# Chapter 3
# Information Source Detection in Social Networks


## 3.1 Introduction

The rising popularity of online social networks has made information generating and sharing much easier than ever before, due to the ability to publish content to large, targeted audiences. Such networks enable their participants to simultaneously become both consumers and producers of content, shifting the role of information broker from a few dedicated entities to a diverse and distributed group of individuals. While this fundamental change allows information propagating at an unprecedented rate [166], it also enables unreliable or unverified information spreading among people, such as rumors [47].

Rumor has been a research subject in psychology and social cognition for a long time [50]. It is often viewed as an unverified account or explanation of events circulating from person to person and pertaining to an object, event, or issue in public concern [152]. Bordia et al. [23] propose that transmission of rumor is probably reflective of a "collective explanation process." Since there is often not enough resource to manually identify rumors or misinformation from the huge volume of fast evolving data, it has become a critical problem to design systems that can automatically detect misinformation and disinformation. Microblogging services, like Twitter, allow small pieces of information to spread quickly to large audiences, allowing rumors to be created and spread in new ways [108].

Current media environment is suitable to the emergence and propagation of rumors that are not limited to insignificant subjects: Rumors can have major consequences on political, strategic, or economical decisions. Increasingly, they are triggered off on purpose for various reasons: campaigns can be carried out in order to discredit a company, endanger strategic choices, or question political decisions. Therefore research on rumor detection has great significance on Web security issues [140].

In recent times, many Web based systems have been developed to detect and evaluate the rumors in social networks. Examples are (1) TwitterTrails.com [133], a



W. Xu, W. Wu, *Optimal Social Influence*, SpringerBriefs in Optimization,
https://doi.org/10.1007/978-3-030-37775-5_3

system which permits users to determine the features of propagated rumors and its falsification, (2) TweedCred [80], an instantaneous system to judge trustworthiness of posts on Twitter, (3) Hoaxy [174], a platform for tracking the misinformation in a social network, (4) Emergent.info [189], a real-time rumor follower that focused on rising tales on the Internet and observes their faithfulness, and (5) Snopes.com [181] and factcheck.org [60], admired websites archiving memes and urban myths. The reality checking abilities of these rumor detection systems validate the authentication of rumors on Web and vary from entirely automatic to semi-automatic. But, these systems do not track or observe the diffusion progress and do not detect all possible source(s).

**Rumor Detection using Machine Learning** Social network analysis about studying rumors often focuses on machine learning techniques such as building classifiers, sentiment analysis, Twitter data mining, and so on. Work in this area includes [59, 116, 158, 160]. Leskovec et al. use the evolution of quotes reproduced online to identify memes and track their spread overtime [116]. Ratkiewicz et al. [160] created the Truthy system, identifying misleading political memes on Twitter using tweet features, including hashtags, links, and mentions. Other projects focus on highlighting disputed claims on the Internet using pattern matching techniques [59]. Qazvinian et al. [158] explore the effectiveness of three categories of features: content based, network based, and microblog specific memes for correctly identifying rumors in microblogs. In these introduced research work, a complete set of social conversations (e.g., tweets) that are actually about the rumor need to be retrieved first.

There have appeared some studies on analyzing rumors and information credibility on Twitter, the world's largest microblogging platform. Castillo et al. [34] focus on automatically assessing the credibility of a given set of tweets. They analyze the collected microblogs that are related to "trending topics," and use a supervised learning method (decision tree) to classify them as credible or not credible. Qazvinian et al. [158] focus on two tasks: The first task is classifying those rumor-related tweets that match the regular expression of the keyword query used to collect tweets on Twitter monitor. The second task is analyzing the users' believing behavior about those rumor-related tweets. They build different Bayesian classifiers on various subsets of features and then learn a linear function of these classifiers for retrieval of those two sets. Mendoza et al. [132] use tweets to analyze the behavior of Twitter users under bombshell events such as the Chile earthquake in 2010. They analyze users' retweeting topology network and find the difference in the rumor diffusion pattern on Twitter environment than on traditional news platforms.

**Rumor Source Detection Based on Information Spreading** Many studies on the problem of information propagation are inspired from the more common issue of contagion and generally use models for viral epidemics in populations such as the susceptible-infected-recovered (SIR) model. On this subject, research has focused on the effects of the topological properties of the network on inferring the source of a rumor in a network. Shah and Zaman [171–173] were the first to study systematically the problem of infection sources estimation which consider

an susceptible-infected (SI) model, in which there is a single infection source, and susceptible nodes are those with at least one infected neighbor, while infected nodes do not recover. Subsequently, [123, 126] consider the multiple sources estimation problem under an SI model; [211] studies the single source estimation problem for the susceptible-infected-recovered (SIR) model, where an infected node may recover but can never be infected again; and [125] considers the single source estimation problem for the susceptible-infected-susceptible (SIS) model, where a recovered node is susceptible to be infected again.

Although all the works listed above answer some fundamental questions about information source detection in large-scale networks, they assume that a complete snapshot of the network is given while in reality a complete snapshot, which may have hundreds of millions of nodes, is expensive to obtain. Furthermore, these works assume homogeneous infection across links and homogeneous recovery across nodes, but in reality, most networks are heterogeneous. For example, people close to each other are more likely to share rumors and epidemics are more infectious in the regions with poor medical care systems. Therefore, it is important to take sparse observations and network heterogeneity into account when locating information sources. In [124, 170, 212], detecting information sources with partial observations in which only a fraction of nodes (observers) can be observed has been investigated. The work in [154] assumes that for each of the observers, the knowledge of the first infected time and from which neighbor the infection is received are given. This assumption is impractical in some cases. For example, it is usually hard to know from which neighbor the infection is coming from in a contagious disease spreading within a community. In [54], the authors have considered the detection rate of the rumor centrality estimator when a priori distribution of the source node is given. Several other source locating algorithms have also been proposed recently, including an eigenvalue based estimator [157], a fast Monte Carlo algorithm [2], and a dynamic message-passing algorithm under the SIR model [2].

Source detection is very significant in various application domains such as medical (to find the source of epidemic), security (to detect the source of virus), large interconnected network (to detect the flaws in power grid network, gas or water pipeline network), social network (to identify the culprits who spread wrong information), financial network (for checking the reasons of cascade failures), etc. Due to its wide scope in different applications, past two decades observed large improvements in source detection techniques. Major research has been done for source identification in different application areas like finding the first patient to control an epidemic of disease [9], source of virus [171], gas leakage source in wireless sensor network [177], propagation sources in complex networks [90], and source of rumor or misinformation in a social network [148, 175, 206, 210] which are directly or indirectly related to rumor source detection.

In Sect. 3.2, we introduce a monitor based approach to detect single rumor source in online social networks and define a probability-score function, named as rumor quantifier, for ranking how likely nodes are the actual rumor source [206]. Given a weighted social graph, we propose a polynomial time algorithm to detect the rumor

source with respect to the popular independent cascade (IC) model in online social networks. Since real social networks are getting bigger with millions of nodes, our algorithm can scale well on real large datasets.

In Sect. 3.3, we consider detecting multiple rumors from a deterministic point of view by modeling it as the set resolving set (SRS) problem [210]. Let $G$ be an undirected graph on $n$ vertices. A vertex subset $K$ of $G$ is SRS of $G$ if any different detectable node sets are distinguishable by $K$. The problem of multiple rumor source detection (MRSD) will be defined as finding an SRS $K$ with the smallest cardinality in $G$. Using an analysis framework of submodular functions, we propose a highly efficient greedy algorithm for MRSD problem on general graphs, which is polynomial time under some reasonable constraints, that is, there is a constant upper bound $r$ for the number of sources. Moreover, we show that our natural greedy algorithm correctly computes an SRS with provable approximation ratio of at most $(1 + r \ln n + \ln \log_2 \gamma)$, given that $\gamma$ is the maximum number of equivalence classes divided by one node-pair. This is the first work providing explicit approximation ratio for the algorithm solving minimum SRS. Therefore the introduced framework suggests a robust approach for MRSD independent of diffusion models in networks.

In the last section, we will summarize our work of rumor source detection and future work in this field will also be discussed.

## 3.2 Single Source Detection

This section studies the problem of identifying source location of single rumor in online social networks in which the spread of information follows the popular independent cascade (IC) model. In the absence of text information, we develop a monitor based approach to evaluate how likely that a piece of information is actually a rumor. Given the underlying social network structure, a number of monitor nodes are injected into the network whose job is to report the data they receive. Based on observing which of monitors received the information and which did not, we propose a polynomial time algorithm to compute rumor quantifier, a reachability based score for ranking the importance of nodes as the rumor source. Extensive simulation results have shown that, with a reasonable number of monitor nodes and appropriate monitor deployment, our rumor source detection algorithm can recognize rumor source effectively and efficiently.

### *3.2.1 Problem Formulation*

We first introduce the propagation model of rumors, then present the formal problem formulation. Opinion dynamics in a social network can be modeled in some cases using independent cascade (IC) Model, which is a classical model in influence

spreading. A individual on Twitter may be influenced by the opinion or posting of someone she is following, thereby becoming "infected" with the same opinion.

*Independent Cascade Model* A social network is modeled as a directed graph $G = (V, E)$ where $V$ is the set of users and $E$ is the set of edges where each edge represents relationship between two individuals. Let $u \in V$ be a rumor source form which a rumor starts spreading at time, say $t$. As basic independent cascade (IC) model operates, the cascade of rumor spreads in the graph. Specifically, $u$ is given a single chance to activate each currently inactive neighbor $v$; it succeeds with a probability $p(u, v)$—a parameter of the system—independently of the history. (If $u$ has multiple inactive neighbors, its attempts are sequenced in an arbitrary order.) If $u$ succeeds, then $v$ will become active in step $t$+1. Again, the process runs until no more activations are possible. In this model, once a node is infected with the rumor, it retains it forever.

Note that cascades in IC model are necessarily trees since if a node, say $s$, gets infected multiple times knowing the node that infected $s$ first is sufficient. Thus, the influence structure of a cascade is given by a directed tree $T$, which is contained in the directed graph $G$, i.e., the graph over which the cascade propagates.

*Problem Definition* Given the above spreading model, the goal of rumor source detection is to identify the rumor source based on the input weighted social graph.

*Monitor* We assume that a set of $k$ pre-selected nodes $M$ ($M \in V$) are our monitors. For rumor investigation purposes, given a specific piece of information (cascade), monitors report whether they have received it or not. We denote the set of monitor nodes who have received the rumor by $M^+$, and the set of monitor nodes who have not received it by $M^-$ (where $M^+, M^- \in M$). We call the former set positive monitors and the latter negative monitors.

*Social Influence Probability* Social influence from node $u$ to $v$, denoted by $p(u, v)$, is a numerical weight associated with the edge $e \in E$. In most cases, the social influence score is asymmetric, i.e., $p(u, v) \neq p(u, v)$. Furthermore, the social influence from node $u$ to $v$ will vary on different types of networks.

Thus based on the above concepts, we can define the tasks of rumor source detection. In this paper, we only discuss the case that there is only one rumor source. Given a weighted social network $G = (V; E; p)$, a propagation model, and a monitor set, the goal is to identify the source node that starts the rumor cascade. The problem definition is as follows.

**Problem 3.1 (Single Rumor Source Detection)** Given an cascade model $m$, we observe the rumor infected graph $G = (V, E, p)$ at some time $t > 0$. We do not know the value of $t$ or the realization of the spreading times on edges $e \in E$; we only know positive monitors $M^+ \subseteq V$ and negative monitors $M^- \subseteq V$. The goal is to find the rumor source $r \in V$ given $G$.

### 3.2.2 Monitor Based Approach

Our main idea is to leverage monitors for rumor source detection. Firstly, we introduce a rumor quantifier $Q(r)$, a probability-score function, for ranking the importance of nodes as the rumor source. Let $Q(r)$ denote how likely that a node $r \in V$ is actually the rumor source. Identifying the rumor source will be formulated as finding a node $r \in V$ that maximizes the rumor quantifier $Q(r)$, which is written as follows:

$$Max \qquad Q(r) \tag{3.1}$$

To identify the source of a rumor, we use the intuition that information will spread more quickly from the source to the positive monitors but more slowly to the negative monitors. Since our model is probabilistic and dynamic, in other words, the cascade must be easier to propagate from the source to positive monitors while harder to propagate from the source to negative monitors. Based on this idea, the equation of $Q(\cdot)$ can be defined as follows:

$$Q(r) = F(p(r, M^+), p(r, M^-)) \tag{3.2}$$

where $p(r, M^+)$ denotes the probability that a cascade spreads from node $r$ to the positive monitor set $M^+$, and $p(r, M^-)$ denotes the probability that the cascade spreads from node $r$ to the negative monitor set $M^-$.

The function $F$ demonstrates that the quantifier $Q(r)$ of a node $r$ is affected by two factors: both $p(r, M^+)$ and $p(r, M^-)$. For each node $r \in V$, the quantifier first considers how likely that a cascade spreads from it to the positive monitor set, $p(r, M^+)$. In specific, the larger of the probability of $p(r, M^+)$, the larger of the value of $Q(r)$. In other word, the value of our quantifier is proportional to the value of $p(r, M^+)$. Also, the smaller of the value of $p(r, M^-)$, the larger of the value of $Q(\cdot)$. Thus, the node with maximum rumor quantifier $Q(\cdot)$ has the maximum likelihood estimation in the context of independent cascade model. The details of the function $F$ will be further depicted in the section of algorithm.

For a cascade, we will first specify the influence probability $p(u, v)$ that describes how likely that node $u$ spreads the cascade to node $v$. Then we will describe the probability $p(r, M^+)$ which specifies the probability that the cascade propagates from node $r$ to the positive monitor set $M^+$. Similarly, we also define $p(r, M^-)$, which describes how likely cascade propagates from node $r$ to the negative monitor set $M^-$ in the network $G$.

For a path $P =< p_1, p_2, \ldots p_i, \ldots, p_m >$, we define the propagation probability of the path $P$ as,

$$p(P) = \prod_{i=1}^{m-1} (p_i, p_{i+1}) \tag{3.3}$$

where the product is over the edges of path $P$. Intuitively the probability that $u$ activates $v$ through path $P$ is $p(P)$, because it needs to activate all nodes along the path. Here, the edges of the path $P$ simply specify how the cascade spreads, i.e., every node gets influenced by its parent.

To approximate the actual expected influence within the social network, we propose to use the maximum influence path (MIP) to estimate the influence from one node to another. Let $P_G(u, v)$ denote the set of all paths from $u$ to $v$ in a graph $G$.

**Definition 3.1 (Maximum Influence Path (MIP))** For a social graph G, we define the Maximum Influence Path $MIP(u, v)$ from $u$ to $v$ in $G$ as

$$MIP(u, v) = argmax_P\{p(P)|P \in P_G(u, v)\}.$$

Ties are broken in a predetermined and consistent way, such that $MIP(u, v)$ is always unique, and any subpath in $MIP(u, v)$ from $x$ to $y$ is also the $MIP(x, y)$. If $P_G(u, v) = \oslash$, we denote $MIP(u, v) = \oslash$.

For any two nodes $u, v \in V$, if there exists no path connecting from $u$ to $v$, then $p(u, v) = 0$ since they cannot influence each other. Otherwise suppose there exists multiple paths connecting from $u$ to $v$, we define $p(u, v)$ according to maximum influence path as follows.

$$p(u, v) = MIP(u, v) \tag{3.4}$$

Now that we have specified the probability $p(u, v)$ for any two nodes $u, v \in V$, next we define the probability of observing cascade propagating from a node $r$ to a monitor set $M$ in a particular tree structure $T$ as

$$p_T(r, M) = \phi_{m_i \in M^+} p(r, m_i) \tag{3.5}$$

A typical way of function $\phi$ is to summarize $p(r, m_i)$ for all the nodes $m_i \in M$. Suppose $|M| = n$, we provide a heuristic in our experiment for function $\phi$, as demonstrated in Eq. (3.6) such that $p(u, M)$ will not exceed 1.

$$\phi_{m_i \in M} p(r, m_i) = 1 - \prod_{i=1}^{i=n}(1 - p(r, m_i)) \tag{3.6}$$

Here we will use a specific example to illustrate Eq. (3.6). Figure 3.1 shows the propagation tree form root node $u$ to the monitor set, say $M^+$, which has three nodes: $m_1, m_2, m_3$. The influence probabilities among nodes are given. According to Eq. (3.6), $P(u, M+) = 1 - (1 - 0.8 * 0.8 * 0.3)(1 - 0.8 * 0.6)(1 - 0.5) = 0.790$.

In this section, we have introduced the rumor quantifier $Q(\cdot)$ for ranking the probability that a node is the rumor source and how to compute it. In the following section, we will develop efficient algorithms to detect the rumor source.

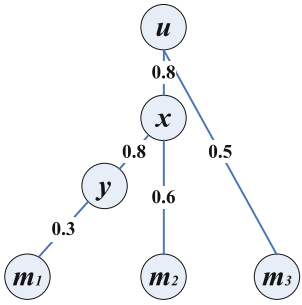


**Fig. 3.1** An example of propagation tree from root node $u$ to monitor node set $\{m_1, m_2, m_3\}$. The influence probabilities are given as edge weights

**Algorithm 1** Maximum propagation tree identification

Input: Root Node $r$, Leaf Nodes $m_i \in M$, Weighted Social Graph $G = (V, E, p)$.
Output: Maximum Propagation Tree $T$

Set $p' = -lnp$.
$G' = (V, E, P')$.
**for all** node $m_i \in M$ **do**
  Find the shortest path $h$ from $r$ to $m_i$ in $G'$.
**end for**
**return** all the paths $h$.

### *3.2.3 Proposed Algorithm*

Given a node $r$ and a monitor node set $M$, there are more than one propagation trees satisfying the condition that the root is $r$ and the leaf node set is $M$. To reduce the computation complexity, instead of searching all the possible propagation trees, we only consider the most likely propagation tree from $r$ to $M$. Here we give a formal definition of the most likely propagation tree, named as maximum propagation tree.

**Definition 3.2 (Maximum Propagation Tree (MPT))** Given a weighted social graph $G = (V, E, p)$, a root node $r$, a monitor node set $M$, a maximum propagation tree ($MPT(r, M)$) consists of all the maximum influence paths from root node $r$ to each node in the set $M$.

In order to find the maximum propagation tree given a root node, leaf nodes (the monitor set) and the underlying graph structure, we propose a polynomial time solution, as demonstrated in Algorithm 1. The general idea is to find all the shortest paths from the root node to all the leaf nodes.

Next we will give a proof that why Algorithm 1 can find the desired maximum propagation tree.

***Proof*** Let $G = (V, E, p)$ be a weighted social graph. In terms of problem definition of MPT, the maximum propagation tree for $G$ is a tree, say $T \subseteq G$, which consists of all the maximum influence paths from root node $r$ to each node in the set $M$. In Algorithm 1, all the weights $p$ is updated to $p' = -lnp$. For each node $m_i \in$

$M^+$, if a shortest path from $r$ to $m_i$ is found, then $p'(r, m_i) = -lnp$ is minimized. Since $p(r, m_i) \in (0, 1)$ (influence probability), $lnp(r, m_i) < 0$, $p' = -lnp > 0$, thus, $p(r, m_i) = \prod_{(i,j) \in edgeconnectingr, m_i}(i, j)$ is maximized. In other word, the maximum influence path from root node $r$ to $m_i \in M^+$ is actually the shortest path between them.

To detect the hidden rumor source, our algorithm is to, for every mode $u \in V$, first calculate the rumor quantifier $Q(u)$, that is $p(u, M^+)$ in our settings here. Rank nodes according to the value of rumor quantifier decreasingly. The node with maximum $Q(u)$ is most likely to be the rumor source. The detailed algorithm is described in Algorithm 2. Given a weighted social graph $G$ in which the weight denote the influence probability between individuals, and monitor sets $M^+$, $M^-$, for each node $u$, we want to find a maximum propagation tree $T \in G$ such that the root is set as node $u$, while the leafs of the tree are predefined as positive monitors (line 2). Based on the found maximum propagation tree, the algorithm computes probabilities $p(u, M^+)$ (line 3). The reason why such tree exists is that, if a node $u$ is the rumor source, $u$ must have paths to all the monitors in $M^+$. Otherwise $u$ cannot be a rumor source. When it comes to a special case that several nodes cannot be distinguished by positive monitor set (say, nodes have the same probability of $p(v, M^+)$) (line 5), the quantifier will consider how likely the cascade spread from them to negative monitor set (line 6). In this case, the node with lower probability of $p(v, M^-)$ will be ranked higher. Therefore, our rumor quantifier relies mainly on the positive monitor set while at the same time it does not neglect the effect of negative monitor set.

### *3.2.4 Experiments*

To test our monitor based method to identify the rumor source, we run our RSD (**R**umor **S**ource **D**etection) algorithm on graphs of a real online social network. We are interested in understanding its behavior in practice and comparing its

**Algorithm 2** Rumor source detection

Input: Monitor set $M^+$, $M^-$, weighted social Graph $G = (V, E, p)$.
Output: Rumor Source node.

```
for all node u ∈ V do
    T = Maximum Propagation Tree Identification (u, M+, G);
    Compute the probability p(u, M+) based on T;
end for
for nodes v ∈ V with the same probability of p(v, M+) do
    Rank them by p(v, M−) increasingly;
end for
return the node ranking first.
```

performance under various monitor deployment. We find that our RSD algorithm achieves significant accuracy (up to 87%) over real datasets of social networks.

#### 3.2.4.1 Dataset

At September 2010, Twitter reports that its users publish nearly 95 million tweets per day [194]. This makes Twitter an excellent case to analyze rumors in social media. In our experiment, we extracted a social graph structure from Twitter using Twitter search API as testbed. The data we collect has 38,484 nodes, 1,364,322 edges where the nodes represent users, the edges represent the friendship or followership among the users. Besides the topology, we also calculated Retweet probability of each edge $x \rightarrow y$ as the ratio of $x$'s tweets retweeted by $y$ to all tweets of $x$. Calculated retweet probabilities were used to simulate independent cascade propagation of rumors.

#### 3.2.4.2 Experimental Evaluation

The goal of the experiments on synthetic data is to understand how the underlying network structure and monitor deployment affect the performance of our algorithm. In general, we proceed experiments as follows: (1) given a weighted social graph extracted from Twitter, we simulate a cascade (a random rumor source is selected); (2) using the retweet probability of each edge, the rumor is propagated according to IC diffusion model; (3) if the rumor fails to reach 1% of all nodes, it is viewed as a negligible rumor and this rumor propagation is discarded. A new rumor is selected and the same process is repeated. For each rumor propagation (cascade), we try to identify the rumor source using our RSD algorithm with different number of monitors. To compute average precision of the algorithm, we simulate cascades 200 times.

For best accuracy, it is important to choose monitors wisely. In this paper, we compare the following three monitor selection methods.

**Random** Random selection method selects $k$ monitors randomly. This means that, for any node $x \in V$, the probability that $x$ is selected as a monitor is $k/ \mid V \mid$.

**Incoming Degree (ID)** In this method, the number of incoming edges of each node is counted. Then, the top $k$ nodes which have largest counts are chosen as monitors.

**Betweenness Centrality (BC)** This method calculates betweenness centrality [145] for each node $v$, which is defined as

$$C(v) \sum_{s \neq t \neq v \in V} \frac{\sigma_{st}(v)}{\sigma_{st}}$$

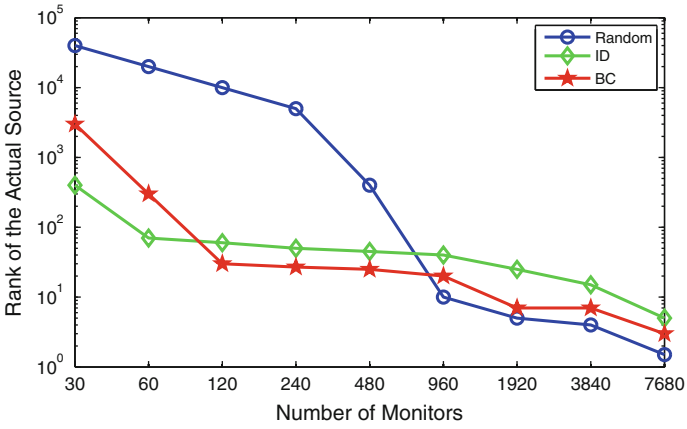


**Fig. 3.2** Average rank of the actual rumor source in the output (out of 38,484 nodes)

where $\sigma_{st}$ is the number of shortest paths from $s$ to $t$ and pass through $v$. Then, the $k$ nodes which have the largest betweenness centrality are chosen as monitors.

#### 3.2.4.3 Experimental Results

Using the method presented in Sect. 3.3, all nodes are sorted in the likelihood that they are the actual rumor source. Figure 3.2 shows the average rank of the actual source in the output. In the ideal case, the rank should be one which means that the top suspect is actually the rumor source. Note that, regardless of the monitor selection method, the rank of the true source generally decreases (i.e., improves by becoming closer to 1) as the number of monitors increases. Here the monitor numbers are set increasingly as [30, 60, 120, 240, 480, 960, 1920, 3840, 7680]. At first when the number of monitor nodes is small, Random performs worst, but it improves as more monitors are added. In contrast, Incoming Degree (ID) performs quite well when there are small number of nodes but is not satisfactory when the number of monitors is very large. The performance of betweenness centrality (BC) lies in between.

Figure 3.3 shows the distance between the top suspect and the actual source of all monitor selection methods. Note that no matter how many monitors are chosen, the average distance is smaller than three steps. In the ideal case, the distance should be zero, meaning the top suspect is the source. Figure 3.3 shows a similar tendency as Fig. 3.2. The distance decreases as more monitors are added. Random has large distances with a small number of monitors, but the distance decreases drastically as the number of monitors increase. BC and ID generally show the smallest distance between the top suspect and the actual source.

We also observe the details of monitors except the number and deployment. Figure 3.4 shows the ratio of experiments in which no monitor received the rumor.

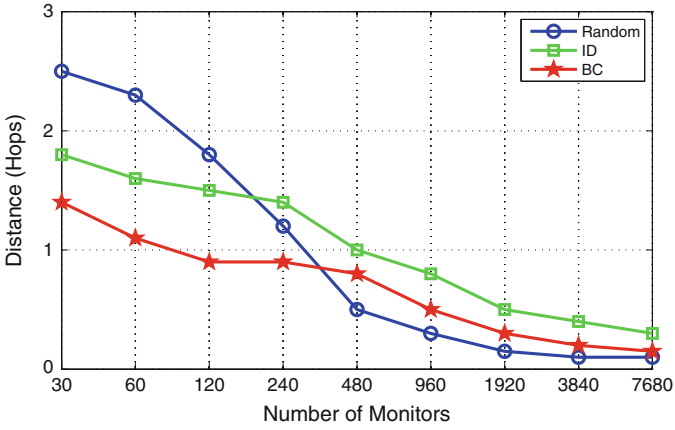


**Fig. 3.3** Average distance between the found rumor source and the actual rumor source

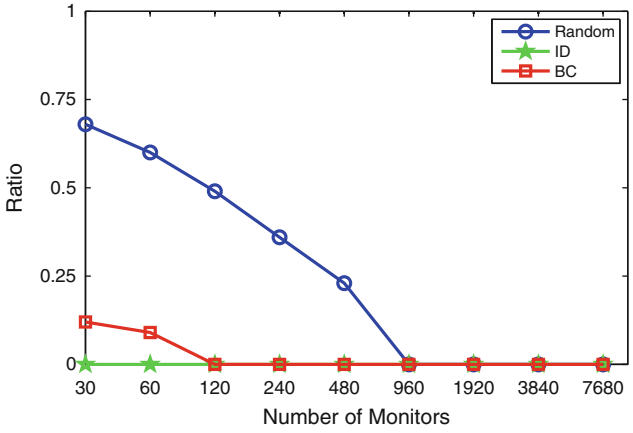


**Fig. 3.4** Ratio of experiments in which no monitor receives a rumor (out of 200 cascades)

In all monitor selection methods, the ratio decreases as the number of monitors increases. Among the three methods compared, the Random selection method has the highest ratio. When the number is monitors is small, Random tends to choose nodes loosely scattered on the boundary of the graph. Therefore, monitors selected by Random have low probability of hearing rumors. The Random selection method also has a high ratio of negative monitors when the number of monitors is small. The other methods (ID, BC) have small ratio compared to Random. When no monitor hears the rumor, it is very hard to find the source accurately as shown in Fig. 3.2 (Random when the number of monitors is 30, for example).

Now we take a detailed look about the accuracy of our RSD algorithm. As Fig. 3.5 demonstrates, the precision of our algorithm increases with the monitor number increases. The precision is defined as the number of experiments when the

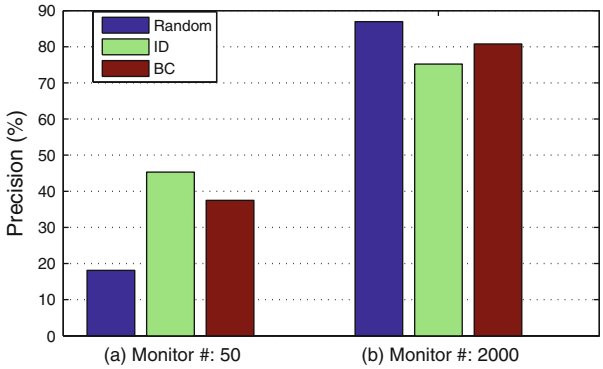


**Fig. 3.5** Average precision of RSD algorithm in finding the actual rumor source (out of four hundreds of experiments). The monitor numbers are set as 50 and 2000

top suspect is the actual rumor source divided by the total number of experiments. When the monitor are randomly chosen, the precision of RSD increases from 18% (monitor number: 50) up to 87% (monitor number: 2000), the latter of which is quite effective in practice. This series of experiments also suggest that when the number of monitor is small, in order to identify rumor source accurately, the monitor can be chosen using ID (with precision 45%, monitor number: 50). In contrast, when the number of monitor is large, random is the best way to choose monitors while at the same maintains high accuracy.

## 3.3 Detecting Multiple Rumor Sources in Networks with Partial Observations

Suppose there are more than one rumor sources in the network; the problem is how to detect all of them based on limited information about network structure and the rumor infected nodes. If each rumor source initiates a different rumor, then the problem can be decoupled to the detection of each rumor source independently. Thus, we assume that one rumor is initiated at a lot of different locations. We place some nodes $v \in K \subseteq V$ as the observers which has a clock that can record the time at which the state of $v$ is changed (e.g., knowing a rumor, being infected or contaminated). Typically, the time when the single source originates is unknown. The monitors/observers can only report the times when they receive the information, but no information about senders (i.e., we do not know who infects whom in epidemic networks or who influences whom in social networks). The information is diffused from the sources to any vertex through shortest paths in the network, i.e., as soon as a vertex receives the information, it sends the information to all its neighbors simultaneously, which takes one time unit. Our goal is to select a subset $K$ of vertices with minimum cardinality such that the source can be uniquely

located by the infected times of vertices in $K$ and network structure. This problem is equivalent to finding a minimum set resolving set (SRS) in networks defined in our models.

In this section, we originally propose the concept of set resolving set (SRS) problem and give the formal definition of it. Based on SRS, we then present a novel approach for locating multiple information sources on general graphs with partial observations of the set of infected nodes at the observation time, without knowing the neighbors from which the infection is received. Our method is robust to network heterogeneity and the number of observed infected nodes. To the best of our knowledge, this paper is the first work to study multiple rumor source detection (MRSD) problem via SRS.

Moreover, we show that our objective function for detecting multiple rumor sources in networks is monotone and submodular. By exploiting the submodularity of the objective, we develop an efficient greedy approximation for MRSD problem, which is theoretically proved to have a $(1+r\ln n+\ln\log_2\gamma)$-approximation ratio in real world, given that $\gamma$ is the maximum number of equivalence classes divided by one node-pair. These guarantees are important in practice, since selecting nodes is expensive, and we desire solutions which are not too far from the optimal solution.

The following section is organized as follows. In Sect. 3.3.1, we present the SRS based model and give a formal problem formulation. In Sect. 3.3.2, we then develop a greedy algorithm, and prove its approximation ratio. To confirm the effectiveness of our algorithm, in Sect. 3.3.3, the performance of our algorithm is evaluated in networks which exemplify different structures.

### 3.3.1 The Model

We start by describing the model and problem statement of multi-rumor-source detection. In the process, we will give the definition of set resolving set (SRS), which is the basis of the model.

If a node $u$ is a rumor source, then we use $s(u)$ to denote the time that it initiates the rumor. If $u$ is not a rumor source, then $s(u)=\infty$. For two nodes $u$ and $v$, the distance between them is denoted as $d(u,v)$. The time that a rumor initiated at node $u$ is received by node $v$ is $r_u(v)=s(u)+d(u,v)$. For a set of rumor sources $A\subseteq V$, the time that the rumor from $A$ is received by node $v$ is $r_A(v)=\min\{r_u(v):u\in A\}$.

**Definition 3.3 (Set Resolving Set (SRS))** Let $K$ be a node subset of $V$. Two node set $A, B\subseteq V$ are *distinguishable* by $K$ if there exist two nodes $x, y\in K$ such that

$$r_A(x)-r_A(y)\neq r_B(x)-r_B(y)$$

For a node set $A\subseteq V$, a node $x\in A$ is detectable if $A$ and $A\backslash\{x\}$ are distinguishable by $V$. Node set $A$ is *detectable* if every node in $A$ is detectable. Let $\mathscr{D}$ be the family of detectable node sets. Node set $K\subseteq V$ is an SRS if any different detectable node sets $A, B\in\mathscr{D}$ are distinguishable by $K$.

Multi-Rumor-Source Detection problem (MRSD): find an SRS $K$ with the smallest cardinality.

The following theorem characterizes the condition under which a node set is detectable. The idea behind the condition is as follows: when a rumor is initiated at a node $x$ after $x$ can receive the same rumor from some other nodes, then one cannot tell whether the rumor is initiated by $x$ or $x$ merely relays the rumor.

**Theorem 3.1** *A node set $A$ is detectable if and only if for every node $x \in A$,*

$$s(x) < r_{A\setminus\{x\}}(x) \tag{3.7}$$

***Proof*** Suppose there is a node $x \in A$ such that $s(x) \geq r_{A\setminus\{x\}}(x)$. Then, there is a node $z \in A \setminus \{x\}$ such that $s(x) \geq r_z(x) = s(z) + d(z, x)$. For any node $y \in V$, $r_x(y) = s(x) + d(x, y) \geq s(z) + d(z, x) + d(x, y) \geq s(z) + d(z, y) = r_z(y)$.

Hence, $r_A(y) = \min\{r_x(y), r_{A\setminus\{x\}}(y)\} = r_{A\setminus\{x\}}(y)$. It follows that $r_A(y_1) - r_A(y_2) = r_{A\setminus\{x\}}(y_1) - r_{A\setminus\{x\}}(y_2)$ for any nodes $y_1, y_2 \in V$, and thus $A$ and $A \setminus \{x\}$ are not distinguishable by $V$. This finishes the proof for the necessity.

To show the sufficiency, notice that $s(x) < r_{A\setminus\{x\}}(x)$ implies that

$$\begin{aligned} r_A(x) &= \min\{r_x(x), r_{A\setminus\{x\}}(x)\} = \min\{s(x), r_{A\setminus\{x\}}(x)\} \\ &= s(x) < r_{A\setminus\{x\}}(x) \end{aligned} \tag{3.8}$$

For any node $y_1 \in A$, choose $y_2 \in A$ such that $s(y_2) = \min_{y\in A\setminus\{y_1\}}\{s(y)\}$. Then

$$r_{A\setminus\{y_1\}}(y_2) = s(y_2) = r_A(y_2) \tag{3.9}$$

This is because of property (3.8) and the observation $s(y_2) = r_{y_2}(y_2) \geq r_{A\setminus\{y_1\}}(y_2) = \min_{y\in A\setminus\{y_1\}}\{s(y) + d(y, y_2)\} \geq \min_{y\in A\setminus\{y_1\}}\{s(y)\} = s(y_2)$. Also by (3.8), we have

$$r_A(y_1) < r_{A\setminus\{y_1\}}(y_1) \tag{3.10}$$

Combining (3.9) and (3.10), we have

$$r_A(y_1) - r_A(y_2) < r_{A\setminus\{y_1\}}(y_1) - r_{A\setminus\{y_1\}}(y_2)$$

So, $A$ and $A \setminus \{y_1\}$ are distinguishable by $y_1$ and $y_2$. The sufficiency follows from the arbitrariness of $y_1$.

*Remark 3.1* If the starting time for all nodes is a constant, then condition (3.7) is satisfied at all nodes. So, this condition does occur in the real world.

**Lemma 3.1** *Let $A, B$ be two detectable node sets with $A \setminus B \neq \emptyset$. Then for any node $x \in A \setminus B$ and any node $y \in B$, node sets $A$ and $B$ are distinguishable by $x$ and $y$.*

***Proof*** Suppose the lemma is not true, then there exists a node $x \in A \setminus B$ and a node $y \in B$ such that

$$r_A(x) - r_A(y) = r_B(x) - r_B(y) \tag{3.11}$$

Since both $A$ and $B$ are detectable, we see from property (3.8) that

$$r_A(x) = s(x) \text{ and } r_B(y) = s(y)$$

Combining these with (3.11), we have

$$s(x) - r_A(y) = r_B(x) - s(y) \leq r_y(x) - s(y) = d(y, x) \tag{3.12}$$

Then,

$$r_x(y) = s(x) + d(y, x) \leq r_A(y) \leq r_x(y) \tag{3.13}$$

It follows that the inequalities in (3.12) and (3.13) become equalities, that is,

$$r_A(y) = r_x(y) \text{ and } r_B(x) = r_y(x)$$

But then,

$$r_A(x) - r_A(y) = s(x) - r_x(y) = -d(x, y) < d(y, x) = r_y(x) - s(y) = r_B(x) - r_B(y)$$

contradicting (3.11). The lemma is proved.

As a consequence of Lemma 3.1, we have the following theorem.

**Theorem 3.2** *Node set $V$ is an SRS.*

Theorem 3.2 shows that $V$ is a trivial solution to the MRSD problem. In next section, we shall present an approximation algorithm for the problem.

### *3.3.2 The Algorithm*

In this section, we present a greedy algorithm for MRSD. The algorithm starts from $\mathscr{T} = \emptyset$, and iteratively adds into $\mathscr{T}$ node-pairs with the highest efficiency (which will be defined later) until all sets can be distinguished by some node-pair in $\mathscr{T}$. The output of the algorithm is $K = \bigcup_{T \in \mathscr{T}} T$.

#### 3.3.2.1 Potential Function

The efficiency of a node-pair is related with a potential function $f$ defined as follows. Two detectable node sets $A$ and $B$ are *equivalent* under $\mathscr{T}$, denoted as $A \equiv_{\mathscr{T}} B$, if $A$ and $B$ are not distinguishable by any node-pair in $\mathscr{T}$. Under $\equiv_{\mathscr{T}}$, detectable node sets $\mathscr{F}$ is divided into equivalence classes. The equivalence class containing detectable node set $A$ is denoted as $[A]_{\mathscr{T}}$. Suppose the equivalence classes under $\equiv_{\mathscr{T}}$ are $\mathscr{F}_1, \ldots, \mathscr{F}_k$. Define $\pi(\mathscr{T}) = \prod_{i=1}^{k} |\mathscr{F}_i|!$ and

$$f(\mathscr{T}) = -\log_2 \pi(\mathscr{T}) \tag{3.14}$$

For a node-pair $T = \{x, y\}$, let

$$\Delta_T f(\mathscr{T}) = f(\mathscr{T} \cup \{T\}) - f(\mathscr{T})$$

We shall show that $f$ is monotone increasing and submodular. The proof idea is similar to the one in [17] which studies group testing. The difference is that in [17], only elements need to be distinguished. While in this paper, distinguishing sets needs more technical details. The following is a technical result of combinatorics (see Fig. 3.6a for an illustration of its conditions).

**Lemma 3.2** *Suppose* $\{h_{ij}\}_{i=1,\ldots,p}^{j=1,\ldots,q}$ *is a set of non-negative integers. For* $i = 1, \ldots, p$, $a_i = \sum_{j=1}^{q} h_{ij}$. *For* $j = 1, \ldots, q$, $b_j = \sum_{i=1}^{p} h_{ij}$. *Furthermore,* $\sum_{i=1}^{p} a_i = \sum_{j=1}^{q} b_j = g$. *Then*

$$\frac{g!}{\prod_{j=1}^{q} b_j!} \geq \frac{\prod_{i=1}^{p} a_i!}{\prod_{i=1}^{p} \prod_{j=1}^{q} h_{ij}!} \tag{3.15}$$

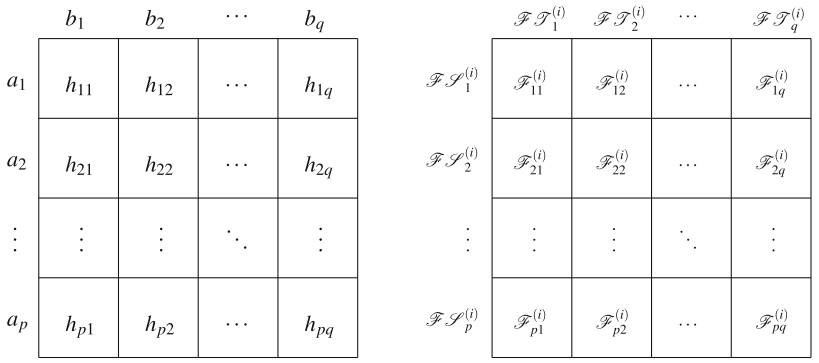


**Fig. 3.6** (**a**) Illustration for the conditions in Lemma 3.2. (**b**) Refinement of equivalence class $\mathscr{F}_i$ by adding $S$ and $T$

***Proof*** Consider the expansion of the following multi-variable polynomial:

$$
\begin{aligned}
&(x_{11} + \cdots + x_{1q})^{a_1} \cdots (x_{p1} + \cdots + x_{pq})^{a_p} \\
&= \left( \sum_{a_{11}+\cdots+a_{1q}=a_1} \frac{a_1!}{\prod_{j=1}^{q} a_{1j}!} x_{11}^{a_{11}} \cdots x_{1q}^{a_{1q}} \right) \cdots \\
&\quad \left( \sum_{a_{p1}+\cdots+a_{pq}=a_p} \frac{a_p!}{\prod_{j=1}^{q} a_{pj}!} x_{p1}^{a_{p1}} \cdots x_{pq}^{a_{pq}} \right) \\
&= \sum \frac{\prod_{i=1}^{p} a_i!}{\prod_{i=1}^{p} \prod_{j=1}^{q} a_{ij}!} x_{11}^{a_{11}} \cdots x_{1q}^{a_{1q}} \cdots x_{p1}^{a_{p1}} \cdots x_{pq}^{a_{pq}}
\end{aligned}
$$

where the sum is over all non-negative integers $\{a_{ij}\}_{i=1,\ldots,p}^{j=1,\ldots,q}$ satisfying $\sum_{j=1}^{q} a_{ij} = a_i$ for $i = 1, \ldots, p$. Setting $x_{1j} = \cdots = x_{pj} = x_j$ for $j = 1, \ldots, q$ in the above equation, we have

$$
\begin{aligned}
&(x_1 + \cdots x_q)^{a_1+\cdots+a_p} \\
&= \sum \frac{\prod_{i=1}^{p} a_i!}{\prod_{i=1}^{p} \prod_{j=1}^{q} a_{ij}!} x_1^{a_{11}+\cdots+a_{p1}} \cdots x_q^{a_{1q}+\cdots+a_{pq}}
\end{aligned} \tag{3.16}
$$

On the other hand,

$$
\begin{aligned}
&(x_1 + \cdots x_q)^{a_1+\cdots+a_p} \\
&= (x_1 + \cdots x_q)^{g} = \sum_{r_1+\cdots r_q=g} \frac{g!}{\prod_{j=1}^{q} r_j!} x_1^{r_1} \cdots x_q^{r_q}
\end{aligned} \tag{3.17}
$$

Comparing the coefficients of $x_1^{b_1} \cdots x_q^{b_q}$ in (3.16) and (3.17), we have

$$
\frac{g!}{\prod_{j=1}^{q} b_j!} = \sum \frac{\prod_{i=1}^{q} a_p!}{\prod_{i=1}^{p} \prod_{j=1}^{q} a_{ij}!} \tag{3.18}
$$

where the sum is over all non-negative integers $\{a_{ij}\}_{i=1,\ldots,p}^{j=1,\ldots,q}$ satisfying $\sum_{j=1}^{q} a_{ij} = a_i$ for $i = 1, \ldots, p$ and $\sum_{i=1}^{p} a_{ij} = b_j$ for $j = 1, \ldots, q$. Since $\{h_{ij}\}_{i=1,\ldots,p}^{j=1,\ldots,q}$ satisfy these restrictions, the right-hand side of (3.15) is one term contained in the right-hand side of (3.18). Then, the Lemma follows.

We shall use the following characterization of monotonicity and submodularity.

**Lemma 3.3 ([55, Lemma 2.25])** *Let $f$ be a function defined on all subsets of a set $U$. Then $f$ is submodular and monotone increasing if and only if for any two subsets $R \subseteq S \subseteq U$ and any element $x \in U$,*

$$\Delta_x f(R) \geq \Delta_x f(S)$$

**Lemma 3.4** *The function $f$ defined in (3.14) is submodular and monotone increasing.*

***Proof*** To use Lemma 3.3, we are to show that for any families of node-pairs $\mathscr{T}_1 \subseteq \mathscr{T}_2$ and any node-pair $T$,

$$\Delta_T f(\mathscr{T}_1) \geq \Delta_T f(\mathscr{T}_2) \tag{3.19}$$

In fact, it suffices to prove (3.19) for the case that $|\mathscr{T}_2 \setminus \mathscr{T}_1| = 1$. Then, induction argument will yield the result for the general case. So, in the following, we assume that $\mathscr{T}_2 = \mathscr{T}_1 \cup \{S\}$, where $S$ is a node-pair. In this case, (3.19) is equivalent to

$$\frac{\pi(\mathscr{T}_1)}{\pi(\mathscr{T}_1 \cup \{T\})} \geq \frac{\pi(\mathscr{T}_1 \cup \{S\})}{\pi(\mathscr{T}_1 \cup \{S, T\})} \tag{3.20}$$

Suppose the equivalence classes under $\equiv_{\mathscr{T}}$ are $\mathscr{F}_1, \ldots, \mathscr{F}_k$. Notice that for any detectable node set $A$, $[A]_{\mathscr{T} \cup \{S\}} \subseteq [A]_{\mathscr{T}}$, that is, adding one node-pair results in a refinement of equivalence classes. Also notice that for any detectable node set $A$,

$$[A]_{\mathscr{T} \cup \{S,T\}} = [A]_{\mathscr{T} \cup \{S\}} \cap [A]_{\mathscr{T} \cup \{T\}}$$

Hence we may assume (see Fig. 3.6b for an illustration) that for each $i = 1, \ldots, k$,

(a) equivalence classes under $\mathscr{T} \cup \{S, T\}$ which are contained in $\mathscr{F}_i$ are $\{\mathscr{F}_{s,t}^{(i)}\}_{s=1,\ldots,l_i}^{t=1,\ldots,m_i}$;
(b) For $s = 1, \ldots, l_i$, let $\mathscr{F}\mathscr{S}_s^{(i)} = \bigcup_{t=1}^{m_i} \mathscr{F}_{s,t}^{(i)}$. Equivalence classes under $\equiv_{\mathscr{T} \cup \{S\}}$ contained in $\mathscr{F}_i$ are $\{\mathscr{F}\mathscr{S}_s^{(i)}\}_{s=1}^{l_i}$;
(c) For $t = 1, \ldots, m_i$, let $\mathscr{F}\mathscr{T}_t^{(i)} = \bigcup_{s=1}^{l_i} \mathscr{F}_{s,t}^{(i)}$. Equivalence classes under $\equiv_{\mathscr{T} \cup \{T\}}$ contained in $\mathscr{F}_i$ are $\{\mathscr{F}\mathscr{T}_t^{(i)}\}_{t=1}^{m_i}$.

Taking $a_l = |\mathscr{F}\mathscr{S}_l^{(i)}|$, $b_j = |\mathscr{F}\mathscr{T}_j^{(i)}|$, $h_{lj} = |\mathscr{F}_{lj}^{(i)}|$, and $g = |\mathscr{F}_i|$, the conditions of Lemma 3.2 are satisfied, and thus

$$\frac{|\mathscr{F}_i|!}{\prod_{j=1}^{q} |\mathscr{F}\mathscr{T}_j^{(i)}|!} \geq \frac{\prod_{l=1}^{p} |\mathscr{F}\mathscr{S}_l^{(i)}|!}{\prod_{l=1}^{p} \prod_{j=1}^{q} |\mathscr{F}_{lj}^{(i)}|!}$$

which is exactly the desired inequality (3.20).

**Lemma 3.5** *Suppose node-pair $T$ divides $\mathscr{F}$ into $k$ equivalence classes. Then*

$$\Delta_T f(\emptyset) \leq |\mathscr{F}| \log_2 k$$

***Proof*** Suppose the equivalence classes under $\equiv_T$ have cardinalities $n_1, \ldots, n_k$, respectively. Then

$$\Delta_T f(\emptyset) = f(\{T\}) - f(\emptyset) = \log_2 \left( \frac{|\mathscr{F}|!}{\prod_{i=1}^{k} n_i!} \right) \leq \log_2 \left( k^{|\mathscr{F}|} \right) = |\mathscr{F}| \log_2 k$$

where the inequality can be seen by setting $x_1 = \cdots = x_k = 1$ in the following equation:

$$(x_1 + \cdots + x_k)^{|\mathscr{F}|} = \sum_{n_1 + \cdots + n_k = n} \frac{|\mathscr{F}|!}{\prod_{i=1}^{k} n_i!} x_1^{n_1} \ldots x_k^{n_k}$$

The lemma is proved.

#### 3.3.2.2 The Algorithm and Its Approximation Ratio

As stated at the beginning of Sect. 3.3.2, an SRS will be derived from a family $\mathscr{T}$ of node-pairs such that

$$\text{node-pairs in } \mathscr{T} \text{ can distinguish all detectable node sets.} \tag{3.21}$$

Call any family of node-pairs as a *test family*, and call a test family satisfying condition (3.21) as a *valid test family*.

**Lemma 3.6** *Suppose $\mathscr{T}$ is a valid test family. Let $K = \bigcup_{T \in \mathscr{T}} T$ and $x$ be an arbitrary node in $K$. Then $\widetilde{\mathscr{T}} = \{(x, y) \colon y \in K \setminus \{x\}\}$ is also a valid text family.*

***Proof*** Observe that if two detectable node sets $A, B$ are distinguished by $\{y, z\} \in \mathscr{T}$, then they can be distinguished by either $\{x, y\}$ or $\{x, z\}$. The lemma follows.

Notice that all node-pairs in $\widetilde{\mathscr{T}}$ have a common element. We call such a test family as a *canonical test family*. Notice that $\bigcup_{T \in \mathscr{T}} T = \bigcup_{T \in \widetilde{\mathscr{T}}} T = K$. Hence $\mathscr{T}$ and $\widetilde{\mathscr{T}}$ are equivalent in the sense that they produce a same SRS. As a consequence, to find an SRS, it suffices to consider canonical test families, that is, to find a node $x$ and a valid test family $\mathscr{T}_x \subseteq \mathscr{P}_x = \{\{x, y\} \colon y \in V \setminus \{x\}\}$.

In order to analyze the approximation ratio, we have to compare the size of the approximation solution with that of an optimal one. Since we do not know which node is in an optimal solution, we have to "guess." To be more concrete, for each node $x \in V$, the algorithm finds a valid test family $\mathscr{T}_x \subseteq \mathscr{P}_x$. Let $K_x = \bigcup_{T \in \mathscr{T}_x} T$.

**Algorithm 3** Greedy algorithm for MRSD

Input: A graph $G = (V, E)$.
Output: A node set $K$ which is an SRS.

```
for all x ∈ V do
    Set 𝒯_x ← ∅.
    while there exists a node-pair T ∈ 𝒫_x such that Δ_T f(𝒯_x) > 0 do
        select node-pair T ∈ 𝒫_x with the maximum Δ_T f(𝒯_x).
        𝒯_x ← 𝒯_x ∪ {T}.
    end while
    K_x = ⋃_{T∈𝒯_x} T.
end for
Output K ← arg min{|K_x| : x ∈ V}.
```

The final output of the algorithm is $K = \arg\min_{x \in V} |K_x|$. The details of the algorithm for MRSD is described in Algorithm 3.

**Lemma 3.7** *A test family $\mathscr{T}$ is valid if and only if $\Delta_T f(\mathscr{T}) = 0$ for any node-pair $T$.*

***Proof*** First, we make some observation. Suppose the equivalence classes under $\mathscr{T}$ are $\mathscr{F}_1, \ldots, \mathscr{F}_k$. For each $i = 1, \ldots, k$, $\mathscr{F}_i$ is refined under $\mathscr{T} \cup \{T\}$ into equivalence classes $\mathscr{F}_1^{(i)}, \ldots, \mathscr{F}_{l_i}^{(i)}$. Then

$$\begin{aligned} \Delta_T f(\mathscr{T}) &= \log_2 \left( \frac{\prod_{i=1}^{k} |\mathscr{F}_i|}{\prod_{i=1}^{k} \prod_{j=1}^{l_i} |\mathscr{F}_j^{(i)}|} \right) \\ &= \log_2 \left( \prod_{i=1}^{k} \frac{|\mathscr{F}_i|}{\prod_{j=1}^{l_i} |\mathscr{F}_j^{(i)}|} \right) \end{aligned} \tag{3.22}$$

Notice that $|\mathscr{F}_i| / \prod_{j=1}^{l_i} |\mathscr{F}_j^{(i)}|$ is the number of ways to put $|\mathscr{F}_i|$ balls into $l_i$ labeled boxes such that the $j$-th box contains $|\mathscr{F}_j^{(i)}|$ balls ($j = 1, \ldots, l_i$). So,

$$|\mathscr{F}_i| / \prod_{j=1}^{l_i} |\mathscr{F}_j^{(i)}| \tag{3.23}$$

is a positive integer which equals 1 if and only if $l_i = 1$.

Notice that $l_i = 1$ implies that adding node-pair $T$ into $\mathscr{T}$ does not incur a strict refinement of $\mathscr{F}_i$.

If $\mathscr{T}$ is a valid test family, then every equivalence class has cardinality 1, and thus $f(\mathscr{T}) = 0$. Combining this with the fact that $f$ is a non-positive monotone

increasing function, we see that the maximum value of $f$ is zero and thus $\Delta_T(\mathscr{T}) = 0$ holds for any node-pair $T$.

If $\mathscr{T}$ is not a valid test family, then there exist two different detectable node sets $A, B$ which cannot be distinguished by $\mathscr{T}$. Since $A \neq B$, we may assume that $A \setminus B \neq \emptyset$. By Lemma 3.1, $A, B$ can be distinguished by a node-pair $\{y, z\}$ with $y \in A \setminus B$ and $z \in B$. Then by Lemma 3.6, $A, B$ can be distinguished by $T = \{x, y\}$ or $\{x, z\}$. In this case, at least one equivalence class under $\mathscr{T}$ is refined by adding $T$. In other words, there is an $i \in \{1, \ldots, k\}$ such that $|\mathscr{F}_i| / \prod_{j=1}^{l_i} |\mathscr{F}_j^{(i)}| > 1$. Then by (3.22), $\Delta_T(\mathscr{T}) > 0$. The lemma is proved.

**Theorem 3.3** *Suppose $\gamma$ is the maximum number of equivalence classes divided by one node-pair. Then, Algorithm 3 correctly computes an SRS with approximation ratio at most* $1 + \ln\left(|\mathscr{F}| \log_2 \gamma\right)$.

***Proof*** By Lemma 3.7, we see that every $\mathscr{T}_x$ computed in the algorithm is a valid test family. The correctness follows.

To analyze the approximation ratio, suppose $K^*$ is an optimal solution to MRSD and $x$ is a node in $K^*$. Let $\mathscr{T}^* = \{\{x, y\} \colon y \in K^* \setminus \{x\}\}$.

Consider the test family $\mathscr{T}_x$ produced by the greedy algorithm for node $x$. We claim that every node-pair $T$ chosen in the algorithm satisfies

$$\Delta_T f(\mathscr{T}_x) \geq 1 \tag{3.24}$$

By expression (3.22), this is equivalent to show

$$\prod_{i=1}^{k} \frac{|\mathscr{F}_i|}{\prod_{j=1}^{l_i} |\mathscr{F}_j^{(i)}|} \geq 2 \tag{3.25}$$

Since every $T$ taken in the algorithm has $\Delta_T f(\mathscr{T}_x) > 0$, which is equivalent to $\prod_{i=1}^{k} \frac{|\mathscr{F}_i|}{\prod_{j=1}^{l_i} |\mathscr{F}_j^{(i)}|} > 1$, we see that at least one $|\mathscr{F}_i| / \prod_{j=1}^{l_i} |\mathscr{F}_j^{(i)}|$ is greater than 1, and thus is at least 2. Inequality (3.25) follows from this observation and property (3.23). Claim (3.24) is proved.

We shall use Theorem 3.7 in [55], which says, using terminologies here, that as long as (3.24) is true, then

$$|\mathscr{T}_x| \leq \left(1 + \ln \frac{f(\mathscr{T}^*) - f(\emptyset)}{|\mathscr{T}^*|}\right) \cdot |\mathscr{T}^*| \tag{3.26}$$

By a property of submodular function (see [55, Lemma 2.23]),

$$\sum_{T \in \mathscr{T}^*} \Delta_T f(\emptyset) \geq \Delta_{\mathscr{T}^*} f(\emptyset) = f(\mathscr{T}^*) - f(\emptyset)$$

Combining this with Lemma 3.5,

$$\begin{aligned}&\frac{f(\mathscr{T}^*)-f(\emptyset)}{|\mathscr{T}^*|}\\&\leq \frac{\sum_{T\in\mathscr{T}^*}\Delta_T f(\emptyset)}{|\mathscr{T}^*|}\leq \max_{T\in\mathscr{T}^*}\{\Delta_T f(\emptyset)\}\leq |\mathscr{F}|\log_2\gamma\end{aligned} \tag{3.27}$$

Combining (3.26), (3.27) with $|K_x| = |\mathscr{T}_x| - 1$ and $|K^*| = |\mathscr{T}^*| - 1$, we have

$$|K_x|\leq \left(1+\ln\left(|\mathscr{F}|\log_2\gamma\right)\right)(|K^*|-1)+1\leq \left(1+\ln\left(|\mathscr{F}|\log_2\gamma\right)\right)|K^*|$$

The approximation ratio follows since the algorithm chooses a node $x_0$ with $|K_{x_0}| = \min_{y\in V}|K_y| \leq |K_x|$.

*Remark 3.2* Notice that in a worst case, the number of detectable node sets is $\Theta(2^n)$. In fact, if the starting time for all nodes is a constant, then by Remark 3.1, every nonempty node set is detectable, and thus $|\mathscr{F}| = 2^n - 1$. In such a case, the approximation ratio is $(1 + n\ln 2 + \ln\log_2\gamma)$, which is no better than a trivial bound $n$. However, in the real world, it is reasonable to assume that the number of rumor sources is at most a constant number $r$, and only those detectable node sets of cardinality at most $r$ are considered. In this case, $|\mathscr{F}| = O(n^r)$, and the approximation ratio is $(1 + r\ln n + \ln\log_2\gamma)$.

*Remark 3.3* Notice that $\gamma \leq 2D + 1$, where $D$ is the diameter of the graph. To see this, suppose $T = \{x, y\}$ is a node-pair, $A$ is a node set, and $r_A(x) - r_A(y) = c$, then a node set $B$ belongs to equivalence class $[A]_{\{T\}}$ if and only if $r_B(x) - r_B(y) = c$. Notice that $c$ has at most $2D + 1$ different values, namely, $\{-D, -(D-1), \ldots, -1, 0, 1, \ldots, D-1, D\}$. So, one node-pair divides $\mathscr{F}$ into at most $2D + 1$ equivalence classes. In a social network, $D$ is a small constant. So, the third term $\ln\log_2\gamma$ in the above approximation ratio is not large.

### 3.3.3 Simulation Results

In this section we experimentally evaluate our greedy algorithm for MRSD, in particular its effectiveness in finding rumor sources—how many sources it identifies, whether it correctly identifies and its scalability.

As discussed in the introduction, the existing proposals for identifying rumor sources consider significantly different problems settings than we do. The rumor centrality of Shah and Zaman [171, 173] can only discover one rumor source, while estimators proposed in [123] consider a completely different infection model from ours. As such it is not meaningful to compare performances, and therefore here we only consider the greedy algorithm for MRSD.

In our study we conduct simulations on synthetic networks exemplifying different types of structure—including geometric trees, regular trees, and small-world networks. In general, we set the rumor sources, simulate diffusion process, and record the times of monitors when they received the rumors. Then, given the times and network structure, we try to infer the number rumor sources and where they are.

As described in Sect. 3.3.1, the diffusion model is implemented as a discrete event in Java. Each hop takes one time unit. Note that the cascade starts from all rumor sources at the same time stamp. The number of rumor sources is set as $k$. For each $k$, we perform large number of simulation runs to get high precession.

#### 3.3.3.1 Effectiveness of Greedy in Identifying How Many

The number of infection sources $k$ are chosen to be 1, 2, 3, and 4. For each type of network and each number of infection sources, we perform 1000 simulation runs with 500 monitors. The estimation results for the number of infection sources in different scenarios are shown in Fig. 3.7. It can be seen that our algorithm correctly finds the number of infection sources more than 95% of the time for geometric trees, and more than 86% of the time for regular trees. The accuracy of about 79% for small-world networks is slightly lower than that for the tree networks, as the node-pair for a small-world network is estimated based on the BFS heuristics, thus additional errors are introduced into the procedure. It also shows the power of our approach, as we can easily identify the true number of seeds for most cases using a principled approach.

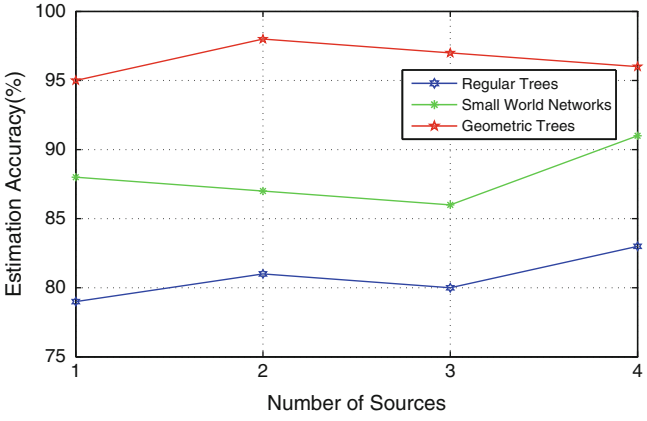


**Fig. 3.7** Estimating the number of rumor sources

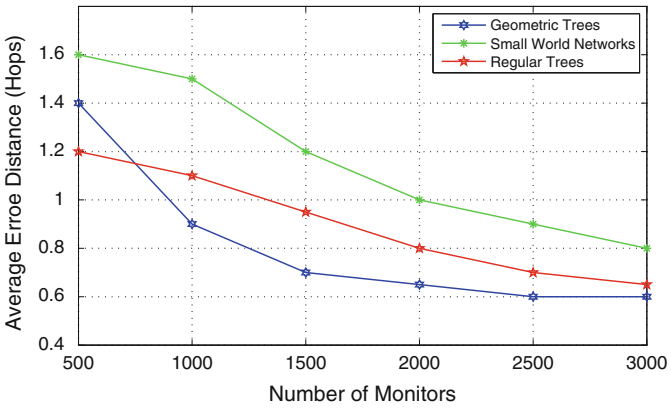


**Fig. 3.8** Estimating the average error distance between the identified source and the actual source

#### 3.3.3.2 Effectiveness of Greedy in Identifying Which Ones

To quantify the performance of identified rumor sources, we propose error distance. Error distance is defined as the average distances between the estimates and the respective rumor sources. To be specific, we match the estimated source nodes with the actual sources so that the sum of the error distances between each estimated source and its match is minimized. If we have incorrectly estimated the number of infection sources, we neglect the extra number of found nodes since here we only focus on the error distance between correct sources. In Fig. 3.8, we see that the proposed algorithm finds rumor sources that have small error distance on average. Note that the reported results here are also based on 1000 trials. For geometric trees, the average error distance lies between 1.4 and 0.6 hop. For regular trees, the error distance decreases from 1.2 to 0.65 hops. For small-world networks, the value is between 1.6 and 0.8. In general, the average error distance is less than two hops. Moreover, as the number of monitors increases, error distance will start to drop.

#### 3.3.3.3 Scalability

Figure 3.9 demonstrates the average computation time of greedy after running it on increasingly larger infected graphs (as the complexity depends on the size of the monitors). We use small-world network graph with $k = 2$. The statistics of running time is based on 10 runs for each graph. As we can see, the running time is linear on the number of edges of the infected graph. Thus, overall our algorithm scales well with high performance in solution quality.

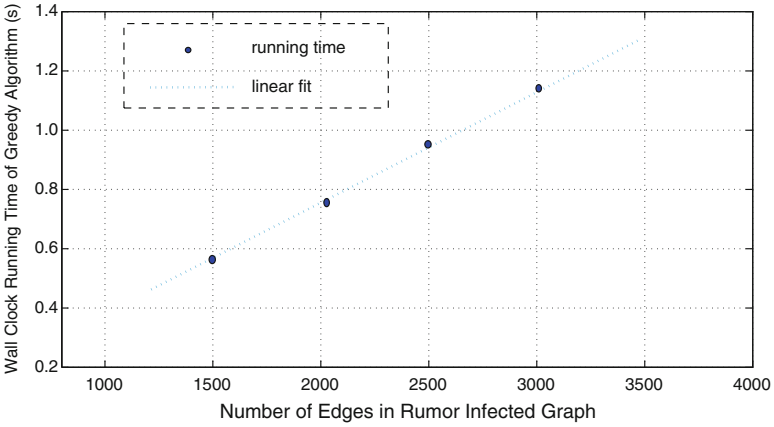


**Fig. 3.9** Wall-clock computation time (in seconds) by our greedy algorithm for increasingly larger infected graphs. $k = 2$. Each point average of 10 runs

## 3.4 Conclusion

In this chapter, we have studied the rumor source detection problem under different scenarios. In Sect. 3.2, we have investigated the problem of rumor source detection in online social networks when lacking of text information. We formalize the problem and define rumor quantifier, a probability based score for ranking how likely a node is going to be the actual rumor source. The idea behind it is simple: a cascade is more likely to spread from a rumor source to the monitors who have received the information but less likely to those who have not. To compute the rumor quantifier of each node, we developed a scalable algorithm, RSD, to detect the rumor source and differentiate the rumors.

We evaluated various monitor deployment on real online social network—Twitter—with rumor propagating according to the popular independent cascade model, and showed that our algorithm, RSD is able to accurately identify the rumor source from a large network when there are reasonable number of monitors. Our future work includes two aspects: first, discuss the situation when there are multiple rumor sources spreading in the network. Second, we will try to develop both structure and content combined method to identify rumor sources.

In Sect. 3.3 we discussed finding multiple rumor sources, the challenging problem of identifying the nodes from which an infection in a graph started to spread. We first gave the definition of set resolving set(SRS) and proposed to employ minimum SRS for identifying the set of rumor sources from which the rest of nodes in the graph can be distinguished correctly. In this framework, the inference is based only on knowledge of the infected monitors and the underlying network structure. We have designed a highly efficient greedy algorithm using submodularity analysis and theoretically proved the performance ratio to be $1 + \ln\left(|\mathscr{F}| \log_2 \gamma\right)$, given that $\gamma$ is the maximum number of equivalence classes divided by one node-pair.

Several improvements and future directions are possible. One direction is to extend our methodology to different applications, including influence maximization, rumor blocking, etc. and see how the proposed methodology leads to deeper insights. Another promising direction is to tackle the MRSD problem in different diffusion models, such as models with transmission probabilities between nodes considered or models without submodularity property.