Inferring Probabilistic Contagion Models Over Networks Using Active Queries

Abhijin Adiga
Virginia Tech
abhijin@vt.edu

Vanessa Cedeno-Mieles*
Virginia Tech
vcedeno@vt.edu

Chris J. Kuhlman
Virginia Tech
ckuhlman@vt.edu

Madhav V. Marathe
Virginia Tech
mmarathe@vt.edu

S. S. Ravi†
Virginia Tech
ssravi0@gmail.com

Daniel J. Rosenkrantz
University at Albany – SUNY
drosenkrantz@gmail.com

Richard E. Stearns
University at Albany – SUNY
thestearns2@gmail.com

ABSTRACT

The problem of inferring unknown parameters of a networked social system is of considerable practical importance. We consider this problem for the independent cascade model using an active query framework. More specifically, given a network whose edge probabilities are unknown, the goal is to infer the probability value on each edge by querying the system. The optimization objective is to use as few queries as possible in carrying out the inference. We present approximation algorithms that provide provably good estimates of edge probabilities. We also present results from an experimental evaluation of our algorithms on several real-world networks.

CCS CONCEPTS

• **Computing methodologies** → *Machine learning algorithms*;

KEYWORDS

Independent cascade model; Active inference; Edge coloring; Approximation algorithms

ACM Reference Format:
Abhijin Adiga, Vanessa Cedeno-Mieles, Chris J. Kuhlman, Madhav V. Marathe, S. S. Ravi, Daniel J. Rosenkrantz, and Richard E. Stearns. 2018. Inferring Probabilistic Contagion Models Over Networks Using Active Queries. In *The 27th ACM International Conference on Information and Knowledge Management (CIKM '18), October 22–26, 2018, Torino, Italy*. ACM, New York, NY, USA, Article 4, 10 pages. https://doi.org/10.1145/3269206.3271790

*Also with Escuela Superior Politécnica del Litoral (ESPOL), Guayaquil, ECUADOR.
†Also with University at Albany – SUNY.

CIKM '18, October 22–26, 2018, Torino, Italy

ACM ISBN 978-1-4503-6014-2/18/10…$15.00
https://doi.org/10.1145/3269206.3271790

1 INTRODUCTION

Background and Motivation. Due to the tremendous increase in the use of networked social systems (e.g., Facebook, Twitter, LinkedIn), researchers are actively studying various aspects of such systems. In particular, the study of contagion propagation in networked systems is an active area of research in many disciplines (e.g., computer science, social science, business, economics) since contagions can be used to model many different phenomena including disease spread, propagation of influence and social trends and flow of information (see e.g., [10, 29]). Many classes of diffusion phenomena over networks have been studied in the literature when the network and the associated parameters are known (e.g., [9, 10]). In actual social systems, many parameters of the network (e.g., behavior characteristics of nodes, transmission probabilities associated with edges) are not generally known. To understand diffusion phenomena over such networks, and subsequently apply the understanding to forecast, maximize influence, control the spread, etc., it is essential to have good estimates of model parameters. For example, for systems where the node behaviors can be captured by appropriate threshold functions, techniques for inferring those functions from media or other observational data have appeared in [2, 13, 26].

In this paper, our focus is on obtaining provably good estimates of the edge (or influence) probabilities of a given directed social network. Researchers have studied this problem under a model where observational data about the dynamics of the system (e.g., log of user activities, a time-ordered trace specifying the set of nodes influenced at each time step) is available (see e.g., [14, 27]). We consider the problem under an *active* query model, where a query specifies state values for the nodes and the response to the query provides the state of each node at the next time step. This active query model is appropriate for networked systems that arise in the context of online social experiments carried out under controlled settings (see e.g., [6, 18, 22]). We develop a precise formulation of the edge probability inference problem under the *independent cascade* (IC) model of diffusion. This diffusion model, which was first considered in the context of interacting particle systems [9, 19], has been widely used in the study of influence and disease propagation (see e.g., [12, 15]). To make our algorithms scale to large social

networks (with millions of nodes and about 200 million edges), we also formulate a problem whose goal is to find an appropriate subgraph of a large network so that the inference algorithm can be applied to the smaller subgraph. This formulation exploits the fact that in several application contexts, edges of a social network are partitioned into classes such that edges within the same class have the same (transmission) probability.

Summary of Results. Our results, summarized below, include provably good approximations of edge probabilities as well as experimental evaluations using several real-world and synthetic networks.

(1) For the IC model, we develop a precise formulation of the edge probability inference problem for a directed network under the active query framework.

(2) Given a directed network and values ϵ and δ, where $0 < \epsilon, \delta < 1$, we present an (ϵ, δ)-approximation algorithm to infer the edge probabilities of G for the IC model. Formally, our algorithm ensures that for every edge e, the probability that the estimated probability $\hat{p}_e$ differs from the actual probability p_e by more than ϵp_e is at most δ. This approximation relies on two algorithmic ideas. First, it uses a stopping criterion for Monte Carlo sampling developed in [7]. Second, to minimize the number of queries used, it uses a novel edge coloring formulation (which we call fan-out edge coloring) for directed graphs.

(3) In practice, edge sets of large social networks are partitioned into classes such that all the edges in the same class have the same transmission probability. We rely on this idea to make our algorithms scale to very large social networks (with millions of nodes and hundreds of millions of edges). In particular, we formulate a combinatorial problem (called the **Minimum Cost Covering Subgraph** or MCCS problem) to identify a subgraph which has a small number of nodes and which contains at least one edge from each class. We show that this problem is **NP**-complete but present an approximation algorithm which provides a performance guarantee of $O(\sqrt{k})$, where k is the number of classes. This allows us to work with the smaller subgraphs generated by the approximation algorithm. It should be noted that our focus is on learning edge probabilities. We assume that a partition of the edges into subsets, where all edges in the same subset have the same probability, is available to our inference algorithms.

(4) We evaluate our algorithm for estimating edge probabilities on many real-world and synthetic networks. In a first set of experiments, we exploit edge labeling and use MCCS to reduce the sizes of large networks (in terms of numbers of nodes and edges) by several orders of magnitude, and infer edge probabilities. We evaluate a second set of intermediate sized-networks to address the case where there are no edge labels. Our assessments of these methods consider accuracy of the probability estimates, number of queries required to obtain these estimates and errors in contagion dynamics on networks under the IC model when using true and estimated probabilities.

Related Work. Many researchers have proposed formal models for contagion propagation in social networks (see e.g., [5, 10, 25]). This line of research generally assumes that all the parameters of the underlying network are known. Recently, there has been a considerable amount of interest in learning the parameters of networked systems. For example, for systems where state changes of nodes are determined by threshold values of nodes (i.e., a node changes to state 1 only when at least a specified number of its neighbors are in state 1), many papers have addressed the problem of learning the node thresholds (see e.g., [2, 13, 26]). The problem of learning influence probabilities in networks has also received attention in the literature. For example, Goyal et al. [14] study the problem assuming that a log of users' actions is available. They develop algorithms for learning the edge probabilities under a variety of influence models. Saito et al. [27] consider the problem of estimating the edge probabilities for the IC model of diffusion. They assume that data in the form of a system trace which gives for each time instant t, the set of nodes which changed to state 1 at t is available. They use an algorithm based on expectation maximization to obtain estimates of edge probabilities. Liu et al. [20] address the probability inference problem for heterogeneous networks. Our work differs from the previous work in that we use an *active* query model (explained in Section 2) instead of observational data. An active query model in a different context (namely, determining users' choices) has been studied recently in [16]. Our query model enables us to obtain provably good estimates of edge probabilities.

Organization. The remainder of this paper is organized as follows. In Section 2, we provide precise specifications of our active query model and the problem of inferring the edge probabilities under the IC model. In Section 3, we present our algorithm for the inference problem. To enable our algorithm to scale to very large social networks, we formulate the minimum covering subgraph problem, establish its complexity and present an efficient approximation algorithm for the problem in Section 4. We report results from our experiments (with and without finding a subgraph) in Section 5. Conclusions and directions for future work appear in Section 6. For space reasons, proofs of many results mentioned in the paper are omitted; they are available in [1].

2 MODEL DESCRIPTION AND PROBLEM FORMULATION

We assume that the underlying social network $G(V, E)$ is *directed*, with V and E denoting the vertex and edge sets respectively. An edge $e = (u, v)$, where u and v are the end points of e, is directed from u to v. In this case, u is an *in-neighbor* of v and v is an *out-neighbor* of u. For a node u, the indegree of u (denoted by $\deg_{\text{in}}(u)$) and outdegree of u (denoted by $\deg_{\text{out}}(u)$) are respectively the number of incoming and outgoing edges. The maximum indegree and outdegree of a graph are denoted by Δ_{in} and Δ_{out} respectively. Each directed edge is associated with a probability value; however, these probability values are not given and the goal is to obtain provably good estimates of those values.

We consider the independent cascade (IC) model of diffusion (defined below). We assume that every node has a state value from $\{0, 1\}$, where the state value 0 indicates that the node is "not infected" (or "not influenced") and 1 indicates that the node is "infected" (or "influenced"). In each time step and for each node v whose state value is 0, certain in-neighbors of v which are in state 1 try to influence v. This is a stochastic process whose nature is governed by the definition of the IC model (given below). A **configuration** of the network at time t is the binary tuple comprising

the state of every node in the network at time t. Given the configuration at time t, the diffusion process specified by the IC model determines a **successor** configuration at time $t+1$. As the system is stochastic, the successor of a configuration need not be unique.

Independent Cascade (IC) model. Let $G(V, E)$ be a directed network where every edge $e \in E$ is associated with a (transmission or influence) probability $p_e > 0$. As mentioned earlier, each node may be in state 1 (influenced/infected) or state 0. At time t, a node v in state 0 is influenced independently by each in-neighbor u which changed to state 1 at time $t-1$ with influence probability $p_{(u,v)}$. Subsequently, if the state of v changes to 1, then v influences its state 0 out-neighbors for exactly one time step, and never changes its state to 0.

Active query model. To estimate the edge probability values, our query model assumes an active form of interaction with the system. This type of query model has been studied in the literature in several contexts, including determining node behaviors [2] and inferring users' choices [16]. In our model, the user issues a query q that specifies a system configuration (i.e., the tuple of state values of all the nodes of the system) at a certain time instant; the response to the query is a successor configuration of q determined by the underlying stochastic process of the IC model. Formally, given a network G, the active query model corresponds to a *query function* $Q_G^{\mathrm{IC}}(\cdot)$, which takes a configuration or *query* q as input and returns a successor $Q_G^{\mathrm{IC}}(q)$ configuration of q as output. As the successor of a given configuration q is, in general, not unique, the system may return any successor of q that is consistent with the underlying stochastic process. Since generating responses to queries can be expensive, it is important to minimize the number of queries used. For a query q and a node v, we use $q(v)$ to denote the 0 or 1 value assigned to v by q. We can now present a precise definition of the problem of inferring edge probabilities under the IC model.

Problem 2.1 (InferIC). *Given a directed network $G(V, E)$ and a query function Q_G^{IC} corresponding to an IC model over G, infer the influence probability p_e for every edge $e \in E$ using a minimum number of queries.*

3 RESULTS FOR THE IC MODEL

Overview. We first propose an (ϵ, δ)-approximation algorithm for the InferIC problem. The algorithm gives the following guarantee: with probability at least $1-\delta$, the estimated influence probability $\hat{p}_e$ is at most ϵp_e away from the actual probability p_e for every edge e. Next, we discuss how our algorithm compares with an optimal solution with respect to the number of queries used. We then consider the special case of a *homogeneous* IC model where all edges have the same influence probability.

Our approach. Consider a directed edge $e = (u, v)$. If u is in state 1, then it may influence v with probability p_e. To estimate p_e, we a design query q in such a manner that $q(u) = 1$, $q(v) = 0$ and u is the only in-neighbor of v that is in state 1. Suppose $q' = Q_G^{\mathrm{IC}}(q)$; that is, q' is the successor of q returned by the system. Then, $q'(v)$ is a $0-1$ random variable with $\Pr(q'(v) = 1) = p_e$. When query q is repeated N times, we obtain in N independent samples of $q'(v)$ which can be used to estimate p_e. To determine a suitable value of N, we employ the stopping criterion discussed in [7].

To minimize the number of queries used, we utilize a specific edge coloring scheme which enables us to simultaneously estimate the influence probabilities of a group of edges. Suppose each edge in E is assigned a color from $\{1, 2, \ldots, \tau\}$ for some positive integer τ. (The coloring must satisfy certain conditions which will be discussed later in this section.) This coloring induces a partition $\{E_1, E_2, \ldots, E_\tau\}$ of E, where E_i is the subset of edges assigned color i, $1 \le i \le \tau$. Then, query q_i is constructed as follows: for every $(u, v) \in E_i$, set $q(u) = 1$ and the rest of the vertices of the graph to 0. We repeatedly try these τ queries until the stopping criterion is satisfied for each edge in that query, thus ensuring the accuracy of the estimate for every edge.

Stopping rule. Suppose $Z_1, Z_2, \ldots$ are i.i.d. random variables distributed according to Z in the interval $[0, 1]$ with mean p. Dagum et al. [7] proposed an (ϵ, δ)-approximation algorithm, which we refer to as StoppingRule (Algorithm 1), to estimate the value p with a near-optimal number of samples. Steps 2 and 9 of this Algorithm use the function $T(x, y)$ defined by

$$T(x, y) = 4(e-2)\log(2/y)/x^2. \tag{1}$$

The following theorem from [7] shows the approximation quality of the estimate and the number of samples used.

Algorithm 1: StoppingRule$(\epsilon, \delta, \{Z_1, Z_2, \ldots\})$ of [7]

Data: Values $\epsilon, \delta \in (0, 1)$ and i.i.d. random variables $Z_1, Z_2, \ldots$ according to Z in the interval $[0, 1]$ with mean p
Result: Estimate $\hat{p}$ of p

1 /*Step 1*/
2 Let $\epsilon' = \min(1/2, \sqrt{\epsilon})$, $T_1 = 1 + (1+\epsilon')T(\epsilon', \delta/3)$;
3 Initialize $N = 0, S = 0$;
4 **while** $S < T_1$ **do**
5 $\quad N = N + 1, S = S + Z_N$;
6 **end**
7 $p' = T_1/N$;
8 /*Step 2*/
9 Let $T_2 = 2(1+\sqrt{\epsilon})(1+2\sqrt{\epsilon})(1+\log(3/2)/\log(2/\delta))T(\epsilon, \delta)$;
10 Let $N' = T_2\epsilon/p'$ and $S = 0$;
11 **for** $i = 1, \ldots, N'$ **do**
12 $\quad S = S + (Z_{N+2i-1} - Z_{N+2i})^2/2$;
13 **end**
14 $\rho' = \max(S/N', \epsilon p')$;
15 /*Step 3*/
16 Let $N'' = T_2\rho'/p'^2$ and $S = 0$;
17 **for** $i = 1, \ldots, N''$ **do**
18 $\quad S = S + Z_{N+N'+i}$;
19 **end**
20 $\hat{p} = S/N''$;

Theorem 3.1. *Let Z be any random variable distributed in $[0, 1]$. Let $\mu_z = E[Z] > 0$ be the mean of Z, σ_Z^2 be the variance of Z and $\rho_Z = \max\{\sigma_Z^2, \epsilon\mu_Z\}$. Let $\tilde{\mu}_Z$ be the approximation produced by*

Algorithm 1 and let N_Z be the number of experiments run by the algorithm with respect to Z for input parameters ϵ and δ. Then (i) $\Pr[\mu_Z(1-\epsilon) \le \tilde{\mu}_Z \le \mu_Z(1+\epsilon)] \ge 1-\delta$ and (ii) there is a universal constant c' such that $E[N_Z] \le c'\, T(\epsilon,\delta)\, \rho_Z/\mu_Z^2$. □

From the above theorem, it can be seen that the estimate $\hat{p}$ produced by Algorithm 1 satisfies $\Pr[|\hat{p}-p| > \epsilon p] \le \delta$; further, the number of samples used is within a constant factor of the minimum number of samples required in expectation.

Fan-out edge coloring. The goal is to obtain a partition of the edge set E into $\{E_1, E_2, \ldots, E_\tau\}$ (for some τ), where E_i has all the edges with color i ($1 \le i \le \tau$). Further, each subset E_i in the partition must satisfy the following condition: for any $e = (u,v) \in E_i$, v does not have any other incoming edge with color i and v does not have any outgoing edge with color i. In such a coloring, any color class induces a graph which is a collection of stars with edges "fanning out" from a central vertex. Hence, we refer to this as a "fan-out edge coloring". Once we have such a coloring, the key idea is that one query is sufficient to estimate the probability for all the edges in the same color class. For each color class E_i ($1 \le i \le \tau$), such a query q_i can be constructed as follows. Let $Y_i \subseteq V$ be the subset of nodes such that for each node $u \in Y_i$, there is an outgoing edge $(u,v) \in E_i$. For each node $u \in Y_i$, we set $q_i(u) = 1$; for all other nodes $w \in V - Y_i$, we set $q(w) = 0$. An example of fan-out edge coloring and the construction of queries from the coloring are presented in Figure 1.

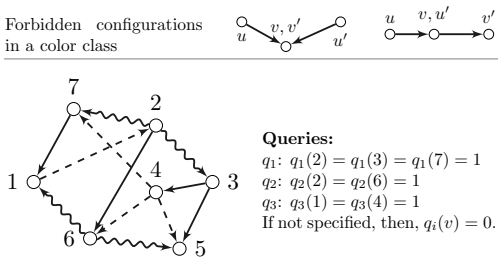

Figure 1: The constraints for fan-out edge coloring, an example of valid coloring and the resulting queries.

We now describe a method to realize a fanout edge coloring. We first construct an undirected graph $\mathcal{E}_G$ from G as follows. The vertex set of $\mathcal{E}_G$ is in one-to-one correspondence with E, the edge set of G. For two edges $e = (u,v)$ and $e' = (u',v')$ in E, their corresponding vertices in $\mathcal{E}_G$ are adjacent iff either (i) $v = v'$ or (ii) $v = u'$ (see Figure 1). Thus, in $\mathcal{E}_G$, two nodes are adjacent iff the corresponding edges in E cannot be assigned the same color. Hence, any proper vertex coloring of $\mathcal{E}_G$ corresponds to an edge coloring of G that satisfies the conditions for a valid fan-out edge coloring mentioned above. We note that $\mathcal{E}_G$ is a variant of the well-known *line graph* defined for an undirected graph. Further, a simple greedy coloring strategy based on Brooks's theorem [31] can be used to color $\mathcal{E}_G$ with at most $\Delta(\mathcal{E}_G)+1$ colors, where $\Delta(\mathcal{E}_G) = \max_{e=(u,v)\in E} \deg_{\text{in}}(u) + \deg_{\text{in}}(v) + \deg_{\text{out}}(v) - 1$.

Our algorithm ApproxInferIC, which uses Algorithm 1, is described in Algorithm 2. The algorithm first constructs a fan-out edge coloring of E as discussed above. It then constructs a query q_i for each color class i and repeatedly obtains a successor of q_i. The number of repetitions is determined using Algorithm 1. We can now show that Algorithm 2 indeed provides a provably good estimate of the true probability for each edge.

Algorithm 2: ApproxInferIC($G, Q_G^{\text{IC}}, \epsilon, \delta$)

Data: Directed graph $G(V,E)$, query function Q_G^{IC} which returns a successor of the input query q and values $\epsilon, \delta \in (0,1)$.
Result: For every $e \in E$, an estimate $\hat{p}_e$ of the influence probability p_e

```
Construct a fan-out edge coloring {E_1, ..., E_τ} of E;
for i = 1 to τ do
    /*Construct query q_i*/
    q_i = 0; /*first set every vertex state to 0 in q_i*/
    for e = (u, v) ∈ E_i do
        q_i(u) = 1;
    end
    /*Query until stopping criterion is satisfied*/
    Initialize N = 0, ∀e ∈ E_i, S_e = ∅;
    while ∀e ∈ E_i, StoppingRule(ε, δ/|E|, S_e) has not terminated do
        q'_N = Q_G^IC(q_i);
        ∀e = (u, v) ∈ E_i, S_e = S_e ∪ q'_N(v);
        N = N + 1;
    end
end
∀e ∈ E, p̂_e is the output of StoppingRule(ε, δ/|E|, S_e).
```

Theorem 3.2. *Let $G(V,E)$ be a directed graph, $\mathcal{E}$ be a fan-out edge coloring of G with τ colors and $\epsilon, \delta \in (0,1)$. For any independent cascade model defined over G, under the active query model, ApproxInferIC estimates the influence probabilities with the following guarantees: (i) $\Pr\left(\forall e \in E, |\hat{p}_e - p_e| < \epsilon p_e\right) \ge 1-\delta$, where for an edge e, p_e and $\hat{p}_e$ are the actual and estimated influence probabilities respectively. (ii) The expected number of queries is at most $c\tau T(\epsilon, \delta/|E|)\rho/p_{\min}$ where c is a positive constant, $p_{\min} = \min_{e\in E}\{p_e\}$ and $\rho = \max(p_{\min}, \epsilon)$.*

Proof. We recall that each $p_e > 0$. First, we show that the algorithm satisfies the following condition for each edge e.

$$\Pr(|\hat{p}_e - p_e| > \epsilon p_e) < \delta/|E|\,. \tag{2}$$

Let $e = (u,v)$, and let $e \in E_i$. By definition, $q_i(u) = 1$ and $q_i(v) = 0$. Let $q' = Q_G^{\text{IC}}(q_i)$. By the definition of the fan-out coloring, e is the only incoming edge of v in E_i, and therefore, u is the only neighbor of v whose state is 1 in q_i. Hence, $q'(v) = 1$ with probability p_e and 0 with probability $1-p_e$. Note that the repetitions of q_i are independent of each other. Therefore, by Theorem 3.1, the estimate of StoppingRule($\epsilon, \delta/|E|, S_e$) satisfies Equation (2). We now use

the union bound [23] to prove Part (i) of Theorem 3.2.

$$\Pr\left(\exists e \in E, |\hat{p}_e - p_e| > \epsilon p_e\right) \leq \sum_{e \in E} \Pr\left(|\hat{p}_e - p_e| > \epsilon p_e\right) \leq \sum_{e \in E} \delta/|E| = \delta\,.$$

Now we prove Part (ii) of the theorem. From Part (ii) of Theorem 3.1, for a random variable Z taking values in $[0, 1]$, with mean p and variance σ^2, the expected number of queries required by the StoppingRule for (ϵ', δ') is at most $f(p, \epsilon', \delta') = cT(\epsilon', \delta') \max(\sigma^2, \epsilon p)/p^2$. Since, in our case, Z is a $0-1$ random variable, it follows that $\sigma^2 = p(1-p)$. Also, it can be verified that when $p_1 > p_2$, for any ϵ and δ satisfying $0 < \epsilon, \delta \leq 1$, $f(p_1, \epsilon, \delta) < f(p_2, \epsilon, \delta)$; that is, the edge that requires the most number of repetitions is the one with the minimum influence probability $p_{\min}$. Therefore, the number of times a query must be repeated is at most $cT(\epsilon, \delta/|E|)\rho/p_{\min}$ in expectation, for some constant $c > 0$. Since there are at most τ distinct queries, the expected total number of queries used is at most $c\tau T(\epsilon, \delta/|E|)\rho/p_{\min}$. □

Bounds on the optimal number of queries. Dagum et al. [7] show that for a single random variable, the expected number of samples used by the stopping rule algorithm (Algorithm 1) is within a constant factor of the optimum expected value. For the InferIC problem, there are two factors that influence the number of queries required: (i) network structure which influences the total number of distinct configurations that can be used for querying, and (ii) the distribution of edge probabilities that determines how many times each query is repeated. The interplay between these two factors makes it challenging to obtain good bounds on the number of queries. Theorem 3.2 gives an upper bound for the optimal number of queries.

For the lower bound, under the assumption that an algorithm must infer the probability of each edge independently, the number of distinct queries required is at least Δ_{in}, the maximum in-degree of the network. This is because, a vertex with in-degree = Δ_{in} has that many incoming edges which must be evaluated in distinct queries. Suppose $p_{\max}$ is the maximum edge probability in the IC model. Then, every query must be repeated at least $c\,\Delta_{\text{in}} T(\epsilon, \delta/|E|)\rho/p_{\max}$ times for some constant c, where T is the function defined in Equation (1).

Uniform probability. When all the edges have the same probability p, the number of queries can be substantially reduced. Firstly, we need not consider all the color classes. It is enough to just use a color class with the maximum number of edges. Secondly, in every query, we obtain s samples of the same random variable, where s is the number of edges in the corresponding color class. Therefore, using a proof similar to that of Theorem 3.2, the expected number of queries used can be seen to be at most $\frac{cT(\epsilon, \delta/|E|)\rho}{p|E_{\max}|}$, where $E_{\max}$ is a color class with the maximum number of edges. If the number of colors is τ, note that $|E_{\max}| \geq |E|/\tau$. Hence, the number of queries is at most $\frac{c\tau T(\epsilon, \delta/|E|)\rho}{p|E|}$, which is $1/|E|$ times the upper bound for the non-uniform probabilities case.

4 MINIMUM COVERING SUBGRAPH PROBLEM

4.1 Motivation for Considering Subgraphs

While the algorithm discussed in the previous section provides a good performance guarantee, it is difficult to use it directly with very large social networks with several million nodes and about 200 million edges. The reason is that with a large number of edges and the number of times a query must be repeated (for each edge) to obtain the desired performance guarantees (in terms of the parameters ϵ and δ), the required number of queries to be tried is prohibitively large. (One can get an idea of this from the experimental results in Section 5.3 for smaller graphs for which the number of edges varies from about 84,000 to about 940,000.) To ensure scalability of our algorithm, we exploit a feature of social networks that is commonly present in practical epidemic and social simulations. In these simulations, the edges of the social network are partitioned into a certain number of classes, and all the edges within the same class have the same (transmission) probability value. Such partitioning schemes rely on the fact that each edge in the social network represents a type of interaction among the two individuals corresponding to the end points of the edge, and each interaction type can be modeled by a single probability value. A good discussion on why interactions and degrees of influence can be grouped and modeled in this manner appears in [8, 24, 28]. The interaction types are generally based on node and edge attributes (e.g., ages of the individuals, the duration of interactions), and the number of types of interactions is much less than the number of edges in the network. Thus, instead of considering a very large social network, one can consider a subgraph which has at least one edge for each type of interaction. The inference algorithm under the IC model from the previous section can be applied on the subgraph and the results can be translated to the larger social network. In Section 5, we describe a scheme that we used in our experiments to partition the edge set into classes. In the next subsection, we provide a precise formulation of the subgraph problem, establish its complexity and present a provably good approximation algorithm for the problem.

4.2 Problem Definition and Results

As mentioned above, we want to find a subgraph G' of a given social network G so that G' contains all the edge types that are in the larger network. To make the problem formulation and analysis easy to understand, we will define this problem for undirected graphs. (The directions of the edges don't play a role in deciding which edges are chosen.) In our experiments, however, we used directed graphs. A general version this problem can be formulated as follows.

Minimum Cost Covering Subgraph (MCCS)
Instance: An undirected graph $G(V, E)$, a nonnegative cost $c(v)$ for each node $v \in V$, a partition of the edge set E into k classes $E_1, E_2, \ldots, E_k$ and a budget $B \leq \sum_{v \in V} c(v)$.
Question: Is there a subset $V' \subseteq V$ such that the total cost of the nodes in V' is at most B and the subgraph $G'(V', E')$ induced on V' contains at least one edge from each class?

The following result points out that MCCS is unlikely to be solvable efficiently.

Theorem 4.1. *MCCS is **NP**-complete even when the underlying graph is bipartite and the cost of each node is 1.*

Proof. (Idea) We prove the **NP**-hardness through a reduction from Minimum Set Cover (MSC) problem which is known to be **NP**-complete [11]. The details appear in [1]. □

Theorem 4.1 shows that the MCCS problem is **NP**-hard even when all the nodes have a cost of 1; that is, the cost of the subgraph G' is the number of nodes in G'. For this version, we now present an approximation algorithm which provides a performance guarantee of $O(\sqrt{k})$, where k is the number of edge classes. In other words, the number of nodes in the subgraph chosen by Approx-MCCS is always within the factor $O(\sqrt{k})$ of the number of nodes in an optimal subgraph. The details of this approximation algorithm, which we refer to as Approx-MCCS, appear in Figure 3. The following theorem, whose proof appears in [1], shows the worst-case performance guarantee provided by Approx-MCCS.

Algorithm 3: Approximation Algorithm for MCCS

Input : An undirected graph $G(V, E)$, a partition of the edge set E into k classes $E_1, E_2, \ldots, E_k$.

Output : A subgraph $G'(V', E')$ of G such that E' contains at least one edge from each class E_i $(1 \leq i \leq k)$ and the number of nodes in V' is as small as possible.

1 Let $C = V' = \emptyset$. (Note: C is the set of classes of edges from which at least one edge has been chosen and V' is the subset of nodes in the resulting subgraph G'.)
2 **while** $(|C| < k)$ **do**
3 Find a node v such that among all the nodes in V, the set of edges incident on v has the largest number of new classes which are not in C.
4 For each new class of edges, choose one edge from that class incident on v and add it to E'.
5 Update C by adding the new classes of edges chosen in Step 4.
6 Add v and the other end points of the edges chosen in Step 4 to V'.
7 **end**
8 Output the subgraph G' of G induced on V'.

Theorem 4.2. *For any instance of MCCS given by a graph $G(V, E)$ where the number of classes into which the edge set E has partitioned is k and the cost of each node is 1, Algorithm Approx-MCCS provides a performance guarantee of $O(\sqrt{k})$.* □

5 EXPERIMENTAL RESULTS

5.1 Overview

Set 1 experiments. We report on two sets of experiments. In the first set, we start with larger synthetic populations, and generate labeled *directed* networks from them with 10^5 to 10^6 nodes and 10^6 to 10^8 edges (see Table 1). We then specify an edge labeling process and use Algorithm 3 to produce much smaller subgraphs that preserve the edge labels from the original graphs. These operations are summarized in the top row of Figure 2, and will be explained below. Each edge label corresponds to an edge probability. We use Algorithm 2 to compute fanout edge colorings and estimated edge probabilities on these subgraphs, and map the results back to the larger graphs. These operations are summarized in the second and third rows of Figure 2, and will be explained below. This entire process demonstrates how to take large labeled networks, generate smaller networks, analyze them, and map the results back to the original large network. That is, this process demonstrates a *scalable* approach to inferring edge probabilities under the IC model.

Table 1: Networks used in our experiments and their properties. Set 1 networks are in rows 1 through 3. Set 2 networks are in rows 4 through 8. To conserve space, we have provided ranges of values for some network families in Set 2. Fan-out coloring is not applicable (N/A) to Set 1 networks because we perform the coloring on subgraphs.

Network	Properties			
	n	$\|E\|$	Δ_{in}, Δ_{out}	Fanout(τ)
NRV	152,661	8,301,322	772	N/A
Miami	2,165,398	108,618,252	846	N/A
Seattle	3,405,279	197,374,320	892	N/A
Enron	33,696	361,622	1,383	1,393
Epinions	75,877	811,478	3,044	3,050
Slashdot0811	77,360	938,360	2,539	2,540
Erdős-Rényi (5)	70,000	$\approx 84K$	29,28-31	30-34
Barabási-Albert (5)	70,000	839928	879-1273	880-1279

Set 2 experiments. In the second set of experiments, we use unlabeled networks; these networks are in rows 4 through 8 of Table 1. Because there are no explicit labels, each edge is treated as possessing a unique label. Hence, we operate on the original networks; consequently, we omit the operations in the first row of Figure 2 and execute the procedures in the second and third rows. While these networks are much smaller than the networks of Set 1, they are bigger than the subgraphs that we operate on in Set 1, and hence represent a different kind of scalability assessment of Algorithm 2.
Types of experimental analyses. In both sets of experiments, we evaluate Algorithm 2 using two performance measures: (i) the quality of the inferred model and (ii) the total number of queries used to obtain the model. Further, the quality in (i) is assessed in two ways: (a) how close the estimated probabilities are to the actual model and (b) how IC (contagion) dynamics on networks compare when using true versus estimated edge probabilities. Due to the stochastic nature of Algorithm 2, for each combination of IC model, ϵ and δ, we obtained 10 estimates of the IC model probabilities. All networks in this study are taken as directed.

5.2 Large Graph Experimental Results

The starting point for this work is step 1 of Figure 2, and we progress through step 5 in generating and evaluating different networks.
Labeled network G generation. The networks in the first three rows of Table 1 are analyzed in this section. We explain the network generation process, which is the first row of Figure 2. These

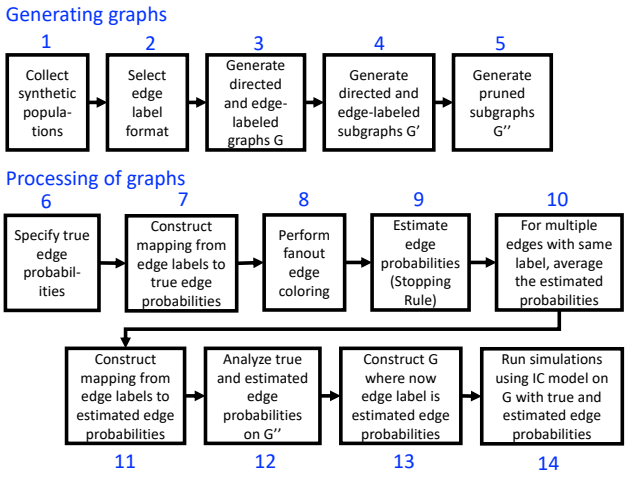

Figure 2: Workflow for experiments per graph of Table 1. The first five steps produce the directed graphs G, G' and G'' from synthetic populations, and are applicable to the first set of experiments. The last nine steps operate on graphs to infer model properties, compare true and estimated IC models (i.e., edge probabilities), and compare dynamics on networks through simulation and are applied to the networks in both sets of experiments. This workflow uses the parameter values in Table 2.

networks G, which have been used to make epidemic and economic assessments to inform public policy [4, 21], are constructed from synthetic human populations that have per-person attributes such as age, gender, family composition, and home location, and a set of daily activities (e.g., go to work, go to school). See [3] for population construction details.

To construct edge labels, we are guided by the influence literature, where age and context affect how individuals influence each other. For example, the degree of influence on a person by others changes between ages 10 to 30 [24, 28], although there are clearly additional complicating factors [8]. Furthermore, a person's interactions with others vary with context [30], which we represent through activity types of individuals' activities. We first bin the ages of people in the population in 10-year increments per row 2 of Table 2. Then, people in a population can have any number of activities of the six types, as specified in the third row of Table 2. A network G is induced on the population in the following way: there is an undirected edge $\{n_1, n_2\}$ between humans n_1 and n_2 if they are co-located (in the same room of the same building during an activity) and the times of their visits to the location overlap. Each undirected edge produces two directed edges. For our purposes, we specify a 4-tuple as an edge label for the directed edge (n_1, n_2); this is shown in the fourth row of Table 2. The set of all possible edge labels is denoted by L, and we let $n_\ell = |L|$. Our methods are agnostic to the particular form of an edge label.

Unpruned subgraph G' generation. For each network, we construct a subgraph according to Algorithm 3. We call this the *unpruned* subgraph, designated G'. The intuition behind this subgraph is as follows. Each edge label in L in the original graph represents a corresponding edge probability in the IC model. Our goal is to determine these probabilities at a minimal cost. To do this, we identify a small subset of nodes of G such that among *all* of the edges between pairs of these nodes, there is at least one edge with each label from L. This enables us to determine the probability corresponding to each edge type. For simplicity, we assume the cost of each node to be 1 so that the goal is to produce a subgraph with a small number of nodes. Also, for the specified nodes, we cannot intervene into the system; e.g., we cannot remove edges (interactions) between pairs of nodes, even when these edges have redundant labels, because we must not disrupt the interactions among people. This is called the **non-intervention** condition. The fanout coloring is used precisely to deal with this constraint, providing a method for specifying system configurations to query.

There are several additions to the implementation of Algorithm 3 in producing G' from G. First, both G and G' are directed. Second, when selecting node v, after the first criterion of selecting a node incident on the greatest number of labels yet to be covered, we prefer out-edges from v rather than in-edges because this can enable more efficient queries: all out-edges from v can be evaluated simultaneously by setting all of the corresponding in-nodes on these edges to state 0. We break ties by selecting a node with the minimum number of edges. The last criterion is to minimize the number of additional edges introduced into G' (that may have redundant labels). After selecting a node v and its neighbors that contribute new edge labels to G', all edges among these nodes, and between these nodes to the existing nodes of G' are formed. This is to satisfy the criterion to not intervene in the system G' by removing interactions.

Table 2: Summary of the parameters and their values used in the experiments. These are used in conjunction with Figure 2.

Parameter	Description
Networks G.	Networks (graphs and subgraphs) in Table 1.
Age bins.	Nodes in labeled graphs are humans. Ages are binned [0,9], [10,19], [20,29], . . ., [80,89], 90+. This gives 10 age bins.
Activity types.	Humans can have six different activity types: home, work, school, college, shopping, other [3].
Edge label.	For directed edge (n_1, n_2), from node n_1 to node n_2, the edge label is the 4-tuple (activity type of n_1, activity type of n_2, age bin of n_1, age bin of n_2).
True edge probabilities.	These are edge probabilities for the IC model. Set 1 experiments: p_e values assigned from $\{0.1, 0.2, 0.3, \ldots, 0.8, 0.9\}$. Set 2 experiments: $\{0.1, 0.2, \ldots, 0.5\}$.
Estimated edge probabilities.	These are edge probabilities for the IC model produced by Algorithm 2.
True instance, t_i.	For each network G' and G'', there are five independent probability assignments made to edges, called true probabilities, to account for stochasticity. For each edge label, a true probability p_e is assigned uniformly at random, for each instance. Values for instances are labeled 0 through 4.
Epsilon, ϵ.	Used in Algorithm 2 that estimates edge probabilities; three values are used: $\epsilon = 0.1, 0.3, 0.5$.
Delta, δ.	Used in Algorithm 2 that estimates edge probabilities; three values are used: $\delta = 0.1, 0.3, 0.5$.
Estimated solutions, t_e.	For each assignment of true edge probabilities, ten estimated solutions are computed according to Algorithm 2 to account for stochasticity. We average results over all ten instances.

Pruned subgraph G'' generation. There is one more step. We may be able to prune edges from G'. A node is **critical** if it is incident on an edge that is the only representative of a label in G'. If a node is only critical as an in-node (i.e., as v in directed edge (u, v)), then any edge for which v is an out-node (i.e., v in (v, u)) can be deleted, because it is redundant. Similarly, for nodes that are only critical as out-nodes, all incoming edges can be removed from G'. The insight for critical out-nodes is as follows. These nodes will be set to state 1 in some queries to determine their effect on their corresponding in-neighbors, to infer those edge probabilities. They never need to be set to state 0 and used to determine the probability on an incoming edge because there is another edge in G' that can be used to infer this probability. The same intuition applies for in-nodes. In this way, we prune G', producing the pruned graph G''. This process only removes edges; no nodes are removed. Also, this process does not violate our non-intervention condition above: these edges are removed because we will never set the out-nodes of these edges to state 1 in any query, so that they will not be used to infer edge probabilities. A similar argument holds for in-nodes.

Results on generating labeled subgraphs. Table 3 shows our results on generating subgraphs G' and G''. The number n_ℓ of labels in G for each network is shown in the rightmost column of that table. The numbers of nodes in these networks are far less (about 5× less) than $2n_\ell \approx 900$, which is the number of nodes required if n_ℓ edges with unique labels formed a perfect matching. Assuming unit cost per node in Algorithm 3, this is a roughly 5× cost savings. Further, the pruning process reduced the number of edges in G'' compared that of G', by about 17%. In turn, this will reduce the number of queries for inferring edge probabilities.

Table 3: Results on subgraphs G' and G'' generated with Algorithm 3 based on the labeling of the directed networks in Set 1 of Table 1. The subgraphs are $10^4\times$ to $10^6\times$ smaller than the original graphs, in terms of numbers of edges. The number of edges in G'' is further reduced by pruning edges in G'. The numbers of edge labels n_ℓ are the same for the original graphs.

Network	Properties					
	n	$\lvert E'\rvert, \lvert E''\rvert$	$\frac{\lvert E'\rvert}{n_\ell}, \frac{\lvert E''\rvert}{n_\ell}$	Δ', Δ''	Fanout′, Fanout″	n_ℓ
NRV	193	792, 659	1.74, 1.45	70, 66	36, 32	456
Miami	172	872, 731	1.82, 1.53	82, 78	42, 38	479
Seattle	165	850, 698	1.78, 1.46	86, 79	44, 37	477

The performance of Algorithm 3 is shown in Figure 3. This figure shows that for all three networks, all n_ℓ edge labels can be covered by selecting nodes in about 20 iterations of the while loop in Algorithm 3 to produce G'. The first couple of iterations reduce the number of uncovered edge labels by about 50%, with asymptotic progress thereafter.

Note that the results in Table 3 and Figure 3 are for evaluating the top row in Figure 2. We now turn to evaluating the IC model inference, the second and third rows of Figure 2.

Results on comparing true p_e and estimated $\hat{p_e}$ edge probabilities in IC model. This experiment covers steps 6 through 12 of Figure 2. The parameters evaluated begin with the *True edge probabilities* in Table 2. A true edge probability in $\{0.1, 0.2, \ldots, 0.9\}$

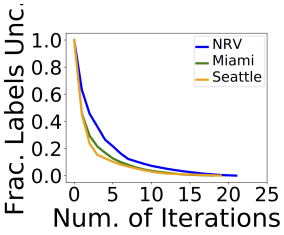

Figure 3: Progress in covering edges and their labels in G to form unpruned subgraphs G', i.e., progress in reducing the fraction of labels uncovered (Frac. Labels Unc.).

is assigned to each edge label in G' and G''. Fanout edge coloring is performed on G' and G'', and Algorithm 2 is used to estimate edge probabilities for the five different true edge probability instances t_i, and for all nine combinations of (ϵ, δ). Because Algorithm 2 is stochastic, we generate 10 estimated edge probability $\hat{p_e}$ solutions for each combination of $(G, t_i, \epsilon, \delta)$ of Table 2. Note that this algorithm is edge-label unaware, so that each edge's probability is estimated independently. Thus, while there is a unique mapping from edge label to p_e, there is no unique mapping from edge label to $\hat{p_e}$. To create this latter mapping, we average the computed $\hat{p_e}$ values on edges that have the same label. We then use these mappings to produce edge probabilities (p_e and $\hat{p_e}$) for all edges in the first three original networks G of Table 1. We will discuss the larger (original) networks below momentarily. Now, we provide results for the comparisons of p_e and $\hat{p_e}$, and for numbers of queries.

Figure 4 provides average absolute errors, computed by averaging values of $|p_e - \hat{p_e}|$ across all edges, as a function of ϵ. These data points, represented as squares, have values around 0.02. That is, the average error between true and estimated probabilities is quite small, considering that $p_e \in [0.1, 0.9]$. However, the maximum error for an edge can be large—roughly 0.8 in this figure. Theses results indicate that while the maximum error in estimated probability can be large, most probability estimates are in very good agreement with the true probabilities, because the average absolute error is quite small. We consider probabilities on G'' (and G') because these are the graphs on which the edge probabilities are estimated.

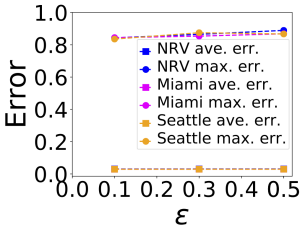

Figure 4: Average absolute errors $|p_e - \hat{p_e}|$ (squares) across all subgraph edges and maximum absolute errors (circles) across all edges, as a function of ϵ, for $\delta = 0.5$. Results are qualitatively similar for other values of δ. These results were generated on G''; results are similar for G'.

Figure 5 shows for G'' of Miami and Seattle networks the number of queries used to achieve the level of accuracy in $\hat{p_e}$ conveyed by ϵ and δ. We note that as ϵ increases the numbers of queries decrease, and for a fixed ϵ, the numbers of queries decrease as δ increases. Results for G'' of NRV are similar. These results make for interesting comparisons with the graphs in Set 2 as discussed in Section 5.3.

Results on comparing simulations on networks using true p_e and estimated $\hat{p_e}$ edge probabilities: a usage scenario. To evaluate the effects of p_e and $\hat{p_e}$ on *population dynamics*, we run simulations on the large NRV, Miami, and Seattle networks of Table 1. We used the following combinations of (ϵ, δ) values: (0.1, 0.1),

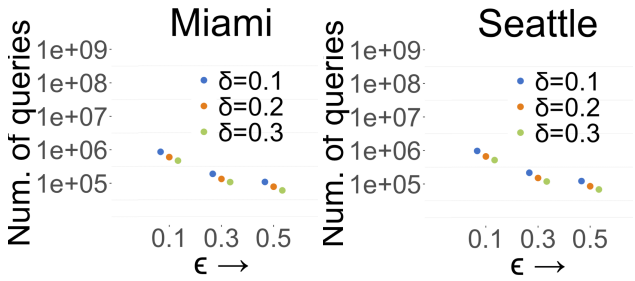

Figure 5: The number of queries required to compute edge probability estimates. The numbers of queries decrease as ϵ and δ increase. The results for NRV are similar to those for Miami and Seattle.

(0.3, 0.3), and (0.5, 0.5). For each (G, p, ϵ, δ) 4-tuple (where p indicates whether true p_e or estimated $\hat{p_e}$ probabilities are used), we run 100 diffusion instances; the initial condition for each run is that one randomly-chosen node is in state 1 and all others are in state 0. The seed node for run j ($1 \leq j \leq 100$) is the same for all (G, p, ϵ, δ).

Results are shown in Figure 6 for the Miami network. Figure 6(a) shows the cumulative fraction of nodes in the network that reach state 1 as a function of time. The three curves that each use estimated edge probabilities are in excellent agreement with the curve generated with true probabilities; the curves overlap. In Figure 6(b), we assess the fraction of nodes changing to state 1 at each time step, and compare these values from the simulations using estimated probabilities to those generated in simulations with the true probabilities. The errors are small—about 1% or less in the predicted fraction of newly infected nodes as a function of time—with the largest errors corresponding to the greatest spreading rate of the contagion. The plots for the other two networks are similar. This use case illustrates how inferred threshold systems can be used to produce real-world predictions of contagion dynamics.

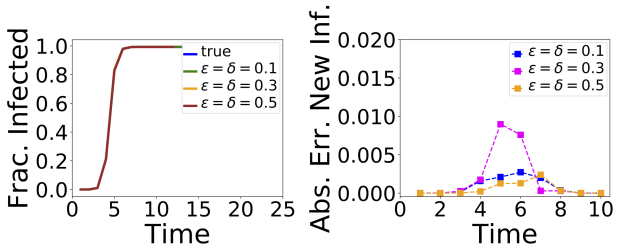

Figure 6: (a) Comparison of agent-based simulation results, using the IC model per agent, for true edge probabilities, versus estimated edge probabilities for the (ϵ, δ) combinations in the legend. It shows the cumulative fraction of nodes (agents) in the Miami network that are infected/activated as a function of time. It can be seen that $\hat{p_e}$ values produce contagion dynamics results similar to those produced by p_e. Each curve is the average of 100 simulation instances. Variances at each time, for the 100 simulations, are very small and are not shown for clarity. The largest variance over all data is 0.028 (at $t = 6$); the great majority of variance values are $< 10^{-4}$. (b) Plot of absolute error in number of new infections/activations per time unit; hence, these data are *temporal* errors in simulation results. These errors are small: less than 1% in the fraction of infected/activated nodes.

5.3 Small Graph Experimental Results

The experiments were performed on two sets of synthetic networks and three real-world networks [17], all of comparable size. They are listed in the last five rows of Table 1. Among the synthetic networks, one set consists of five replicates of directed Erdős-Rényi graphs. The other set has five replicates of the Barabási-Albert graph; each replicate was obtained by first constructing an undirected graph with $70K$ nodes and average degree 6, and then replacing each edge with a pair of bidirectional edges. We used the same approach to obtain directed versions of the real-world networks. For space reasons, we present representative results for selected networks. Other networks exhibit similar behavior. For our experiments, each IC model corresponding to a network was obtained by drawing the edge probabilities uniformly at random from the discrete set $\{0.1, 0.2, 0.3, 0.4, 0.5\}$. For each network, we considered five replicates of the IC model. It should be noted that the probability inference algorithm (Algorithm 2 in Section 3) was run on these graphs directly; we didn't use the algorithm to find a subgraph (Algorithm 3 of Section 4).

Fan-out edge coloring. The number of colors used τ is listed in Table 1. In all cases, the quantities τ and $\Delta_{\text{in}} + 1$ were very close; the maximum difference between τ and $\Delta_{\text{in}} + 1$ was 9. Recalling the discussion on the optimal number of queries required (Section 3), we note that in practice the approach of constructing the line graph $\mathcal{E}_G$ and using the greedy vertex coloring strategy seems to yield near-optimal results.

Accuracy of the estimates. In Figure 7(a), we compare the estimated influence probabilities with the reference IC model using two measures, namely mean absolute error (boxes) and maximum error (vertical lines). Here, the error corresponding to an edge probability p and its estimate $\hat{p}$ is $|\hat{p} - p|$. For each (ϵ, δ) pair, the mean absolute error is much lower than ϵ; it is roughly 0.1ϵ. Also, we observe that even the maximum error is comparable to ϵ, indicating that the estimates are much more accurate than the performance guarantees given by our theoretical results even for high values of δ. Also, we did not see much variation across networks since network structure has no role to play in this analysis. (Network structure only affects the number of queries required.)

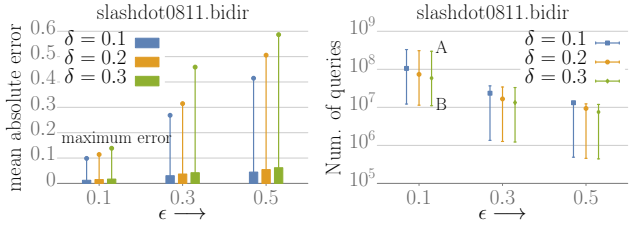

Figure 7: (a) The accuracy of the estimated models and (b) the number of queries required to obtain these estimates. The results shown here for Slashdot0811 are representative of the results for all networks. In (b) $A = 3\tau T(\epsilon, \delta/|E|)\rho/p_{\min}$ and $B = \Delta_{\text{in}}\tau T(\epsilon, \delta/|E|)\rho/p_{\max}$, the upper and lower bounds for the expected number of queries respectively.

Number of queries required. In Figure 7(b), the average number of queries required to infer the given absolute model for different ϵ, δ values is shown for a representative network. This is compared with two values based on the discussion in Section 3. Consistently, we note that the number of queries required is close

to $3\tau T(\epsilon, \delta/|E|)\rho/p_{\min}$. Even though the number of colors τ is close to the lower bound, since the probabilities were drawn uniformly from $\{0.1, \ldots, 0.5\}$, under the assumption that most color classes have many edges in them, the probability that every color class has an edge with probability $p_{\min} = 0.1$ is high. Under this observation, the lower bound significantly improves ($p_{\min}$ replaced by $p_{\max}$). We note that the number of queries used here (from about 10^7 to 5×10^8) is much larger compared to those in Section 5.2 since the subgraph generation algorithm was not used.

Dynamical analysis. For every IC model (both true and inferred), we simulated the spread on all networks by seeding one vertex at random in each instance. The results are averaged over 100 iterations. Figure 8 (left) summarizes the error in the fraction of influenced nodes in the inferred models when compared with the reference model. We recall that there are five IC models for every network with probabilities drawn from the same distribution. The large variance in mean absolute error suggests that the probability assignment plays an important role in inference. Also, an increase in ϵ does not show a corresponding increase in the error. Figure 8 (right) shows average curves of cumulative fraction of nodes influenced as a function of time for the 5 networks. The variance in inferred models rises at large outbreak sizes.

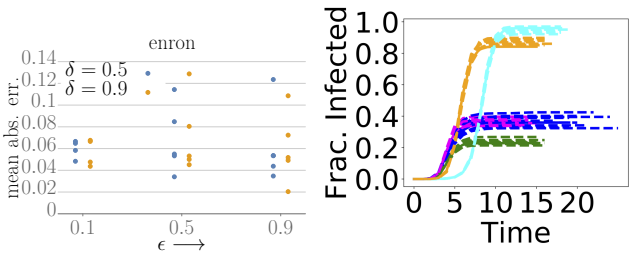

Figure 8: Comparison of dynamics between inferred models and reference model: We show the error in the average fraction of infections. Representative plots for Enron network are shown (left). Also shown are the cumulative infection plots (right) for various networks as a function of time for $\epsilon = \delta = 0.5$ (Enron–blue, Epinion–green, Slashdot–magenta, ER–cyan, and BA–orange). Variances within 100 iterations of one simulation are about 0.08 maximum; we show differences across runs with multiple curves.

6 FUTURE WORK

Our work suggests several future research directions under the active query framework. For example, one can investigate the edge probability inference problem for other diffusion models such as the SIR model [10] and its variants (e.g., SEIR model). Another direction is to obtain probability estimates for a maximum number of edges under a budget on the number of queries. Finally, it is of interest to develop a more sophisticated stopping criterion to further reduce the number of queries needed to obtain provably good performance guarantees.

Acknowledgments: We thank the referees of CIKM 2018 for providing valuable suggestions. We thank Dominik Borkowski, William Miles Gentry, Jeremy Johnson, William Marmagas, Douglas McMaster, Kevin Shinpaugh, and Robert Wills. This work has been partially supported by DTRA CNIMS (Contract HDTRA1-11-D-0016-0001), NSF DIBBS Grant ACI-1443054, NSF BIG DATA Grant IIS-1633028 and NSF EAGER Grant CMMI-1745207. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

REFERENCES

[1] A. Adiga, V. Cedeno-Mieles, et al. 2018. Active Query Based Inference of Probabilistic Contagion Models Over Networks. NDSSL Tech. Rep. 2018-48, Virginia Tech, Blacksburg, VA.
[2] A. Adiga, C. J. Kuhlman, M. V. Marathe, S. S. Ravi, D. J. Rosenkrantz, and R. E. Stearns. 2017. Inferring local transition functions of discrete dynamical systems from observations of system behavior. *Theor. CS.* 679 (2017), 126–144.
[3] C. Barrett, R. Beckman, M. Khan, V. S. A. Kumar, M. Marathe, P. Stretz, T. Dutta, and B Lewis. 2009. Generation and Analysis of Large Synthetic Social Contact Networks. In *Proceedings of the Winter Simulation Conference*. 1003–1014.
[4] C. Barrett, K. Bisset, J. Leidig, A. Marathe, and M. Marathe. 2011. Economic and social impact of influenza mitigation strategies by demographic class. *Epidemics* 3 (2011), 19–31.
[5] C. Barrett, H. B. Hunt, M. V. Marathe, S. S. Ravi, D. J. Rosenkrantz, and R. E. Stearns. 2011. Modeling and analyzing social network dynamics using stochastic discrete graphical dynamical systems. *Theor. CS* 412, 30 (2011), 3932–3946.
[6] D. Centola. 2010. The Spread of Behavior in an Online Social Network Experiment. *Science* 329 (2010), 1194–1197.
[7] P. Dagum, R. Karp, M. Luby, and S. Ross. 2000. An optimal algorithm for Monte Carlo estimation. *SIAM Journal on computing* 29, 5 (2000), 1484–1496.
[8] L. M. S. Dekkers, A. Bexkens, A. D. Hofman, et al. 2017. Formal Modeling of the Resistance to Peer Influence Questionnaire: A Comparison of Adolescent Boys and Girls With and Without Mild-to-Borderline Intellectual Disability. *Assessment* 43 (2017), 1–14.
[9] R. Durrett. 1988. *Lecture Notes on Particle Systems and Percolation.* Wadsworth.
[10] D. Easley and J. Kleinberg. 2010. *Networks, Crowds, and Markets: Reasoning About a Highly Connected World.* Cambridge University Press.
[11] M. R. Garey and D. S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-completeness.* W. H. Freeman & Co., San Francisco.
[12] J. Goldenberg, B. Libai, and E. Muller. 2001. Talk of the Network: A Complex Systems Look at the Underlying Process of Word-of-Mouth. *Marketing Letters* 12, 3 (2001), 211–223.
[13] S. González-Bailón, J. Borge-Holthoefer, A. Rivero, and Y. Moreno. 2011. The dynamics of protest recruitment through an online network. *Scientific Reports* (2011).
[14] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan. 2010. Learning Influence Probabilities in Social Networks. In *WSDM*. 241–250.
[15] D. Kempe, J. Kleinberg, and E. Tardos. 2003. Maximizing the Spread of Influence Through a Social Network. In *Proc. 9th ACM SIGKDD*. 137–146.
[16] J. Kleinberg, S. Mullainathan, and J. Ugander. 2017. Comparison-Based Choices. arXiv:1705.05735v1 [cs.DS]. 20 pages.
[17] J. Leskovec and A. Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection.
[18] S. Li, X. Gao, W. Bao, and G. Chen. 2017. FM-Hawkes: A Hawkes Process Based Approach for Modeling Online Activity Patterns. In *CIKM*. 1119–1128.
[19] T. M. Liggett. 1985. *Interacting Particle Systems.* Springer.
[20] L. Liu, J. Tang, J. Han, and S. Yang. 2012. Learning Influence from Heterogeneous social networks. *Data Mining and Knowledge Discovery* 25 (2012), 511–544.
[21] A. Marathe, B. Lewis, J. Chen, and S. Eubank. 2011. Sensitivity of Household Transmission to Household Contact Structure and Sizes. *PLoS One* 6 (2011), e22461–1–e22461–7.
[22] W. Mason and D. J. Watts. 2012. Collaborative Learning in Networks. *PNAS* 109, 3 (2012), 764–769.
[23] M. Mitzenmacher and E. Upfal. 2005. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis.* Cambridge University Press.
[24] K. C. Monahan, L. Steinberg, and E. Cauffman. 2009. Affiliation With Antisocial Peers, Susceptibility to Peer Influence, and Antisocial Behavior During the Transition to Adulthood. *Dev. Psych.* 45 (2009), 1520–1530.
[25] H. Mortveit and C. Reidys. 2007. *An Introduction to Sequential Dynamical Systems.* Springer, New York, NY.
[26] D. Romero, B. Meeder, and J. Kleinberg. 2011. Differences in the mechanics of information diffusion across topics: Idioms, political hashtags, and complex contagion on Twitter. In *Proc. 20th WWW*. ACM, 695–704.
[27] K. Saito, R. Nakano, and M. Kimura. 2008. Prediction of Information Diffusion Probabilities for Independent Cascade Model. In *Proc. KES*. 67–75.
[28] L. Steinberg and K. C. Monahan. 2007. Age Differences in Resistance to Peer Influence. *Dev. Psych.* 43 (2007), 1531–1543.
[29] H. Tong, B. A. Prakash, T. Eliassi-Rad, M. Faloutsos, and C. Faloutsos. 2012. Gelling, and melting, large graphs by edge manipulation. In *CIKM*. 245–254.
[30] J. C. Turner, P. J. Oakes, S. A. Haslam, and C. McGarty. 1994. Self and Collective: Cognition and Social Context. *Personality and Social Psychology Bulletin* 20, 5 (1994), 454–463.
[31] D. B. West. 2001. *Intro. to Graph Theory.* Prentice-Hall.