A Comparative Analysis of Source Identification Algorithms

Pablo A. Curiel, Richard C. Tillquist

California State University, Chico, USA

Abstract

Identifying the source of a spread in a network, often referred to as the patient-zero problem, is a difficult task when only given the subgraph of infected nodes. Since 2011, several algorithms have been created to address this problem. Their success depends on many factors, such as the graph's size and structure, the infection rate, and the time since the start of the spread. This paper empirically compares four prominent source identification algorithms when applied to randomly-generated graphs and to a real-world airport network.

1. Introduction

The importance of modeling spread processes on networks cannot be overstated. Applications related to the spread of diseases in a physical-contact network, rumors, knowledge, or memes in a social network, electricity in a power grid network, and malware attacks in a computer-system network [1] all require a deep understanding of the relationship between network structure and spread dynamics.

In recent years, determining the source of a spread (often referred to as **patient-zero**) has become a popular area of research in network science. This is largely due to how networks capture important features necessary for spread. Many traditional models rely on the assumption of homogeneous mixing in the population, which states that all individuals have the same probability of coming into contact with any other individual. This assumption implies that all individuals are connected and disregards the true contact network. Determining the source of a spread has many critical real-world applications, such as contact tracing, optimizing the allocation of immunization resources, mitigating the spread of rumors or misinformation, and defending against malware attacks.

This paper focuses on comparing four state-of-the-art methods of identifying the source of a spread when applied to random graphs of varying type and size, as well as a real-world U.S. airport network [2].

2. Background

Traditional spread models often utilize continuous-time differential equations to simulate a spread. Perhaps the most popular of these models is the **S**usceptible-**I**nfected model (SI-model), where individuals in a population are divided into two distinct classes or states [3]. There are many variations of the SI-model, which include different numbers or orderings of states. Some of these variations include the SIR-model that adds a **R**ecovered (or removed) state, the SIS-model that allows an infected individual to return to a **S**usceptible state, and the SEIR-model that includes a pre-infectious **E**xposed state.

These traditional models often rely on two fundamental assumptions: compartmentalization and homogeneous mixing [1]. Compartmentalization refers to the classifying of individuals in a population based on their disease state. Homogeneous mixing, also known as well-mixed, allows any pair of individuals to interact. Such models help capture the broader picture of a spread, but their assumption of a well-mixed population reduces their realism. This is where modeling spread using networks becomes important.

In contrast to traditional models, modeling spread on networks takes into account each individual's contacts. This helps provide a more realistic representation of a spread, as individuals can only interact with their direct connections. In addition, spread models on networks use discrete time-steps instead of continuous time. For the SI-model on a network, a single source node is chosen at time $t = 0$ and classified as infected. At each following time-step, an infected node can infect any of its susceptible one-hop neighbors with probability β, the infection rate. Given a connected network G and $\beta > 0$, $I_N \to N$ as $t \to \infty$, where I_N is the number of infected nodes in the network and N is the total number of nodes in the network. An important quantity when discussing spread in networks is the transmission rate $\beta\langle k\rangle$, where $\langle k\rangle$ is the average degree of the network. This can be thought of as the speed of the spread through a network and is analogous to the basic reproductive rate R_0 in the context of epidemic spreads. More precisely, $R_0 = \frac{\beta\langle k\rangle}{\mu}$, where μ represents a recovery rate. When $R_0 > 1$ we expect an epidemic spread, and

when $R_0 < 1$ we expect a spread to die out [1, 3].

3. Related Work

3.1. Rumor Centrality (2011)

In 2011, Shah and Zaman published the first paper focused on identifying the source of a spread in a network [4]. Their primary contribution was a maximum likelihood estimator for the source node called "rumor centrality". They define $R(v, G_I)$ as the number of permitted permutations of an infected subgraph G_I centered at some node v. A permitted permutation describes all of the possible orderings of nodes that led to a given infected subgraph starting with some node. Rumor centrality is calculated for all nodes in the infected subgraph, and the source, or rumor center, is estimated to be the node with the maximum rumor centrality. In other words, for an estimated source node v^*, $R(v^*, G_I) \geq R(v, G_I)\ \forall v \in G_I$. Their method is designed for tree-structured graphs. For general graphs, the method finds an infected subgraph using a Breadth First Search (BFS) tree rooted at each node, then calculates the rumor centrality for each BFS-tree. This method has a time complexity of $O(|V_I|)$ for tree graphs and $O(|V_I|^2)$ for general graphs, where V_I is the set of infected nodes.

3.2. Jordan Centrality (2017)

Jordan centrality, another measure of node importance with respect to a spread, refers to a node's eccentricity, or maximum hop-distance to all other nodes in the network [5, 6]. For an infected subgraph I and a given node v, Jordan centrality can be defined as

$$J(v, I) = \max_{u \in I} d(v, u)$$

where $d(v, u)$ represents the hop-distance between two infected nodes v and u. A "Jordan infection center" describes an infected node with the smallest Jordan centrality out of all other nodes in an infected subgraph. For a Jordan infection center v^*,

$$v^* = \arg\min_{v \in I} J(v, I)$$

such that $J(v^*, I) \leq J(u, I)\ \forall u \in I$. This node is estimated to be the source of the spread. This method has a time complexity of $O(|V_I||E_I|)$, where E_I is the set of edges in the infected subgraph.

3.3. NETSLEUTH (2014)

The NETSLEUTH algorithm [7] focuses on the minimum description length principle. When applied to source inference, it is referred to as the minimal infection description problem. Given an infected subgraph G_I, the goal of this algorithm is to find a proper set of seed (or source) nodes S and valid propagation ripple R that minimize the total description length $\mathcal{L}(G_I, S, R)$. More precisely,

$$\mathcal{L}(G_I, S, R) = \mathcal{L}(S) + \mathcal{L}(R|S)$$

where $\mathcal{L}(S)$ denotes the encoded length of the seed set S, and $\mathcal{L}(R|S)$ denotes the encoded length of a ripple R starting at a seed set S. A valid propagation ripple can be thought of as a permitted permutation from the Shah and Zaman paper [4]. In the case of a single-source spread, $|S| = 1$ for any seed set S.

The NETSLEUTH algorithm iteratively reduces the total description length of an infected subgraph $\mathcal{L}(G_I, S, R)$ by considering potential sources given the current set of infected nodes until the most succinct encoding is found. The method has a time complexity of $O(|E_I| + |E_F| + |V_I|)$, where E_F denotes the set of edges connecting uninfected nodes with infected neighbors to infected nodes. Notably, this algorithm is linear in the size of the graph.

3.4. LISN (2019)

Nie and Quinn [8] provide an algorithm that calculates a maximum likelihood estimator of each node in the infected subgraph being the source. Their algorithm assumes that the time it takes for the spread to propagate across edges in the graph is exponentially distributed with rate parameter β. As the spread propagates along n edges, the algorithm sums n exponential random variables, which results in a gamma distribution with shape parameter n and rate parameter β. They iteratively update each node's probability of being the source using a gamma distribution.

The node maximizing this probability is chosen as the estimated source. The algorithm does not have a name and is referred to as "LISN" following the title of the paper in which it was introduced [8]. Its time complexity is $O(|V_I|(|V| + |E|))$, where V and E are the set of nodes and edges in the entire graph, respectively.

4. Experiments

4.1. Methods

Using the cosasi Python package [9], we tested the four source inference methods described in the previous section under different conditions. This package allows users to simulate a spread on a NetworkX [10] graph, apply several source inference methods, and obtain results related to the spread and inference methods. For all simulations, a single source node was chosen at random, and the spread was simulated for up to 100 time-steps. Each source inference method was applied at time-steps $10, 20$, and 30.

Experiments were conducted on four random network types and on one real-world network. Barabási-Albert random graphs [11] exhibit scale-free degree distributions, a feature common to many real-world networks. Here we consider parameter values $m = 1$, which produces trees, and $m = 3$. Erdős-Rényi random graphs [12] serve as a good general baseline. We set the parameter $p = \frac{\ln(N)+1}{N}$ to fall in the connected regime. Watts-Strogatz random graphs have a small diameter and high clustering coefficient for the chosen parameter values $p = 0.01$ and $k = 4$. Graphs of size $100, 250$, and 500 were considered.

Spread simulations were conducted using five different infection rates: $0.1, 0.2, 0.3, 0.4$, and 0.5. A total of $6,000$ spread simulations were conducted on the random networks with $1,500$ for each random graph type. For each set of parameters (graph type, graph size, β-value), 100 spread simulations were run. Spreads across a real-world network of U.S. airports [2] were also simulated with the same set of infection rates and observation times. Due to the size of this network ($1,572$ nodes, $17,214$ edges) and to computational limitations, twenty spreads were simulated for each infection rate. All code is available on GitHub: `https://github.com/pacuriel/cscsu-2023`

4.2. Results

Several metrics were collected to analyze the success of each source inference method including the estimated source's distance from the true source (in number of hops), the average distance from the estimated source to the true source, the true source's rank, and the average of the true source's rank. Rank is a measure of a node's likelihood to be the source node. Ideally, the lower a node's rank is, the more likely it is to be the true source node.

The importance of the infection rate β and the observation times is highlighted in figures 1 and 2. Figure 1 shows the frequency of the estimated source's distance from the true source for the LISN method when applied to Barabási-Albert random graphs with $m = 3$ where $N = 500, \beta = 0.1$, and the observation time t varies. When $t = 10$, the method is able to correctly identify the source node approximately 20% of the time. As t increases to 20 and 30, the method is no longer able to correctly identify the source node, which is shown by the absence of a distance $d = 0$. Similarly, figure 2 shows the frequency of the estimated source's distance from the true source for the rumor centrality method applied to Barabási-Albert random graphs with $m = 1$ where $N = 500, t = 10$, and the infection rate β varies. Rumor centrality is known to excel when applied to tree-structured graphs, as shown by its approximately 40% accuracy of identifying the source when $\beta = 0.1$. However, its top-1 accuracy drops below 10% when $\beta = 0.3$ and below 5% when $\beta = 0.5$.

Table 1 reinforces the significance of the infection rates and observation times, while showing the importance of graph structure. The rows of these tables are source inference methods, and the columns are the ratios of average rank $\langle R \rangle$ to graph size N and average distance between the true and estimated source $\langle d \rangle$ to average diameter $\langle diam(G) \rangle$. All table values were obtained from simulations on graphs with $N = 500$ nodes and varying β and time values.

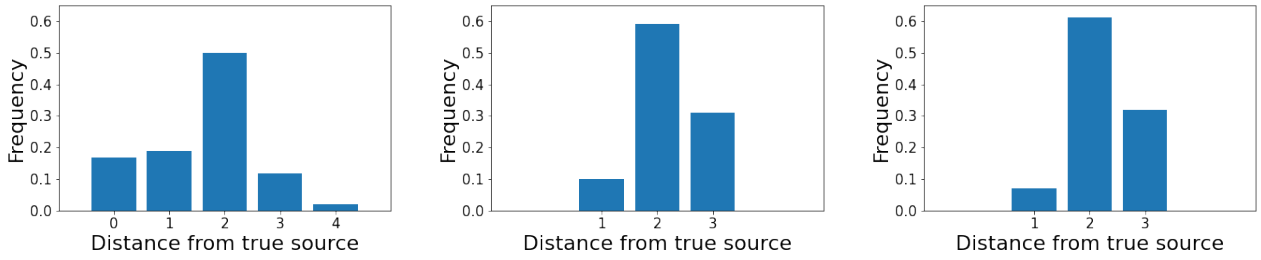
When infection rate $\beta = 0.5$ and time $t = 30$, most algorithms had poor average ranks for the true source node, as well as average distance between the estimated and true sources except on Watts-Strogatz random graphs. With a weaker infection rate of $\beta = 0.1$, we observe marginal improvements with respect to these metrics on most graph types. Notably, Barabási-Albert random graphs with $m = 1$ showed more substantial improvements. With a weak infection rate of $\beta = 0.1$ and an early observation time of $t = 10$, there are marked improvements in measured values for all graph types.

5. Discussion

The experiments conducted here show how the success of each source inference method depends on several factors, including graph structure, infection rate, and the time since the spread began. Methods known to perform well under certain conditions exhibit much poorer performance as infection rate and observation time increase. Interestingly, Watts-Strogatz random graphs displayed the most promising results across all inference methods and spread parameters. Their average distance between estimated and true sources were often relatively low. This is likely related to the overall small average path lengths and high clustering coefficient exhibited by these graphs with the given parameters.

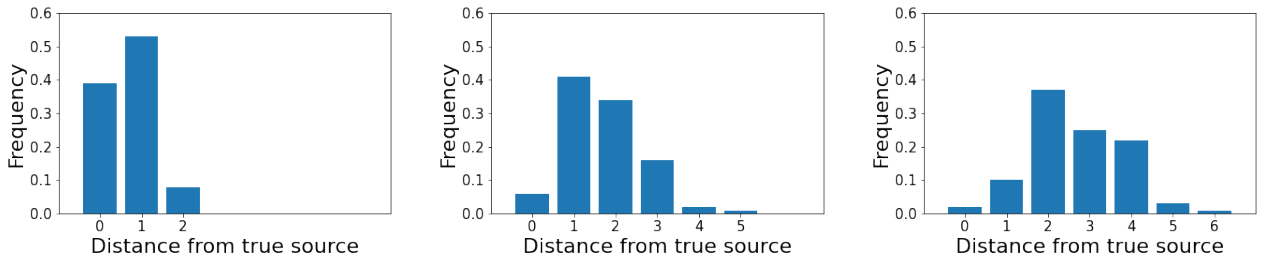
Notably, there was little-to-no change in the results for Barabási-Albert random graphs with $m = 3$, Erdős-Rényi random graphs, or the real-world U.S. airport network after lowering the infection rate. This likely has to do with the structure of these graphs. In particular, the scale-free nature of Barabási-Albert random graphs and the U.S. airport network likely had a large influence on the spread. The hubs associated to graphs of this type can be thought of as super-spreaders in an epidemic context. Their many connections allow them to quickly become infected and propagate the spread to the rest of their uninfected connections. This highlights the importance of taking action early, before it gets into contact with hubs.

The results obtained here are similar to those in the original algorithm papers and other experimental papers. Rumor centrality, Jordan centrality, and LISN performed best on tree-like graphs. Rumor centrality also performed well on Watts-Strogatz random graphs both in the original paper [4] and these experiments. A 2020 paper [13] ran exper-



(a) (b) (c)

Figure 1: LISN algorithm; Barabási-Albert ($m = 3$), $N = 500, \beta = 0.1$; (a): $t = 10$, (b): $t = 20$, (c): $t = 30$.



(a) (b) (c)

Figure 2: Rumor centrality; Barabási-Albert ($m = 1$), $N = 500, t = 10$; (a): $\beta = 0.1$, (b): $\beta = 0.3$, (c): $\beta = 0.5$.

Graph Type	**Method**	**a.)** $\beta = 0.5, t = 30$		**b.)** $\beta = 0.1, t = 30$		**c.)** $\beta = 0.1, t = 10$	
		$\frac{\langle R \rangle}{N}$	$\frac{\langle d \rangle}{\langle diam(G) \rangle}$	$\frac{\langle R \rangle}{N}$	$\frac{\langle d \rangle}{\langle diam(G) \rangle}$	$\frac{\langle R \rangle}{N}$	$\frac{\langle d \rangle}{\langle diam(G) \rangle}$
Barabási-Albert ($m = 1$)	Rumor	0.50	0.25	0.09	0.13	**0.02**	0.06
	Jordan	0.51	0.25	0.08	0.14	**0.02**	0.10
	NETSLEUTH	**0.48**	0.29	0.10	0.16	**0.02**	0.13
	LISN	0.49	0.25	**0.07**	0.16	**0.02**	0.10
Barabási-Albert ($m = 3$)	Rumor	0.50	0.36	0.52	0.36	0.16	0.36
	Jordan	0.51	0.36	0.54	0.36	0.19	0.36
	NETSLEUTH	0.51	0.36	**0.44**	0.36	**0.15**	0.36
	LISN	**0.48**	0.36	0.54	0.36	0.17	0.36
Erdős-Rényi ($p = \frac{\ln(N)+1}{N}$)	Rumor	0.55	0.48	0.50	0.37	**0.06**	0.32
	Jordan	0.50	0.40	0.50	0.32	0.08	0.32
	NETSLEUTH	**0.45**	0.48	**0.48**	0.40	0.14	0.40
	LISN	0.53	0.40	**0.48**	0.40	0.09	0.32
Watts-Strogatz ($p = 0.01, k = 4$)	Rumor	0.12	0.15	**0.02**	0.05	**0.01**	0.02
	Jordan	**0.04**	0.12	**0.02**	0.05	**0.01**	0.02
	NETSLEUTH	0.26	0.19	0.04	0.07	0.02	0.02
	LISN	0.11	0.15	**0.02**	0.03	**0.01**	0.02
US Airport Network	Rumor	**0.43**	0.25	0.40	0.25	0.32	0.25
	Jordan	0.44	0.25	**0.39**	0.25	0.36	0.25
	NETSLEUTH	0.47	0.25	0.47	0.25	0.47	0.25
	LISN	0.46	0.25	**0.39**	0.25	**0.28**	0.13

Table 1: Ratio of average rank of true source $\langle R \rangle$ to graph size N and the ratio of average distance between the estimated and true sources $\langle d \rangle$ to average diameter $\langle diam(G) \rangle$ for varying graph types where $N = 500$. Values for β and t vary and are listed above each column.

iments on Rumor centrality, Jordan centrality, and NETSLEUTH that showed the three methods behaved similarly, with NETSLEUTH slightly outperforming the other two. The results in Table 1 support these conclusions. In the original LISN paper, the authors compare LISN to Jordan centrality. Their results show similar behavior for the two algorithms across several random graph types. In a real-world network experiment, LISN noticeably outperformed Jordan centrality on a Facebook social network. Here, we see that, for a small infection rate and early observation time, LISN outperforms all other methods on a U.S. airport network.

It is important to mention that each of the methods studied perform best under different conditions. For example, rumor centrality is known to perform well on tree-like graphs when there is a single-source [4]. In contrast, NETSLEUTH is capable of inferring multiple sources [7]. Depending on the context, one method may be the best option for inferring the source of a spread.

In terms of the implications for real-world applications, the results presented here highlight the importance of applying source identification algorithms swiftly once a harmful spread is detected. Real-world networks are often scale-free, like Barabási-Albert random graphs or the U.S. airport network. The results suggest that the four algorithms studied have similar performance on these types of graphs. Therefore, it would be beneficial to apply any of these algorithms in the event of a single-source spread.

6. Future Work

There are many possible directions for future work in this area. Here we focused on four source inference methods, two primary metrics related to performance, and five types of networks including one real-world network. However, studying other techniques for estimating the source on a wider array of network structures and including more real-world networks would provide more insight into their characteristics. In particular, comparing the performance of these algorithms more rigorously with statistical methods would allow for more precise comparisons.

As the field of deep learning continues to grow, learning how to best apply these methods to the source identification problem could help improve overall accuracy and efficiency. Research in this area has already begun [14], and it will be interesting to see how it performs once perfected.

7. Conclusion

In this paper, we conducted extensive experiments on four state-of-the-art source inference methods. These experiments varied greatly in both graph and spread parameters. Simulations were conducted on several random graph types and a real-world U.S. airport network. The success of these inference methods was shown to depend on several factors, such as graph structure, graph size, infection rates, and time since the start of the spread.

References

[1] Barabási A, Pósfai M. Network Science. Cambridge University Press, 2016. ISBN 9781107076266. URL https://books.google.com/books?id=iLtGDQAAQBAJ.

[2] Kunegis J. Konect: The Koblenz Network Collection. In Proceedings of the 22nd International Conference on World Wide Web. 2013; 1343–1350. URL http://konect.cc/networks/opsahl-usairport/.

[3] Murray JD. Mathematical Biology: I. An Introduction. Springer, 2002.

[4] Shah D, Zaman T. Rumors in a network: Who's the culprit? IEEE Transactions on Information Theory 2011; 57(8):5163–5181.

[5] Ying L, Zhu K. Diffusion source localization in large networks. Synthesis Lectures on Communication Networks 2018;11(1):1–95.

[6] Luo W, Tay WP, Leng M. On the universality of jordan centers for estimating infection sources in tree networks. IEEE Transactions on Information Theory 2017; 63(7):4634–4657.

[7] Prakash BA, Vreeken J, Faloutsos C. Efficiently spotting the starting points of an epidemic in a large graph. Knowledge and Information Systems 2014;38(1):35–59.

[8] Nie G, Quinn C. Localizing the information source in a network. In TrueFact 2019: KDD 2019 Workshop on Truth Discovery and Fact Checking: Theory and Practice. 2019; .

[9] McCabe LH. cosasi: Graph diffusion source inference in python. Journal of Open Source Software 2022;7(80):4894.

[10] Hagberg A, Swart P, S Chult D. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab (LANL), Los Alamos, NM (United States), 2008.

[11] Barabási AL, Albert R. Emergence of scaling in random networks. science 1999;286(5439):509–512.

[12] Erdős P, Rényi A. On random graphs i. Publicationes mathematicae 1959;6(1):290–297.

[13] Choi J. Epidemic source detection over dynamic networks. Electronics 2020;9(6). ISSN 2079-9292. URL https://www.mdpi.com/2079-9292/9/6/1018.

[14] Shah C, Dehmamy N, Perra N, Chinazzi M, Barabási A, Vespignani A, Yu R. Finding patient zero: Learning contagion source with graph neural networks. CoRR 2020; abs/2006.11913. URL https://arxiv.org/abs/2006.11913.

Address for correspondence:

Pablo A. Curiel
Computer Science Department
400 W. First St, Chico, CA, 95929
pacuriel@csuchico.edu