(1 + \sum_{v \in G/v^*} \prod_{j \in \mathcal{P}(v^*,v)} \frac{T_j^{v^*}}{N - T_j^{v^*}}\right) \\ &= R(v^*,G) Z_0(G) \end{aligned} \tag{4.4}$$

We will analyze this normalized partition function $Z_0(G)$ because it provides us with the attachment probabilities in a simple form. For example, the attachment probability for the superstar is simply the reciprocal of this function.

$$\mathbf{P}(v^*|G) = \frac{1}{Z_0(G)}$$

and the attachment probability for any other node v is

$$\begin{aligned}\mathbf{P}(v|G) &= \frac{R(v,G)}{R(v^*,G)Z_0(G)} \\ &= \frac{1}{Z_0(G)} \left(\prod_{j \in \mathcal{P}(v^*,v)} \frac{T_j^{v^*}}{N - T_j^{v^*}} \right).\end{aligned}$$

Now we look at the value of $Z_0(G)$ for the proposed fixed point network. First note that all but a finite number of the superstar's neighbors' subtrees have size of order $o\left(N^{-1/2}\right)$. Therefore, for these subtrees we can approximate the fraction terms as

$$\begin{aligned}\frac{T_j^{v^*}}{N - T_j^{v^*}} &= \frac{\frac{T_j^{v^*}}{N}}{1 - \frac{T_j^{v^*}}{N}} \\ &= \frac{T_j^{v^*}}{N}\left(1 + o\left(N^{-1/2}\right)\right)\end{aligned}$$

where we have used the fact that for $0 < x < 1$

$$\frac{1}{1-x} = \sum_{k=0}^{\infty} x^k.$$

For the finite number of subtrees with size of order $\Theta\left(N^{-1/2}\right)$, we find with a similar analysis that the fraction terms become

$$\frac{T_j^{v^*}}{N - T_j^{v^*}} = \frac{T_j^{v^*}}{N}\left(1 + \Theta\left(N^{-1/2}\right)\right)$$

Let us first look at the contribution to $Z_0(G)$ from the superstar v^*. This is by definition 1. Next, let us look at the contribution to $Z_0(G)$ from all the neighbors of the superstar. We define $\mathcal{N}(v^*)$ as the set of neighbors of v^*. Then the contribution of these nodes to $Z_0(G)$ is

$$\sum_{v \in \mathcal{N}(v^*)} \frac{T_j^{v^*}}{N} \left(1 + \Theta\left(N^{-1/2}\right)\right) = \left(1 + \Theta\left(N^{-1/2}\right)\right) \frac{N-1}{N}$$
$$= 1 - \Theta\left(N^{-1}\right) + \Theta\left(N^{-1/2}\right) - \Theta\left(N^{-3/2}\right)$$
$$= 1 + \Theta\left(N^{-1/2}\right)$$

Therefore, the contribution from the neighbors of v^* is $1 + \Theta\left(N^{-1/2}\right)$.

Finally, consider any node more than one hop from v^*. The contribution to $Z_0(G)$ from any one of these nodes is given by

$$\prod_{j \in \mathcal{P}(v^*, v)} \frac{T_j^{v^*}}{N - T_j^{v^*}} \leq \frac{O(\log(N))}{N} \frac{\Theta\left(N^{1/2}\right)}{N}$$
$$= O\left(N^{-3/2} \log(N)\right)$$

There are up to $\Theta(N)$ of these nodes, so their net contribution to $Z_0(G)$ is $O\left(N^{-1/2} \log(N)\right)$.

Combining all these terms, we find that the normalized partition function for the fixed point network is

$$Z_0(G) = 1 + 1 + \Theta\left(N^{-1/2}\right) + O\left(N^{-1/2} \log(N)\right)$$
$$= 2 + O\left(N^{-1/2} \log(N)\right) \tag{4.5}$$

With the normalized partition function, we now calculate the attachment probability of the superstar v^*.

$$\mathbf{P}(v^*|G) = \frac{1}{Z_0(G)}$$
$$= \frac{1}{2} - O\left(N^{-1/2} \log(N)\right)$$
$$= \frac{1}{2} - o(1) \tag{4.6}$$

For the neighbors of the superstar which are the root of a subtree of size $\Theta\left(N^{1/2}\right)$, the

Node	Attachment probability
Superstar	$1/2 - o(1)$
Node with $\Theta(N^{1/2})$ size subtree	$\Theta(N^{-1/2})$
Node with $o(N^{1/2})$ size subtree	$o(N^{-1/2})$
Node with $O(\log(N))$ size subtree	$O(N^{-3/2}\log(N))$

Table 4.2: Attachment probabilities for nodes in TNG with rumor centrality fixed point network with N nodes.

attachment probability will be

$$\begin{aligned}\mathbf{P}(v|G) &= \frac{1}{Z_0(G)}\frac{\Theta(N^{1/2})}{N-\Theta(N^{1/2})} \\ &= \Theta(N^{-1/2})\end{aligned}$$

and for any neighbor of the superstar which is the root of a subtree of size $o\left(N^{1/2}\right)$, the attachment probability will be

$$\begin{aligned}\mathbf{P}(v|G) &= \frac{1}{Z_0(G)}\frac{o(N^{1/2})}{N-o(N^{1/2})} \\ &= o(N^{-1/2}).\end{aligned}$$

For any node which is the root of a subtree of size $O(\log N)$, the attachment probability will be upper bounded by

$$\begin{aligned}\mathbf{P}(v|G) &\leq \frac{1}{Z_0(G)}\frac{O(\log N)}{N-O(\log N)}\frac{\Theta\left(N^{1/2}\right)}{N-\Theta\left(N^{1/2}\right)} \\ &= O(N^{-3/2}\log N).\end{aligned}$$

We collect these results in Table 4.2 and will use them next to establish the invariance of the fixed point.

4.4.3 Invariance of Fixed Point Network

With the attachment probabilities for the nodes in the fixed point network characterized, we now show how this leads to the invariance of the fixed point. We start with an N node fixed

point network and then add N nodes to it as per the TNG with rumor centrality model. In this new network with $N' = 2N$ nodes, we will show that with high probability all fixed point properties remain invariant. That is, whatever property held for N will also hold for N'.

To describe N new nodes joining the fixed point network, we can think of each node v in the fixed point network as tossing N coins which have probability $\mathbf{P}(v|G)$ of coming up heads. For each coin that comes up heads, a new node connects to v. Thus, the change in the degree of a node v when a new node joins the network is an independent Bernoulli random variable with parameter $p = \mathbf{P}(v|G)$. These probabilities will change as more nodes join the network, so we must take care in our analysis. Our approach will be to add N nodes to the network in small groups. We will show that when adding nodes in this manner, the attachment probabilities will not change appreciably. With this we can show that the change in the degree of a node when we add the small group of nodes nodes concentrates about a certain value with high probability. We then repeat this for each group until N nodes are added, and then we show that with high probability the fixed point properties of the network remain invariant.

To establish the invariance of the fixed point we need to show that the sum of N independent Bernoulli random variables concentrates around its mean. The following theorem provides us with the required concentration.

Theorem 12. *[46, Theorem 4.1 (pg. 70) and Theorem 4.3 (pg. 72)] Let $X_1, X_2, ...X_M$ be a sequence of independent Bernoulli random variables such that for $1 \leq i \leq M$, $\mathbf{P}(X_i = 1) = p_i$. Then, for $S_M = \sum_{i=1}^{M} X_i$, $\mu = \mathbb{E}[S_M] = \sum_{i=1}^{M} p_i$, and $0 < \epsilon \leq 1$ we have that*

$$\mathbf{P}(|S_M - \mu| > \epsilon\mu) \leq 2\exp\left(-\frac{\mu\epsilon^2}{4}\right). \tag{4.7}$$

Also, for any $\epsilon > 0$

$$\mathbf{P}(S_M > (1+\epsilon)\mu) \leq \left(\frac{e^{\epsilon}}{(1+\epsilon)^{1+\epsilon}}\right)^{\mu}. \tag{4.8}$$

Invariance of Superstar

First we consider the superstar v^*. We add N nodes to the network in groups of size $N^{1/2}$. Let X_i be a Bernoulli random variable which is 1 if node i joins the superstar and 0 otherwise. We start with $\mathbf{P}(X_1 = 1) = p_1 = \mathbf{P}(v^*|G_N) = 1/2 - o(1)$ (from Table 4.2). If we add $N^{1/2}$ nodes, then the degree of the superstar will still be $N/2 - o(N)$. Therefore, the attachment probability will still be $1/2 - o(1)$. We set $M = N^{1/2}$, $\epsilon = N^{-1/6}$, and

$$\begin{aligned}\mu &= \sum_{i=1}^{N^{1/2}} p_i \\ &= N^{1/2}\left(1/2 - o(1)\right) \\ &= \frac{N^{1/2}}{2} - o\left(N^{1/2}\right)\end{aligned}$$

Now let M, μ and ϵ be as defined above and define the following event for $1 \leq j \leq N^{1/2}$:

$$\mathcal{E}_j = \left\{ \left| \sum_{i=(j-1)M+1}^{jM} X_i - \mu > \right| < \mu\epsilon \right\}.$$

This event says that in block j of $N^{1/2}$ nodes, the number of nodes which join the superstar deviates from $\mu = \frac{1}{2}N^{1/2} - o\left(N^{1/2}\right)$ by at most $\mu\epsilon = \frac{1}{2}N^{1/3} - o\left(N^{1/3}\right)$. We want to show that all of the $\mathcal{E}_j$ occur for $1 \leq j \leq N^{1/2}$. That is, we want lower bounds on $\mathbf{P}(\mathcal{E}_1)$ and $\mathbf{P}(\mathcal{E}_j|\mathcal{E}_{j-1})$ for $2 \leq j \leq N^{1/2}$. These will have the same lower bound because they will have the same attachment probability. We can use Theorem 12 to lower bound $\mathbf{P}(\mathcal{E}_1)$, and equivalently $\mathbf{P}(\mathcal{E}_j|\mathcal{E}_{j-1})$.

$$\begin{aligned}\mathbf{P}(\mathcal{E}_1) &= 1 - \mathbf{P}\left(\left|\sum_{i=1}^{M} X_i - \mu > \right| > \mu\epsilon\right) \\ &\geq 1 - 2\exp\left(-\frac{1}{4}\left(\frac{N^{1/2}}{2} - o\left(N^{1/2}\right)\right)N^{-1/3}\right) \\ &\geq 1 - 2\exp\left(-\Theta\left(N^{1/6}\right)\right)\end{aligned}$$

Now we use this to show that with high probability all the $\mathcal{E}_j$ occur.

$$\begin{aligned}
\mathbf{P}\left(\bigcap_{j=1}^{N^{1/2}} \mathcal{E}_j\right) &= \mathbf{P}\left(\mathcal{E}_1\right) \prod_{j=2}^{N^{1/2}} \mathbf{P}\left(\mathcal{E}_j | \mathcal{E}_{j-1}\right) \\
&\geq \left(1 - 2\exp\left(-\Theta\left(N^{1/6}\right)\right)\right)^{N^{1/2}} \\
&\geq 1 - 2N^{1/2}\exp\left(-\Theta\left(N^{1/6}\right)\right) \\
&\geq 1 - \exp\left(-\Theta\left(N^{1/6}\right)\right)
\end{aligned}$$

Above we have used Bernoulli's inequality which states that for every real number $x \geq -1$ and every integer r

$$(1+x)^r \geq 1 + rx$$

Therefore, with high probability, $N/2 - o\left(N\right)$ nodes join the superstar so its degree becomes $N - o(N) = N'/2 - o(N')$. Therefore, we see that the superstar remains invariant.

Invariance of Nodes with $\Theta\left(N^{1/2}\right)$ Size Subtrees

Consider a node v that is root of a subtree with size $\Theta\left(N^{1/2}\right)$. Let X_i be a Bernoulli random variable which is 1 if node i joins the superstar and 0 otherwise. We start with $\mathbf{P}\left(X_1 = 1\right) = p_1 = \mathbf{P}(v^*|G_N) = \Theta\left(N^{-1/2}\right)$ (from Table 4.2). We add N nodes to the network in groups of size $N^{1/2+a}$ for $0 < a < 1/2$. We are going to look at the event that in each group at most $\Theta\left(N^a\right)$ nodes join v, so the degree of v will still be $\Theta\left(N^{1/2}\right)$. Therefore, the attachment probability will still be $\Theta\left(N^{-1/2}\right)$. We set $M = N^{1/2+a}$, $\epsilon = 2e - 1$, and

$$\begin{aligned}
\mu &= \sum_{i=1}^{N^{1/2+a}} p_i \\
&= N^{1/2+a}\left(\Theta\left(N^{-1/2}\right)\right) \\
&= \Theta\left(N^a\right)
\end{aligned}$$

Now let M, μ and ϵ be as defined above and define the following event for $1 \leq j \leq N^{1/2-a}$:

$$\begin{aligned}\mathcal{E}_j &= \left\{ \sum_{i=(j-1)M+1}^{jM} X_i < (1+\epsilon)\,\mu \right\} \\ &= \left\{ \sum_{i=(j-1)M+1}^{jM} X_i < 2eN^a \right\}.\end{aligned}$$

This event says that in block j of $N^{1/2+a}$ nodes, the number of nodes which join v is upper bounded by $2eN^a$ for $0 < a < 1/2$. We want to show that all of the $\mathcal{E}_j$ occur for $1 \leq j \leq N^{1/2-a}$. That is, we want lower bounds on $\mathbf{P}(\mathcal{E}_1)$ and $\mathbf{P}(\mathcal{E}_j|\mathcal{E}_{j-1})$ for $2 \leq j \leq N^{1/2-a}$. These will have the same lower bound because they will have the same attachment probability. We can use Theorem 12 to lower bound $\mathbf{P}(\mathcal{E}_1)$, and equivalently $\mathbf{P}(\mathcal{E}_j|\mathcal{E}_{j-1})$.

$$\begin{aligned}\mathbf{P}(\mathcal{E}_1) &= 1 - \mathbf{P}\left(\sum_{i=1}^{M} X_j > 2eN^a \right) \\ &\geq 1 - \left(\frac{e^{2e-1}}{(2e)^{2e}} \right)^{N^a} \\ &\geq 1 - e^{-N^a}\end{aligned}$$

Now we use this to show that with high probability all the $\mathcal{E}_j$ occur.

$$\begin{aligned}\mathbf{P}\left(\bigcap_{j=1}^{N^{1/2-a}} \mathcal{E}_j \right) &= \mathbf{P}(\mathcal{E}_1) \prod_{j=2}^{N^{1/2-a}} \mathbf{P}(\mathcal{E}_j|\mathcal{E}_{j-1}) \\ &\geq \left(1 - e^{-N^a}\right)^{N^{1/2-a}} \\ &\geq 1 - N^{1/2-a} e^{-N^a} \\ &\geq 1 - e^{-\Theta(N^a)}\end{aligned}$$

Above we have used Bernoulli's inequality. Therefore, with high probability, we have $N^{1/2-a}N^a = \Theta\left(N^{1/2}\right)$ nodes join v so its degree becomes $\Theta\left(N^{1/2}\right) + \Theta\left(N^{1/2}\right) = \Theta\left(N'^{1/2}\right)$. Therefore, we see that this fixed point property remains invariant.

Invariance of Nodes with $o\left(N^{1/2}\right)$ Size Subtrees

Consider a node v that is root of a subtree with size $o\left(N^{1/2}\right)$. Let X_i be a Bernoulli random variable which is 1 if node i joins the superstar and 0 otherwise. We start with $\mathbf{P}\left(X_1 = 1\right) = p_1 = \mathbf{P}(v^*|G_N) = o\left(N^{-1/2}\right) = \Theta\left(N^{-1/2-a/2}\right)$ for some $0 < a < 1/2$ (from Table 4.2). We add N nodes to the network in groups of size $N^{1/2+a}$. We are going to look at the event that in each group at most $\Theta\left(N^{a/2}\right)$ nodes join v, so the degree of v will still be $o\left(N^{1/2}\right)$. Therefore, the attachment probability will still be $\Theta\left(N^{-1/2-a/2}\right)$. We set $M = N^{1/2+a}$, $\epsilon = 2e - 1$, and

$$\begin{aligned}\mu &= \sum_{i=1}^{N^{1/2+a}} p_i \\ &= N^{1/2+a}\left(\Theta\left(N^{-1/2-a/2}\right)\right) \\ &= \Theta\left(N^{a/2}\right)\end{aligned}$$

Now let M, μ and ϵ be as defined above and define the following event for $1 \leq j \leq N^{1/2-a}$:

$$\begin{aligned}\mathcal{E}_j &= \left\{\sum_{i=(j-1)M+1}^{jM} X_i < (1+\epsilon)\,\mu\right\} \\ &= \left\{\sum_{i=(j-1)M+1}^{jM} X_j < 2eN^{a/2}\right\}.\end{aligned}$$

This event says that in block j of $N^{1/2+a}$ nodes, the number of nodes which join the superstar is upper bounded by $2eN^{a/2}$ for $0 < a < 1/2$. We want to show that all of the $\mathcal{E}_j$ occur for $1 \leq j \leq N^{1/2-a}$. That is, we want lower bounds on $\mathbf{P}\left(\mathcal{E}_1\right)$ and $\mathbf{P}\left(\mathcal{E}_j|\mathcal{E}_{j-1}\right)$ for $2 \leq j \leq N^{1/2-a}$. These will have the same lower bound because they will have the same attachment probability. We can use Theorem 12 to lower bound $\mathbf{P}\left(\mathcal{E}_1\right)$, and equivalently

$\mathbf{P}\left(\mathcal{E}_j|\mathcal{E}_{j-1}\right)$.

$$\begin{aligned}\mathbf{P}\left(\mathcal{E}_1\right) &= 1-\mathbf{P}\left(\sum_{i=1}^{M} X_j > 2eN^{a/2}\right)\\ &\geq 1-\left(\frac{e^{2e-1}}{(2e)^{2e}}\right)^{N^{a/2}}\\ &\geq 1-e^{-N^{a/2}}\end{aligned}$$

Now we use this to show that with high probability all the $\mathcal{E}_j$ occur.

$$\begin{aligned}\mathbf{P}\left(\bigcap_{j=1}^{N^{1/2-a}} \mathcal{E}_j\right) &= \mathbf{P}\left(\mathcal{E}_1\right)\prod_{j=2}^{N^{1/2-a}} \mathbf{P}\left(\mathcal{E}_j|\mathcal{E}_{j-1}\right)\\ &\geq \left(1-e^{-N^{a/2}}\right)^{N^{1/2-a}}\\ &\geq 1-N^{1/2-a}e^{-N^{a/2}}\\ &\geq 1-e^{-\Theta\left(N^{a/2}\right)}\end{aligned}$$

Above we have used Bernoulli's inequality. Therefore, with high probability, we have $N^{1/2-a}N^{a/2} = o\left(N^{1/2}\right)$ nodes join v so its degree becomes $o\left(N^{1/2}\right) + o\left(N^{1/2}\right) = o\left(N'^{1/2}\right)$. Therefore, we see that this fixed point property remains invariant.

Invariance of Nodes with $O\left(\log(N)\right)$ Size Subtrees

We use a different approach to show the invariance of nodes with subtrees of size $O\left(\log(N)\right)$ due to the small size of their attachment probabilities. Let us consider one of these nodes v whose attachment probability is $O\left(N^{-3/2}\log\left(N\right)\right)$. We will add N nodes to the network in $\log(N)$ groups of size $N/\log(N)$. Because this group is fairly large, it is possible that the attachment probability will increase quite a bit. However, we will now show that this will not happen with high probability. To that end, let us define the following event for group j of the nodes which are added, $1 \leq j \leq \log(N)$.

$$A_j = \{1 \text{ or less of the } N/\log(N) \text{ nodes in group } j \text{ join } v\}.$$

Now, if A_j occurs, then v's subtree size will still be $O(\log(N))$, so the attachment probability will still be $O\left(N^{-3/2}\log(N)\right)$. We will use this fact to show that in each group, with high probability no more than 1 node joins v. That is, we are interested in $\mathbf{P}(A_1)$ and $\mathbf{P}(A_j|A_{j-1})$ for $2 \leq j \leq \log(N)$. It is clear that $\mathbf{P}(A_j|A_{j-1})$ and $\mathbf{P}(A_1)$ will have the same upper bound because the attachment probability will be $O\left(N^{-3/2}\log(N)\right)$ in each case. Therefore, we will show the calculation for $\mathbf{P}(A_1)$, but an identical calculation can be done for $\mathbf{P}(A_j|A_{j-1})$ which gives the same upper bound.

$$\begin{aligned}\mathbf{P}(A_1) &= \left(1 - O\left(N^{-3/2}\log(N)\right)\right)^{N/\log(N)} + \\ &\quad \frac{N}{\log(N)} O\left(N^{-3/2}\log(N)\right)\left(1 - O\left(N^{-3/2}\log(N)\right)\right)^{N/\log(N)-1} \\ &\geq 1 - O\left(N^{-1/2}\right)\end{aligned}$$

Above we have use Bernoulli's inequality. Now, we wish to show that in each group of size $N/\log(N)$, with high probability no more than 1 node joins v. That is, we want all of the A_j to occur. This happens with probability

$$\begin{aligned}\mathbf{P}\left(\bigcap_{j=1}^{\log(N)} A_j\right) &= \mathbf{P}(A_1)\prod_{j=2}^{\log(N)} \mathbf{P}(A_j|A_{j-1}) \\ &\geq \left(1 - O\left(N^{-1/2}\right)\right)^{\log N} \\ &\geq 1 - O\left(N^{-1/2}\log(N)\right)\end{aligned}$$

Above we again used Bernoulli's inequality. Therefore, with high probability, $O(\log(N))$ nodes join v, and hence its subtree size increases to $O(\log(N)) + O(\log(N)) = O(\log(N'))$ and this fixed point property remains invariant.

4.4.4 Origin of Superstar

We have seen that the superstar node v^* has an attachment probability of $1/2 - o(1)$ in the fixed point network. It is an interesting property of rumor centrality that lets so much probability mass reside at the superstar and not escape in the fixed point. Here we give a simpler explanation as to how a superstar emerges. Consider the simple star network in Figure 4-13 with a central superstar node as the hub with N neighbors. The rumor centrality of the superstar is $N!$, and the rumor centrality of every other node is $(N-1)!$. With these values we have that

$$\begin{aligned}\mathbf{P}\left(v^*|G_N\right) &= \frac{R(v^*, G_N)}{\sum_{v \in G_N} R(v, G_N)} \\ &= \frac{N!}{N! + N(N-1)!} \\ &= \frac{1}{2}.\end{aligned}$$

Thus, in this simple star network the attachment probability is $1/2$ exactly, while in the fixed point it is slightly perturbed to $1/2 - o(1)$. If in this star network a node joins the superstar (which happens with probability $1/2$) then the attachment probability of the superstar remains $1/2$. If a node joins one of its neighbors, the attachment probability will decrease slightly by $o(1)$. If many new nodes were to join a single neighbor of the superstar, then it could overtake the superstar. However, as we have just seen, the chance of this happening is quite low, so with high probability the superstar will remain a superstar. A more visual display of this superstar node's growth is in Figure 4-14, where we show snapshots of a TNG rumor centrality network at various stages in its growth. The emergence of the superstar can be clearly seen from these network snapshots.

4.4.5 Origin of Power-Law

The power-law degree distribution can be understood if we look at the ratio of rumor centralities of non-superstar nodes. Consider two nodes u and v which neighbor the superstar, as shown in Figure 4-15. Using equation (4.3) we find that the ratio of the rumor centralities

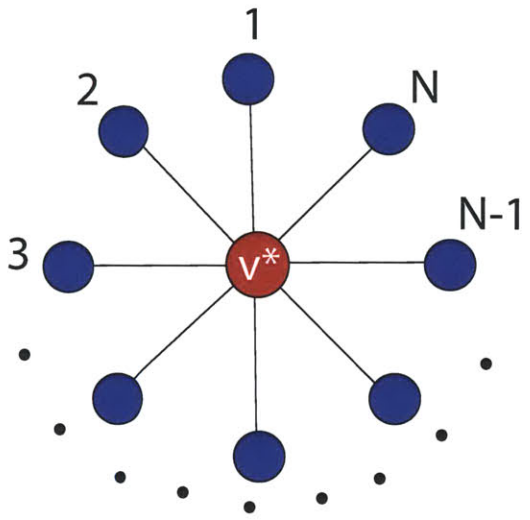

Figure 4-13: Star network with one superstar v^* and N leaf nodes.

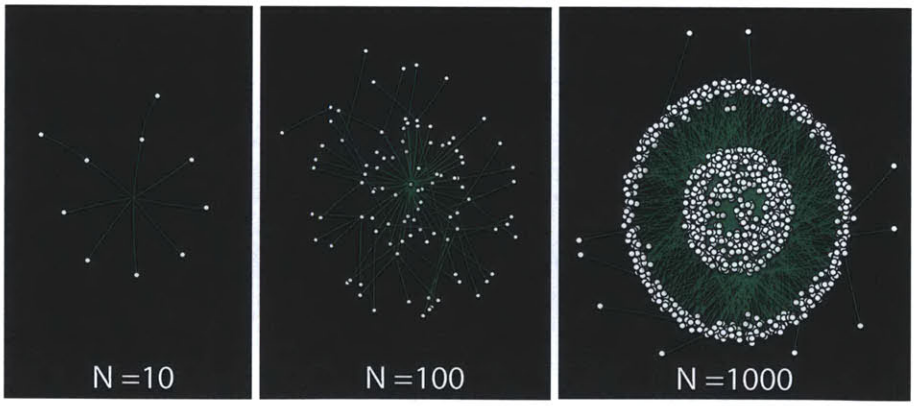

Figure 4-14: Snapshots in time of a TNG with rumor centrality network. The snapshots occur at (left) $N = 10$, (middle) $N = 100$, and (right) $N = 1000$

of u and v is given by

$$\begin{aligned}\frac{R(v,G)}{R(u,G)} &= \frac{T_v^{v^*}(N - T_u^{v^*})}{T_u^{v^*}(N - T_v^{v^*})} \\ &= \frac{T_v^{v^*}}{T_u^{v^*}}\left(1 - \frac{T_u^{v^*}}{N}\right)\left(1 + \frac{T_v^{v^*}}{N} + \Theta\left(N^{-1/2}\right)\right) \\ &= \frac{T_v^{v^*}}{T_u^{v^*}}\left(1 + \frac{T_v^{v^*} - T_u^{v^*}}{N} + \Theta\left(N^{-1/2}\right)\right).\end{aligned}$$

Above we have used the fact that each subtree has size at most $\Theta(N^{1/2})$. Because of the structure of the fixed point network, the subtree sizes of u and v are very close to their degree. With these observations, we have that

$$\frac{R(v,G)}{R(u,G)} \approx \frac{\deg(v)}{\deg(u)}\left(1 + \frac{T_v^{v^*} - T_u^{v^*}}{N} + \Theta\left(N^{-1/2}\right)\right)$$

where $\deg(\cdot)$ is the degree of the node. Therefore, the ratio of the attachment probability of nodes neighboring the superstar is approximately equal to the ratio of their degrees, just as in the PA model. The only difference is a slight bias in this ratio towards the larger subtree, give by the term $\left(T_v^{v^*} - T_u^{v^*}\right) N^{-1}$. These nodes effectively follow PA amongst themselves, and they constitute over half of the network. This explains the near equivalence of the power-law exponent of PA and TNG with rumor centrality.

4.5 Random Walk Interpretation of TNG with Rumor Centrality

We have seen in Section 2.3.6 that there is a random walk interpretation of rumor centrality in terms on an appropriately defined Markov chain. The transition probabilities from a node v to a node u for this walk were proportional to how many nodes were in u's subtree (how many nodes the rumor could reach from u once v had given it a rumor). We termed this the *trendy gossiper model*. This view of rumor centrality can give us a better understanding as to why TNG with rumor centrality models Twitter so well.

Before explaining how this random walk model explains Twitter, we first look at a

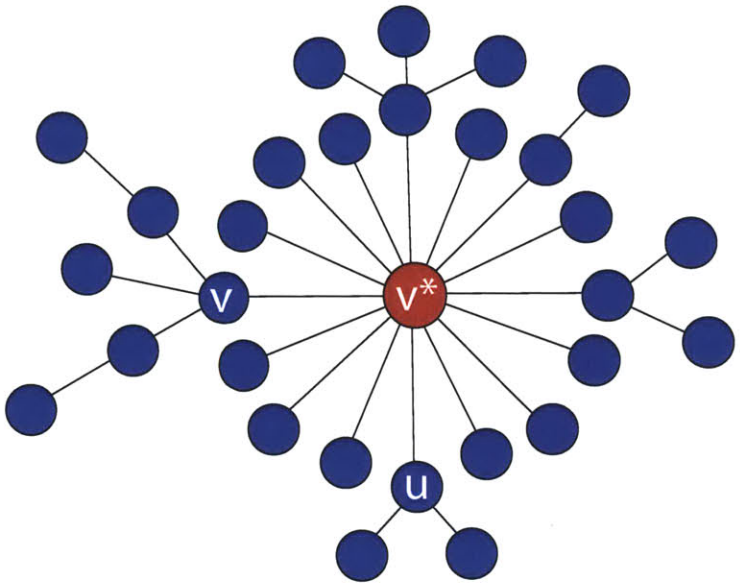

Figure 4-15: Network with nodes u and v neighboring superstar v^*.

similar random walk view of PageRank. PageRank emerges as the stationary distribution of a random walk/Markov chain with the following transition probabilities. The probability of going from a node to any of its neighbors is uniform and approximately equal to the reciprocal of its out-degree, while there is a small probability of randomly jumping to an arbitrary node in the network. This model is known as the random web-surfer model [53]. It can be thought of a web-surfer clicking links on webpages randomly, surfing through the Internet. The PageRank of a website measures the fraction of time this random surfer lands on that website. Therefore, the higher a website's PageRank, the more likely it will be visited by a web surfer, and the more important or relevant it is.

For rumor centrality a similar analogy can be constructed. We showed that rumor centrality was the stationary distribution of a random walk on a network where transition probabilities were proportional to the subtree sizes. Another way to state is that the transition probabilities were proportional to potential reach of a node. By reach we mean that if this node had a rumor, to how many other nodes could it spread the rumor. Now consider a retweet network in Twitter. Imagine the trendy gossiper starts on someone's Twitter page and then clicks on the profiles of other users in that user's Twitter feed who tweeted about some specific topic. This is how the gossiper does a random walk on Twitter. Now unlike the random surfer model, here the gossiper is biased towards users with more potential

reach. That is, biased towards people with more influence or fame. The rumor centrality of a user on this network tells us the fraction of time the gossiper lands on his or her Twitter profile page. Therefore, the higher a user's rumor centrality, the more likely he or she will be visited by this trendy gossiper and subsequently retweeted, and the more influential the user is. This interpretation gives an intuitive explanation for the TNG with rumor centrality model.

The trendy gossiper's bias for users with greater reach is an important difference from the random surfer's uniform transition probabilities. It provides a qualitative explanation for why social networks such as Twitter exhibit non-local phenomena such as superstars. The transitions depend on non-local information (reach/subtree size). In practice people do not ever actually calculate the reach of users on Twitter, but it may occur subconsciously. That is, people who are famous are more likely to have greater reach, and this may be why they are retweeted so often. Their approximate reach is known at least qualitatively through social interactions and exposure to media, and this may drive retweeting behavior on Twitter.

4.6 Trumor: A Dynamic Influence Tracking Engine

The accuracy with which the TNG with rumor centrality model describes retweet networks led us to develop a system for tracking influence on Twitter. Our system is called *Trumor* and works as follows. There is a Twitter crawler which collects tweets from the Twitter streaming API [5] and stores them in a database. A user goes to the Trumor homepage and enters a query along with a time window. Trumor then takes the following actions.

1. Construct the retweet network using all tweets in the database which match the user's query and time window.

2. Calculate the Trumor score of everyone on this retweet network.

3. Return to the user a list of all people on the retweet network, ranked by their Trumor score. Also return all tweets of these people corresponding to the user's query and time window.

There are two key ideas to Trumor. The first is the construction of dynamic, query specific retweet networks. This allows very precise influence information to be obtained. The second is the Trumor score, which provides the correct quantification of this influence. This score is based upon rumor centrality with an appropriate normalization. Very often there are multiple connected components in a retweet network. We scale the rumor centralities by the size of the connected components to obtain the Trumor scores. Assume a user v belongs to a certain connected component of the retweet graph G. Let us refer to the subgraph corresponding to this connected component as G_c and denote the number of nodes it contains as N_c node. Let the Trumor score of v be $TR(v, G)$. Then this score is given by

$$TR(v, G) = 2(N_c - 1)\frac{R(v, G_c)}{\sum_{u \in G_c} R(u, G_c)} \tag{4.9}$$

The Trumor score of a node is its attachment probability under TNG with rumor centrality, scaled by the size of its connected component. To understand this score better, consider the retweet network with three connected components in Figure 4-16. The Trumor scores of these nodes are shown in the figure as well. As can be seen, for this simple example, the Trumor score roughly corresponds to the degree of the nodes. An exception to this is node 2 which has a Trumor score higher than its degree. This is because this node has a strong strategic position, as it connects two hubs. Thus, the Trumor score does more than identify influential nodes by their degree. It also takes into account the global structure of the network. Also, it gives more weight to nodes in larger connected components. This makes sense, because these nodes have a larger potential reach than nodes in smaller connected components.

A screenshot of Trumor is shown in Figure 4-17. The search results are for the query "S&P" from August 6th, 2011 to August 7th, 2011. This topic is significant because on August 5th, the rating agency S&P downgraded the U.S. government's credit rating. There are 485 users in this retweet network. Trumor was able to process the query (construct retweet graph and calculate Trumor scores) and return the results in under one second on a standard PC with a 1.8 GHz processor. The Trumor results identify users who have been

heavily retweeted on this topic, such as BreakingNews and thinkprogress. We list the top Trumor scoring users for this query and time window in Table 4.3. To show the dynamic nature of influence that Trumor captures, we also show the Trumor scores for the same query, but from August 5th, 2011 to August 6th, 2011. The scores are relatively constant, but thinkprogress has a higher score than BreakingNews and WSJ's score is higher than CNBC. Therefore, we see that influence depends upon the time window used.

We compare these results with results from Twitter's search engine using its "top" search feature, which are shown in 4-18. The Twitter search results contain recent tweets (less than a few hours old), but only one heavily retweeted tweet (from tinyrevolution). In contrast, Trumor produces heavily retweeted users from over two days. Historical search is not available on Twitter's search engine, in contrast to Trumor. Twitter's search is geared more towards real time tweets than to historic influential tweets. Also, it is difficult to directly compare these results as Trumor uses the Twitter public streaming API which only provides 1% of the total volume of data on Twitter, so many tweets are missing from the Trumor database. Despite this sparsity of data, Trumor still produces relevant tweets from influential users, in contrast to Twitter which focuses on real time search.

The more important aspect of Trumor is the flexibility it provides the user as well as the quantitative information it provides. With Trumor, a user can obtain numerical values for the influence of users on specific topics. This information can be used for many different applications. For example, the Trumor scores can be used to determine how to allocate resources for users in any sort of marketing campaign. Also, tracking the scores over time can allow a user to determine whose influence is rising or falling. We emphasize again that all this information is topic specific, thereby providing very fine grained influence information.

Finally, we note that the Trumor scores are based on rumor centrality, which as we saw were proportional to retweet probabilities in the TNG with rumor centrality model. Therefore, one could potentially use the Trumor scores to predict the reach of tweets by different users. The Trumor scores obtain the retweet probabilities using only the retweet network structure. Our empirical analysis indicates that these probabilities are accurate and have the potential to be incorporated into a system that can predict retweets and reach on

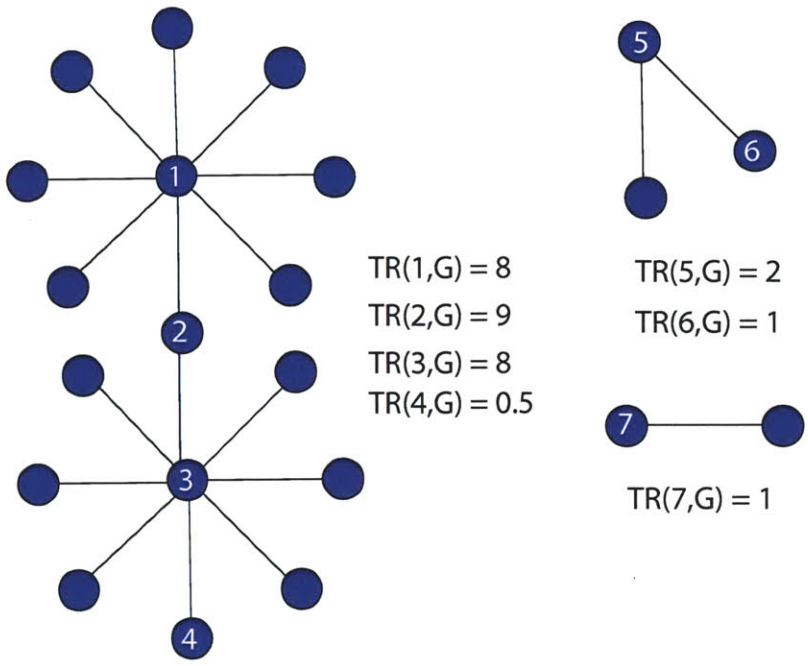

Figure 4-16: An example network with the corresponding Trumor scores for select nodes.

User	Trumor Score 8/6/11 to 8/7/11	Trumor Score 8/5/11 to 8/6/11
BreakingNews	19.0	20.0
thinkprogress	18.0	23.0
cnnbrk	16.0	17.0
Reuters	12.0	15.0
fivethirtyeight	12.0	12.0
CNBC	8.0	8.0
WSJ	7.0	9.0

Table 4.3: Top Trumor scoring users for query "S&P".

Twitter.

4.7 Chapter Summary

We have studied the problem of finding influentials in Twitter in this chapter. Our approach was empirically based, using real retweet networks from Twitter. We found that these networks exhibited two important properties: a power-law degree distribution and a super star node. These properties emerged in retweet networks on different topics ranging from music to sports. Power-laws have been observed in other networks and were explained

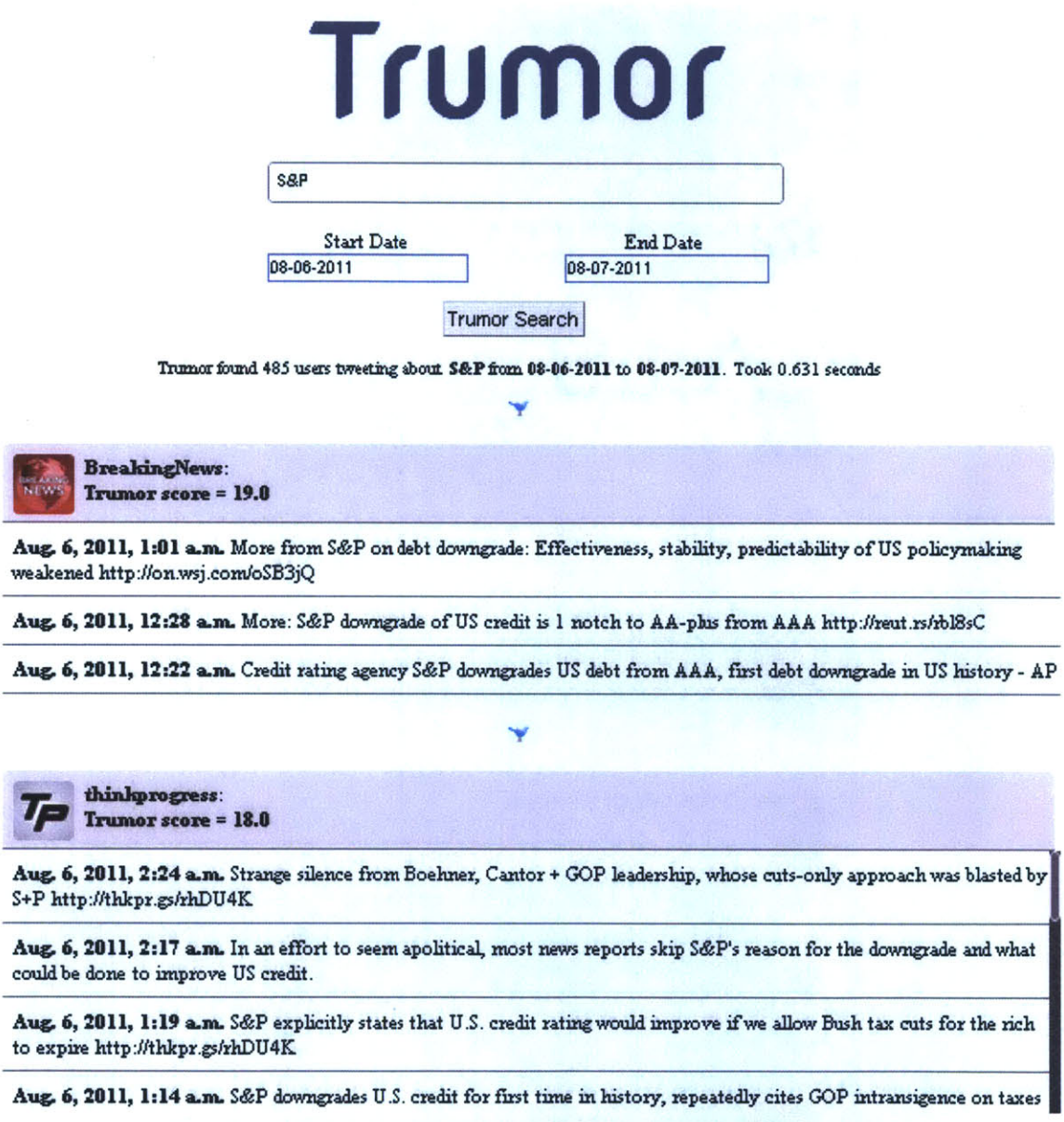

Figure 4-17: A screenshot of Trumor search results for the query "S&P" from August 6th, 2011 to August 7th, 2011.

Figure 4-18: A screenshot of Twitter search results for the query "S&P" on August 7th, 2011.

using the PA model. However, the superstar is a non-local property which has not been observed before.

To explain the superstar and the power-law, we proposed the TNG model which allowed one to grow a network using an arbitrary influence measure. If one used degree centrality, then this model reduced to PA. However, if one used rumor centrality as the influence measure, then this model produced both the superstar and the power-law. Even more striking, this model accurately described the degree distribution of the actual retweet networks across different topics. This suggested that rumor centrality is the proper influence measure for Twitter.

We analyzed the TNG with rumor centrality model to understand the origins of the superstar and power-law. Analysis of this model is quite challenging due to the global nature of rumor centrality. We conjectured that there was a certain fixed point structure that emerged in this model, and we were able to prove that this structure was invariant. Having established the invariance of the fixed point structure, we then went on to show how it led to the observed superstar and power-law. It is still an open question as to whether or not there are other fixed point network structures for this model.

Motivated by the success of TNG with rumor centrality to model Twitter, we created a dynamic influence tracking engine for Twitter based upon rumor centrality which we called Trumor. We found that Trumor does an effective job of identifying influential individuals in Twitter. More importantly, it provides quantitative measures of influence for users specified by time window and topic. The scores provided by Trumor are related to retweet probabilities in TNG with rumor centrality, so it would be an interesting next step to use these scores to actually predict retweets and the reach of users on Twitter.

Chapter 5

Learning Community Structure

This chapter focuses on the problem of learning the community structure of a network. In computer science this problem is also known as clustering a network and it arises in many different contexts. The term community makes sense when one speaks of people in a social network, while the term cluster is more general and can be used for networks where the nodes are not necessarily people. We will use the terms communities and clusters interchangeably here.

To understand the type of situation where we wish to find communities, consider the following example. N people participate in a college culture show that contains k different performances, with people allowed to be in more than one performance. Imagine that we only know which people were in the same performance, but we do not know which performance. Then can we recover who was in each performance? That is, can we find the overlapping clusters that correspond to the performances?

The people and performances form a bipartite network, with people and performances being the two classes of nodes with an edge connecting each performance with the people who performed in it as shown in Figure 5-1. If we project this network onto the N people we obtain a network of N nodes with an edge between people in the same performance, also shown in Figure 5-1. In the original bipartite network, each performance node corresponds to a cluster which becomes a clique in the projected network. Therefore, the projected network is a collection of k cliques which can overlap if people are in more than one performance. This type of network structure is typical of social networks. The

performance nodes generalize to any type of social group such as family members, friends from work, etc. In practice, one does not have the bipartite network of people and social groups. Instead, one usually has access to the projected network of social ties (consider for example friends on Facebook). What we would like is an algorithm that can efficiently find the overlapping clusters in this type of network without any extra information such as the number of clusters.

To solve this problem we create the *leader-follower algorithm* (LFA) for finding overlapping clusters in networks. The algorithm is based upon the notions of leaders (nodes connecting clusters) and followers (nodes unique to a cluster). It makes use of network centrality in a very different manner from what we have seen in Chapters 2 and 3. The LFA uses the very simple degree centrality in a clever manner to identify these leaders and followers. Earlier we had seen how the actual value of rumor centrality was used to identify the most likely rumor source or quantify the influence of individuals. In the LFA, we use degree centrality in a differential manner. That is, we use local minima of degree centrality to separate leaders from followers. The actual value of the centrality is not important. Rather, it is the local minima that is important.

The use of degree centrality in this manner provides several advantages for the LFA. First, it makes the algorithm truly nonparametric, requiring no knowledge of the number or size of clusters in the network or any other parameter value. Second, because we are using a network centrality as simple as degree, the algorithm is very fast, having a runtime linear in the network size. We are also able to establish theoretical conditions under which the LFA produces exact clusterings and approximate clusterings. Experiments performed on real networks show that the LFA produces better clusterings than other, more common algorithms such as spectral clustering. We use the LFA to perform exploratory analysis on a Facebook network and a human brain network where there is no true clustering known. For the Facebook network, it reveals relevant social groups and for the human brain it reveals different functional lobes.

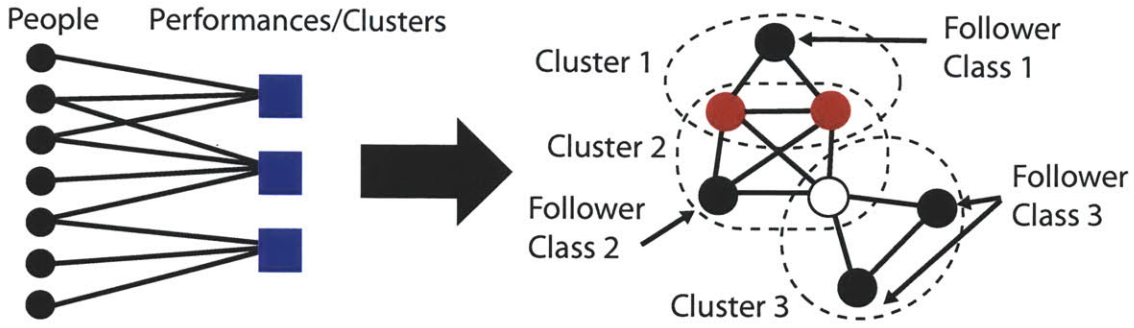

Figure 5-1: (left) Bipartite network of people and performances/clusters. (right) Projected network where the clusters are cliques. The follower classes are labeled for each cluster.

5.1 Previous Work

5.1.1 Spectral Approaches

One of the most common methods for clustering networks or graphs is spectral clustering. This algorithm is based on minimizing weighted graph-cuts [38], [57], which is in general NP-hard [26], but an approximation can be found using spectral clustering. Here one finds the eigenvectors of the graph Laplacian matrix and clusters them using techniques such as k-means clustering. Spectral clustering can be used for clustering data points with non-convex clusters if one uses a network derived from the data points [52]. Theoretical analysis of spectral clustering has been done to understand for what types of networks it works well [52], [48]. These results basically show that spectral clustering is best for graphs with densely connected clusters that are sparsely connected to other clusters.

The one drawback of spectral clustering is that it is a parametric algorithm that requires one to specify the number of clusters. This is a problem in situations where this information is not known beforehand. It would be ideal to have an algorithm that could learn this information from the network structure. Also, we will see in Section 5.5 that spectral clustering has difficulty resolving smaller clusters in densely connected networks.

5.1.2 Bayesian Approaches

Statistical inference is another approach for clustering that involves using nonparametric Bayesian techniques [17], [60]. While these methods allow for the number of clusters

to be learned from the data, they still involve specifying some distribution parameters. Also, the statistical inference approach could end up being very slow for large datasets. Nonparametric Bayesian methods generally involve calculating complex, high dimensional sums or integrals to find posterior distributions. Approximate methods such as Markov chain Monte Carlo (MCMC) [49] or variational inference [61] are generally used. Even these can be computationally demanding. Using MCMC involves running a Markov chain to generate samples of the posterior distribution over cluster assignments. It may take a long time for the Markov chain to reach its stationary distribution (burn-in) and a large number of samples may need to be generated to get a good approximation to the posterior distribution. Variational inference is generally faster that MCMC, but it is not guaranteed to recover the posterior distribution and also requires one to solve a large optimization problem that may contain several local maxima. Therefore, for clustering large networks these techniques may not scale with the data size.

5.1.3 Modularity Optimization

Another approach to clustering is based on optimizing a certain function known as modularity [51]. Modularity is a function of a network $G = (V, E)$ and a partition of the nodes of the network $\mathcal{C} = \{C_1, C_2, ...C_k\}$. Let the cluster membership of a node v be $c(v)$. For example, if node v is in cluster C_1, then $c(v) = C_1$. Let $\mathbf{A}$ be the adjacency matrix of the network. That is, $A_{uv} = 1$ if $(u, v) \in E$, and 0 otherwise. Also, let $\deg(v)$ denote the degree of node v. Then the modularity of the partition $\mathcal{C}$ of G is given by

$$M(\mathcal{C}, G) = \frac{1}{2|E|} \sum_{u,v \in V} \left[A_{uv} - \frac{\deg(u)\deg(v)}{2|E|} \right] \delta\left(c(u), c(v)\right) \tag{5.1}$$

where $\delta\left(c(u), c(v)\right) = 1$ if u and v are in the same cluster and 0 otherwise. Modularity is generally maximized when densely connected nodes are placed together in the same cluster. Therefore, by finding the clustering or partition that maximizes modularity, one expects to obtain a good clustering of the network.

The difficulty with modularity is that optimizing it is NP-complete [23]. Therefore, one must generally resort to approximation schemes. There are spectral based techniques

for optimizing modularity [50] as well as greedy optimization schemes [18]. However, no theoretical guarantees are known for these techniques.

In addition to being hard to optimize, there are arguments that optimizing modularity may not even produce a good clustering. For instance, modularity has a well known resolution limit. It has been shown that on large networks composed of many small clusters, the partition which maximizes modularity ends up grouping the smaller clusters into larger ones [31]. Therefore, for large networks, optimizing modularity may not detect finer structure and end up merging smaller clusters with larger ones. Ideally, we would like a clustering method that is scale invariant and can find both large and small clusters.

5.1.4 Correlation Clustering

Another type of clustering problem involves networks with positive and negative weights on edges. Consider the following scenario. One is given a network which is a complete graph with edges labeled $+$ and $-$. The goal is to cluster the nodes of the graph such that within a cluster either the number of disagreements ($-$ edges) are minimized or the number of agreements ($+$ edges) are maximized. This is known as *correlation clustering* because one thinks of the edge weights as correlation values. One wishes to cluster positively correlated nodes together and separate negatively correlated nodes.

It has been shown that correlation clustering is NP-complete [13]. However, the authors are able to obtain a constant factor approximation for the problem of minimizing disagreements on a complete graph. That is, if the minimum number of disagreements for the graph is m^*, then their algorithm produced a clustering with Km^* disagreements for some constant $K > 1$. They also obtained a polynomial time approximation scheme (PTAS) for maximizing agreements. In [6] the authors present a very simple randomized algorithm for obtaining a 3-approximation for minimizing disagreements.

These algorithms for correlation clustering have the common feature of being non-parametric. They can find any number and size of clusters and do not take any inputs. The basic idea of these algorithms is to pick a random node as a *pivot* and form a cluster from it and its neighbors with positive edge weights. Here we see how they learn the number of

clusters from the network data. They do not assume any number of clusters a priori, but rather this number emerges as the algorithm is run. Now correlation clustering is concerned with weighted edges on a complete graph. However, we are looking at networks which are not complete graphs and have unweighted edges, such as Facebook. Therefore, we cannot directly apply these algorithms. However, we will see that the LFA uses a similar technique as these algorithms to learn the number of clusters naturally from the network structure. The only difference is that the LFA uses degree centrality in a novel way to decide which nodes are the *pivots*, which we refer to as *leaders*.

5.1.5 Overlapping Clusters

Many of the methods for clustering focus on disjoint clusters. However, overlapping clusters will be a natural feature of social networks, so methods are needed which go beyond disjoint clusters. For finding general overlapping clusters in networks, the k-clique percolation method [54] can be used. This algorithm searches for k-cliques in a network, where a k-clique is defined as a subgraph of the network where each node is part of a clique of size k and one can reach any node in the subgraph via a path of cliques which share $k-1$ nodes. The main drawback of this method is that it must be given the size of the k-cliques and it is hard to know which value of k is optimal. This is similar to the type of challenge encountered with spectral clustering. Our goal is to have an algorithm which is completely parameter free and can learn all of the clusters using only the network data.

Despite requiring an input parameter, k-clique percolation does provide important insights for overlapping clusters. These insights will be a key aspect of the LFA. However, we will show how to find overlapping clusters without needing to specify any k-clique size by using degree centrality in a novel manner.

5.2 Intuition for Overlapping Clusters

In this section we will examine the structure of overlapping clusters in order to understand their detectable features. Clusters are typically densely connected subgraphs of the underlying network. Consider for the moment that the clusters are cliques. In this simplified

scenario, what will distinguish the individual clusters? An example of a network with three overlapping clique clusters is shown in Figure 5-1. Notice that each cluster has some nodes that are shared between clusters, and some that are only contained within the cluster. We refer to the nodes with no neighbors outside the clusters as *followers*, and those with connectivity between clusters as *leaders*. The followers within a cluster form what we refer to as a *follower class*. In this simple network, the follower classes identify the clusters. A leader node's neighboring follower classes identify its cluster memberships. For example, in the figure, the node colored white is a member of the clusters defined by follower classes 2 and 3.

If we could identify the follower classes, we could then find the clusters. In the network in Figure 5-1 each leader has a higher degree than at least one neighbor. This is due to the fact that within a cluster all nodes are connected, but the leaders additionally have connections to other clusters. Therefore, in this network we could identify leaders as nodes with degree greater than at least one neighbor, and any node without this property would be a follower. Also note that the nodes that constitute a follower class in these clique clusters form a clique themselves. Therefore, once followers are found, a follower class can be identified as a follower and its neighboring followers.

In the next section we will use this intuition for overlapping clusters to develop the LFA for finding clusters in networks. It is an efficient and non-parametric algorithm. While the intuition upon which the algorithm is based is in the context of clusters which are cliques, we will see in Section 5.4 that the algorithm produces very good clusterings even in networks where the clusters are not cliques.

5.3 Leader-Follower Algorithm

We now present the *leader-follower algorithm* (LFA) for detecting overlapping clusters in networks. Roughly speaking, there are three stages in the algorithm. First, we distinguish leaders from followers in the network. Second, we identify all follower classes. Third, we assign leaders to follower classes in order to form the clusters. We now describe these steps in more detail. Figure 5-2 shows the steps of the LFA on an example network. Throughout

this section we will assume we are looking for clusters in a network $G(V, E)$ with vertex set V and edge set E. Also, we will define the degree of a node v as $\deg(v)$ and the set of neighbors of node v as $\mathcal{N}(v) = \{u \in V : (u, v) \in E\}$.

5.3.1 Leader and Follower Detection

We use degree centrality to identify leaders and followers. We first calculate the degree of each node in the network. Then, leaders will be any node whose degree is greater than at least one of their neighbors, and the remaining nodes will be followers. Thus, we identify local minima of degree centrality as followers. This local minima idea is an important concept for the LFA. It says that we can detect clusters by the presence of these locally less important nodes, where importance is measured by degree. In essence, these local minima of degree centrality (followers) are the detectable signal of a cluster. In Figure 5-2, nodes 1, 2, and 3 are identified as leaders, and the remaining nodes are followers.

Summary of Leader and Follower Detection Step

1. For each $v \in V$, calculate its degree $\deg(v)$.
2. Define leader nodes as $\mathcal{L} = \{v \in V : \text{there exists } u \in \mathcal{N}(v) \text{ such that } \deg(v) > \deg(u)\}$ and follower nodes as $\mathcal{F} = V \backslash \mathcal{L}$.

5.3.2 Follower Class Detection

Next we must partition the followers $\mathcal{F}$ into follower classes. We do this by starting with the highest degree follower. This node and all of its neighboring followers form the first follower class. We remove all of these nodes from $\mathcal{F}$. We then repeat this process with next highest degree node in $\mathcal{F}$ and continue until $\mathcal{F}$ is empty. At the end of this stage, we return all follower nodes back to $\mathcal{F}$. We will end up with a follower class for each time we remove a follower and its neighboring followers from $\mathcal{F}$. In Figure 5-2 there are four different follower classes identified.

Summary of Follower Class Detection Step

1. Let the follower class counter $i = 1$. While $\mathcal{F} \neq \emptyset$ do the following:

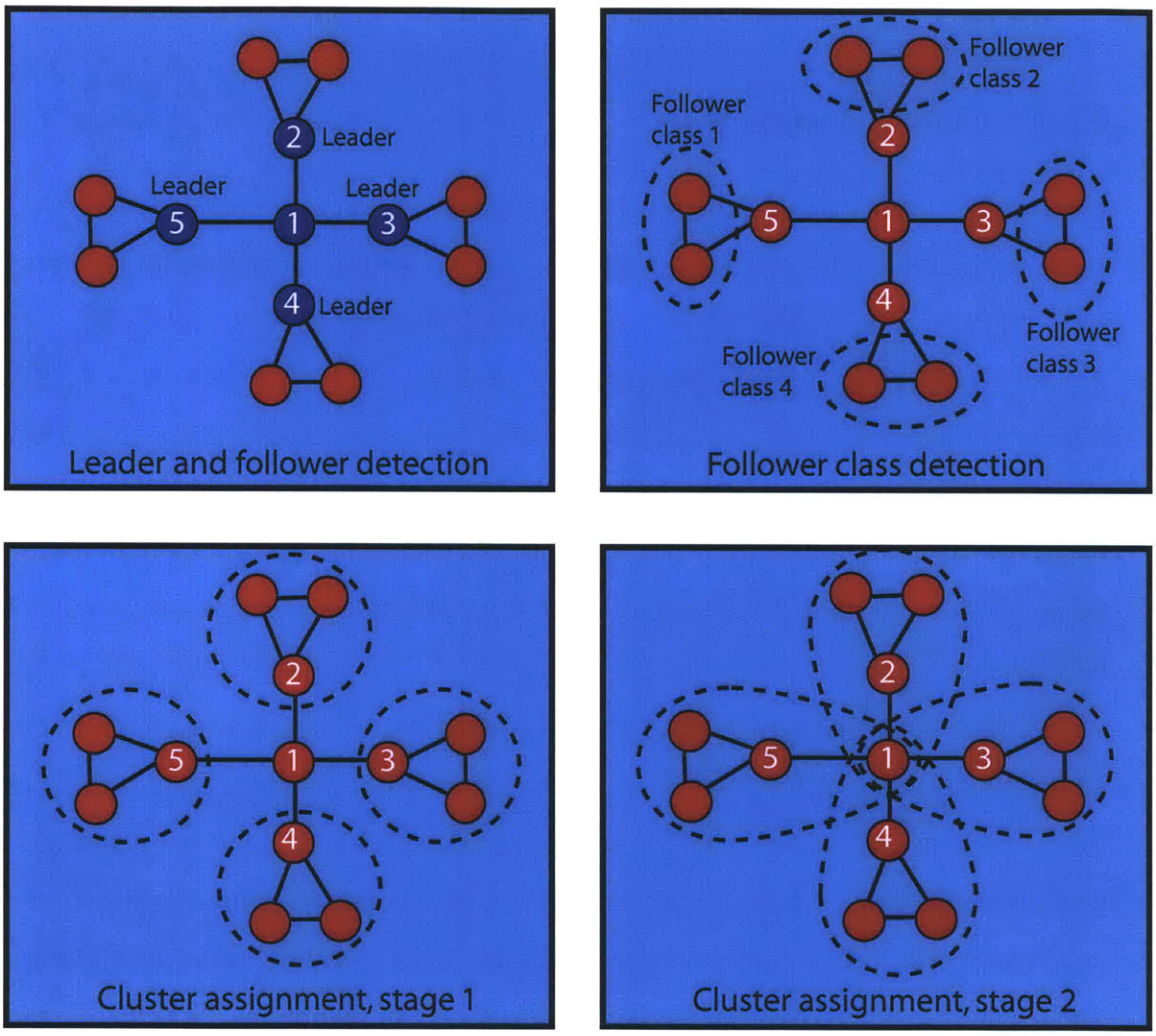

Figure 5-2: Steps of LFA on example network. (top left) Leader and follower detection step, (top right) follower class detection step, (bottom left) first stage of cluster assignment step, (bottom right) second stage of cluster assignment step.

- Let $f_i = \arg\max_{f \in \mathcal{F}} \deg(f)$
- Define follower class i, $\mathcal{F}_i$ as $\mathcal{F}_i = \{f_i\} \bigcup \{\mathcal{F} \bigcap \mathcal{N}(f_i)\}$.
- Set $i = i + 1$ and set $\mathcal{F} = \mathcal{F} \backslash \mathcal{F}_i$.

2. Set $\mathcal{F} = \bigcup_{j=1}^{i-1} \mathcal{F}_j$.

5.3.3 Cluster Assignment

Finally, with leaders and follower classes identified, we can assign nodes to clusters. Assume we found k follower classes. Then we define a cluster membership function $M : V \rightarrow \mathcal{P}(\{1, 2, ..., k\})$ where $\mathcal{P}(\{1, 2, ..., k\})$ is the power set of $\{1, 2, ..., k\}$. This function initially has value $\{i\}$ for any follower in follower class $\mathcal{F}_i$ ($i = 1, 2, ..., k$), and has value $\emptyset$ for any leaders. What $M(v)$ does is map each node v to a subset of the k clusters. $M(v)$ maps nodes to the power set $\mathcal{P}(\{1, 2, ..., k\})$ because we allow for overlapping clusters, so nodes can have multiple cluster memberships. Initially, all followers belong to the cluster associated with their respective follower class and all leader nodes have a null cluster membership.

We start by defining two queues $\mathcal{B}' = \emptyset$ and $\mathcal{B} = \mathcal{F}$ with the nodes of $\mathcal{F}$ ordered arbitrarily. We also define a set $\mathcal{A} = \mathcal{F}$ which consists of all nodes assigned to clusters. First we will use followers to assign leaders to clusters. While $\mathcal{B}$ is not empty, we pop each node from the front of $\mathcal{B}$ and pass its cluster membership to its neighbors not in $\mathcal{A}$. Any node which absorbed a membership is then added to the end of $\mathcal{B}'$ if it is not already in $\mathcal{B}'$. The result of this part of the clustering stage is shown in Figure 5-2. Nodes 1, 2, 3, and 4 join their neighboring follower classes.

Next, we will cluster the remaining nodes. We first add all nodes in $\mathcal{B}'$ to $\mathcal{A}$. Then, while $\mathcal{B}'$ is not empty, we pop each node from the front of $\mathcal{B}'$, add it to $\mathcal{A}$, and pass its cluster membership to its neighbors not in $\mathcal{A}$. Any node which absorbed a membership is added to the end of $\mathcal{B}'$ if it is not already in $\mathcal{B}'$. We continue doing this until $\mathcal{B}'$ is empty, at which point all nodes will be assigned to at least one cluster and the LFA terminates. In Figure 5-2 this part of the clustering stage puts node 5 in each four of the follower classes.

We have a slight difference in these two stages because we want leaders neighboring followers to receive cluster memberships from followers only. The first stage lets leaders neighboring followers absorb multiple cluster memberships, while the second stage prevents these nodes from absorbing any more cluster memberships.

Summary of Cluster Assignment Step

1. Define a set $\mathcal{A} = \mathcal{F}$ and two queues $\mathcal{B}' = \emptyset$ and $\mathcal{B} = \mathcal{F}$ with the nodes of $\mathcal{F}$ ordered arbitrarily. Define $M : V \rightarrow \mathcal{P}(\{1, 2, ..., k\})$ as follows

$$M(v) = \left\{ \begin{array}{ll} \{i\} & \forall v \in \mathcal{F}_i, i = 1, 2, ..., k \\ \emptyset & \text{else} \end{array} \right\}.$$

2. While $\mathcal{B} \neq \emptyset$ do the following:

   - Pop node v from front of $\mathcal{B}$.
   - For each node $u \in \mathcal{N}(v) \bigcap \mathcal{A}^c$, set $M(u) = M(u) \bigcup M(v)$ and add u to end of $\mathcal{B}'$ if it is not already in $\mathcal{B}'$.

3. Set $\mathcal{A} = \mathcal{A} \bigcup \mathcal{B}'$. While $\mathcal{B}' \neq \emptyset$ do the following:

   - Pop node v from front of $\mathcal{B}'$ and set $\mathcal{A} = \mathcal{A} \bigcup v$.
   - For each node $u \in \mathcal{N}(v) \bigcap \mathcal{A}^c$, set $M(u) = M(u) \bigcup M(v)$ and place u at end of $\mathcal{B}'$ if it is not already in $\mathcal{B}'$.

5.3.4 Runtime Analysis

We now analyze the runtime of the LFA on a network $G(V, E)$ with $|V|$ nodes and $|E|$ edges. We establish the following result for the LFA runtime, which is proved in Section 5.6.

Theorem 13. *For an input network $G(V, E)$, the LFA will terminate in $O(|E|)$ time.*

This is very fast and represents the limit of how fast clustering algorithms can run because $|E|$ is the size of the input network. Also note that the runtime does not depend on any other input parameters such as the number of clusters. This is in contrast to spectral clustering, where finding k clusters has a runtime of $O(k|E|)$ [64]. This could result in a very slow runtime if the number of clusters k scaled with the network size, in contrast to the LFA which is independent of the number of clusters found.

To show the scale of network that can be efficiently clustered with the LFA, we compare the algorithm's runtime against spectral clustering for synthetic networks of various size which contained randomly connected cliques. Table 5.1 shows the number of vertices ($|V|$), edges ($|E|$), and clusters (k) in each network, along with the runtime of the LFA and spectral clustering on these networks. These experiments were both run on MATLAB on a Pentium Core 2 Duo 1.8 GHz desktop machine to allow for fair comparison. The true number of clusters was given as the input to the spectral clustering algorithm. As can be seen, for a very large network with over $440,000$ nodes and 3 million edges, the LFA runs in 4 minutes, while spectral clustering could not finish in an hour. The speed-up of the LFA as compared to spectral clustering is due to the inherent simplicity of the algorithm. Whereas spectral clustering involves finding eigenvectors of the network and applying an algorithm such as k-means clustering, the LFA only involves simple edgewise comparisons.

Table 5.1: Runtime of the LFA and spectral clustering on synthetic networks

$\|V\|$	$\|E\|$	k	Spectral Clustering Runtime	LFA Runtime
636	16,750	100	1.4 seconds	0.4 seconds
5,154	138,080	500	4.3 minutes	1.5 seconds
7,525	202,272	1,000	16.9 minutes	2.5 seconds
74,750	715,572	10,000	$>$ 60 minutes	1.1 minutes
149,758	1,434,544	20,000	$>$ 60 minutes	3.8 minutes

5.4 Theoretical Guarantees

We will next establish theoretical performance guarantees for the LFA. The main result presented here is that when the clusters are clique-like, then as long as they have followers, the LFA can produce a good *approximate* clustering. As a corollary to this, we show that

the LFA has *exact* detection in networks where the clusters are cliques with followers. Our results show that the LFA is a robust algorithm capable of producing good clusterings for a broad class of networks.

5.4.1 Clustering Distance Function

First, in order to compare clusterings, we need to define an appropriate distance function. This is difficult because we are allowing overlapping clusters. We choose to follow a simple matching approach from [11] with a penalty for any excess clusters. We refer to a clustering $\mathcal{C}$ as a set of clusters: $\mathcal{C} = \{C_1, C_2, ..., C_k\}$. We define the following distance function for two clusterings $\mathcal{C}$ and $\mathcal{C}'$ on a network with $|V| = N$ nodes. Let $k = |\mathcal{C}|$ and $k' = |\mathcal{C}'|$ and assume that $k \leq k'$. Then we define the distance between these clusterings as

$$d(\mathcal{C}, \mathcal{C}') = \frac{1}{kN} \min_{\sigma \in I_{kk'}} \sum_{i=1}^{k} \left(|C_i| + |C'_{\sigma(i)}| - 2 \left| C_i \bigcap C'_{\sigma(i)} \right| \right) + \frac{|k' - k|}{N} \tag{5.2}$$

where $I_{kk'}$ is the set of injections from k to k' elements. The first term in this distance is the average fraction of erroneous nodes per cluster under an optimal injection from $\mathcal{C}$ to $\mathcal{C}'$. The second term is the number of remaining unmatched clusters in $\mathcal{C}'$ normalized by the network size N. With this distance measure, we define the closeness of two clusterings.

Definition 4. *Two clusterings $\mathcal{C}$ and $\mathcal{C}'$ are ϵ-close if $d(\mathcal{C}, \mathcal{C}') \leq \epsilon$.*

5.4.2 Clustering Accuracy

In [11], if a set of data points satisfied certain properties, then a graph could be constructed from the data that would satisfy certain structural properties which allowed for accurate clustering. We will focus on this graph structure and slightly modify it. We say this modified graph structure satisfies the ϵ-clique with follower property, which we now define.

Definition 5. *Let $\epsilon \geq 0$. A network $G(V, E)$ satisfies the* ϵ-clique with follower *property if its nodes are in k (possibly overlapping) clusters $C_1, C_2, ...C_k$ such that $V = \bigcup_{i=1}^{k} C_i$ and each cluster C_i satisfies the following conditions.*

1. *C_i contains a set of nodes $X_i \subseteq C_i$ which form a clique. These are* good *nodes.*

2. *Let the $B = V \setminus \bigcup_{i=1}^{k} X_i$ be the set of* bad *nodes. Then $|B| \leq \epsilon |V|$.*

3. *There is at least one node $f_i \in X_i$ such that for all $y \in V \setminus X_i$, $(f_i, y) \notin E$.*

The first two conditions of this network are the same as the network in [11]. They state that the network has an ϵ fraction of *bad* nodes which are not in cliques and the remaining $(1-\epsilon)$ fraction of *good* nodes form cliques within their clusters. We add the third condition to the network in [11] in order to require each cluster to have a follower. Also, we remove the condition in [11] that required that there be no edges between good nodes in different clusters. We allow good nodes to have neighbors in other clusters because we are allowing overlapping clusters, whereas in [11] the focus was on disjoint clusters.

In this network structure clusters are not cliques, but clique-like. Even under this relaxed condition, the LFA can still find a good approximation to the true underlying clustering. Formally, we establish the following result, which we prove in Section 5.6.

Theorem 14. *For a network which satisfies the ϵ-clique with follower property, the LFA will find a clustering that is 2ϵ-close to the true clustering.*

If we set $\epsilon = 0$ then we obtain a network in which each cluster is a clique with at least one follower. For these networks, we are able to obtain exact clustering.

Corollary 3. *For a network which satisfies the ϵ-clique with follower property for $\epsilon = 0$, the LFA will find a clustering that is equal to the true clustering.*

While having $\epsilon = 0$ may seem like a stringent condition, we will see in Section 5.5 that this network structure is indeed observed in real social networks.

5.5 Empirical Results

5.5.1 Social Networks with Known Clustering

We first look at two networks of performers from a college culture show. The networks are from shows performed in 2010 and 2011. The 2010 network has 153 nodes (performers) and 15 clusters (performances) and the 2011 network has 138 nodes (performers) and

13 clusters (performances). An edge is placed between two performers if they are in the same performance. In this way, each cluster (performance) is a clique. We therefore expect the LFA to find those performances which have followers. The 2010 network is plotted in Figure 5-3. To visualize the overlapping cluster structure, we also plot the 2010 cluster overlap network where each node is a cluster with the node size proportional to the cluster size. There is an edge between clusters if they share nodes, and the edge thickness is proportional to the amount of overlap between the clusters. As can be seen, there is substantial overlap among the clusters of this network.

We show the error (as defined in equation (5.2)) of the LFA for these networks in Figure 5-4. For the 2010 network, the LFA finds all clusters exactly, while for the 2011 network, it detects all but one of the clusters exactly. We have exact detection for the 2010 network because it satisfied the ϵ-clique with follower property for $\epsilon = 0$. For the 2011 network, the cluster that the LFA did not detect lacked any followers and therefore we would not expect it to be detectable by the LFA.

We compared the LFA with spectral clustering on these networks. The error of spectral clustering as a function of the number of input clusters is shown in Figure 5-4. As can be seen, while spectral clustering has minimum error when it is given the true number of clusters as input, it still cannot produce the correct clustering. This is expected, because spectral clustering produces disjoint clusters, while this network has overlapping clusters. For comparison, the LFA error is also shown in this plot and is always less that the spectral clustering error.

We also compared the LFA to the k-clique percolation algorithm which was designed to find overlapping clusters in networks. The error of k-clique percolation as a function of the size of the k-cliques is shown in Figure 5-4. As can be seen, the error does not go to zero over the range of k-clique sizes used and is always greater than the LFA error. Therefore, we see that the LFA also has superior performance to this algorithm on these real networks.

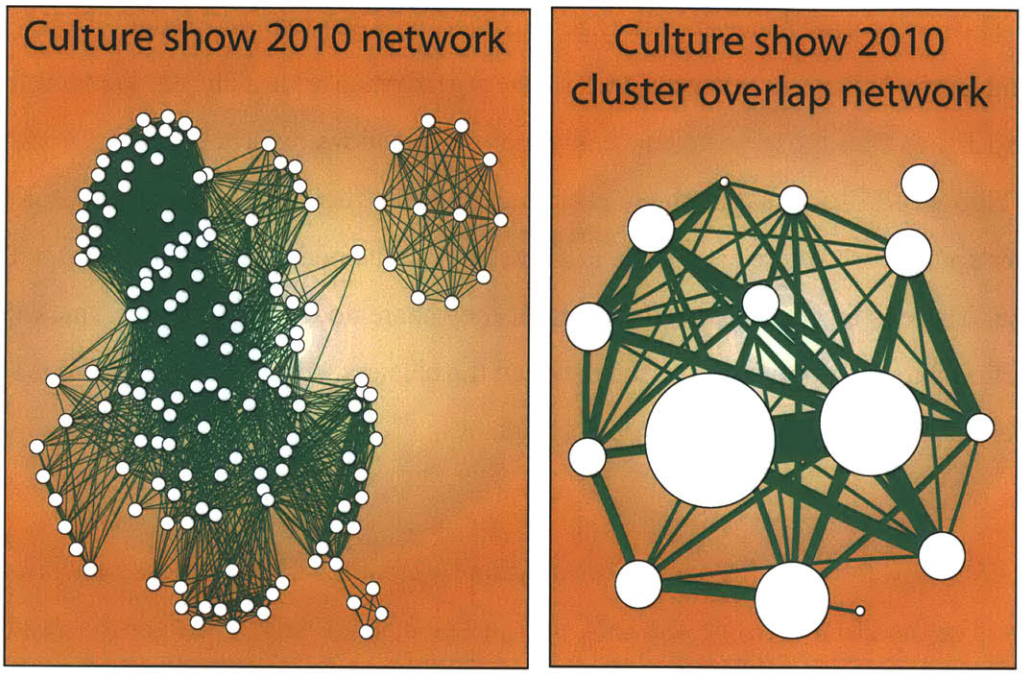

Figure 5-3: (left) 2010 culture show network and (right) 2010 true cluster overlap network.

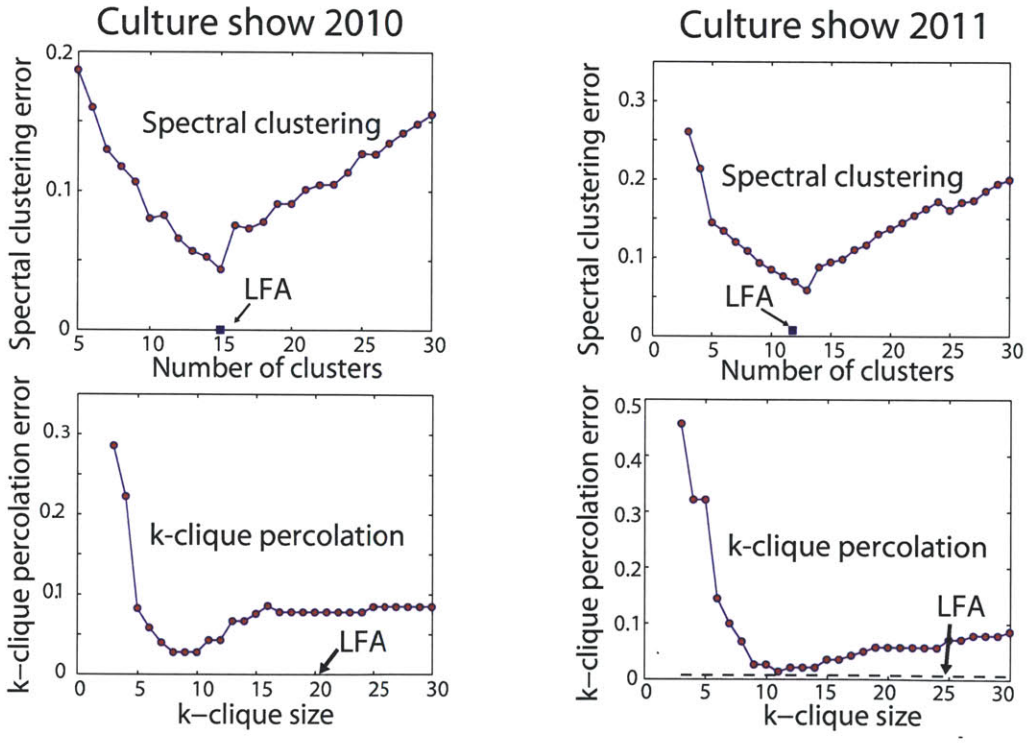

Figure 5-4: Error of spectral clustering versus number of input clusters and error of k-clique percolation versus k-clique size on (left) 2010 culture show network and (right) 2011 culture show network. The LFA error is also shown.

5.5.2 Exploratory Cluster Analysis

When there is no known clustering for a network, applying a clustering algorithm can help reveal the latent structure. We performed this type of exploratory analysis on two different networks. The first is a Facebook network, and the second is a human brain network.

First, we look at a real Facebook network which contains 131 nodes. This network does not have a true underlying clustering like the culture show networks. However, it is informative to see the different types of clusters found by the LFA. It finds 55 clusters in this network and we will look more closely at the clusters in two subgraphs of this network shown in Figure 5-5.

The first subgraph shown on the left of Figure 5-5 corresponds to a set of graduate students and contains four overlapping clusters. There are two people common to each of the four clusters, which correspond to different research groups. Two of the clusters actually correspond to a single research group, but the followers of these two clusters were not Facebook friends. This is an example of what the LFA does with incomplete network data. The other two clusters correspond to common friends the two people have in two other research groups.

The second subgraph shown on the right of Figure 5-5 has three overlapping clusters. There is a single person that is in all three of these clusters. The larger cluster corresponds to this person's college friends, while the other two correspond to two sets of friends from this person's home town. These hometown clusters also have a three node overlap.

We next applied the LFA to a human brain network obtained from fMRI data [3]. Edges in this network correspond to points in the brain whose correlation exceeded a threshold value. This network has 4221 nodes. There were 20 clusters detected by the LFA and much to our surprise, three of the larger clusters corresponded to major lobes of the brain which are related to different types brain functions: the frontal lobe, the occipital lobe, and the temporal lobe. This result indicates that the cluster structure we defined may apply to a broader class of networks beyond just social networks. Figure 5-6 shows the labellings of the clusters (lobes) we detected using the LFA. We also applied spectral clustering to this brain network with the number of clusters found by the LFA as input. Spectral clus-

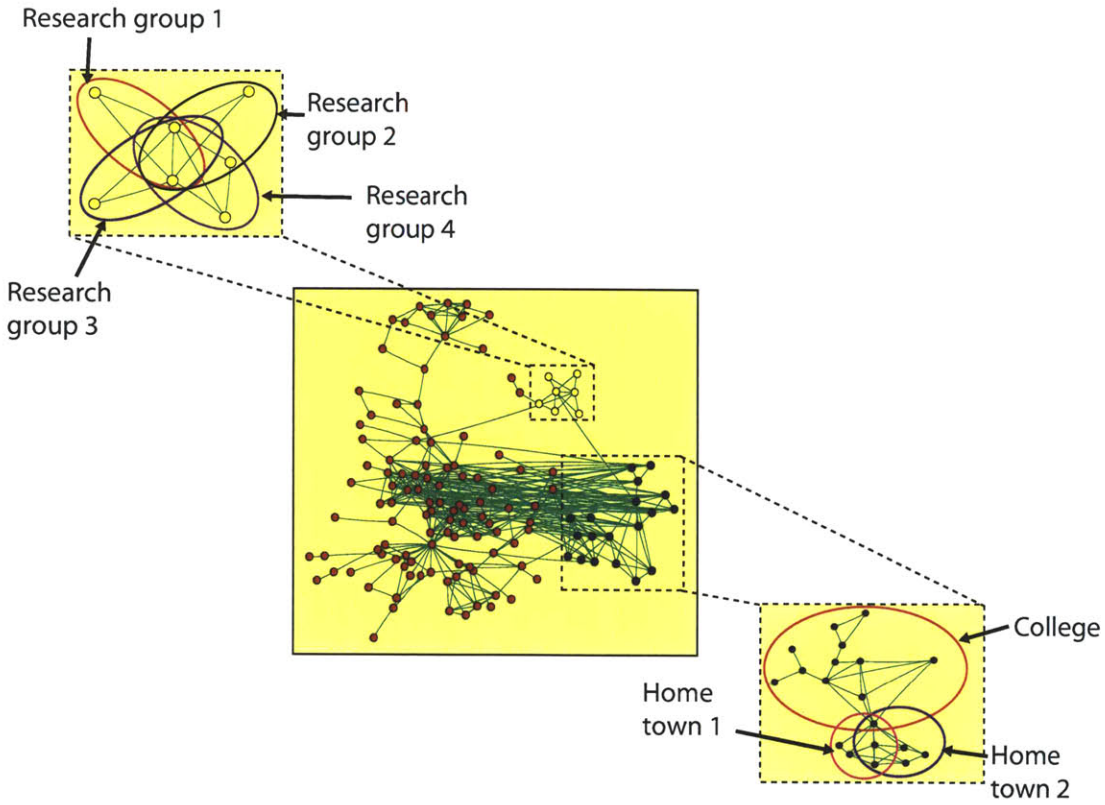

Figure 5-5: A Facebook network with two of its subgraphs expanded. The clusters found with the LFA in these subgraphs are labeled with the appropriate social identification.

tering was not able to resolve the temporal lobe in this network, but instead grouped it with the frontal lobe. Thus, we see the LFA resolving a finer cluster structure than spectral clustering.

5.6 Proofs

5.6.1 Proof of Theorem 13

We wish to find the runtime of the LFA, which we refer to as $T(LFA)$. The LFA has has three stages: leader and follower detection, follower class detection, and cluster assignment. We will look at the runtime of each stage individually first and then combine these runtimes to get the overall runtime of the LFA.

1. **Leader and follower detection:** In the leader and follower detection stage, each

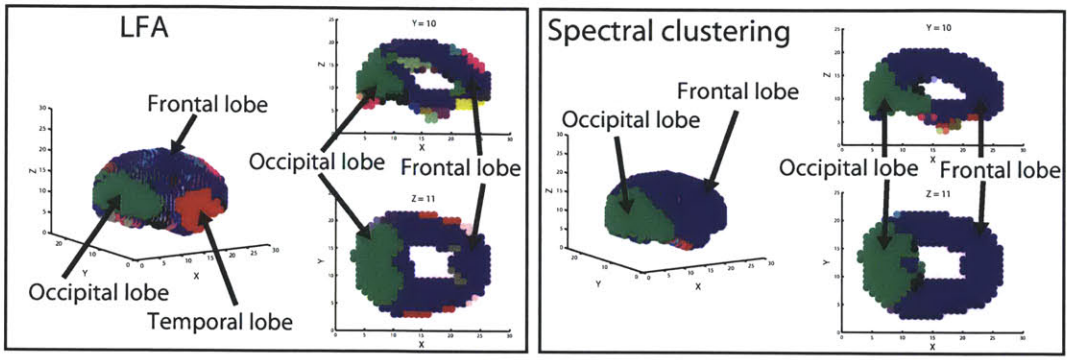

Figure 5-6: Lobes detected in a human brain network using (left) the LFA and (right) spectral clustering.

node must compare its degree with all of its neighbors. This stage involves calculating every node's degree, which will involve counting each edge at most twice, and then a maximum of two comparisons on each edge in the network to identify leaders and followers. Therefore, the leader and follower detection stage will take $O(|E|)$ time

2. **Follower class detection:** The second stage is the detection of follower classes. Again, it involves each follower checking if its neighbors are also followers. This step takes at most $O(|E|)$ time because it involves comparisons on each edge twice in the worst case.

3. **Cluster assignment:** The cluster assignment stage involves spreading cluster memberships through the network. As we go through the first queue $\mathcal{B}$, at each iteration we always remove one node from it. Each iteration involves checking the neighboring edges of a node. For the queue $\mathcal{B}'$ we will remove a node each time, and may or may not add a new node. Because each node can be in $\mathcal{B}'$ only once, we will never have more than $|V|$ iterations through the queue. Again, each iteration involves checking the neighboring edges of a node. Overall, this stage involves checking all edges of the network a constant number of times and in the worst case will have a runtime of $O(|E|)$.

Putting together the runtime of these individual stages, we obtain the runtime of the entire algorithm.

$$T(LFA) = O(|E|) + O(|E|) + O(|E|)$$
$$= O(|E|)$$

Therefore, the LFA runtime is $O(|E|)$.

5.6.2 Proof of Theorem 14

To prove Theorem 14 we first assume we have a network $G(V, E)$ that satisfies the ϵ-clique with follower property. Let the (possibly overlapping) true clusters be $\mathcal{C}_{true} = \{C_1, C_2, ...C_k\}$ such that $V = \bigcup_{i=1}^{k} C_i$. In the ϵ-clique with follower networks, the LFA will find at least k clusters. This is because each cluster C_i contains one good node with no neighbors outside of the clique of good nodes X_i. The degree of this node is less than or equal to all of its neighbors, and so it will define a follower class in the LFA. If there is more than one node in X_i with no neighbors outside X_i, then, because they form a clique, they will identify a follower class. This gives a lower bound of k follower classes.

Let us define k_{bad} as the number of follower classes in excess of the k true clusters. We then have that $k_{bad} \leq \epsilon N$. The upper bound for k_{bad} arises when each bad node forms its own follower class. This can happen if, for example each bad node has degree 1 and is only connected to a single good node.

Each of the k true clusters will contain all good nodes within the cluster. This is because each good node neighbors the follower class for the cluster, so they join this cluster in the assignment step. We can have at most ϵN incorrect nodes in each of these k clusters. This occurs if all bad nodes are assigned to one of the true clusters. The maximum error from excess clusters occurs when k_{bad} is equal to its maximum value of ϵN. With these bounds, the distance between the LFA clustering $\mathcal{C}_{LFA}$ and the true clustering $\mathcal{C}_{true}$ is upper

bounded by

$$\begin{aligned} d(\mathcal{C}_{LFA}, \mathcal{C}_T) &\leq \frac{k\epsilon N}{kN} + \frac{k_{bad}}{N} \\ &\leq \epsilon + \frac{\epsilon N}{N} \\ &\leq 2\epsilon \end{aligned}$$

5.7 Chapter Summary

We presented here the leader-follower algorithm (LFA) for learning community structure in networks. The algorithm allows one to find overlapping clusters in a network, an important aspect of real networks, especially social networks. The algorithm requires no input parameters and learns the number of clusters naturally from the network. It accomplishes this using degree centrality in a clever manner. It identifies local minima of degree centrality as followers which belong only to one cluster, and the remaining nodes are leaders which connect clusters. In this way, the number of clusters can be learned using only the network structure.

We proved that the LFA finds good approximations to the true clustering for a fairly broad class of networks. This class of networks possess properties that are typical in real social networks. Empirical comparisons on real social networks with more common clustering algorithms such as spectral clustering found that the LFA exceeded their performance. Exploratory analysis of social and biological networks showed that the LFA can reveal important communities or clusters in networks.

The LFA is also an extremely fast algorithm, having runtime linear in the network size. Thus, this algorithm can be used to efficiently cluster extremely large networks. This is an important aspect of the algorithm as modern networks are incredibly large and any practical clustering algorithm must be able to scale to these sizes.

In summary, we have shown that a centrality measure as simple as degree centrality can be used to construct an efficient, accurate, non-parametric community detection algorithm for networks.

Chapter 6

Conclusion

This thesis has shown the potential power of network centralities for solving network problems. We looked at finding rumor sources, measuring influence, and learning community structure. These are three very different problems which we were able to solve using network centralities. In each problem, we saw network centrality applied in a very different manner, but in each case leading to an effective solution.

We developed a new network centrality we called rumor centrality to find rumor sources in networks. Rumor centrality was proven to be an exact maximum likelihood estimator for random regular graphs. That is, the rumor centrality of a node was proportional to the probability of it being the rumor source. This showed that rumor centrality was in a sense the correct centrality measure for the rumor source detection problem. We showed an efficient distributed algorithm for calculating rumor centrality, demonstrating its practicality for large networks. We also saw many interesting connections of rumor centrality to Markov chains, combinatorics, and other network centrality measures.

The performance of the rumor centrality based estimator was thoroughly characterized for a wide array of networks. We found that there is an interesting threshold phenomenon for rumor source detection. For networks which grow like a line detection was impossible, but for networks which grow faster than a line non-trivial detection was possible. In fact, we were able to show that non-trivial detection of the rumor source was possible for any random spreading time distribution on any random tree which grew faster than a line. For random regular graphs with exponential spreading times, we were able to show that the

error of rumor centrality decayed exponentially fast. We used different analysis techniques to establish these results. For networks which grew polynomially, we used concentration based techniques, while for networks with expansion, we used techniques from the analysis of Polya urn models and branching processes. These analysis techniques represent an important contribution to not only the study of rumor source detection, but also other problems which involve networks with expansion.

We found another use for rumor centrality in the problem of measuring influence in networks. Specifically, to explain the structure of topic specific retweet networks in Twitter we developed the topological network growth model which could use different network centralities. We found that if one used rumor centrality, then this model produced two important features we observed in Twitter networks: a power-law degree distribution and a superstar node with very high degree. Using this observation, we argued that rumor centrality was correctly quantifying the influence of users on Twitter on specific topics. We then developed a dynamic influence tracking engine for Twitter we called Trumor which allowed one to measure the influence of users on Twitter specified by topic and time.

The final problem we studied was that of learning the community structure of a network. Using arguments based on social interactions, we determined that degree centrality could be used to detect communities. Specifically, we identified local minima of degree centrality as followers which are nodes which only belong to a single community, and any other node was a leader which connected communities. We used this to develop the leader-follower algorithm (LFA) which could learn the overlapping community structure in networks. The LFA is very fast, having a runtime linear in the network size. It is also non-parametric, in the sense that it can learn both the number and size of communities naturally from the network structure. We proved that it was very robust and learned accurate community structure for a certain class of network which is fairly broad and is observed in real social networks. We found that the LFA did a better job of learning community structure on real social networks than more common algorithms such as spectral clustering. Also, we showed that the LFA could learn relevant community structure in real networks such as Facebook and the human brain.

6.1 Future work

Going forward, there are several aspects of this thesis to build upon. We presented creative uses of network centrality for different problems. However, in doing so, we also opened up several interesting research questions.

For rumor source detection, an open theoretical question is to understand the error probability for rumor centrality on general networks. Thus far, we showed that the error decayed exponentially on random regular graphs. Also, we have seen empirically that on certain random trees the error does decay exponentially. Proving that this occurs for general networks would be an important advancement to this theoretical analysis. Also, our analysis here focused on trees and tree-like networks. It would be a major contribution to prove any type of result for non-tree networks.

We saw how rumor centrality could be used to measure influence on Twitter. Along the way to finding this result, we saw that rumor centrality was actually telling us the probability of a user being retweeted. This opens to door to the very interesting problem of predicting the spread of information on Twitter. That is, rumor centrality could be used to actually predict retweets and the subsequent spread of information. Coupling these probabilities with further temporal analysis of how quickly retweets occur could allow one to predict not only the reach of information, but also the speed with which it spreads. Predicting information spreading in social networks is by no means a simple problem, but our results in this thesis present one important step forward in formulating a solution.

To learn the community structure of networks, we developed the leader-follower algorithm (LFA) which used degree centrality. The LFA was designed for unweighted, undirected networks, as occur in social networks such as Facebook. A natural next step for this problem would be to extend this algorithm to weighted networks. The LFA was developed by understanding the form of communities in social networks. A similar type of analysis would be needed to understand the properties of communities in weighted networks. Once the structure of these communities is understood, an appropriate network centrality measure can be used or created in order to extend the LFA to weighted networks. Extending the LFA to weighted networks would greatly broaden its scope of application and provide

a deeper understanding of the structure of communities in more general networks.

Appendix A

Appendix

A.1 Proof of Theorem 10

We recall that Theorem 10 stated that for a rumor spreading for time t as per the SI model with a general distribution with mean spreading time μ the rumor graph on a geometric tree is full up to a distance $\frac{t}{\mu}(1-\epsilon)$ and does not extend beyond $\frac{t}{\mu}(1+\epsilon)$, for $\epsilon = t^{-1/2+\delta}$ for some positive $\delta \in (0, 1/2)$. To establish this, we shall use the following well known concentration property of renewal processes. We provide its proof later for completeness.

Theorem 15. *Consider a renewal process $P(\cdot)$ with holding times with mean μ. Then for any $t > 0$, any $0 < \gamma < 1$, there exists a positive constant c such that*

$$\mathbf{P}\left(\left|P(t) - \frac{t}{\mu}\right| \geq \frac{t\gamma}{\mu}\right) \leq 2e^{-\frac{\gamma^2\mu}{8c}t}$$

Now we use Theorem 15 to establish Theorem 10. Recall that the spreading time along each edge is an independent and identically distributed random variable with mean μ. Now the underlying network graph is a tree. Therefore for any node v at distance r from source node v^*, there is a unique path (of length r) connecting v and v^*. Then, the spread of the rumor along this path can be thought of as a renewal process, say $P(t)$, and node v is infected by time t if and only if $P(t) \geq r$. Therefore, from Theorem 15 it follows that for

any node v that is at distance $\frac{t}{\mu}(1-\epsilon)$ for $\epsilon = t^{-\frac{1}{2}+\delta}$ for some $\delta \in (0, 1/2)$,

$$\begin{aligned}\mathbf{P}\big(v \text{ is not rumor infected}\big) &\leq 2e^{-\frac{\epsilon^2 \mu t}{8c}} \\ &= 2e^{-\frac{\mu}{8c}t^{2\delta}}.\end{aligned}$$

Now the number of such nodes at distance $\frac{t}{\mu}(1-\epsilon)$ from v^* is at most $O\left(\left(\frac{t}{\mu}\right)^{\alpha+1}\right)$ (which follows from arguments similar to those in the proof of Theorem 5). Therefore, by an application of the union bound it follows that

$$\begin{aligned}&\mathbf{P}\left(\text{a node at distance } \frac{t}{\mu}(1-\epsilon) \text{ from } v^* \text{ isn't infected}\right) \\ &= O\left(2\left(\frac{t}{\mu}\right)^{\alpha+1} e^{-\frac{\mu}{8c}t^{2\delta}}\right) \\ &= O\left(e^{-\frac{\mu}{8c}t^{\delta}}\right).\end{aligned}$$

Using similar argument and another application of Theorem 15, it can be argued that

$$\begin{aligned}&\mathbf{P}\left(\text{a node at distance } t(1+\epsilon) \text{ from } v^* \text{ is infected}\right) \\ &= O\left(e^{-\frac{\mu}{8c}t^{\delta}}\right).\end{aligned}$$

Since the rumor is a 'spreading' process, if all nodes at distance r from v^* are infected, then so are all nodes at distance $r' < r$ from v^*; if all nodes at distance r from v^* are not infected then so are all nodes at distance $r' > r$ from v^*. Therefore, it follows that with probability $1 - O\left(e^{-\frac{\mu}{8c}t^{\delta}}\right)$, all nodes at distance up to $\frac{t}{\mu}(1-\epsilon)$ from v^* are infected and all nodes beyond distance $\frac{t}{\mu}(1+\epsilon)$ from v^* are not infected. This completes the proof of Theorem 10.

A.2 Proof of Theorem 15

We wish to provide bounds on the probability of $P(t) \leq \mu t(1-\gamma)$ and $P(t) \geq \mu t(1+\gamma)$ for a renewal process $P(\cdot)$ with holding times with mean μ. Define nth arrival time S_n as

$$S_n = \sum_{i=1}^{n} X_i$$

where X_i are non-negative i.i.d. random variables with a well defined moment generating function $M_X(\theta)$ and mean $\mathbb{E}[X_i] = \mu > 0$. We can relate the arrival times to the renewal process by the following relations:

$$\mathbf{P}(P(t) \leq n) = \mathbf{P}(S_n \geq t)$$

and

$$\mathbf{P}(P(t) \geq n) = \mathbf{P}(S_n \leq t)$$

The first relations says that the probability of less than n arrivals in time t is equal to the probability that the nth arrival happens after time t. The second relations says that the probability of more than n arrivals in time t is equal to the probability that the nth arrival happens before time t.

We now bound $\mathbf{P}(S_n \geq t)$. To that end, for $\theta > 0$ it follows from the Chernoff bound that

$$\begin{aligned}\mathbf{P}(S_n \geq t) &= \mathbf{P}\left(e^{\theta S_n} \geq e^{\theta t}\right) \\ &\leq M_X(\theta)^n e^{-\theta t}\end{aligned}$$

We can use the following approximation for $M_X(\theta)$ which is valid for small θ.

$$\begin{aligned}M_X(\theta) &= 1 + \theta\mu + \theta^2 \frac{\mathbb{E}[X^2]}{2} + \theta^3 \sum_{i=3}^{\infty} \theta^{i-3} \frac{\mathbb{E}[X^i]}{i!} \\ &\leq 1 + \theta\mu + c_1\theta^2\end{aligned}$$

for some finite positive constant c_1. Using this along with the inequality $\log(1+x) \leq x$ for $-1 < x$, we obtain

$$\begin{aligned}
\log\left(\mathbf{P}\left(S_n \geq t\right)\right) &\leq n\log\left(M_X\left(\theta\right)\right) - \theta t \\
&\leq n\log\left(1 + \theta\mu + c_1\theta^2\right) - \theta t \\
&\leq \theta\left(\mu n - t\right) + nc_1\theta^2
\end{aligned}$$

To minimize this probability, we find the θ that minimizes $\theta\left(\mu n - t\right) + nc_1\theta^2$. This happens for $\theta = \frac{1}{2c_1}\left(\frac{t}{n} - \mu\right)$. We set $n = \frac{t}{\mu}\left(1-\gamma\right)$, so the minimum value is achieved for $\theta = \frac{\gamma\mu}{2c_1(1-\gamma)}$. Using this value, we obtain

$$\begin{aligned}
\log\left(\mathbf{P}\left(S_{\frac{t}{\mu}(1-\gamma)} \geq t\right)\right) &\leq -\frac{\gamma\mu}{2c_1(1-\gamma)}\left(\gamma t\right) + \frac{tc_1}{\mu}(1-\gamma)\frac{\gamma^2\mu^2}{4c_1^2\left(1-\gamma\right)^2} \\
&\leq -\frac{\gamma^2\mu t}{2c_1(1-\gamma)} + \frac{\gamma^2\mu t}{4c_1(1-\gamma)} \\
&\leq -\frac{\gamma^2\mu t}{4c_1(1-\gamma)} \\
&\leq -\frac{\gamma^2\mu t}{8c_1}
\end{aligned}$$

With this result, we obtain

$$\mathbf{P}\left(P(t) \leq \frac{t}{\mu}\left(1-\gamma\right)\right) \leq e^{-\frac{\gamma^2\mu t}{8c_1}}$$

For the upper bound, we have for $\theta > 0$

$$\begin{aligned}
\mathbf{P}\left(S_n \leq t\right) &= \mathbf{P}\left(e^{-\theta S_n} \geq e^{-\theta t}\right) \\
&\leq M_X\left(-\theta\right)^n e^{\theta t}
\end{aligned}$$

We can use the following approximation for $M_X(-\theta)$ which is valid for small $\theta > 0$.

$$\begin{aligned} M_X(-\theta) &= 1 - \theta\mu + \theta^2 \frac{\mathbb{E}[X^2]}{2} - \theta^3 \sum_{i=3}^{\infty} \theta^{i-3} (-1)^{i-3} \frac{\mathbb{E}[X^i]}{i!} \\ &\leq 1 - \theta\mu + c_2\theta^2 \end{aligned}$$

for some finite positive constant c_2. Using this we obtain

$$\begin{aligned} \log(\mathbf{P}(S_n \leq t)) &\leq n \log(M_X(-\theta)) + \theta t \\ &\leq n \log\left(1 - \theta\mu + c_2\theta^2\right) + \theta t \\ &\leq \theta(t - \mu n) + n c_2 \theta^2 \end{aligned}$$

To minimize this probability, we find the θ that minimizes $\theta(t - \mu n) + n c_2 \theta^2$. This happens for $\theta = \frac{1}{2c_2}\left(\mu - \frac{t}{n}\right)$. We set $n = \frac{t}{\mu}(1+\gamma)$, so the minimum value is achieved for $\theta = \frac{\gamma\mu}{2c_2(1+\gamma)}$. Using this value, we obtain

$$\begin{aligned} \log\left(\mathbf{P}\left(S_{\frac{t}{\mu}(1+\gamma)} \leq t\right)\right) &\leq -\frac{\gamma\mu}{2c_2(1+\gamma)}(\gamma t) + \frac{t c_2}{\mu}(1+\gamma)\frac{\gamma^2\mu^2}{4c_2^2(1+\gamma)^2} \\ &\leq -\frac{\gamma^2\mu t}{2c_2(1+\gamma)} + \frac{\gamma^2\mu t}{4c_2(1+\gamma)} \\ &\leq -\frac{\gamma^2\mu t}{4c_2(1+\gamma)} \\ &\leq -\frac{\gamma^2\mu t}{8c_2} \end{aligned}$$

With this result, we obtain

$$\mathbf{P}\left(P(t) \geq \frac{t}{\mu}(1+\gamma)\right) \leq e^{-\frac{\gamma^2\mu t}{8c_2}}$$

If we set $c = \max(c_1, c_2)$ and combine the upper and lower bounds then we obtain

$$\mathbf{P}\left(\left|P(t) - \frac{t}{\mu}\right| \geq \frac{t\gamma}{\mu}\right) \leq 2e^{-\frac{\gamma^2\mu}{8c}t}$$

This completes the proof of Theorem 15.

Bibliography

[1] Data courtesy Microsoft Research Cambridge.

[2] The CAIDA as relationships dataset. http://www.caida.org/data/active/as-relationships/, August 2009.

[3] Private communication with Boon Thye Thomas Yeo and Bruce Fischl, October 2009.

[4] http://askobama.twitter.com/, July 2011.

[5] Twitter streaming API. https://dev.twitter.com/docs/streaming-api, July 2011.

[6] Nir Ailon, Moses Charikar, and Alantha Newman. Aggregating inconsistent information: ranking and clustering. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, STOC '05, pages 684–693, New York, NY, USA, 2005. ACM.

[7] K. B. Athreya and P. E. Ney. *Branching Processes*. Springer-Verlag, 1972.

[8] M. D. Atkinson. On computing the number of linear extensions of a tree. *Order*, 7:23–25, 1990.

[9] N. T. J. Bailey. *The Mathematical Theory of Infectious Diseases and its Applications*. Griffen, 1975.

[10] E. Bakshy, J. M. Hofman, W. A. Mason, and D. J. Watts. Everyone's an influencer: Quantifying influence on Twitter. In *Proc. WSDM*, 2010.

[11] M. F. Balcan, A. Blum, and A. Gupta. Approximate clustering without the approximation. In *Proc. of FOCS*, volume 15, pages 5249–5262, 2010.

[12] C. Ballester, A. Calvo-Armengol, and Y. Zenou. Who's who in networks. Wanted: The key player. *Econometrica*, 74:1403–1417, 2006.

[13] N. Bansal, A. Blum, and S. Chawla. Correlation clustering. *Machine Learning*, 56:89–113, 2004.

[14] A. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

[15] E. Bender. and E. R. Canfield. The asymptotic number of non-negative integer matrices with given row and column sums. *Jour. of Combinatorial Theory, Series A*, 24:296–307, 1978.

[16] D. Bertsekas and J. N. Tsitsiklis. *Probabilistic System Analysis, 2nd Edition.* Aetna Publications, 2009.

[17] D. M. Blei, T. Griffiths, M. I. Jordan, and J. Tenenbaum. Hierarchical topic models and the nested Chinese restaurant process. In *Proc. Neural Information Processing Systems (NIPS)*, 2004.

[18] Vincent D Blondel, Jean loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, P10008, 2008.

[19] B. Bollobas. A probabilistic proof of an asymptotic formula for the number of labeled regular graphs. *Preprint Series, Matematik Institut, Aarhus Universitet*, 1979.

[20] B. Bollobas and O. Riordan. The diameter of a scale-free random graph. *Combinatorica*, 24:5–34, 2004.

[21] P. Bonacich. Factoring and weighting approaches to status scores and clique identification. *Journal of Mathematical Sociology*, 2:113–120, 1972.

[22] P. Bonacich. Power and centrality: A family of measures. *American Journal of Sociology*, 92:1170–1182, 1987.

[23] U. Brandes, D. Delling, M. Gaertler, R. Grke, M. Hoefer, Z. Nikoloski, and D. Wagner. On modularity – NP-completeness and beyond. Technical Report 2006-19, ITI Wagner, Faculty of Informatics, Universitt Karlsruhe (TH), 2006.

[24] G. Brightwell and P. Winkler. Counting linear extensions is #P-complete. pages 175–181, 1991.

[25] M. Cha, H. Haddadi, F. Benevenuto, and K. P. Gummadi. Measuring user influence in Twitter: the million follower fallacy. In *Proc. AAAI Conf. on Weblogs and Social Media*, 2010.

[26] F. Chung. *Spectral Graph Theory*, volume 92 of *CBMS Regional Conference Series in Mathematics*. American Mathematical Society, 1997.

[27] comScore. Google reaches 1 billion global visitors. http://www.comscoredatamine.com/2011/06/google-reaches-1-billion-global-visitors/, June 2011.

[28] C. Daskalakis, E. Mossel, and S. Roch. Phylogenies without branch bounds: Contracting the short, pruning the deep. *SIAM J. on Discrete Mathematics*, 25:872–893, 2011.

[29] P. L. Erdos, M. A. Steel, L. A. Szekely, and T. A. Warnow. A few logs suffice to build (almost) all trees (part 2). *Theor. Comput. Sci*, 221:77–118, 1999.

[30] W. Evans, C. Kenyon, Y. Peres, and L. Schulman. Broadcasting on trees and the Ising model. *Ann. Appl. Prob.*, 10:410–433, 2000.

[31] S. Fortunato and M. Barthelemy. Resolution limit in community detection. *Proc. Natl. Acad. of Sciences*, 104:36–41, 2007.

[32] L. C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40:35–41, 1977.

[33] L. C. Freeman, S. Borgatti, and D. R. White. Centrality in valued graphs: A measure of betweenness based on network flow. *Social Networks*, 13:141–154, 1991.

[34] A. Ganesh, L. Massoulie, and D. Towsley. The effect of network topology on the spread of epidemics. In *Proc. 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, volume 2, pages 1455–1466, 2005.

[35] A. Gerschenfeld and A. Montanari. Reconstruction for models on random graphs. In *Proc. 48th IEEE Symp. Found. Comp. Sci. (FOCS)*, pages 194–204, 2007.

[36] M. Gomez-Rodriguez, J. Leskovec, and A. Krause. Inferring networks of diffusion and influence. In *Proc. Knowledge Discovery and Data Mining (KDD)*, 2010.

[37] Jennifer Van Grove. Twitter sets new record: 3,283 tweets per second. http://mashable.com/2010/06/25/tps-record/, June 2010.

[38] L. Hagen and A. B. Kahng. New spectral methods for the ratio cut partitioning and clustering. *IEEE Trans. Compututer Aided Design*, 11:1074–1085, 1992.

[39] S. Johnson. *The Ghost Map: The Story of London's Most Terrifying Epidemic - and How it Changed Science, Cities and the Modern World.* Riverhead Books, 2006.

[40] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46:604 – 632, 1999.

[41] H. Kwak, C. Lee, H. Park, and S. Moon. What is Twitter, a social network or a news media? In *Proc. WWW*, 2010.

[42] Yanhong Li. Toward a qualitative search engine. *Internet Computing, IEEE*, 2(4):24 – 29, 1998.

[43] C. Moore and M. E. J. Newman. Epidemics and percolation in in small-world networks. *Phys. Rev. E*, 61:5678–5682, 2000.

[44] D. Mosk-Ayoma and D. Shah. Fast distributed algorithms for computing separable functions. *IEEE Trans. on Information Theory*, 54:2997–3007, 2008.

[45] E. Mossel. Reconstruction on trees: Beating the second eigenvalue. *Ann. Appl. Prob.*, 11:285–300, 2001.

[46] R. Motwani and P. Raghavan. *Randomized algorithms*. Chapman & Hall/CRC, 2010.

[47] S. A. Myers and J.Leskovec. On the convexity of latent social network inference. In *Proc. Neural Information Processing Systems (NIPS)*, 2010.

[48] B. Nadler and M. Galun. Fundamental limitations of spectral clustering. In *Proc. Neural Information Processing Systems (NIPS)*, 2007.

[49] R. M. Neal. Markov chain sampling methods for Dirichlet process mixture models. *Journal of Computational and Graphical Statistics*, pages 249–265, 2000.

[50] M. E. J. Newman. Modularity and community structure networks. *Proc. Natl. Acad. of Sciences*, 103:8577–8582, 2006.

[51] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69:026113, 2004.

[52] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: analysis and an algorithm. In *Proc. Neural Information Processing Systems (NIPS)*, 2002.

[53] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: Bringing order to the web. Technical report, Stanford University InfoLab, 1999.

[54] G. Palla, I. Derenyi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435:814–818, 2005.

[55] R. Pastor-Satorras and A. Vespignani. Epidemic spreading in scale-free networks. *Phys. Rev. Lett.*, 86:3200–3203, 2001.

[56] G. Sabidussi. The centrality index of a graph. *Psychometrika*, 31:581–603, 1966.

[57] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22:888–905, 2000.

[58] Maggie Shiels. Twitter co-founder Jack Dorsey rejoins company. http://www.bbc.co.uk/news/business-12889048, March 2011.

[59] J. Snow. *On the Mode of Communication of Cholera*. John Churchill, New Burlington Street, 1855.

[60] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei. Sharing clusters among related groups: Hierarchical Dirichlet processess. In *Proc. Neural Information Processing Systems (NIPS)*, 2005.

[61] M. Wainwright and M. Jordan. Graphical models, exponential families, and variational inference. Technical Report 649, U.C. Berkeley, Dept. of Statistics, 2003.

[62] D. J. Watts and S. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998.

[63] J. Weng, E. Lim, J. Jiang, and Q. He. Twitterrank: Finding topic-sensitive influential twitterers. In *Proc. WSDM*, 2010.

[64] S. White and P. Smyth. A spectral clustering approach to fining communities in graphs. In *Proc. of SIAM International Conference on Data Mining*, pages 76–84, 2005.