Reconstructing an Epidemic Outbreak Using Steiner Connectivity

---

A
Thesis
Presented to
the faculty of the School of Engineering and Applied Science
University of Virginia

In partial fulfillment
of the requirements for the degree

Master of Science

in

Computer Science

by

Ritwick Mishra

December 2022

APPROVAL SHEET

This

Thesis

is submitted in partial fulfillment of the requirements
for the degree of

Master of Science

Author: Ritwick Mishra

This Thesis has been read and approved by the examining committee:

Advisor: Anil Vullikanti

Advisor: Abhijin Adiga

Committee Member: Jundong Li

Committee Member: Madhav Marathe

Committee Member:

Committee Member:

Committee Member:

Committee Member:

Accepted for the School of Engineering and Applied Science:

Jennifer L. West, School of Engineering and Applied Science

December 2022

© Copyright by Ritwick Mishra 2022

This work is licensed under the Creative Commons Attribution 4.0 International License. To view a copy of this license, visit http://creativecommons.org/licenses/by/4.0/ or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

Acknowledgements

I would like to sincerely thank my advisors Professors Anil Vullikanti and Abhijin Adiga for their constant guidance and encouragement throughout my time working with them. It is only due to their steadfast support that I have been able to meet the myriad challenges involved in writing and defending this Thesis. I would like to thank my collaborators Gursharn Kaur and Jack Heavey for their valuable inputs and for our fruitful discussions.

I would like to thank Professors Jundong Li and Madhav Marathe for taking the time to be on my Thesis examining committee. I am privileged to be part of the Biocomplexity Institute here at UVA. I would also like to thank my professors at the department of Computer Science for creating an environment for learning and research.

I dedicate this Thesis to my mother, whose love and support is my foundation. This work would not be possible without the blessings of my grandparents.

Abstract

Only a subset of infections is actually observed in an outbreak, due to multiple reasons such as asymptomatic cases and under-reporting. Therefore, inferring the state of the epidemic, given some observed cases, is an important step in formulating our response. We consider two approaches for reconstructing a cascade and inferring epidemic properties.

First, we consider the problem of finding a maximum likelihood (MLE) solution to the cascade reconstruction problem for the Independent Cascade (IC) model (referred to as CASCADEMLE). This is a common approach in many inference problems, and can be shown to be a variation of the classical Steiner subgraph problem, which connects a subset of observed infections. In contrast to prior works on epidemic reconstruction, which consider the standard Steiner tree objective, we show that a solution to CASCADEMLE, based on the actual MLE objective, has a very different structure. We design a logarithmic approximation algorithm for CASCADEMLE, and evaluate it on multiple synthetic and social contact networks, including a contact network constructed for the University of Virginia (UVA) hospital. Our algorithm has significantly better performance compared to a prior baseline.

The MLE solution might not be very informative in many regimes, as our experiments show. In such settings, it is useful to consider the actual distribution of cascades, which are consistent with the observed infections. This can be shown to correspond to the problem of sampling Steiner trees which contain a set of terminal nodes, with a probability that depends on both the edges in the tree and the edges not in the tree. While there has been a lot of work on random sampling of spanning trees, random sampling of Steiner trees is open. A prior approach considers random generation of spanning trees with probability proportional to the product of probabilities of edges in the tree. We show that this gives a very different distribution on Steiner trees. We discuss partial progress in generating Steiner trees with the correct distribution.

Table of Contents

List of Figures

Chapter 1

Introduction

In most kinds of outbreaks, including COVID-19 [16], Hospital Associated Infections (HAIs), such as Methicillin-resistant *Staphylococcus aureus* (MRSA), plant and livestock diseases [12], it is important to have adequate knowledge of the spread of the disease to respond effectively. In most instances, only a subset of infections is known at any time, due to multiple reasons such as prevalence of asymptomatic cases [6], delays in discovery of pests and pathogens, etc. Such outbreak events require a prompt response, such as isolation of infected patients or quarantining infested farms. However, responding to such outbreaks is hampered by the lack of knowledge about the state of the epidemic. If we could reconstruct the cascade (or the who-infected-who subgraph corresponding to a diffusion outcome), for instance, by finding the most likely cascade or by sampling a probable cascade, we would be able to not only identify the undetected infections but also gain insight into how the infection spreads. This inferred knowledge could play a crucial role in both surveillance and control of the epidemic.

The problem of cascade reconstruction as well as the closely related problem of source detection has been studied extensively for several classes of network diffusion models on networks [14, 6, 20, 15]. These works assume partial information is available about the cascade, e.g., a subset of nodes which are known to be infected [6], or both infection state and time of infection [14, 20, 15]. In Rozenshtein et al. (2016), the input consists of a temporal interaction network, and a sample of nodes observed as infected. Their goal is to recover the flow of the spread, which would allow them to discover the sources, as well as the missing

infections. They use a directed Steiner tree based approach in which they minimize a cost depending only on the edges within the tree. Jang et al. (2021) argue for the consideration of node attributes in the problem of recovering the asymptomatic infections. They formulate this problem as a directed prize-collecting Steiner tree problem, in which the cost is still based only on the edges contained in the tree.

The related problem of sampling probable cascades from a distribution consistent with the observed infections can be especially important in cases where the Maximum Likelihood cascade is not very informative. Sampling cascades can also be used to find the probability of infection of nodes for which there is no state information [19]. This problem is less well-studied in comparison to the problem of finding the maximum likelihood cascade.

The simplest SIR type network model, also referred to as the independent cascade (IC) model [8] describes an epidemic process where an infection spreads on each edge e of a contact network $G = (V, E)$ with probability p_e. In this process, each infected node gets one chance to infect its susceptible neighbors, after which it is removed from the process. The probability of a cascade generated by such a process depends not only on the infections that succeeded but also on the attempts which failed. Thus we must take into account the edges that make up the cascade as well as the edges not in the cascade across which infection attempt failed.

Consider a cascade T_r rooted at root r on a network $G = (V, E)$ under this diffusion model. Let δ_{T_r} be the set of edges not in T_r with exactly one endpoint in T_r, i.e., $\delta_{T_r} = \{(u, v) \in E \setminus E(T_r) : u \in V(T_r), v \notin V(T_r)\}$. Let λ_{T_r} be the set of edges not in T_r with both endpoints in T_r, i.e. $\lambda_{T_r} = \{(u, v) \in E \setminus E(T_r) : u, v \in V(T_r)$. Let $d_{T_r}(r, u)$ denote the distance between root node r and u, in the subgraph T_r. Under the IC dynamics, the probability of the cascade subgraph T_r is:

$$\overline{P}(T_r) = \prod_{e \in E(T_r)} p_e \prod_{e \in \delta_{T_r}} (1 - p_e) \prod_{\substack{e=(u,v) \in \lambda_{T_r} \\ d_{T_r}(r,u) \neq d_{T_r}(r,v)}} (1 - p_e) \tag{1.1}$$

The first term corresponds to the contribution of edges in the subgraph. Since every infected node gets a single chance to infect a susceptible neighbor, we have two kinds of $(1 - p_e)$ terms contributed by edges not in T_r: (a) with exactly one endpoint in T_r, and (b) with both endpoints in T_r, which are at different distances from the root. Please see Figure 1.1 for an illustrative example.

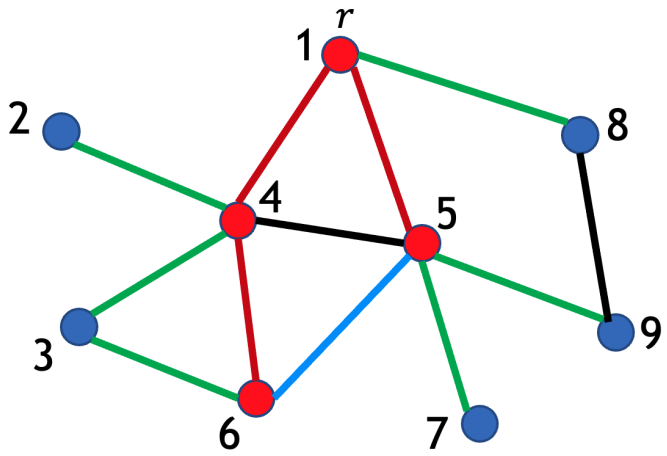

Figure 1.1: In this example, node 1 is the source (or root), the red nodes are the infections, and the red edges represent the infection cascade T_r. Here, $\delta_{T_r} = \{(2,4),(3,4),(1,8),(5,9),(5,7)\}$, and $\lambda_{T_r} = \{(4,5),(6,5)\}$. Each edge e in T_r contributes p_e to $\overline{P}(T_r)$, while each edge e in δ_{T_r} and λ_{T_r} contributes $(1-p_e)$, except for edge $(4,5)$. Neighbors 4 and 5 get infected at the same time, so they cannot attempt to infect each other.

Let $S \subseteq V$ be the subset of observed infections. The cascade reconstruction problem (referred to as CASCADEMLE) involves finding a connected subgraph T_r of G such that $S \subseteq V(T_r)$. Using the natural maximum likelihood estimation (MLE) approach, the goal of the CASCADEMLE problem is to find a T_r which maximizes the probability as given by Equation 1.1. Most prior work on reconstructing epidemic cascades has mainly been restricted to the regular Steiner tree objective, e.g., [6, 14], which corresponds to the the product of probabilities of edges only within T_r. This immediately connects with the vast literature on algorithms for the Steiner tree problem. We note that Zhu and Ying (2014) consider the actual cost from the source detection perspective, but mainly focus on trees for rigorous analysis (which is then extended to general graphs through various heuristics).

The MLE solution may not always be informative. In some regimes, the MLE cascade does not recover any part of the ground-truth cascade as we show in our study. In such settings, it is useful to consider the actual distribution of cascades, which are consistent with the given partial observations. Instead of finding the MLE cascade, we could sample from this distribution to gain information not provided by the MLE and make inferences about missing infections. This problem of sampling cascades was first studied

in Xiao et al. (2018). However they do not consider the actual probability distribution which consists of contributions not only from edges within the cascade but also from edges outside the cascade corresponding to *unsuccessful* infections, as specified by Equation 1.1.

1.1 Our Contributions

We study the problem of finding an MLE solution for reconstructing an epidemic cascade for the IC model, referred to as the CASCADEMLE problem, in Chapter 2.

- We show that a solution to CASCADEMLE can have very different structure from one found using the regular Steiner tree objective. In particular, there exist instances where the solution to CASCADEMLE has diameter $\Theta(n)$, where as the Steiner tree solution can have constant diameter. We observe a significant difference in real instances as well.
- We study the conditions under which the MLE based solution to the cascade reconstruction problem will fail. We find that the MLE based approach is not good when the graph is very dense.
- The CASCADEMLE hasn't been studied before. For the independent cascade model, we present an algorithm with a logarithmic approximation factor, under natural assumptions about the structure of social contact networks.
- Finally, we evaluate our formulation and algorithms for several synthetic and realistic contact networks, including a contact network for the University of Virginia (UVA) Hospital, constructed using Electronic Health Record (EHR) data. Our results show improved performance compared to a prior baseline in identifying missing infections.

In our next study, we consider the problem of sampling probable cascades spanning the observed infections, and present some partial results in Chapter 3.

- We show that the correct distribution of cascades, under IC dynamics, is very different from the distribution of Steiner trees as studied in the prior work [19].

- We observe that Propp and Wilson's algorithm of sampling random spanning trees, when adapted for Steiner trees, has a distribution which is very different to the one claimed in Xiao et al. (2018).
- We propose a sampling importance resampling method to correct for the bias in the distribution of the adapted Propp and Wilson's algorithm.

1.2 Related work

1.2.1 Cascade reconstruction and source detection

Several types of inference problems have been studied in the context of diffusion processes on networks. These works vary in terms of what is observed, knowledge of network and diffusion model class, inference objective etc. A popular inference problem is the source detection problem which is closely related to the cascade reconstruction problem. Shah and Zaman (2011) were the first to study the source detection problem. They consider the SI model, and assume that all the infections are given but the infection times are unknown, and the goal is to determine the source. They study the ML estimator for the source detection problem, and show that it can be solved exactly on trees using a notion of rumor centrality. Zhu and Ying (2014) extend this work to the SIR model and formulate the true ML estimator under this model. They recognize the intractability of this optimization problem and propose a simpler and more tractable version. In this sense, their formulation of the source detection problem is similar to our formulation of the CASCADEMLE problem. In their work, they assume that they know the network and all the infected nodes, and the goal is then to infer the source of the infection. There are numerous formulations and estimators proposed for the source detection problem over the past decade [7].

The problem of cascade reconstruction has been less well-studied in comparison. Rozenshtein et al. (2016) introduce a directed Steiner tree based algorithm, CULT, for reconstructing an epidemic cascade, when the underlying network is dynamic, and a subset of infections, along with their times, are given. They do not make any assumptions about the diffusion model and achieve the current state-of-the-art performance among Steiner tree-based cascade reconstruction frameworks. Jang et al.(2021) improve on this approach especially for HAIs, by formulating the missing infection detection problem as a directed prize-collecting

Steiner tree problem. They use this approach to detect asymptomatic cases of HAIs when the hospital mobility log is represented as a temporal network and a (hidden) disease model that starts independently from multiple sources. They argue that it is important to take into account individual risk factors in the form of node attributes in addition to the disease spread along the edges. They note that in the absence of node attributes, their method reduces to CuLT. Both these works consider a general cost structure, but only incorporate costs of edges in the subgraph; in contrast, we consider the true MLE cost.

Another approach to reconstruct cascades is by sampling from the set of epidemic cascades, instead of finding the MLE solution. Xiao et al.(2018) map this to the problem of sampling Steiner trees. However, they only consider the probability of edges within the tree, and ignore the contribution of edges on which transmission failed. We examine this work in detail as part of our research into sampling probable cascades.

1.2.2 Connections to Steiner tree problems

Steiner trees have found natural connection to the problem of cascade reconstruction. Given the vast amount of literature on Steiner tree problems, there is considerable value in formulating cascade reconstruction as a Steiner tree problem. Rozenshtein et al. (2016) adapt the best known approximation algorithm for the directed Steiner tree problem given by Charikar et al. (1999), into a more-efficient form by leveraging the specific structure of their problem. The Charikar algorithm is a greedy algorithm which recursively constructs ρ-level trees by joining subtrees based on minimizing the marginal normalized length. It has an approximation guarantee of $\rho(\rho-1)|S|^{1/\rho}$, where ρ is a depth of the recursion and S is the set of terminals. We can trade efficiency for quality by our choice of ρ. Rozenshtein et al. propose a approximation-preserving improvement to the efficiency of this algorithm by pre-computing all the shortest path distances for all pairs of nodes.

Jang et al. (2021) formulate the asymptomatic infection detection problem as a directed prize-collecting Steiner tree problem. By taking into account the special aspects of their problem setting, they reduce this to a directed Steiner tree problem, following which they also use the Charikar algorithm to approximately solve their problem.

In our work, we map the CascadeMLE to the node-weighted Steiner tree problem, under the independent cascade (IC) model. The first nearly best-possible approximation algorithm for the node-weighted

Steiner tree problem is given by Klein and Ravi (1995). This polynomial-time algorithm to approximate the minimum weighted Steiner tree has a logarithmic approximation factor of $2 \ln |S|$, where S is the set of terminals. They also note that due to its connection to the set cover problem, there is no polynomial-time algorithm that achieves a better approximation factor than logarithmic. Guha and Khuller (1999) improve the approximation factor slightly by generalizing some of the notions presented in Klein and Ravi [9].

1.2.3 Sampling trees

The problem of sampling trees and other combinatorial structures from graphs has been extensively researched in the community [2, 13]. Of relevance is the problem of sampling random spanning trees. Propp and Wilson (1998) present a loop-erased random walk (LERW)-based algorithm to generate random spanning anti-arborescences of a directed graph in accordance with the distribution given by the product of the weights on its edges. They analyze this LERW-based algorithm by showing that it is equivalent to a cycle-popping algorithm based on a deterministic view of the random walk called the stack model. Using this equivalence, they show that the output of the LERW algorithm is governed by the proper distribution. The problem of sampling Steiner trees is the main subject of Xiao et al. (2018), and we examine their proposed algorithms in detail as part of our research. Besides this, there seem to be no other works discussing this problem, as far as we know.

Chapter 2

Finding the MLE cascade

2.1 Preliminaries

2.1.1 Spread model

We consider the simplest form of the Susceptible-Infected-Recovered (SIR) model called the *Independent Cascade* model on a contact graph $G = (V, E)$. In this model, each node is in one of Susceptible (S), Infectious (I) or Recovered (R) state. We start off with all nodes in Susceptible state except the *source* nodes which are in Infected state. At each time-step, an infected node u can infect each susceptible neighbor v with probability $p_{u,v}$, independent of other neighbors of v. Each infected node gets only one opportunity to spread the infection following which they enter into Recovered state. Here, we state the likelihood and problem for the IC model.

2.1.2 Probability of a cascade

An outbreak, starting at a node r, is referred to as a *cascade*, and can be represented as a subgraph $T_r = (V(T_r), E(T_r))$, rooted at r. Let δ_{T_r} be the set of edges not in T_r with exactly one endpoint in T_r, i.e., $\delta_{T_r} = \{(u, v) \in E \setminus E(T_r) : u \in V(T_r), v \notin V(T_r)\}$. Let λ_{T_r} be the set of edges not in T_r with both endpoints in T_r, i.e. $\lambda_{T_r} = \{(u, v) \in E \setminus E(T_r) : u, v \in V(T_r)$. Under the IC dynamics, the probability of

the cascade T_r is:

$$\overline{P}(T_r) = \prod_{e \in E(T_r)} p_e \prod_{e \in \delta_{T_r}} (1 - p_e) \prod_{\substack{e=(u,v) \in \lambda_{T_r}, \\ d_{T_r}(r,u) \neq d_{T_r}(r,v)}} (1 - p_e) \tag{2.1}$$

where $d_{T_r}(r, u)$ denotes the distance between root node r and u, in the subgraph T_r. The first term corresponds to the contribution of edges in the subgraph. Since every infected node gets a single chance to infect a susceptible neighbor, we have two kinds of $(1 - p_e)$ terms contributed by edges not in T_r: (a) with exactly one endpoint in T_r, and (b) with both endpoints in T_r, which are at different distances from the root. Please see Figure 1.1 for an example.

2.1.3 MLE solution

We assume that subsets S_0, S_1 are given where S_0 is a set of nodes which are *known not to be infected* in the outbreak, while S_1 is a set of nodes *known to be infected*. We also assume the outbreak starts at a single node, which need not be in S_1. We say that a cascade T_r is *consistent* with (S_0, S_1) if $S_1 \subset V(T)$ and $S_0 \subset V - V(T)$. The MLE problem involves finding a connected subgraph $T_r = (V(T_r), E(T_r))$ rooted at a node r which is consistent with the given (S_0, S_1), and maximizes $\overline{P}(T_r)$; this is equivalent to the optimal sample path detection problem as described in [20]. Taking the log of the probabilities in $\overline{P}(T_r)$, we can define the cost of T_r as

$$\overline{\text{Cost}}(T_r) = \sum_{e \in E(T_r)} c_e + \sum_{e \in \delta_{T_r}} d_e + \sum_{\substack{e=(u,v) \in \lambda_{T_r}, \\ d_{T_r}(r,u) \neq d_{T_r}(r,v)}} d_e \tag{2.2}$$

Here, $c_e = -\log p_e$ is the cost of including an edge e in the subgraph and $d_e = -\log(1 - p_e)$ is the cost of excluding an edge e from the subgraph.

2.1.4 The CASCADEMLE problem

Given subsets S_0, S_1, the goal is to find a connected subgraph T_r rooted at some node r, which is consistent with (S_0, S_1), and minimizes $\overline{\text{Cost}}(T_r)$.

We say a solution T_r is an α-approximation if $\overline{\text{Cost}}(T_r) \leq \alpha \overline{\text{Cost}}(T^\star_{r\star})$, where $T^\star_{r\star}$ is an optimal solution to the instance of CASCADEMLE. Note that the root of T_r and $T^\star_{r\star}$ need not be the same; we only need that T_r be consistent with (S_0, S_1).

Remark: In practice, the costs of exclusion of the edges between same-level nodes in the cascade, $\sum_{(u,v)\in\lambda_T\ , d_T(r,u)=d_T(r,v)} d_{(u,v)}$, is a very small fraction of $\overline{\text{Cost}}(T_r)$ (as we verify in our experiments). This is also supported by the analysis of [1] that many realistic social and information networks are tree-like, where this condition will hold. In such a setting, $P(T)$ is a good approximation to $\overline{P_r}(T_r)$:

$$P(T) = \prod_{e\in E(T)} p_e \prod_{e\in\delta_T} (1-p_e) \prod_{e\in\lambda_T} (1-p_e). \tag{2.3}$$

Observe that $P(T)$ does not depend on the root. We consider the corresponding cost,

$$\text{Cost}(T) = \sum_{e\in E(T)} c_e + \sum_{e\in\delta_T} d_e + \sum_{e\in\lambda_T} d_e \tag{2.4}$$

and will focus on minimizing $\text{Cost}(T)$.

Our algorithm (described in section 2.3) considers the setting where we are given an undirected contact graph $G = (V, E, p)$ with edge infection probabilities p, and a set of observed infections S. The set of observed infected nodes S forms the terminal set in the output Steiner tree. Next we show that with a slight modification, this approach extends to the setting where we are also given the set of observed uninfected nodes. For a node u, let $\mathcal{N}_e(u)$ denote the set of edges incident on it.

2.2 Difference between CASCADEMLE and Steiner tree solutions

As mentioned earlier, previous works [14, 6] minimize the regular Steiner tree cost which consists of only the first term in $\overline{\text{Cost}}(T)$, namely $\text{Cost}_{st}(T) = \sum_{e\in E(T)} c_e$. Here we show that a solution which minimizes $\text{Cost}_{st}(T)$, can have a very different structure compared to the CASCADEMLE solution. We also observe that there exist instances in which the CASCADEMLE solution does not recover the true cascade.

Observation 2.2.1. *There exist instances in which a* CASCADEMLE *solution* $T = \arg\min_{T'} \overline{\mathrm{Cost}}(T')$ *has diameter* $\Theta(n)$*, while the Steiner tree solution* $T_{st} = \arg\min_{T'} \mathrm{Cost}_{st}(T')$ *has diameter* $\Theta(1)$*.*

Proof. Consider the class of graphs in Figure 2.1 where $A_1 \cup A_2$ form a complete bipartite graph with $|A_1| = |A_2| = N + 1$ and terminal node set $S = \{r, t\}$. Assume the homogeneous setting i.e. $c_e = c, d_e = d$ for every $e \in E$. Consider trees $T_1 = (r, w_1, w_1', t)$, and $T_2 = (r, u_1, \ldots, u_N, t)$. Observe that $\mathrm{Cost}_{st}(T_1) = 3c$ and $\mathrm{Cost}_{st}(T_2) = (N+1)c$. It can be verified that T_1 minimizes the $\mathrm{Cost}_{st}(\cdot)$ objective. On the other hand, we have $\overline{\mathrm{Cost}}(T_1) = 3c + 2Nd + 2d$ and $\overline{\mathrm{Cost}}(T_2) = (N+1)c + 2d$. It can be verified that there exists a sufficiently low value of p for which T_2 minimizes $\overline{\mathrm{Cost}}(\cdot)$, and thus, is the CASCADEMLE solution. Hence there exist regimes in which the CASCADEMLE solution has a diameter $\theta(n)$, while the Steiner tree solution has a diameter $\theta(1)$. □

Observation 2.2.2. *There exist instances in which a* CASCADEMLE *solution does not recover the true cascade.*

Proof. Consider the class of graphs in Figure 2.1. Suppose T_1 is the ground-truth cascade comprising nodes $\{r, t, w_1, w_1'\}$. Given terminal set $S = \{r, t\}$, the MLE approach will pick T_2 over T_1, unless the cost of excluding the bipartite edges is insignificant, i.e., p is small enough. Thus, there exist regimes in which the MLE solution fails to recover any part of the true cascade. □

2.3 Our approach

Assumption 1. *For every edge in the network,* $p_e \leq 1/2$ *or, equivalently,* $c(e) \geq d(e)$*, for all* $e \in E$*.*

Lemma 2.3.1. *Under assumption 1, a* CASCADEMLE *solution* $T^\star$ *is a tree.*

Assumption 1 states that the cost of including an edge is greater than than the cost of excluding it. This implies that an optimal subgraph is a tree, as we can always reduce the cost of the subgraph by excluding (rather than including) any edge that forms part of a cycle. Without this assumption, there exist instances where an optimal solution could have cycles. For example, in the homogeneous setting where all edges have the same costs c and d and $c < d$, an optimal solution could be one which spans the whole graph. Thus,

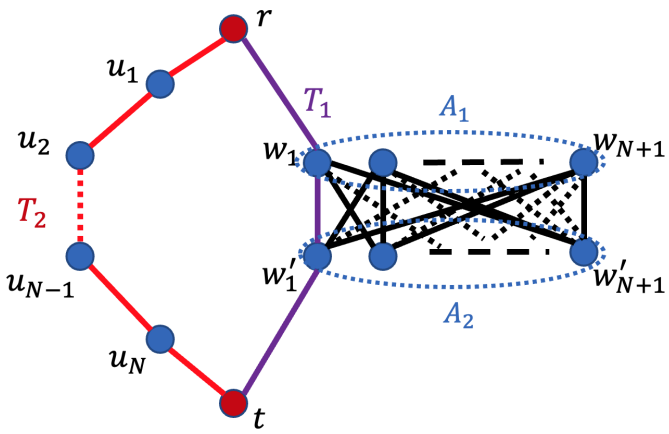

Figure 2.1: In this example, node r is the root, and $S = \{r, t\}$ is the set of terminals. $A_1 \cup A_2$ is a complete bipartite graph on nodes $w_1, \ldots, w_{N+1}$, and nodes $w'_1, \ldots, w'_{N+1}$. T_1 is the purple path between r, t through w_1, w'_1, while T_2 is the red path between r, t through nodes $u_1, .., u_N$.

under Assumption 1, we formulate the problem of finding an optimal subgraph, consistent with observed infected nodes S, as the problem of finding a minimum weighted Steiner tree with terminal nodes S on a node-weighted graph, in our algorithm MINCOSTSTEINERTREE.

In Algorithm 1, we weigh each node by the sum of the costs of exclusion of each of its incident edges, and each edge by the difference between its costs of inclusion and exclusion. Note that under Assumption 1, the edge weights would always be non-negative. Following the reduction to a purely node-weighted graph, this becomes a node weighted Steiner tree problem, where our goal is to find the minimum-weighted Steiner tree with terminal set S.

Algorithm 1 runs in polynomial time as we can construct the node-weighted graph in polynomial time and the Klein and Ravi (1995) algorithm has a polynomial time implementation. We now prove that this algorithm has a logarithmic approximation factor.

Theorem 2.3.2. *Let $\hat{T}$ be the tree returned by Algorithm 1, and let T^* be an optimal solution to the* CASCADEMLE *instance. Then $\hat{T}$ is consistent with S, and*

$$\text{Cost}(T^*) \leq \text{Cost}(\hat{T}) \leq 4 \ln |S| \cdot \text{Cost}(T^*) \tag{2.5}$$

Algorithm 1 MINCOSTSTEINERTREE

Input: An undirected contact graph $G = (V, E, p)$ and a set of observed infected nodes S
Output: Tree T_r consistent with S

for each edge e **do**
  Compute the cost of inclusion $c_e = -\log p_e$ and cost of exclusion $d_e = -\log(1 - p_e)$
end for
Construct a node and edge-weighted graph G' from G, by assigning weights as below:
for each node u **do**
  $w(u) \leftarrow \sum_{e \in \mathcal{N}_e(u)} d_e$
end for
for each edge e **do**
  $w(e) \leftarrow c_e - d_e$
end for
Convert G' to a purely node-weighted graph $\hat{G}$ by splitting each edge with a new node having the same weight.
Find the minimum weighted Steiner tree $\hat{T}$ in $\hat{G}$ with terminal set S, using Klein and Ravi's (1995) algorithm for the node-weighted Steiner tree problem.
Let r be any node in $\hat{T}$
return $\hat{T}$, with root r

Proof of Theorem 2.3.2. For any Steiner tree T,

$$\begin{aligned}
\sum_{u \in V(T)} w(u) + \sum_{e \in E(T)} w(e) &= \sum_{u \in V(T)} \sum_{e \in \mathcal{N}_e(u)} d_e + \sum_{e \in E(T)} (c_e - d_e) \\
&= \sum_{u \in V(T)} \sum_{e \in \mathcal{N}_e(u) \cap E(T)} d_e + \sum_{u \in V(T)} \sum_{e \in \mathcal{N}_e(u) \cap \lambda_T} d_e \\
&\quad + \sum_{u \in V(T)} \sum_{e \in \mathcal{N}_e(u) \cap \delta_T} d_e + \sum_{e \in E(T)} (c_e - d_e) \\
&= 2 \sum_{e \in E(T)} d_e + 2 \sum_{e \in \lambda_T} d_e + \sum_{e \in \delta_T} d_e + \sum_{e \in E(T)} (c_e - d_e) \\
&= \sum_{e \in E(T)} c_e + \sum_{e \in E(T)} d_e + 2 \sum_{e \in \lambda_T} d_e + \sum_{e \in \delta_T} d_e
\end{aligned}$$

$$= \text{Cost}(T) + \sum_{e \in E(T)} d_e + \sum_{e \in \lambda_T} d_e \tag{2.6}$$

$$\Rightarrow \sum_{u \in V(T)} w(u) + \sum_{e \in E(T)} w(e) \geq \text{Cost}(T) \tag{2.7}$$

Continuing from (2.6) and using Assumption 1,

$$\begin{aligned} \sum_{u \in V(T)} w(u) + \sum_{e \in E(T)} w(e) &\leq \text{Cost}(T) + \sum_{e \in E(T)} c_e + \sum_{e \in \lambda_T} d_e \\ &= 2\ \text{Cost}(T) - \sum_{e \in \delta_T} d_e \\ &\leq 2\ \text{Cost}(T) \end{aligned} \tag{2.8}$$

From 2.7 and 2.8, for any Steiner tree T, we have

$$\text{Cost}(T) \leq \sum_{u \in V(T)} w(u) + \sum_{e \in E(T)} w(e) \leq 2\ \text{Cost}(T) \tag{2.9}$$

This holds for $\hat{T}$, the Steiner tree returned by the Algorithm.

$$\text{Cost}(\hat{T}) \leq \sum_{u \in V(\hat{T})} w(u) + \sum_{e \in E(\hat{T})} w(e) \tag{2.10}$$

Klein and Ravi's algorithm has a worst-case approximation factor of $2 \ln |S|$. Hence,

$$\begin{aligned} \sum_{u \in V(\hat{T})} w(u) + \sum_{e \in E(\hat{T})} w(e) &\leq 2 \ln |S| \left(\sum_{u \in V(T^*)} w(u) + \sum_{e \in E(T^*)} w(e) \right) \\ &\leq 4 \ln |S|\ \text{Cost}(T^*) \end{aligned} \tag{2.11}$$

Combining (2.10) and (2.11),

$$\text{Cost}(\hat{T}) \leq 4 \ln |S|\ \text{Cost}(T^*) \tag{2.12}$$

□

2.3.1 Observed uninfected nodes

Our approach extends easily to the setting where the set of observed uninfected nodes S_0 is given in addition to the set of observed infected nodes S_1. Our goal is to find an optimal subgraph consistent with S_0, S_1. Note that we assume the observed uninfected nodes were never part of the cascade, i.e., they remained uninfected throughout the spreading process.

Let $\kappa = \{(u, v) \in E \setminus E(T) : u \in S_0, v \in T\}$ be the set of edges with one endpoint in S_0 and the other in the cascade T. In this setting, define δ_T the set of edges not in cascade with one endpoint in cascade and the other in set $V \setminus (S_0 \cup V(T))$, i.e., $\delta_T = \{(u, v) \in E \setminus E(T) : u \in V(T), v \in (V \setminus (S_0 \cup V(T))\}$. Here λ_T is the same as before, i.e, $\lambda_T = \{(u, v) \in E \setminus E(T) : u, v \in T\}$. Then the cost of a subgraph T, under IC dynamics, can be defined as:

$$\mathrm{Cost}(T) = \sum_{e \in E(T)} c_e + \sum_{e \in \lambda_T} d_e + \sum_{e \in \delta_T} d_e + \sum_{e \in \kappa} d_e \tag{2.13}$$

Our goal is to find a connected subgraph consistent with S_0, S_1, and which minimizes $\mathrm{Cost}(T)$. Here we present algorithm MINCOSTSTEINERTREE-OBS-UNINFECTED, which is only a slight modification of our previous algorithm: we remove the nodes known to be uninfected, after constructing the node and edge-weighted graph. This ensures that the returned tree is consistent with S_0.

Algorithm 2 has the same approximation ratio as the previous algorithm 1.

Theorem 2.3.3. *Let $\hat{T}$ be the tree returned by Algorithm 2, and let T^* be an optimal solution to the* CASCADEMLE *instance. Then $\hat{T}$ is consistent with S_0, S_1, and*

$$\mathrm{Cost}(T^*) \leq \mathrm{Cost}(\hat{T}) \leq 4 \ln |S_1| \cdot \mathrm{Cost}(T^*) \tag{2.14}$$

Algorithm 2 MINCOSTSTEINERTREE-OBS-UNINFECTED

Input: An undirected contact graph $G = (V, E, p)$, set of observed uninfected nodes S_0, a set of observed infected nodes S_1

Output: Tree T_r consistent with S_0, S_1.

for each edge e **do**
  Compute the cost of inclusion $c_e = -\log p_e$ and cost of exclusion $d_e = -\log(1 - p_e)$
end for
Construct a node and edge-weighted graph G' from G, by assigning weights as below:
for each node u **do**
  $w(u) \leftarrow \sum_{e \in \mathcal{N}_e(u)} d_e$
end for
for each edge e **do**
  $w(e) \leftarrow c_e - d_e$
end for
Remove all the nodes in S_0 (and their edges) from G'.
Convert G' to a purely node-weighted graph $\hat{G}$ by splitting each edge with a new node having the same weight.
Find the minimum weighted Steiner tree $\hat{T}$ in $\hat{G}$ with terminal set S_1, using Klein and Ravi's (1995) algorithm for the node-weighted Steiner tree problem.
Let r be any node in $\hat{T}$.
return $\hat{T}$, with root r.

Proof for Theorem 2.3.3. For any Steiner tree T,

$$\begin{aligned}
\sum_{u \in V(T)} w(u) + \sum_{e \in E(T)} w(e) &= \sum_{u \in V(T)} \sum_{\mathcal{N}_e(u)} d_e + \sum_{e \in E(T)} (c_e - d_e) \\
&= \sum_{u \in V(T)} \sum_{\mathcal{N}_e(u) \cap E(T)} d_e + \sum_{u \in V(T)} \sum_{\mathcal{N}_e(u) \cap \lambda_T} d_e + \sum_{u \in V(T)} \sum_{\mathcal{N}_e(u) \cap \delta_T} d_e \\
&\quad + \sum_{u \in V(T)} \sum_{\mathcal{N}_e(u) \cap \kappa} d_e + \sum_{e \in E(T)} (c_e - d_e)
\end{aligned}$$

$$= 2 \sum_{e \in E(T)} d_e + 2 \sum_{e \in \lambda_T} d_e + \sum_{e \in \delta_T} d_e + \sum_{e \in \kappa} d_e + \sum_{e \in E(T)} (c_e - d_e)$$

$$= \sum_{e \in E(T)} c_e + \sum_{e \in E(T)} d_e + 2 \sum_{e \in \lambda_T} d_e + \sum_{e \in \delta_T} d_e + \sum_{e \in \kappa} d_e$$

$$= \text{Cost}(T) + \sum_{e \in E(T)} d_e + \sum_{e \in \lambda_T} d_e \tag{2.15}$$

$$\Rightarrow \sum_{u \in V(T)} w(u) + \sum_{e \in E(T)} w(e) \geq \text{Cost}(T) \tag{2.16}$$

Continuing from 2.15 and using Assumption 1,

$$\sum_{u \in V(T)} w(u) + \sum_{e \in E(T)} w(e) \leq \text{Cost}(T) + \sum_{e \in E(T)} c_e + \sum_{e \in \lambda_T} d_e + \sum_{e \in \delta_T} d_e + \sum_{e \in \kappa} d_e - \sum_{e \in \delta_T} d_e - \sum_{e \in \kappa} d_e$$

$$= 2 \text{ Cost}(T) - \sum_{e \in \delta_T} d_e - \sum_{e \in \kappa} d_e$$

$$\leq 2 \text{ Cost}(T) \tag{2.17}$$

From (2.16) and (2.17), for any Steiner tree T, we have

$$\text{Cost}(T) \leq \sum_{u \in V(T)} w(u) + \sum_{e \in E(T)} w(e) \leq 2 \text{ Cost}(T)$$

This holds for $\hat{T}$, the Steiner tree returned by the algorithm 2.

$$\text{Cost}(\hat{T}) \leq \sum_{u \in V(\hat{T})} w(u) + \sum_{e \in E(\hat{T})} w(e) \tag{2.18}$$

Klein and Ravi's algorithm has a worst-case approximation factor of $2 \ln |S_1|$. Hence,

$$\sum_{u \in V(\hat{T})} w(u) + \sum_{e \in E(\hat{T})} w(e) \leq 2 \ln |S_1| \left(\sum_{u \in V(T^*)} w(u) + \sum_{e \in E(T^*)} w(e) \right)$$

$$\leq 4 \ln |S_1| \text{ Cost}(T^*) \tag{2.19}$$

Combining (2.18) and (2.19),

$$\mathrm{Cost}(\hat{T}) \leq 4 \ln |S_1| \, \mathrm{Cost}(T^*) \tag{2.20}$$

□

2.4 Experimental Results

2.4.1 Dataset and methods

We experimentally study the CASCADEMLE problem and evaluate the performance of MINCOSTSTEINERTREE algorithm on several real-world and synthetic networks. The networks are listed in Table 2.1 and described here.

1. arXiv High Energy Physics-Theory (HEP-TH): This is an academic collaboration network in the High Energy Physics-Theory community based on the citations in the arXiv preprints published between January 1993 and April 2004 [4, 10]. Taking the largest connected component, we generate a subgraph with $n = 500$ nodes, obtained by BFS starting from a random node. We refer to it as `arxiv`.

2. Erdős-Rényi random graphs: We generate several $G(n, q)$ graphs for evaluating the performance of our method and include results for $G(n = 300, q = 0.02)$.

3. Hospital ICU network: This is a contact network of patients and healthcare providers built using the Electronic Health Records (EHR) of the UVA Hospital's ICU between Jan 1, 2018 and Jan 8, 2018. We choose the largest connected component for our experiments and refer to it as `hospital-icu`.

4. Power-law networks: We generate power-law networks with $n = 1000$ nodes, varying the exponent γ in the range $[1.5, 3.5]$.

First, we study the error in approximation of the cost by comparing the true $\overline{\mathrm{Cost}}$ and our approximation Cost. The MINCOSTSTEINERTREE algorithm is evaluated with respect to network structure, diffusion

model parameters, and the observation set. We compare our method against CuLT [14] which is the state-of-the-art Steiner tree-based cascade reconstruction method. Since CuLT takes an additional time-of-report information while MinCostSteinerTree does not, we consider three variants of this method:

1. CuLT-Del: all nodes are reported as infected at the last time step,
2. CuLT-Rand: each node is reported as infected at a time step chosen randomly in between the time of infection and the last time step, and
3. CuLT-NoS: all nodes reported as infected at the time step of infection (NoS means No-Shift).

Note that these CuLT variants are ordered in the increasing amount of information provided to the algorithm. We use the homogeneous probability setting for our experiments where we set the diffusion probability p across all edges to be the same. We generate the infection cascades under IC dynamics for a single source chosen uniformly at random. In our experiments for evaluating the algorithm, we have considered cascade sizes to be within $(0.02n, 0.1n)$, where n is the network size so that sufficient number of observed nodes can be extracted from the cascade.

Next, to create the observation node sets from the generated infection cascades, we use two different schemes:

(a) `random`: We randomly sample a fixed % of nodes from the infected node set to form the observed node set.

(b) `frontier`: Here, nodes in the cascade at a distance at least d from the source are chosen as observed. This corresponds to the scenario in which we have observed the more recent infections and our goal is to infer the rest.

These schemes are inspired from Rozenshtein et al. (2016), and can help evaluate the performance of our method in two distinct observational settings.

We choose *Matthews correlation coefficient* (MCC) [11] and F1-score as in [14, 6], to evaluate the quality of the reconstructed cascades with the ground-truth. All reported values are averaged over 100 trials.

Graph Name	Nodes	Edges	Clustering coefficient	Average shortest path length
$G(n, q)$ random graph	300	897*	0.015	3.45
`arxiv`	500	895	0.52	12.5
`hospital-icu`	879	3575	0.59	4.31
Power-law networks	1000	660–6613	–	–

*Average value reported.

Table 2.1: Networks and their properties

2.4.2 Results

Difference between $\mathrm{Cost}(T)$ and $\overline{\mathrm{Cost}}(T)$

We created several power-law networks on 1000 nodes for various values of the power-law exponent γ in the range $[1.5, 3.5]$. We generated cascades starting from a source chosen uniformly at random, varying the probability p from 0.05 to 0.49. For each such cascade, we computed the error between the two costs. Representative results are in Figure 2.2 (the results are consistent across replicates of the networks). We observe that the difference in the costs depends on both the probability p as well as the network structure (which is decided by γ). We recall that the difference between the two costs is $d = -\log(1-p)$ times the number of node pairs in the cascade that satisfy the property that both nodes are at equal distance from the source. For very low values of p, the difference is low across networks as the cost of including an edge $c = -\log(p) \gg d$. As p increases, we observe that the network structure comes into play. For very low values of the power-law exponent γ, there are several nodes with high degree leading to the presence of dense subgraphs. This increases the chances of node pairs where the nodes at equal distance from the source of the cascade, and in turn, leads to a larger difference in the two costs. On the other hand, for lower γ (2.5–3.5), the graph is more tree-like, and therefore, we see a very low error even for probability approaching 0.5. For $\gamma = 2$ in particular, we note that the error is 10% for p as high as 0.3.

Performance of MINCOSTSTEINERTREE

In Figure 2.3, the MCC scores are plotted for MINCOSTSTEINERTREE and CULT for the two observation schemes and a diffusion probability of 0.1. We observe that MINCOSTSTEINERTREE performance is superior compared to CULT-DEL across observation schemes, networks and diffusion probabilities. We recall

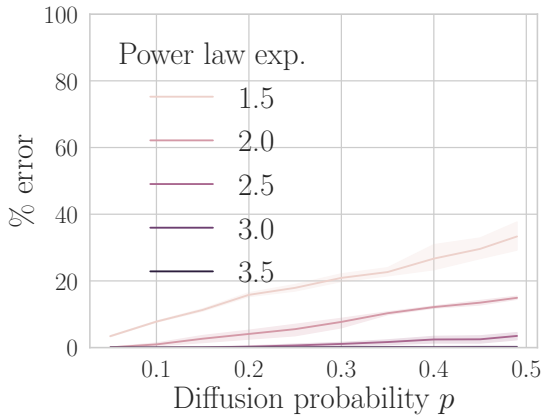

Figure 2.2: Percentage error between $\mathrm{Cost}(T)$ and $\overline{\mathrm{Cost}}(T)$ for cascades on random power-law graphs with varying diffusion probability p and power-law exponent.

that the MINCOSTSTEINERTREE algorithm accounts for the diffusion model while the versions of CULT do not. However, CULT-NOS and, to some extent CULT-RAND, account for the time of infection. In particular, for the $G(300, 0.02)$ graph and the `hospital-icu`, we observe that the performance of MINCOSTSTEINERTREE is much better than that of CULT. We note that `arxiv` has a large average shortest path length (and low diameter) compared to the other two networks even though its clustering coefficient is large. Even though `hospital-icu` has a large clustering coefficient, it has a small average shortest path length like $G(300, 0.02)$. In Figure 2.4, we have representative plots of the MCC and F1-scores under diffusion probabilities 0.05 and 0.20. The performance is similar to that in Figure 2.3 for $G(300, 0.02)$ and `hospital-icu`. For the higher probability, we observe inferior performance in the case of `arxiv` as the distance from source increases under the `frontier` observation scheme.

Impact of different types of observations

For the `random` observation scheme, we observe that MINCOSTSTEINERTREE performance drastically increases with increase in the number of observed nodes. This is particularly true for the real-world networks. Typically, we see good performance when at least 40% of the infected nodes are observed. In the case of FRONTIER observation scheme, we observe that when the distance from the source is ≥ 4, the cascades constructed are quite inferior. This puts emphasis on early discovery of the outbreak.

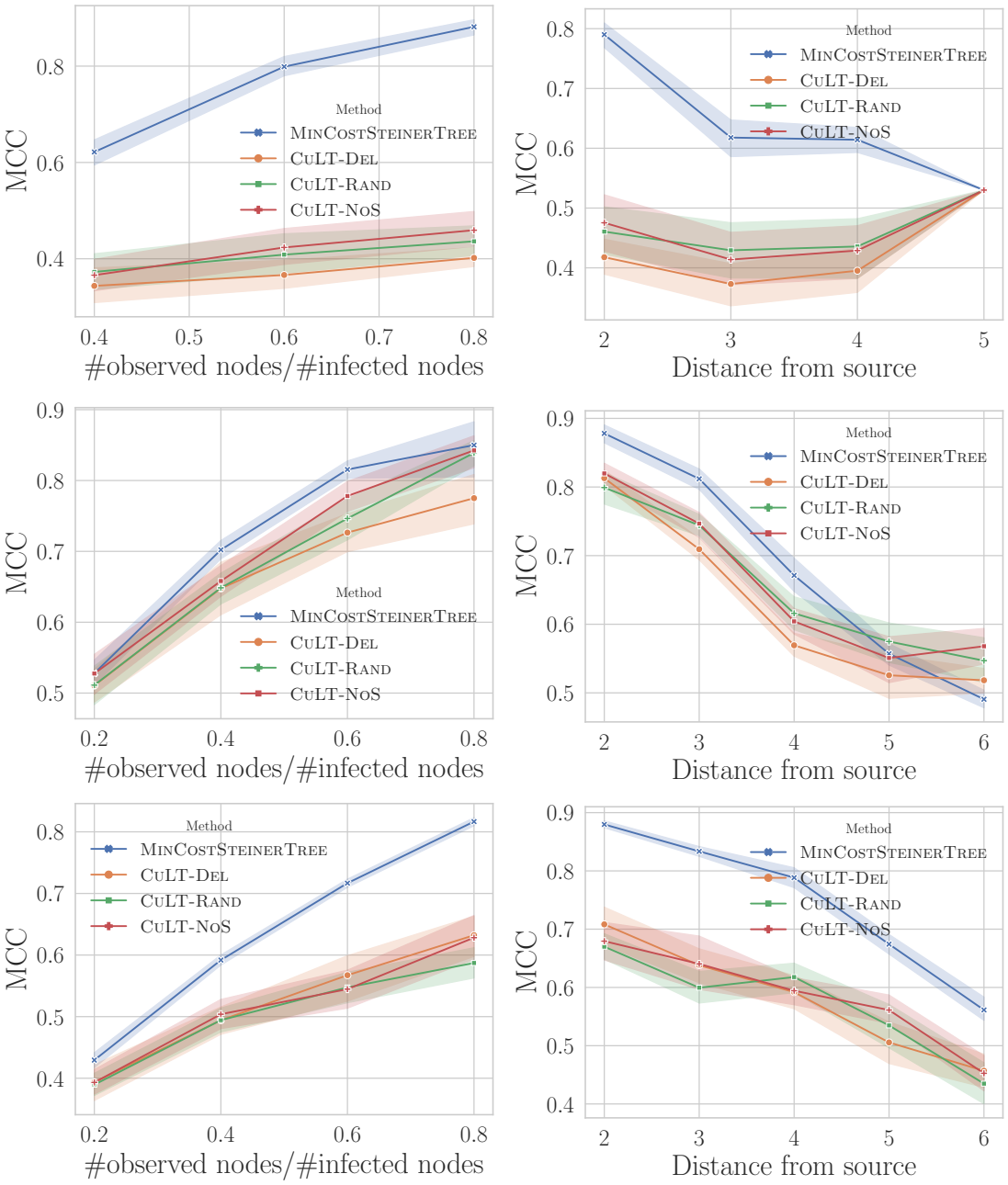

Figure 2.3: Performance of MinCostSteinerTree for `random` and `frontier` observation schemes. (a) $G(300, 0.02)$; (b) `arxiv`; and (c) `hospital-icu`, with fixed $p = 0.10$

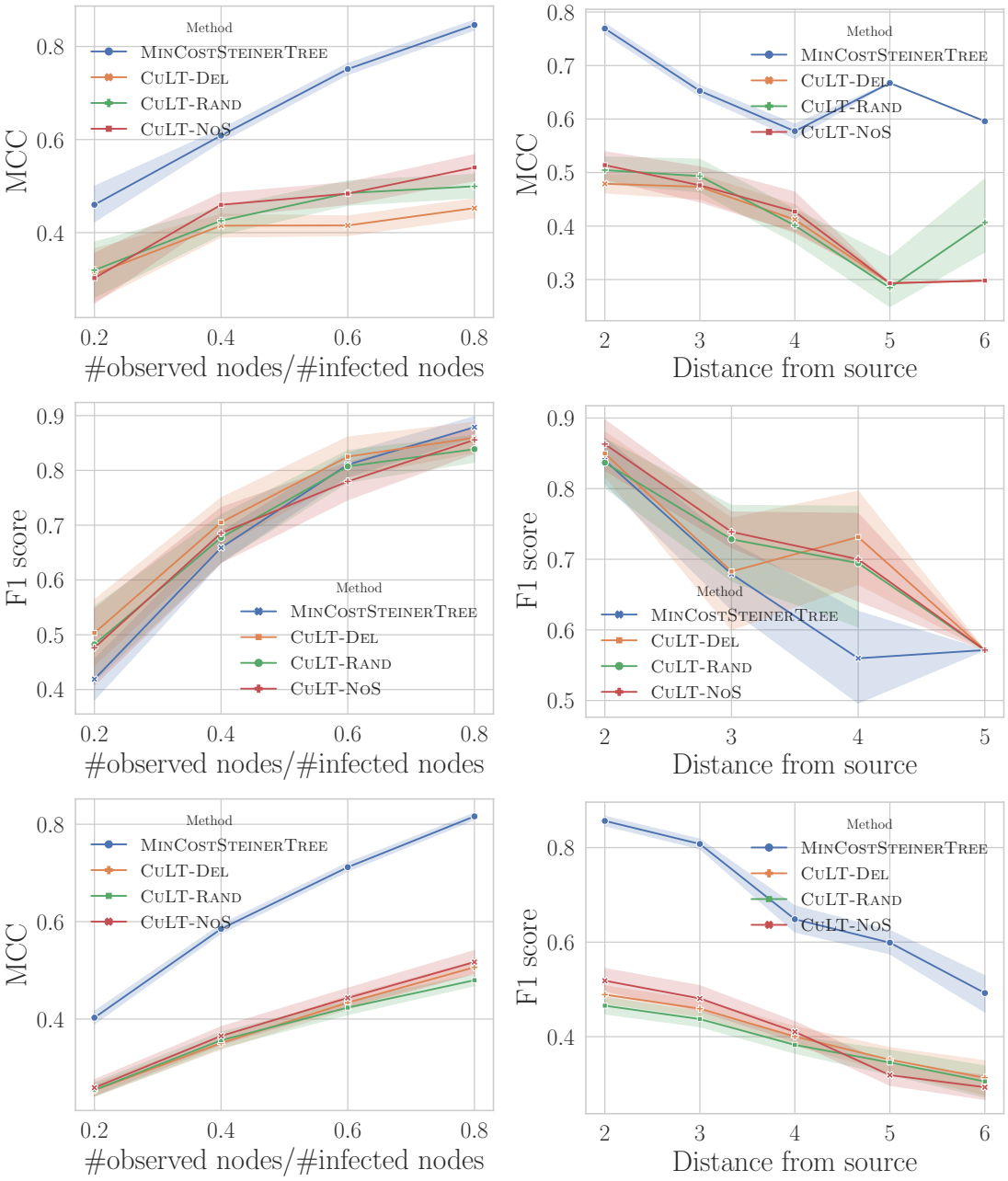

Figure 2.4: Performance of MinCostSteinerTree for (a) $G(300, 0.02)$ with $p = 0.05$; (b) `arxiv` with $p = 0.20$; and (c) `hospital-icu` with $p = 0.05$.

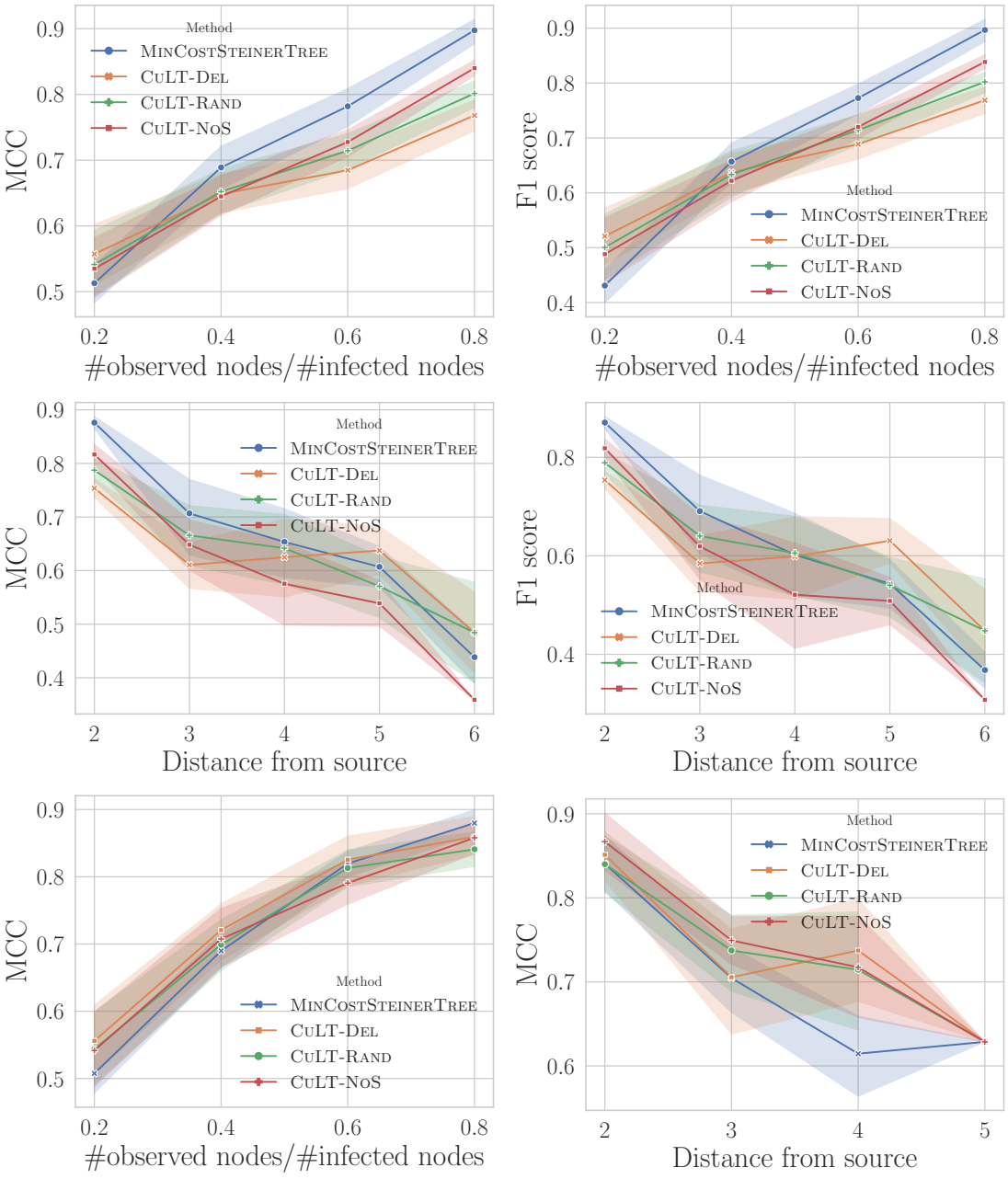

Figure 2.5: More results for performance of MinCostSteinerTree for `random` and `frontier` observation schemes, with 25 trials. Network: `arxiv` with transmission probability: sub-figures (1-4) 0.05, (5-6) 0.20.

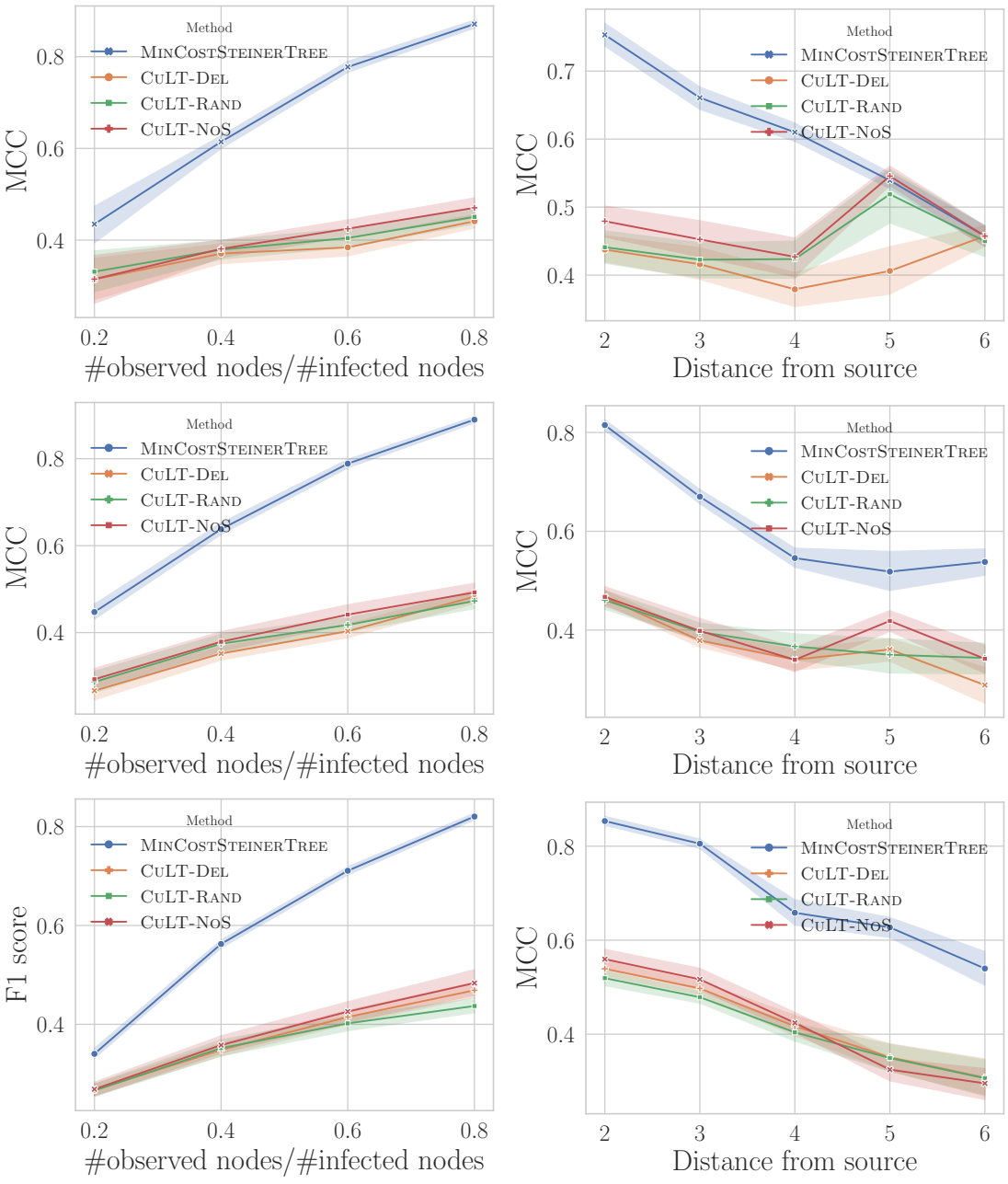

Figure 2.6: More results for performance of MinCostSteinerTree for `random` and `frontier` observation schemes, with 50 trials. (1-4) Averaged over 5 random $G(300, 0.02)$ networks: with diffusion probability $p =$ (a) 0.10, (b) 0.20; and (5-6) `hospital-icu` for $p = 0.05$.

Chapter 3

Sampling Cascades

In this chapter, we take a different view of the cascade reconstruction problem. Instead of finding the MLE cascade, we consider the actual distribution of cascades, which are consistent with the partial observations. Our goal now is to sample cascades from this distribution, according to their probability which is composed of contributions from edges corresponding to both successful and failed infection attempts. These generated cascades can be utilized for a variety of purposes. We can use them to estimate the probability of a node being infected or to identify edges which play an important role in the spread, which can inform our monitoring and control strategies.

As we saw in Section 2.2, the MLE cascade may not recover the ground-truth cascade in some situations. In such settings, it could be valuable to consider the actual distribution of cascades. We could sample cascades from this distribution to glean information about the possible pathways of infection spread and about missing infections, which we could not attain from the MLE solution alone.

A brute-force approach would be to run the simulator a number of times, reject instances that are not consistent with the partial observations, and obtain the required statistic from the remaining samples. However, this is not scalable as running simulations for complex models on large networks is an expensive process, and this rejection sampling might require a very large number of replicates. Thus the goal is to efficiently sample cascades consistent with the observations.

Xiao et al. (2018) map the problem of sampling probable cascades given observations to the problem of sampling Steiner trees given the terminals. Their goal is to sample probable cascades in order to do Monte-Carlo estimation of the probability of a node being infected, given the partial observations. However, there are two issues. Firstly, the distribution considered in their work is different from the actual distribution under the IC model. In their approach, they apply a loop-erased random walk-based (LERW) approach [13] for sampling Steiner trees. The second issue is that this approach samples Steiner trees according to a distribution different from the one intended. We show that their early-termination adaptation of Propp and Wilson's LERW algorithm samples Steiner trees according to a different distribution than has been claimed. We also outline some preliminary approaches to generate Steiner trees with the correct distribution.

3.1 Difference in the cascade distributions

Xiao et al. [19] consider the following problem:

Problem 3.1.1. *Given a weighted directed contact network $G = (V, E, p)$, a root node $r \in V$, and a set $S \subset V$ of terminal nodes, sample a directed Steiner tree T_S, rooted at r, spanning terminal nodes S, with probability proportional to*

$$\Pr(T_S) \propto \prod_{(u,v)\in T_S} p_{uv}$$

When they map the problem of sampling cascades to Problem 3.1.1, they make some implicit assumptions:

- The structure of the cascade is a directed tree.
- The probability of the cascade depends only on the probability of the transmission of infection across the edges contained in it.

We argue that these assumptions do not hold for many commonly-used disease models. Consider the Independent Cascades (IC) model. To formalize the problem of sampling cascades under IC dynamics, given observed infections, we can write the following:

Problem 3.1.2. *Given a weighted directed contact network $G = (V, E, p)$, a source node $r \in V$, and a set $S \subset V$ of observed infections, sample an IC cascade C_S, with source r, spanning observed infected nodes S, with probability proportional to*

$$\Pr(C_S) \propto \prod_{(u,v)\in C_S} p_{uv} \prod_{\substack{(u,v)\in G\setminus C_S \\ u\in V(C_S), v\in V(G\setminus C_S)}} (1-p_{uv}) \prod_{\substack{(u,v)\in G\setminus C_S \\ u,v\in V(C_S) \\ d_{C_S}(r,u)\neq d_{C_S}(r,v)}} (1-p_{uv})$$

where $d_{C_S}(r, u)$ is the distance of node u from source r in the cascade C_S.

The target distribution in Problem 3.1.2 consists of probability contributions from edges in the cascade (the p_{uv} terms) as well as from edges outside the cascade corresponding to failed infection attempts (the $1 - p_{uv}$ terms). We observe that the target distributions in the Problems 3.1.1 and 3.1.2 are quite different. Firstly, the IC cascade need not be a directed tree, as there can be more than one parent for a node in the IC cascade. Secondly, the probability of the sampled cascade must take into account not only the edges in the cascade but also the edges outside it across which the attempted infection must have failed, so as to result in the sampled cascade.

3.2 Sampling minimal Steiner trees

We call a Steiner tree, given a set of terminals S, *minimal* if all its leaves are terminals A Steiner tree is *minimal* in the way each edge is on at least one of the directed paths from the root to the terminals. Xiao et al. [19] propose an early-termination version of Propp and Wilson [13]'s algorithm (presented here as Algorithm 3) which we find only samples *minimal* Steiner trees, although they do not make this key observation. Moreover, we show that this algorithm does not generate minimal Steiner trees according to the target distribution in their problem (presented here as Problem 3.1.1).

3.2.1 Preliminaries

The random walk sampler in Propp and Wilson [13] generates in-directed trees, also called, anti-arborescences, where every edge points inwards towards the root. Even though we would like to sample directed trees with

edges pointing outwards from the root, i.e. arborescences, we can employ this random walk model by simply reversing the edges in the original graph and then undoing the reversal in the sampled anti-arborescence. We must also transform the given graph into a Markov chain in which every node's weighted out-degree is required to be 1. Using the transformation described in Xiao et al. [19], we set the edge-weights in the Markov chain $\tilde{G}$ as $w(u,v) = p_{vu}/p(u)$ where $p(u) = \sum_{v \in N_{in}(u)} p_{vu}$ i.e. the weighted in-degree of u in the original graph G. Propp and Wilson [13] 's random walk method samples a tree $\tilde{T}$ on Markov chain $\tilde{G}$, according to the weight of the tree given by $w(\tilde{T}) = \prod_{(u,v)\in T} w(u,v)$. This normalization introduces a bias:

$$\prod_{(u,v)\in T} p_{uv} = w(\tilde{T}) \prod_{u \in V[\tilde{T}]\setminus r} p(u) \tag{3.1}$$

Algorithm 3 EARLY-TERMINATION LERW [19]

Input: A Markov chain $\tilde{G} = (V, E, w)$, a root node r, and a terminal set S.
Output: A random minimal Steiner tree T_S

Let V be numbered from 1 to n.
Let InTree be an array to denote if node is in current tree.
Let Parent be an array to denote the parent of each node in current tree.
for i $\leftarrow 1...n$ **do**
    InTree[i] $\leftarrow false$
end for
Parent[r] $\leftarrow nil$
InTree[r] $\leftarrow true$
for $i \in S$ **do**
    $u \leftarrow i$
    while not InTree[u] **do**
        Parent[u] $\leftarrow$ RandomSuccessor(u)
        $u \leftarrow$ Parent[u]
    end while
    $u \leftarrow i$
    while not InTree[u] **do**
        InTree[u] $\leftarrow true$
        $u \leftarrow$ Parent[u]
    end while
end for
return Parent

Thus, after sampling the tree $\tilde{T}$ according to its weight $w(\tilde{T})$, we must resample with probability proportional to $\prod_{u \in V[\tilde{T}] \setminus r} p(u)$, using the *sampling importance resampling* method [17]. As the final step, we reverse the direction of the edges in $\tilde{T}$ to get the tree T on the original graph.

We denote the set of spanning trees of $\tilde{G}$, rooted at r, and containing T_S as a sub-tree, as $\mathcal{S}_{T_S}(\tilde{G})$. In order to analyze the distribution of the proposed algorithm, we describe a contraction scheme as introduced by Xiao et al. [19]. We denote the contracted graph of T_S on $\tilde{G}$ as $\tilde{G}_c[T_S]$. In this contraction, we merge all the nodes of T_S into a meta-node. All the now-parallel edges between T_S and rest of $\tilde{G}$ are also merged by summing up the weights of edges between the same pair of endpoints and introducing a label denoting the number of such parallel edges. Let $N_{out}(u)$ denote the set of outgoing edges from node u. We define the Laplacian matrix of $\tilde{G}$ as:

$$L_{ij}(\tilde{G}) = \begin{cases} \sum_{v_k \in N_{out}(v_i)} w(v_i, v_k) & \text{if } i = j \\ -w(i,j) & \text{if } i \neq j \text{ and } (v_i, v_j) \in E \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

Let $L_k(\tilde{G})$ denote the Laplacian matrix of $\tilde{G}$ with k-th row and k-th column removed. We can now proceed to the analysis.

3.2.2 Analysis of the early-termination LERW algorithm

We show here that the early-termination LERW algorithm (3) proposed by Xiao et al. [19] samples minimal Steiner trees with a different distribution than has been claimed.

Theorem 3.2.1. *The early-termination LERW algorithm (3) returns a random Steiner tree T_S rooted at r with probability proportional to $w(T_S).\det(L_r(\tilde{G}_c[T_S])$, where $w(T_S)$ is given by the product of the weights of its constituent edges and $\det(L_r(\tilde{G}_c[T_S])$ is the determinant of the Laplacian matrix of $\tilde{G}_C[T_S]$ with row and column r removed.*

Proof of Theorem 3.2.1. We will use the fact that the LERW of Xiao et al. [19] is simply an early-termination version of the original LERW algorithm of Propp and Wilson [13]. Wilson's original algorithm samples

random spanning trees with probability proportional to their weight given any arbitrary ordering of the vertices. Imagine an ordering of the vertices such that all the terminals are at the beginning (in any order), followed by the rest of the vertices (in any order). Note that if we run Wilson's algorithm with this ordering and stop after $|S|$ iterations, i.e., when all the terminals are part of the current tree, it would return a Steiner tree T_S. This is exactly what the LERW algorithm of [19] does. Now if we resume Wilson's algorithm and proceed till it ends, it would return a random spanning tree $\tau \in \mathcal{S}_{T_S}(\tilde{G})$, with probability proportional to its weight $w(\tau)$.

Claim 3.2.2. *The current tree in the algorithm is a Steiner tree T_S after the first $|S|$ iterations if and only if the eventual spanning tree τ returned by the algorithm belongs to the set $\mathcal{S}_{T_S}(\tilde{G})$.*

Proof of Claim 3.2.2. Every spanning tree corresponds to exactly one minimal Steiner tree. This means that if we get a spanning tree from $\mathcal{S}_{T_S}(\tilde{G})$ at the end, then we must have had the Steiner tree T_S at some point as the then-current tree. The current tree after $|S|$ iterations is the union of all loop-erased random-walks starting from one of the terminals to the root. This implies that all the leaves are terminals, which means that the current tree is a minimal Steiner tree, containing all the given terminals.

The converse is true because if, at some point, we have the Steiner tree as the current tree, then it can only grow from that point into a spanning tree from the set $\mathcal{S}_{T_S}(\tilde{G})$, as we never remove any edges in the LERW algorithm. □

Claim 3.2.2 implies that the event of getting the Steiner tree T_S after the first $|S|$ iterations is the union of the mutually exclusive events of getting each of the spanning trees in set $\mathcal{S}_{T_S}(\tilde{G})$, at the end of the algorithm. Let the event of getting Steiner tree T_S after $|S|$ iterations be denoted by $\mathcal{E}(T_S)$. Let the event of ending with the spanning tree τ be $\mathcal{E}_\tau$. Thus,

$$\mathcal{E}(T_S) = \bigcup_{\tau \in S_{T_S}(\tilde{G})} \mathcal{E}(\tau) \tag{3.3}$$

By the probability rule of sum,

$$\begin{aligned}
\Pr(T_S) &= \sum_{\tau \in \mathcal{S}_{T_S}(\tilde{G})} \Pr(\tau) \\
&\propto \sum_{\tau \in \mathcal{S}_{T_S}(\tilde{G})} w(\tau) \qquad (3.4) \\
&= \sum_{T_c \in \mathcal{S}(\tilde{G}_c[T_S])} w(T_S) w(T_c) \\
&= w(T_S) \sum_{T_c \in \mathcal{S}(\tilde{G}_c[T_S])} w(T_c) \\
&= w(T_S) \det(L_r(\tilde{G}_c[T_S]) \qquad (3.5)
\end{aligned}$$

□

In step 3.4, we use the fact that the Propp and Wilson [13] LERW algorithm generates random spanning trees proportional to the product of the weights of their constituent edges. In step 3.5, we use Tutte et al. (1984)'s Matrix Tree Theorem for spanning trees on weighted directed graphs, which states that the sum of the weights of all spanning trees rooted at k is the determinant of the k-th row and column removed Laplacian of the graph. These steps are taken from the analysis found in Xiao et al. [19] for the distribution of the other algorithm that they proposed called TRIM.

3.2.3 Our approach

In order to correct for the bias term in the early-termination LERW algorithm as given by Theorem 3.2.1, we can use the method of sampling importance resampling.

Algorithm 4 SAMPLEMINIMALSTEINER

Input: A Markov chain $\tilde{G} = (V, E, w)$, a root node r, and a terminal set S.
Output: A random Steiner tree T_S from the space of minimal Steiner trees rooted at r the probability of sampling which is proportional to $w(T_S) = \prod_{e \in T_S} w_e$

1: Sample a large number N of random Steiner trees using the early-termination LERW algorithm 3.
2: **for** each tree T_S **do**
3: Compute its weight $w(T_S) = \prod_{e \in T_S} w(e)$.
4: Compute $\det(L_r(\tilde{G}_c[T_S])$ by first contracting T_S on $\tilde{G}$, and then computing the determinant of the Laplacian after deleting the r-th row and column.
5: **end for**
6: To each tree $T_S^{(i)}$, assign sampling weights equal to

$$\frac{\sum_{j=1}^{N} w(T_S^{(j)}) \det(L_r(\tilde{G}_c[T_S^{(j)}])}{\det(L_r(\tilde{G}_c[T_S^{(i)}]) \sum_{j=1}^{N} w(T_S^{(j)})}$$

or, in other words, inversely proportional to $\det(L_r(\tilde{G}_c[T_S^{(i)}])$
7: Return a random minimal Steiner tree by resampling according to the assigned weights.

Theorem 3.2.3. *The algorithm* SAMPLEMINIMALSTEINER *returns a random Steiner tree T_S from the space of all minimal Steiner trees rooted at r with probability proportional to $w(T_S)$.*

Proof. The probability of a Steiner tree T_S returned by SAMPLEMINIMALSTEINER is given by:

$$\begin{aligned}
&\Pr(T_S \text{ returned by Algorithm 3})\Pr(T_S \text{ is resampled} \mid \text{the set of N Steiner trees}) \\
&= \frac{w(T_S) \det(L_r(\tilde{G}_c[T_S]))}{\sum_{j=1}^{N} w(T_S^{(j)}) \det(L_r(\tilde{G}_c[T_S^{(j)}])} \frac{\sum_{j=1}^{N} w(T_S^{(j)}) \det(L_r(\tilde{G}_c[T_S^{(j)}])}{\det(L_r(\tilde{G}_c[T_S]) \sum_{j=1}^{N} w(T_S^{(j)})} \\
&= \frac{w(T_S)}{\sum_{j=1}^{N} w(T_S^{(j)})} \\
&\propto w(T_S)
\end{aligned}$$

□

Note that we would have to do another round of sampling importance resampling in order to correct for the bias in the transformation back to the original contact graph, as given by equation 3.1.

3.3 Ongoing work

Having studied the problem of sampling minimal Steiner trees, we suggest that it is important to consider the distribution of all Steiner trees. This problem of sampling any Steiner tree, with a probability proportional to the product of its constituent edges, has not been studied in any previous work as far as we know. We have some initial ideas towards solving this problem.

We could first sample a minimal Steiner tree, and in the next step, randomly *grow* it to a bigger Steiner tree, which may no longer be minimal. This would make use of our previously presented approach, although we would have to take care of any bias terms this may introduce. One way to randomly grow a minimal Steiner tree could be to contract it to a meta-node and initiate a diffusion process from this meta-node. Another approach could be to iteratively choose new edges to include in the *growing* tree from the set of all edges originating from nodes in the tree and terminating at nodes outside it.

However, these approaches would need to be further modified when we consider our original problem of sampling from the actual distribution of cascades as stated in Problem 3.1.2. A naive but inefficient approach could be to run many simulations of the IC process from the source and accept only those which contain all the observed infected nodes. Thus, the problem is to find an efficient algorithm which would return cascades containing the observed infections and governed by the correct distribution.

Chapter 4

Summary and Future Work

In Chapter 2, we studied the problem of reconstructing an epidemic cascade given a subset of infections as observed nodes under IC dynamics. We presented an algorithm with a logarithmic approximation factor using a node-weighted Steiner tree approach, and evaluated its performance on several synthetic and real-world networks. An important future direction is to extend our approach to reconstruct cascades resulting from more complex SEIR processes with delayed recovery, SI, and the SIS models. Another direction is to incorporate additional information about the cascade such as reporting time or order of infections that can help overcome the limits of the MLE problem studied here. It might also be worthwhile to extend our methods to directed graphs and time-expanded graphs, as this would broaden their applicability.

Chapter 3 covers our study of the problem of sampling probable cascades, given observed infections. We showed that the prior approach leads to a very different distribution than had been claimed. We presented a sampling importance resampling method to correct for the bias in the distribution of the early-termination LERW algorithm. We also outlined some preliminary ideas in our ongoing work of sampling cascades according to the correct distribution.

References

[1] Aaron B. Adcock, Blair D. Sullivan, and Michael W. Mahoney. Tree-like structure in large social and information networks. In *2013 IEEE 13th International Conference on Data Mining*, pages 1–10, 2013. doi: 10.1109/ICDM.2013.77.

[2] Andrei Z. Broder. Generating random spanning trees. *30th Annual Symposium on Foundations of Computer Science*, pages 442–447, 1989.

[3] Moses Charikar, Chandra Chekuri, To yat Cheung, Zuo Dai, Ashish Goel, Sudipto Guha, and Ming Li. Approximation algorithms for directed steiner problems. *Journal of Algorithms*, 33(1):73–91, 1999. ISSN 0196-6774. doi: https://doi.org/10.1006/jagm.1999.1042. URL https://www.sciencedirect.com/science/article/pii/S0196677499910428.

[4] Johannes Gehrke, Paul H. Ginsparg, and Jon M. Kleinberg. Overview of the 2003 kdd cup. *SIGKDD Explor.*, 5:149–151, 2003.

[5] Sudipto Guha and Samir Khuller. Improved methods for approximating node weighted steiner trees and connected dominating sets. *Inf. Comput.*, 150:57–74, 1999.

[6] Hankyu Jang, Shreyas Pai, Bijaya Adhikari, and Sriram V. Pemmaraju. Risk-aware temporal cascade reconstruction to detect asymptomatic cases : For the cdc mind healthcare network. In *2021 IEEE International Conference on Data Mining (ICDM)*, pages 240–249, 2021. doi: 10.1109/ICDM51629.2021.00034.

[7] Rong Jin and Weili Wu. Schemes of propagation models and source estimators for rumor source detection in online social networks: A short survey of a decade of research. *Discrete Mathematics, Algorithms and Applications*, 13(04):2130002, 2021. doi: 10.1142/S1793830921300022. URL https://doi.org/10.1142/S1793830921300022eprint={https://doi.org/10.1142/S1793830921300022}.

[8] David Kempe, Jon Kleinberg, and Éva Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, page 137–146, New York, NY, USA, 2003. Association for Computing Machinery. ISBN 1581137370. doi: 10.1145/956750.956769. URL https://doi.org/10.1145/956750.956769.

[9] P. Klein and R. Ravi. A nearly best-possible approximation algorithm for node-weighted steiner trees. *Journal of Algorithms*, 19(1):104–115, 1995. ISSN 0196-6774. doi: https://doi.org/10.1006/jagm.1995.1029. URL https://www.sciencedirect.com/science/article/pii/S0196677485710292.

[10] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. http://snap.stanford.edu/data, 2014. Accessed: 2022-7-25.

[11] B.W. Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure*, 405(2):442–451, 1975. ISSN 0005-2795. doi: https://doi.org/10.1016/0005-2795(75)90109-9. URL https://www.sciencedirect.com/science/article/pii/0005279575901099.

[12] Joseph McNitt, Young Yun Chungbaek, Henning Mortveit, Madhav Marathe, Mateus R. Campos, Nicolas Desneux, Thierry Brévault, Rangaswamy Muniappan, and Abhijin Adiga. Assessing the multi-pathway threat from an invasive agricultural pest: Tuta absoluta in Asia. *Proceedings of the Royal Society B: Biological Sciences*, 286(1913):20191159, October 2019. ISSN 0962-8452. doi: 10.1098/rspb.2019.1159. Publisher: The Royal Society.

[13] James Gary Propp and David Bruce Wilson. How to Get a Perfectly Random Sample from a Generic Markov Chain and Generate a Random Spanning Tree of a Directed Graph. *Journal of Algorithms*, 27(2):170–217, May 1998. ISSN 01966774. doi: 10.1006/jagm.1997.0917. URL https://linkinghub.elsevier.com/retrieve/pii/S0196677497909172.

[14] Polina Rozenshtein, Aristides Gionis, B. Aditya Prakash, and Jilles Vreeken. Reconstructing an epidemic over time. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 1835–1844, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342322. doi: 10.1145/2939672.2939865. URL https://doi.org/10.1145/2939672.2939865.

[15] Devavrat Shah and Tauhid Zaman. Rumors in a network: Who's the culprit? *Information Theory, IEEE Transactions on*, 57:5163 – 5181, 09 2011. doi: 10.1109/TIT.2011.2158885.

[16] Jeffrey Shaman et al. An estimation of undetected covid cases in france. *Nature*, 590:38–39, 2020.

[17] Adrian F. M. Smith and Alan E. Gelfand. Bayesian statistics without tears: A sampling-resampling perspective. *Quality Engineering*, 37:645–648, 1992.

[18] W.T. Tutte, G.C. Rota, and C.S.J.A. Nash-Williams. *Graph Theory*. Encyclopedia of Mathematics and its Applications. Cambridge University Press, 1984. ISBN 9780521302418. URL https://books.google.com/books?id=9p_GjgEACAAJ.

[19] Han Xiao, Çigdem Aslay, and A. Gionis. Robust cascade reconstruction by steiner tree sampling. *2018 IEEE International Conference on Data Mining (ICDM)*, pages 637–646, 2018.

[20] Kai Zhu and Lei Ying. Information source detection in the sir model: A sample-path-based approach. *IEEE/ACM Transactions on Networking*, 24(1):408–421, 2014.