

Seeding with Costly Network Information

Dean Eckles^{1,2*}, Hossein Esfandiari[†], Elchanan Mossel^{2,3*}, and M. Amin Rahimian^{1,2*}

We study the choice of k nodes in a social network to seed a diffusion with maximum expected spread size. Most of the previous work on this problem (known as influence maximization) focuses on efficient algorithms to approximate the optimal seed sets with provable guarantees, while assuming knowledge of the entire network. However, in practice, obtaining full knowledge of the network is very costly. To address this gap, we propose algorithms that make a bounded number of queries to the graph structure and provide almost tight approximation guarantees. We test our algorithms on empirical network data to quantify the trade-off between the cost of obtaining more refined network information and the benefit of the added information for guiding improved seeding policies.

Keywords: Viral marketing, influence maximization, social networks, submodular maximization, query oracle

1 INTRODUCTION

Decision-makers in marketing, public health, development, and other fields often have a limited budget for interventions, such that they can only target a small number of people for the intervention. Thus, in the presence of social or biological contagion, they strategize about where in a network to intervene — often where to seed a behavior (e.g., product adoption) by engaging in an intervention (e.g., giving a free product) [9, 22, 24, 28, 33, 43]. The influence maximization literature is devoted to the study of algorithms for finding a set of k seeds so as to maximize expected adoption, given a known network and a model of how individuals are affected by the intervention and others' adoptions [22]. However, finding the best k seeds for many models of social influence is NP-hard, as shown by Kempe, Kleinberg and Tardos [33]. Much of the subsequent influence maximization literature is concerned with developing efficient approximation algorithms with theoretical guarantees to make use of desirable properties of the influence function such as submodularity [17, 57].

With few exceptions [46, 60], the seeding strategies that are studied in the influence maximization literature require explicit and complete knowledge of the network. However, collecting the entire network connection data can be difficult, costly, or impossible. For example, in development economics, public health, and education, data about network connections is often acquired through costly surveys (e.g., [8, 13, 14, 50]). Indeed, to reduce the cost of such surveys a few seeding strategies have been proposed to avoid collecting the entire network information by relying on stochastic ingredients, such as one-hop targeting, whereby one targets random network neighbors of random individuals [14, 18, 36]. Moreover, such methods have the advantage of scalability, since they can be implemented without mapping the entire network. This is particularly important in online social networks with billions of edges, where working with the entire contact lists might be impractical. Although the importance of influence maximization with partial network information has been noted and there are a few papers considering this problem, none of these previous work comes with provable quality guarantees for general graphs.

In this work, we address the problem of influence maximization using partial information of the network which has attracted attention recently [45, 46, 54, 60]. We enhance the theoretical

^{*}¹Sloan School of Management, ²Institute for Data, Systems and Society, and ³Department of Mathematics, Massachusetts Institute of Technology, {eckles,elmos,rahimian}@mit.edu.

[†]Google Research, esfandiari@google.com.

Authors are listed in alphabetical order.

The authors gratefully acknowledge the research assistantship of Md Sanzeed Anwar through the MIT IDSS UROP. Elchanan Mossel is supported by ONR grant N00014-16-1-2227, NSF grant CCF-1665252 and ARO MURI grant W911NF-19-0217.

foundations of what is achievable in this context by proposing two models of querying the graph structure.

In the first model (Section 3), we assume random access to graph edges by probing the nodes. Each node reveals its incident edges with probability p , which is the same as the independent cascade probability along each edge. Accessing the graph information by performing edge queries is a common technique in sublinear time algorithms that provide an output after inspecting a small portion of their input [2, 3, 15, 23, 30]. In the second model (Section 4), each query consists of a spread whereby a signal node is seeded and the identities of the resultant adopters is observed. With both models, we propose algorithms that provide almost tight approximation guarantees for seeding with upper-bounds on the number of queries. In Section 5, we explain how our bounded-query framework can be applied to quantify the value of network information for seeding with costly network information. We provide concluding remarks, and a discussion of the open problems and future directions in Section 6. The detailed proofs are provided in Appendix A. We present all of our results for the case of symmetric influences (in an undirected graph with the same cascade probability p on all the edges). In Appendix B, we discuss the extension of our results to asymmetric influences with weighted, directed graphs. In Appendix C, we discuss the extension of our results to other diffusion models, beyond independent cascade.

1.1 Related work

Motivated by the difficulties of acquiring complete network data, we are interested in influence maximization methods that do not make explicit use of the full graph. Such methods have roots in applied work — for vaccination [21] and sensing [19], in addition to seeding. One approach that has received substantial attention is a “one-hop” strategy (sometimes called “nomination” [36] or “acquaintance” targeting [14, 21]) that selects as seeds the neighbors of random nodes; this exploits a version of the friendship paradox that whereby the friend of a random individual is expected to have more friends than a random individual [40]. For example, Kim et al. [36] report on the results of field experiments that target individuals for delivery of public health interventions. They argue that that one-hop targeting (whereby a random individual nominates a friend to be targeted) leads to increased adoption rates, compared with random or in-degree targeting. Some other empirical work has been less encouraging ([18], cf. [39]). While there are results about how these short random walks affect the degree distribution of selected nodes [38], one-hop seeding currently lacks any theoretical guarantees under models of contagion. Furthermore, given the collection of data about the network neighborhoods of k nodes, it is natural to consider whether this data can be more effectively used than just locally taking a random step, ignoring data collected from the other $k - 1$ neighborhoods.

To address the challenges of seeding when obtaining network information is costly, we offer a framework for influence maximization using a bounded number of queries to the graph structure. In this framework, we investigate the expected spread size versus the increasing number of queries as we obtain more information about the network. Recently, other frameworks have been also proposed to study the trade-off between the cost of seeding and the benefits of faster and further diffusion, e.g., [1, 44]. We postpone a detailed comparison with these works until Section 5, after clarifying our modeling assumptions and results.

Particularly relevant to our present study are the recent works on influence maximization for unknown graphs [45, 46, 54, 59, 60]. Mihara et al. [45] use a biased sampling strategy to greedily probe and seed nodes with the highest expected degree, given the activated nodes from the previous step. They later propose to include random flights to improve this heuristic [46]. Stein et al. [54] explore applications of common heuristics and other known algorithms in scenarios where parts of the network is completely unobservable. Although simulations of influence spread on synthetic and

real social networks provide some evidence, none of these results come with provable performance guarantees in general graphs.

To the best of our knowledge, the only available guarantee for influence maximization with unknown graphs is due to Wilder et al. [60]. However, they only analyze their algorithm for graphs that are generated from a stochastic block model. Roughly speaking, they use the outcome of the queries to estimate the size of each block and choose nodes to seed the largest blocks. Such an analysis does not apply to general graphs and the techniques that we use to provide performance guarantees for general graphs are significantly different. They also provide an impossibility result for general graphs by lower-bounding the required number of node queries. We extend these results by proposing a complementary setup that addresses edge queries in general graphs.

The main technique at the heart of our results is a sketching to summarize influence functions. We adopt some high level ideas from [10, 11] and bound the number of queries to the graph structure for constructing a sketch. The authors in [20] also develop a sketch-based algorithm for influence maximization to bound the running time, with approximation guarantees. However, their aim is to design a fast algorithm and they are not concerned with query complexity. Even though their algorithm serves their purpose, it may require to query all of the edges in the input. In another related work, Borgs et al. query the graph structure by running reverse spreads and give a quasi-linear time algorithm for influence maximization [12]. We postpone a detailed comparison between our work and theirs until Section 4, after clarifying our query model and seeding algorithm.

Our influence maximization guarantees also relate to the recent developments in stochastic submodular maximization [32], as well as optimization from samples [5, 6]. The key difference is that in our algorithms we make explicit use of the combinatorial structure of the observed spreads. This is in contrast with the prior literature [5, 6, 32], where only the sampled values of the submodular function are observed. Consequently, we are able to provide guarantees for arbitrary inputs, and avoid some of the limitation of the optimization from samples [7].

In fact, our results fall in the general category of work that addresses how lack of perfect knowledge about the spreading process affects the design and performance of seeding strategies. This question has been studied by researchers from many other angles, including algorithmic stability and robustness [27, 31]. Some other prior work has focused on uncertainty in spreading model parameters, for example, by bounding the worst-case performance when the true edge influence probabilities are constrained to an interval [16]. Others have considered such uncertainties in dynamic settings [61] or with changing network graph [63].

Learning is another popular approach for influence maximization in uncertain environments [5, 26, 58]. In online-learning and bandit-based approaches, the seed sets are selected across different stages and the feedback from the previous stage can be used to guide future seed selections [41]. This is also related to adaptive seeding, where the initial choice of seeds influences what becomes available for seeding in a followup stage [29, 53]. The ability to seed nodes adaptively makes such setups incomparable to ours.

1.2 Our contributions

We consider the independent cascade model of social contagion [33] that is fairly well-studied. In this model, each active agent has a single chance to activate each of its neighboring agents with probability p , independently at random. Motivated by applications to product and technology adoption, we refer to activated nodes as adopters. Starting from a set of initial adopters, following the independent cascade model, the adoption propagates through the network and the process terminates after a finite number of steps. The k -Influence Maximization, or k -IM problem, refers to the choice of k initial adopters to maximize the expected number of adoptions. Let L be the optimum value for this problem. A μ -approximation algorithm outputs a set of k initial adopters to

guarantee that the expected number of adoptions is at least μL . In this work we assume a query oracle to the graph structure. We study the k -Influence Maximization problem with limited number of queries.

In the first model, we consider edge queries. In particular, a query is of the form (v, i) and returns the i 'th edge of node v , where the edges are ordered arbitrarily. One natural question that arises here is to study the relation between the required number of queries and the cascade probability. In particular, is it possible to find an *approximately optimal* seed set using sub-quadratic number of queries when p is desirably small? We resolve this question by developing a non-trivial, dependent sampling of the edges of the network that approximately preserves the solution of the k -Influence Maximization problem.

THEOREM (APPROXIMATION GUARANTEE WITH BOUNDED EDGE QUERIES). *For any arbitrary $0 < \epsilon \leq 1$, there exist a polynomial-time algorithm for influence maximization that covers $(1 - 1/e)L - \epsilon n$ nodes in expectation, using $\tilde{O}_\epsilon(pn^2 + \sqrt{pn}^{1.5})$ queries, where L is the expected number of nodes covered by the optimum seed set.*

To achieve this result, we apply some subsampling and stopping constraints that enable us to *approximately simulate* $\tilde{O}(k)$ independent realizations of the cascade over the network, using only $\tilde{O}(pn^2 + \sqrt{pn}^{1.5})$ queries. Notice that a *single* simulation of cascade over the entire network (without using our subsampling and stopping constraints) requires $\Omega(pn^2)$ queries. In fact, we show that in the worst case one needs to query $\Omega(n^2)$ edges to guarantee that the expected number of covered nodes is at least a constant fraction of that of the optimum solution. The following theorem states our claim formally.

THEOREM (LOWER BOUND ON REQUIRED EDGE QUERIES). *Let μ be any constant. There is no μ -approximation algorithm for influence maximization using sub-quadratic number of queries.*

In the next step, we present a complementary model of querying the spreading process. In this model, we assume that we can pay a cost to learn the outcome of a spreading process when a node is seeded. Accordingly, we learn the identity of the final adopters, but we do not observe the network edges through which the influence spread. In practical terms, one can seed a random individual with cheap traceable coupons that she can distribute to the people under her influence, and so forth. We can then observe the adoption status of the entire network by observing the use of the coupons. Repeating this process r times we observe r independent outcomes from seeding r (potentially different) nodes. We refer to this setup as spread query. Note that such queries are just used to learn the network, i.e., to find a seeding strategy of size k . Interestingly, we show that running the spreading process $\tilde{O}(k^2)$ times is enough to provide a k -IM solution with almost tight approximation guarantees. Notice that the number of queries depends poly-logarithmically on n , and hence in the case that k is poly-logarithmic, our algorithm only requires poly-logarithmic number of spread queries to find an almost efficient seeding strategy for the entire network.

THEOREM (APPROXIMATION GUARANTEE WITH BOUNDED SPREAD QUERIES). *For any arbitrary $0 < \epsilon \leq 1$, there exist a polynomial-time algorithm for k -influence maximization that covers $(1 - 1/e)L - \epsilon n$ nodes in expectation using no more than $\tilde{O}_\epsilon(k^2)$ spread queries.*

For example on a star, with high probability all of our spread queries are leaves of the star. However, based on the results of the queries our algorithm finds and seeds the center of the star. To complement this, we show that an additive loss (e.g., ϵn) is necessary, given $o(n)$ queries.

THEOREM (LOWER BOUND ON REQUIRED SPREAD QUERIES). *Let μ be an any constant. There is no μ -approximation algorithm for influence maximization using $o(n)$ spread queries.*

Finally, we test the performance of our bounded-query algorithms on empirical social networks, and study the trade-off between cost of acquiring network information and benefit of improved seeding.

2 NOTATION AND SET UP

Consider graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with the set of nodes \mathcal{V} and the set of edges \mathcal{E} . Consider a seed set \mathcal{S} . Starting from the seeded agents in \mathcal{S} , adoption spreads in graph \mathcal{G} according to the independent cascade model. In this model each agent, upon adoption, has a “one time” chance of converting each of her non-adopter neighbors. Each conversion is successful with probability p , independently of others. Given \mathcal{S} , for $v \in \mathcal{V}$, let $\phi(v, \mathcal{S})$ be the probability that v adopts when the nodes in \mathcal{S} are seeded. Let $\Gamma(\mathcal{S}) = \sum_{v \in \mathcal{V}} \phi(v, \mathcal{S})$ be the value (influence) of the seed set \mathcal{S} , which is the expected number of nodes that adopt if the nodes in set \mathcal{S} are seeded.

DEFINITION 1 (k -IM). *Given graph \mathcal{G} , the k -Influence Maximization (k -IM) problem is to choose a seed set $\mathcal{S} \subset \mathcal{V}$, $\text{card}(\mathcal{S}) = k$ to maximize $\Gamma(\mathcal{S})$. We use $\Lambda = \text{argmax}_{\mathcal{S}, \text{card}(\mathcal{S})=k} \Gamma(\mathcal{S})$ to denote any such solution. Moreover, we use $L = \Gamma(\Lambda)$ to denote the maximum expected spread size, using an optimal seed set of size k .*

DEFINITION 2 (APPROXIMATIONS). *Given graph \mathcal{G} , any set of size k , $\Lambda^\alpha \subset \mathcal{V}$, $\text{card}(\Lambda^\alpha) = k$, satisfying $\Gamma(\Lambda^\alpha) \geq \alpha L$ is an α -approximate solution to k -IM.*

Given the knowledge of graph structure \mathcal{G} , submodular maximization algorithms (such as the simple greedy) guarantee a $(1 - 1/e)$ -approximate solution to k -IM. Here we achieve *roughly* the same guarantee using only partial information about the graph \mathcal{G} .

In our approach, rather than optimizing the influence function on the original graph we do so on a subgraph that is properly sampled from the original graph. As its main property, we show that for the appropriate choice of α and ϵ , an α -approximate solution to k -IM on this subgraph, has an influence on the original graph that is lower-bounded by $\alpha L - \epsilon n$, where L is the optimal value of k -IM. We can thus achieve similar worst-case guarantees using only partial information about the network.

3 APPROXIMATION GUARANTEE WITH BOUNDED EDGE QUERIES

In this section we present our edge query model, as well as the algorithm to output an approximate seed set given the outcome of the edge queries. The main idea is to simulate multiple independent spreads by querying each node about their edges. The way the queries are organized is by exploring the extended neighborhoods of nodes. We start from a random set of initial nodes that are fixed for subsequent steps (Subsection 3.1). We then proceed to probe their neighborhoods by asking them to reveal their neighbors with probability p , and repeating the same process for the revealed neighborhoods, etc. (Subsection 3.2). This way we obtain multiple sub-sampled copies of the graph, each of which corresponds to a cascade starting from the same set of initial nodes (see Figure 1). In Subsection 3.3, we argue that one does not need to continue probing the extended neighborhood of an initial node if the size of its connected component is large enough. This is a key observation because it allows us to upper-bound the total number of edge queries. In Subsection 3.4, we propose an algorithm to choose k seeds (approximately optimally), based on the outcome of the edge queries. The crux of the argument is in using the number of initial nodes in the connected components of each node to infer its value as a potential seed. We also prove our guarantees on the number of edge queries (Subsection 3.4.1) and the run-time (Subsection 3.4.2).

In Figure 1, we depict an example of three cascades that are obtained through edge queries. Consider the node that is marked in black. The connected component of the marked node has

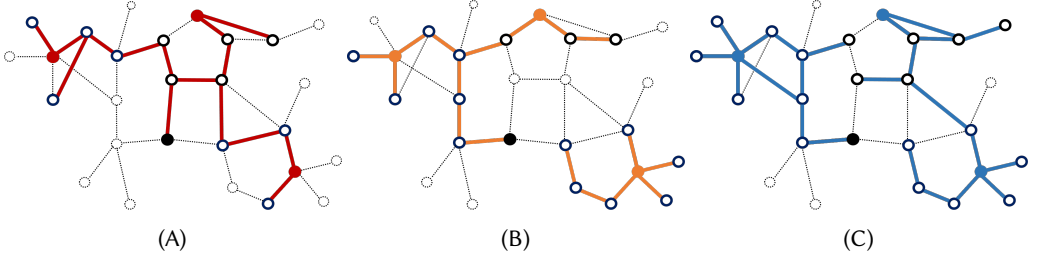


Fig. 1. Three cascades are depicted in (A) red, (B) orange, and (C) blue. All cascades start from the same random initial nodes which are marked in the same color as the cascades. The node that is marked in black scores as high as or higher than other nodes across the three cascades. The dotted sections consist of unrealized influences and nodes that are not influenced in each cascade.

three initial nodes in 1(A), two initial nodes in 1(B), and one initial node in 1(C). The value of each connected component is the number of initial nodes in that component. The total value of the marked node is $1 + 2 + 3 = 6$, which is computed by adding the values of all of the components containing that node. To prevent overlap with already chosen seeds, we set the value of a connected component to zero if one of its nodes is chosen as a seed. Using these valuations, we can approximate the greedy algorithm by adding the most valuable node to the seed set.

The hallmark of our analysis is in identifying an auxiliary submodular function $\Gamma_\delta : 2^{\mathcal{V}} \rightarrow \mathbb{R}$ to approximate our submodular function of interest $\Gamma : 2^{\mathcal{V}} \rightarrow \mathbb{R}$. The approximation is such that $|\Gamma_\delta(\mathcal{S}) - \Gamma(\mathcal{S})| \leq \epsilon n$ for all seed sets \mathcal{S} of size k , with high probability. Here ϵ is the quality of approximation and it depends on δ , which parameterizes the approximator Γ_δ . Following the notation introduced in Definitions 1 and 2, let us use Λ_δ and Λ to denote the maximizers of Γ_δ and Γ with constrained size k . The following Lemma (proved in Appendix A.1) is true for any set function Γ and its approximator Γ_δ . It allows us to bound the loss that is incurred from optimizing Γ_δ in place of Γ .

LEMMA 3. *Consider set functions Γ_δ and Γ that map subsets of \mathcal{V} to \mathbb{R} with k -IM optimal values L_δ and L . Assume that for all seed sets \mathcal{S} of size k we have $|\Gamma_\delta(\mathcal{S}) - \Gamma(\mathcal{S})| \leq \epsilon n$. Then any approximate solution Λ'_δ of k -IM for Γ_δ , satisfying $\Gamma_\delta(\Lambda'_\delta) \geq \alpha L_\delta - \beta n$, also satisfies $\Gamma(\Lambda'_\delta) \geq \alpha L - (\beta + (\alpha + 1)\epsilon)n$.*

3.1 Sampling the nodes

Recall our goal is to choose a seed set that (approximately) maximizes the influence function Γ . In this subsection, we show that we can estimate the value of Γ by choosing a large enough set of nodes uniformly at random. To begin, fix $0 < \rho < 1$ and choose $n\rho$ nodes uniformly at random. We call these the *initial nodes* and denoted them by \mathcal{V}_ρ . Given \mathcal{V}_ρ , for any set $\mathcal{S} \subset \mathcal{V}$ we estimate the value of $\Gamma(\mathcal{S}) = \sum_{v \in \mathcal{V}} \phi(v, \mathcal{S})$ by:

$$\Gamma_\rho(\mathcal{S}) := \frac{1}{\rho} \sum_{v \in \mathcal{V}_\rho} \phi(v, \mathcal{S}). \quad (1)$$

To proceed, also define

$$\rho_{\epsilon, \delta}^{n, k} := \frac{(2 + \epsilon)(k\delta \log n + \log 2)}{2\epsilon^2 n}.$$

In the next Lemma, we bound the difference between Γ and Γ_ρ for $\rho \geq \rho_{\epsilon, \delta}^{n, k}$. The proof is in Appendix A.2. In the proof, we use a standard concentration argument to control the deviation of $\Gamma_\rho(\mathcal{S})$ from $\Gamma(\mathcal{S})$ for a fixed \mathcal{S} , and then a union bound to make the inequality true for any \mathcal{S} .

LEMMA 4 (BOUNDING THE SAMPLING LOSS). *Let $\rho_{\epsilon, \delta}^{n, k} \leq \rho \leq 1$. With probability at least $1 - e^{-\delta}$, for all seed sets \mathcal{S} of size k we have $|\Gamma_{\rho}(\mathcal{S}) - \Gamma(\mathcal{S})| \leq \epsilon n$.*

3.2 Probing the neighborhoods

Note that our definition of Γ_{ρ} in (1) is in terms of $\phi(v, \mathcal{S})$. The latter can only be computed given the knowledge of the entire graph. However, with only partial network information we need to replace $\phi(v, \mathcal{S})$ by a proper estimate. To this end, we sample the graph edges through a probing procedure. Consider the $n\rho$ initial nodes in \mathcal{V}_{ρ} . For each initial node, we probe its neighborhood, keeping the edges with probability p . We then proceed to probe the neighborhoods of the revealed nodes, etc. We never probe a node more than once and each edge receives at most one chance of being sampled. The probing stops after a finite number of steps (bounded by n). We repeat this probing procedure T times and obtain T subsampled graphs that we denote by $\mathcal{G}_{\rho}^{(1)}, \dots, \mathcal{G}_{\rho}^{(T)}$.

We can now estimate $\phi(v, \mathcal{S})$ using the T copies $\{\mathcal{G}_{\rho}^{(1)}, \dots, \mathcal{G}_{\rho}^{(T)}\}$ as follows. For $i = 1, \dots, T$, set $Y^{(i)}(v, \mathcal{S}) = 1$ if v has a path to \mathcal{S} in $\mathcal{G}_{\rho}^{(i)}$, otherwise set $Y^{(i)}(v, \mathcal{S}) = 0$. Our estimate for $\phi(v, \mathcal{S})$ is

$$\phi^{(T)}(v, \mathcal{S}) := \frac{1}{T} \sum_{i=1}^T Y^{(i)}(v, \mathcal{S}). \quad (2)$$

We can similarly construct an estimate for the influence function that we want to optimize:

$$\Gamma_{\rho}^{(T)}(\mathcal{S}) := \frac{1}{\rho} \sum_{v \in \mathcal{V}_{\rho}} \phi^{(T)}(v, \mathcal{S}). \quad (3)$$

To proceed, define

$$T_{\epsilon, \delta}^{n, k} := \frac{3(\delta + \log 2)(k + 1) \log n}{\epsilon^2}.$$

Our next result bounds the difference between $\Gamma_{\rho}^{(T)}$ and Γ_{ρ} for $T \geq T_{\epsilon, \delta}^{n, k}$. In the proof, we use concentration and union bound to ensure that $\phi^{(T)}(v, \mathcal{S})$ remains close to $\phi(v, \mathcal{S})$ for all v and \mathcal{S} . The proof details are in Appendix A.3.

LEMMA 5 (BOUNDING THE PROBING LOSS). *Let $T \geq T_{\epsilon, \delta}^{n, k}$. With probability at least $1 - e^{-\delta}$, for all sets \mathcal{S} of size k we have $|\Gamma_{\rho}^{(T)}(\mathcal{S}) - \Gamma_{\rho}(\mathcal{S})| \leq \epsilon n$.*

3.3 Limiting the probed neighborhoods

Here we consider a variation of the probing procedure described in the previous subsection whereby we stop probing once there are more than a threshold τ nodes in a connected component. Note that the probing may stop even before hitting τ nodes if no new edges are activated. Limiting the probed neighborhoods in this manner helps us bound the total number of edges that are used in our sketch (see Subsection 3.4.1 and Theorem 8). In fact, we show that it is safe to stop probing when there are $\tau = \tau_{\epsilon}^{n, k}$ nodes in a connected component, where

$$\tau_{\epsilon}^{n, k} := \frac{n \log(1/\epsilon)}{\epsilon k}.$$

Let us denote the T subsampled graphs obtained through limited probing by $\mathcal{G}_{\rho, \epsilon}^{(1)}, \dots, \mathcal{G}_{\rho, \epsilon}^{(T)}$. Moreover, let $\Gamma_{\rho, \epsilon}^{(T)}$ be our estimate of the influence function that is constructed based on $\mathcal{G}_{\rho, \epsilon}^{(1)}, \dots, \mathcal{G}_{\rho, \epsilon}^{(T)}$ in the exact same way as in (2) and (3). This new estimator is, itself, a submodular function since it can be expressed as sum of coverage functions. Our following result ensures that by optimizing $\Gamma_{\rho, \epsilon}^{(T)}$

instead of $\Gamma_\rho^{(T)}$, we do not lose more than $(1 - \epsilon)$ in our approximation factor. The proof follows a probabilistic argument similar to [10, Lemma 2.4]. The crux of the argument is in constructing a random set whose expected value on $\Gamma_{\rho, \epsilon}^{(T)}$ is no less than $(1 - \epsilon)$ of the optimum on $\Gamma_\rho^{(T)}$. We do so by starting from the optimum set on $\Gamma_\rho^{(T)}$ and replacing ϵk of its nodes at random. Taking τ large enough allows us to show that those nodes whose connections are affected by limiting the probed neighborhoods belong to a large component – of size $\tau = \tau_\epsilon^{n, k}$ – that is likely to be covered by one of the ϵk random nodes. The complete proof details are in Appendix A.4.

LEMMA 6 (BOUNDING THE LOSS FROM LIMITED PROBING). *For $0 \leq \rho \leq 1$ and $0 < \epsilon \leq 1$, consider the limited probing procedure with the probing threshold set at $\tau = \tau_\epsilon^{n, k}$. Then any α -approximate solution to k -IM for $\Gamma_{\rho, \epsilon}^{(T)}$ is an $\alpha(1 - \epsilon)$ -approximate solution to k -IM for $\Gamma_\rho^{(T)}$.*

3.4 Influence Maximization on the sampled graph

Algorithm 1, below, summarizes the limited probing approach for performing edge queries on the input graph. The output is a sketch comprised of the T independent copies $\mathcal{G}_{\rho, \epsilon}^{(1)}, \dots, \mathcal{G}_{\rho, \epsilon}^{(T)}$ that fully determine the estimator $\Gamma_{\rho, \epsilon}^{(T)}$.

Algorithm 1: PROBE (ρ, T, τ)

Input: Query access to graph \mathcal{G}

Output: $\mathcal{G}_{\rho, \epsilon}^{(1)}, \dots, \mathcal{G}_{\rho, \epsilon}^{(T)}$

- (1) **SAMPLE:** Choose $n\rho$ nodes uniformly at random and call them \mathcal{V}_ρ .
- (2) **PROBE:** For i from 1 to T , do
 - (a) Probe every node in \mathcal{V}_ρ by asking them to reveal their neighbors with probability ρ .
 - (b) For any revealed neighbor that is not probed before, add the corresponding edge to $\mathcal{G}_{\rho, \epsilon}^{(i)}$ and proceed to probe them.
 - (c) Stop probing if there are no more new nodes to probe or if the size of the revealed component exceeds τ .

Lemmas 4, 5, and 6 provide the following appropriate choices of the algorithm parameters for PROBE (ρ, T, τ):

$$\begin{aligned} \rho &= \rho_{\epsilon, \delta}^{n, k} = \frac{(2 + \epsilon)(k\delta \log n + \log 2)}{2\epsilon^2 n}, \\ T &= T_{\epsilon, \delta}^{n, k} = \frac{3(\delta + \log 2)(k + 1) \log n}{\epsilon^2}, \\ \tau &= \tau_\epsilon^{n, k} = \frac{n \log(1/\epsilon)}{\epsilon k}. \end{aligned}$$

The following theorem (proved in Appendix A.5) combines our results so far (Lemmas 3 to 6) to show that any α -approximate solution Λ^* that we obtain for the k -IM problem on $\Gamma_{\rho, \epsilon}^{(T)}$, indeed, satisfies $\Gamma(\Lambda^*) \geq \alpha' L - \epsilon' n$ for appropriate choices of α' and ϵ' , providing an approximate solution to the original k -IM problem on Γ .

THEOREM 7 (BOUNDING THE TOTAL APPROXIMATION LOSS). *Consider any $0 < \epsilon, \alpha < 1$, and fix $\rho = \rho_{\epsilon, \delta}^{n, k}$ and $T = T_{\epsilon, \delta}^{n, k}$. Moreover, let $\alpha' = \alpha(1 - \epsilon)$ and $\epsilon' = 2(\alpha(1 - \epsilon) + 1)\epsilon$. With probability at least $1 - 2e^{-\delta}$, any α -approximate solution to k -IM problem on $\Gamma_{\rho, \epsilon}^{(T)}$, has value at least $\alpha' L - \epsilon' n$ on the original problem.*

In Subsection 3.4.1, we bound the total number of edges that are queried by PROBE (ρ, T, τ) . The output of the PROBE algorithm is the set of T copies $\mathcal{G}_{\rho, \epsilon}^{(1)}, \dots, \mathcal{G}_{\rho, \epsilon}^{(T)}$. From these T copies, we construct the estimate $\Gamma_{\rho, \epsilon}^{(T)}$ and can, then, use a submodular maximization algorithm to find a $(1 - 1/e - \epsilon')$ -approximate solution for any $\epsilon' > 0$. In Subsection 3.4.2, we describe a fast implementation of submodular maximization on the sketch (the output of PROBE) that runs in $\tilde{O}(pn^2 + \sqrt{pn}^{1.5} + nk)$ time.

3.4.1 Bounding the total number of edge queries. To proceed define

$$\begin{aligned} E_{\epsilon, p}^{n, k} &:= p\tau_{\epsilon}^{n, k}(\tau_{\epsilon}^{n, k} - 1)/2, \\ C_{\epsilon, \delta}^{n, k} &:= n\rho_{\epsilon, \delta}^{n, k}T_{\epsilon, \delta}^{n, k} \left(E_{\epsilon, p}^{n, k} + \sqrt{\delta(\tau_{\epsilon}^{n, k} \log(n) + \log T_{\epsilon, \delta}^{n, k})E_{\epsilon, p}^{n, k}} \right). \end{aligned} \quad (4)$$

In Theorem 8, we bound the total number of edge queries, denoted by q , in terms of $E_{\epsilon, p}^{n, k}$ and $C_{\epsilon, \delta}^{n, k}$. The proof in Appendix A.6 relies critically on how we limit the probed neighborhoods (Subsection 3.3). Roughly speaking, the total number of connected components in the output of PROBE (ρ, T, τ) is at most $n\rho T$, and each component contains at most τ nodes. Moreover, since each edges is revealed with probability p , the expected number of edges in each of these components is at most $p\tau(\tau - 1)/2$. Subsequently, concentration allows us to give a high probability upper-bound on the total number of edges that appear in the output of PROBE (ρ, T, τ) . We can, similarly, also bound the total number of edges that are queried but discarded since they have been pointing to already probed nodes — see step (2-a) of the PROBE algorithm.

THEOREM 8 (BOUNDING THE EDGE QUERIES). *For $\rho = \rho_{\epsilon, \delta}^{n, k}, T = T_{\epsilon, \delta}^{n, k}$ and $\tau = \tau_{\epsilon}^{n, k}$, with probability at least $1 - 3e^{-\delta}$ the total number of edge queries (q) can be bounded as follows:*

$$q \leq 2C_{\epsilon, \delta}^{n, k} + \left(2 + \sqrt{2}\right) T_{\epsilon, \delta}^{n, k} n \sqrt{\delta + \log T_{\epsilon, \delta}^{n, k}} \in \tilde{O}_{\epsilon, \delta}(pn^2 + \sqrt{pn}^{1.5}).$$

In Appendix A.7, we prove a matching hardness (or rather, impossibility) result to show that, in the worst case, it is impossible to approximate the problem using $o(n^2)$ edge queries for constant p . To show that there are no μ -approximation algorithms making $o(n^2)$ edge queries, we consider an arbitrary algorithm that makes less than $C_{\mu}n^2$ edge queries, for some constant C_{μ} that is specified in Appendix A.7. Our hard example consists of a collection of $9/\mu^2$ cliques of size $n\mu^2/9$ each. We choose $3/\mu$ of these cliques at random and connect them as in Figure 2. With $k = p = 1$, an optimal algorithm will seed one of the nodes in the connected cliques and achieves $(3/\mu)(n\mu^2/9) = n\mu/3$ spread size. However, an algorithm that makes less than $C_{\mu}n^2$ queries cannot detect the connected clique with probability more than $\mu/3$. In Appendix A.7, we show that the expected spread size from seeding the output of any such algorithm is less than $n\mu^2/3$ — a factor μ of the optimum.

THEOREM 9. *Let μ be any constant. There is no μ -approximation algorithm for influence maximization using sub-quadratic number of edge queries.*

This result extends the previous result of Wilder et al. [60] that lower-bounds the number of nodes needed to be queried. Moreover, as opposed to the previous work of Wilder et al. the spread size in our hard example is linear in the total number of nodes (see Figure 2). This means that our impossibility result holds even if, for some $\epsilon \ll \mu$, an ϵn additive loss is tolerated. Indeed, the hard example used in our impossibility result is adversarially tuned to require $\Omega(n^2)$ queries. In particular, we set the cascade probability p to 1. Although one can adjust the hard example to work for smaller constant cascade probabilities, the example fails when p is sub-constant. In fact, in many realistic applications of the k -IM, the cascade probability is relatively small. For example, if

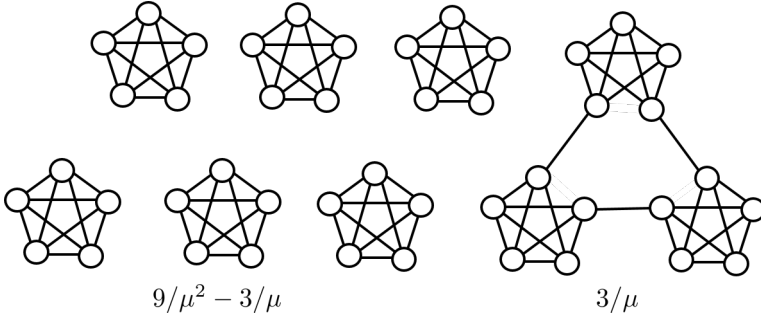


Fig. 2. To lower-bound the required number of edge queries, our hard example consists of $9/\mu^2$ cliques, $3/\mu$ of which are connected in a circle. An algorithm that makes $o(n^2)$ edge queries may detect the connected cliques with probability at most $\mu/3$. The expected performance of any such algorithm is worse than a factor μ of the optimum.

someone tweets about a product, not all of her tweeter followers are subjected to this influence, or convinced to buy, or retweet about it.

3.4.2 Bounding the total running time. In this subsection, we provide a fast implementation of our algorithm for influence maximization on the sampled graph. In fact, we can achieve a running time that is linear in the number of queried edges. First note that $\Gamma_{\rho, \epsilon}^{(T)}$ is, by definition, a coverage function, ergo a submodular function. Hence, we can use the randomized greedy algorithm of [47] to provide a $(1 - 1/e - \epsilon')$ approximation guarantee. We start with $\mathcal{S} = \emptyset$ and as in any greedy algorithm, we only use two types of operations:

- We query the marginal increase of a node v on the current set \mathcal{S} , denoted by:
 $\Delta(v|\mathcal{S}) := \Gamma_{\rho, \epsilon}^{(T)}(\mathcal{S} \cup \{v\}) - \Gamma_{\rho, \epsilon}^{(T)}(\mathcal{S})$.
- We choose a node v^* with maximal marginal increase and add it to the seed set:
 $\mathcal{S} \leftarrow \mathcal{S} \cup \{v^*\}$.

The only difference is that the search for the node v^* is restricted to a subset \mathcal{R} of size $(n/k) \log(1/\epsilon')$ that is drawn uniformly at random from $\mathcal{V} \setminus \mathcal{S}$.

Algorithm 2: SEED (ϵ')

Input: The T copies: $\mathcal{G}_{\rho, \epsilon}^{(1)}, \dots, \mathcal{G}_{\rho, \epsilon}^{(T)}$

Output: Λ^* , $(1 - 1/e - \epsilon')$ -approximate solution to k -IM for $\Gamma_{\rho, \epsilon}^T$

- (1) Find the connected components of $\mathcal{G}_{\rho, \epsilon}^{(1)}, \dots, \mathcal{G}_{\rho, \epsilon}^{(T)}$.
- (2) For every connected component in each of the T copies initialize the current value of the component equal to the number of sampled initial nodes (belonging to \mathcal{V}_ρ) in that component.
- (3) Initialize $\Lambda^* = \emptyset$.
- (4) For i form 1 to k , do
 - (a) Choose a random subset, $\mathcal{R} \subset \mathcal{V} \setminus \Lambda^*$, $\text{card}(\mathcal{R}) = (n/k) \log(1/\epsilon')$.
 - (b) For each $v \in \mathcal{R}$ compute $\Delta(v|\Lambda^*)$ by adding the current values of the connected components containing v and set $v^* = \text{argmax}_{v \in \mathcal{R}} \Delta(v|\mathcal{S})$
 - (c) Add $\Lambda^* \leftarrow \Lambda^* \cup \{v^*\}$ and set the current value of the connected components containing v^* to zero.

In Algorithm 2: SEED (ϵ'), we provide efficient implementations for the above operations. Our implementations are based on the structure of $\Gamma_{\rho, \epsilon}^{(T)}$, as determined by the T copies, $\mathcal{G}_{\rho, \epsilon}^{(1)}, \dots, \mathcal{G}_{\rho, \epsilon}^{(T)}$. First using a graph search (e.g., DFS) we find the connected components of each of the T subsampled graphs and count the number of initial nodes (belonging to \mathcal{V}_ρ) in each connected component. We refer to this count for each connected component as the "value" of that component. The main idea is that maximizing $\Gamma_{\rho, \epsilon}^{(T)}$ is equivalent to finding a seed set \mathcal{S} , such that the total value of all connected components containing at least one seed is maximized. If a connected component already contains (i.e., is covered by) some nodes in \mathcal{S} , then the marginal increase due to that component should be zero. This is achieved by setting the value of a component to zero after adding a node from that component to the seed set \mathcal{S} .

Our next result combines our conclusions from Theorems 7 and 8, as well as the analysis of the performance of fast submodular maximization (randomized greedy) in [47]. The proof is in Appendix A.8.

THEOREM 10. *For any arbitrary $0 < \epsilon \leq 1$, there exist an algorithm for influence maximization that covers $(1 - 1/e)L - \epsilon n$ nodes in expectation in $\tilde{O}_\epsilon(pn^2 + \sqrt{pn}^{1.5} + nk)$ time, using no more than $O_\epsilon(pn^2 + \sqrt{pn}^{1.5})$ edge queries.*

4 APPROXIMATION GUARANTEE WITH BOUNDED SPREAD QUERIES

Here we present a complementary setup to the edge query model of Section 3. In this section, we assume that we can pay a cost to learn the outcome of a spreading process when a single node is seeded. Repeating this process gives us independent outcomes from randomly seeding a node. We refer to this type of query as spread query, and ask how many times we should run the spreading process with a randomly seeded node to be able to provide a k -IM approximate solution whose value is at least $(1 - 1/e)L - \epsilon n$. Our following algorithm outputs one such solution with the desired guarantee for

$$\rho = \rho_\epsilon^{n,k} = \frac{81k \log \frac{6nk}{\epsilon}}{\epsilon^3},$$

where ρ is the number of spread queries we make to add one seed. Hence the total number of spread queries of our algorithm is

$$r_\epsilon^{n,k} = k\rho_\epsilon^{n,k} = \frac{81k^2 \log \frac{6nk}{\epsilon}}{\epsilon^3}.$$

The main idea is to use the identity of adopters from $k\rho = \tilde{O}(k^2)$ independent cascades to seed k nodes with optimal approximation guarantees. Nodes that appear most in different cascades are the best candidates for seeding. For example, in Figure 3 the black node is the only node that appears in all three cascades, each starting from a random initial node. To prevent overlap with already chosen seeds, we discard those cascades that intersect with the chosen seeds.

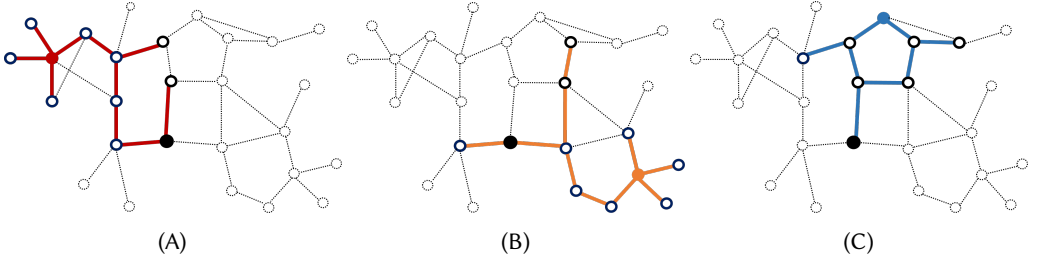


Fig. 3. Three cascades are depicted in (A) red, (B) orange, and (C) blue. In each case, the initial node is marked in the same color as the cascade. The node that appears most across different cascades is marked in black. The dotted sections consist of unrealized influences and nodes that do not adopt in each cascade.

Algorithm 3: SPREAD (ρ)

Input: Spread query access to graph \mathcal{G} with independent cascade probability p

Output: Λ^* , approximate seed set of size k with value at least $(1 - 1/e)L - \epsilon n$

For i form 1 to k , do

- (1) **SAMPLE:** Choose ρ nodes uniformly at random and call them u_1^i, \dots, u_ρ^i .
- (2) **SPREAD:** Run the spreading process ρ times, each time with one of the sampled nodes (u_j^i) seeded. Call the resultant set of adopters A_1^i, \dots, A_ρ^i . For any $j = 1, \dots, \rho$, if $A_j^i \cap \Lambda^* \neq \emptyset$, set $A_j^i = \emptyset$.
- (3) **SEED:** For each node $u \in \mathcal{V} \setminus \Lambda^*$ and $j = 1, \dots, \rho$, set $X_{u,j}^i = 1$ if $u \in A_j^i$ and $X_{u,j}^i = 0$, otherwise. Set $X_u^i = \sum_{j=1}^{\rho} X_{u,j}^i$. Choose $v^* = \operatorname{argmax}_{u \in \mathcal{V} \setminus \Lambda^*} X_u^i$. Add $\Lambda^* \leftarrow \Lambda^* \cup \{v^*\}$.

Algorithm 3: SPREAD (ρ) follows [12, Algorithm 1] but is adapted to undirected networks. To work with directed graphs, the spread queries in [12, Algorithm 1] are performed on the transposed graph where the direction of influences is reversed. Such queries serve their purpose by summarizing the graph information for fast influence maximization. However, our interest is in queries as a costly method of acquiring network information. For our purpose, running spread queries in a transposed graph is hard to motivate since one needs to conceive a mechanism to implement cascades in reverse (see Appendix B for a discussion of extensions to directed graphs).

By focusing on undirected graphs, we can propose an algorithm for influence maximization using spread queries on the original graph (as opposed to its transpose). Moreover, by allowing for an ϵn additive loss in our approximation guarantee, we only need $O_\epsilon(k^2 \log(n)) = o(n)$ spread queries, whereas [12, Algorithm 1] needs $\Theta_\epsilon(kn \log n) = \omega(n)$ spreads to achieve the nearly optimal approximation factor $(1 - 1/e - \epsilon)$. Achieving an approximation guarantee with the fewer than $o(n)$ queries is important in our setup where acquiring network information is costly. More recently, Sadeh et al. [51] provide a significantly improved sample complexity bound of $O(k\tau \log n)$ if the diffusion is stopped after τ steps (e.g., in time-constrained applications or small-world networks).

In the next theorem, we formalize our approximation guarantee and bound on the query complexity for Algorithm 3: SPREAD (ρ). The proof is in Appendix A.9. They crux of the argument is in realizing that with X_u^i defined in step (3), $(n/\rho)\mathbb{E}[X_u^i]$ is the expected marginal gain from adding node u to Λ^{*i-1} — the seed set in step i . Therefore, we can approximate one step of the greedy algorithm by choosing $v^* \in \mathcal{V} \setminus \Lambda^{*i-1}$ to maximize X_u^i while using concentration to control the deviation of X_u^i from $\mathbb{E}[X_u^i]$.

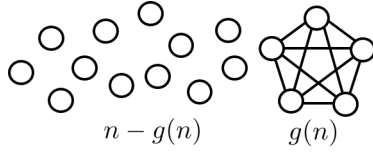


Fig. 4. Using $f(n) \in o(n)$ spread queries on a graph comprised of a clique of size $g(n) = \sqrt{n/f(n)}$ and $n - g(n)$ isolated nodes, one cannot achieve an approximation factor that is better than $o(1)$.

THEOREM 11. *For any arbitrary $0 < \epsilon \leq 1$, there exist a polynomial-time algorithm for influence maximization that covers $(1 - 1/e)L - \epsilon n$ nodes in expectation in $\tilde{O}_\epsilon(nk^2)$ time, using no more than $81k^2 \log(6nk/\epsilon)/\epsilon^3 \in \tilde{O}(k^2)$ spread queries.*

Next we show that an additive loss on the quality of the solution is inevitable with $o(n)$ spread queries in the worst case. The proof details are in Appendix A.10. Our hard example consist of a graph with a small clique and many isolated nodes (see Figure 4). In such a structure, using $o(n)$ spread queries one cannot achieve better than an $o(1)$ approximation factor. The hard example used in this impossibility result is similar to that of Wilder et al. [60, Theorem 1]; however, with a more careful analysis, we improve their $O(n^{1-\epsilon})$ bound to $o(n)$.

THEOREM 12. *Let μ be any constant. There is no μ -approximation algorithm for influence maximization using $o(n)$ spread queries.*

5 THE VALUE OF NETWORK INFORMATION: JUST A FEW QUERIES

We can study the value of network information by considering how the expected spread size from seeding the output of our algorithm increases with increasing number of input queries. Our simulations of spread sizes with increased queries on real networks indicate the existence of an inflection point, whereby the first few queries improve the performance significantly before hitting a plateau – Figures 6C and 7A. Therefore, we can extract the benefits of the network

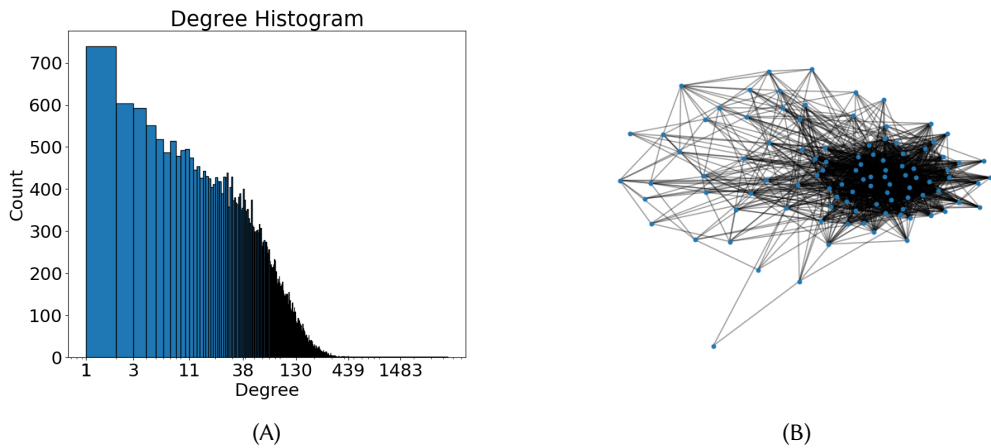


Fig. 5. (A) The degree distribution of the University of Pennsylvania Facebook social network with 41, 536 nodes, average degree 65.59, and a total of 1, 362, 220 edges (B) The connections between the one hundred highest degree nodes in this network

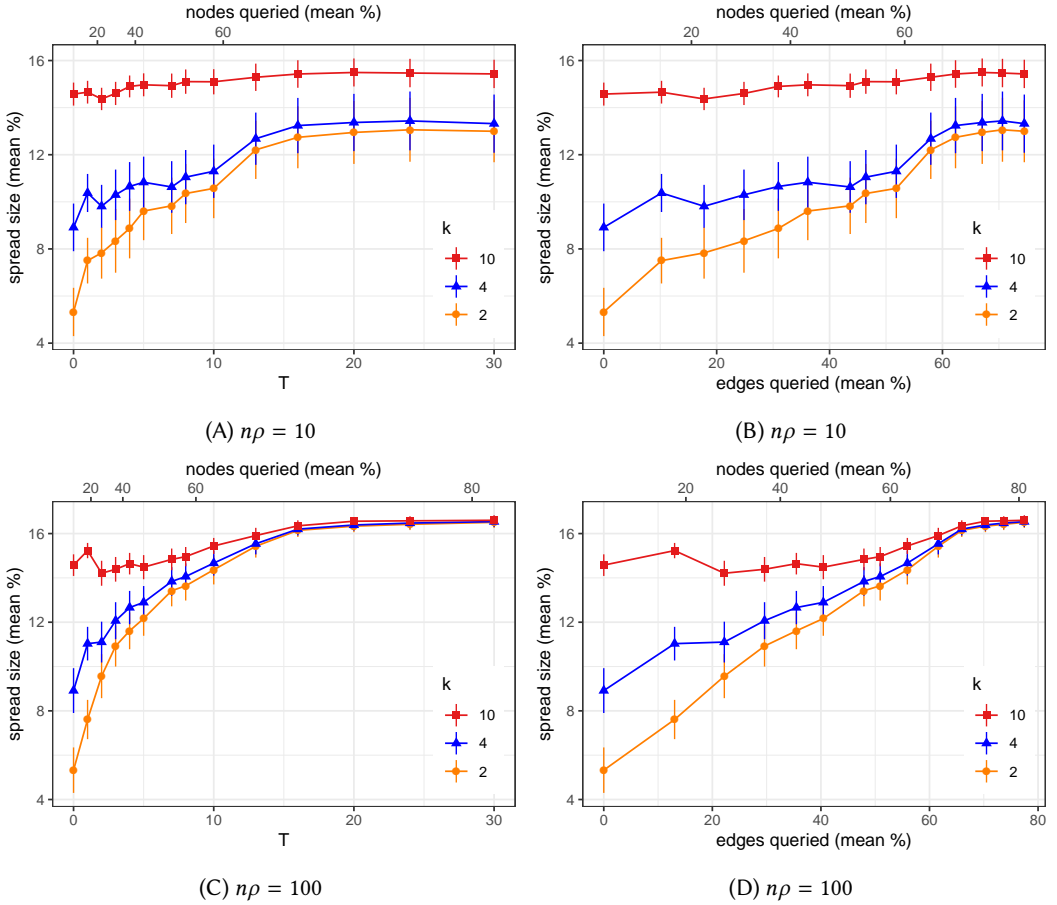


Fig. 6. The spread sizes from seeding the output of Algorithm 2 applied to the University of Pennsylvania Facebook social network. To estimate the influence of each output seed set, we average the spread sizes over 500 independent cascades with $p = 0.01$. To generate the T copies $\mathcal{G}_{\rho, \epsilon}^{(1)}, \dots, \mathcal{G}_{\rho, \epsilon}^{(T)}$ that are input to Algorithm 2, we run Algorithm 1 (PROBE) starting from np initial nodes (randomly chosen). We test the performance for two values of $np = 10$, (A and B), and $np = 100$, (C and D). We vary T over a logarithmic scale: $T = 0, 1, 2, 3, 4, 5, 7, 8, 10, 13, 16, 20, 24, 30$. Note that the $T = 0$ case corresponds to random seeding (using no network information at all). For each T , we run the PROBE Algorithm 50 times to generate 50 random inputs for Algorithm 2. The vertical axes show the mean spread sizes and confidence intervals that are computed over the 50 outputs of Algorithm 2 for each T . Figures (A) and (C) show the performance with increasing T for 10 and 100 initial nodes, respectively. Their top axes show the average number of revealed nodes that is computed over the 50 random inputs for each T . Figures (B) and (D) show the mean spread sizes versus the average number of nodes (top axis) and edges (bottom axes) that are revealed in the input at each T .

information using just a few spread queries. We present our simulation results on the University of Pennsylvania Facebook social network (Figure 5) that is the largest network in a dataset of Facebook social networks in 100 U.S. universities and colleges [55].

In Figure 6, we show the performance of Algorithms 1 and 2 (PROBE & SEED) on the University of Pennsylvania Facebook social network. Running the PROBE algorithm with higher values of T

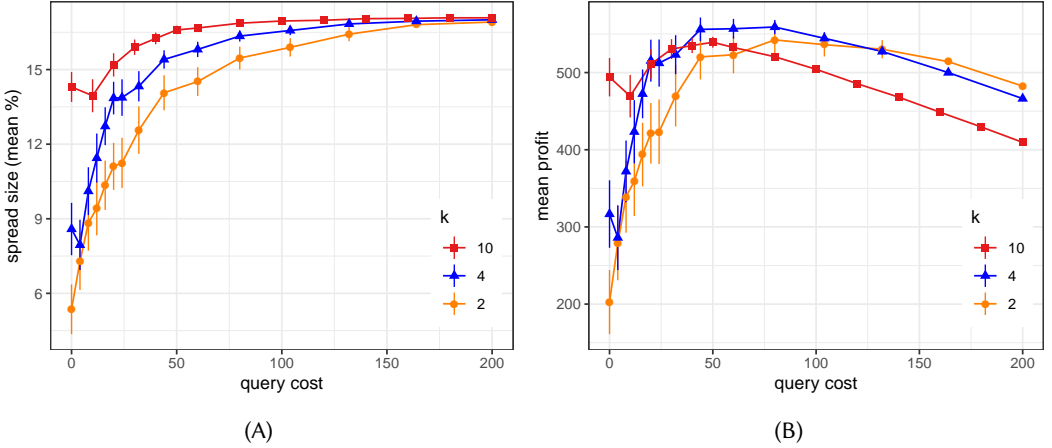


Fig. 7. (A) The spread sizes over the University of Pennsylvania Facebook social network ($p = 0.01$) from seeding the output of Algorithm 3 with different number of spread queries in the input (the query cost). The mean spread sizes and their confidence intervals are computed over 50 executions of the algorithm at each query cost. For each seed set that the algorithm outputs, we run the spread 500 times to estimate its influence (expected spread size). In (B) we assume a cost $c_s = 10$ per seed and another cost $c_q = 1$ per query, as well as a revenue $r = 0.1$ per each adopter. The vertical axis shows the mean profits and confidence intervals that are computed from 50 executions of the algorithm at each query cost. Note that the maximum expected profit is achieved at $k = 4$ with 44 spread queries.

leads to discovery of more nodes and edges from the social network. The top and bottom horizontal axes in Figures 6B and 6D show the average number of nodes and edges that are revealed in the input for different values of T , with 10 and 100 initial nodes respectively. The vertical axes show the mean spread sizes from seeding the output of Algorithm 2 for each T using 50 random inputs. Recall that each input is a set of T probed samples $\mathcal{G}_{\rho, \epsilon}^{(1)}, \dots, \mathcal{G}_{\rho, \epsilon}^{(T)}$ that is obtained through Algorithm 1. The output performance improves with increasing T , since with more nodes and edges that are revealed from the social network, the output seed set can be better optimized; nevertheless, there are diminishing returns to the increasing network information. It is worth noting that the randomness in the algorithm output also decreases with the increasing network information in the input. There are two sources of randomness in the algorithm's output: $T < \infty$ and $\rho < 1$. The output variance for large T remains non-vanishing in Figure 6A; however, increasing the number of initial nodes, i.e. the size of the sample set \mathcal{V}_ρ in Subsection 3.1, from 10 nodes in Figure 6A to 100 nodes in Figure 6C allows us to remove the remnant randomness from the algorithm output at large T .

In Figure 7, we show the mean spread sizes from seeding the output of Algorithm 3 with increasing number of input queries. Figure 7A shows that there are diminishing returns to increasing queries, and we can extract the benefits of complete network information using just 200 spread queries: With enough information, the mean spread size from seeding the output of the algorithm saturates at the complete-information (deterministic) greedy algorithm output, and acquiring more network information does not improve the performance beyond that. The number of queries before each graph hits its inflection point is higher for larger seeds sets; however, in general, *one can extract the benefits of complete network information with just a few queries*. Subsequently, if we assume a cost for each seeded node and another cost for running each spread query, as well as a revenue per each unit spread, then there is a number of queries that is profit maximizing for a given number of

seeds (Figure 7B). In Figure 7B, the maximum expected profit at $k = 4$ is achieved with 44 queries and it dominates the optimum for other smaller or larger seed sets ($k = 2$ or $k = 10$).

Akbarpour et al. [1] show that seeding $k + x$ nodes at random (using no information about the network) for some $x = \omega(1)$ is enough to outperform the optimum spread size with k nodes, as the network size increases ($n \rightarrow \infty$). They conclude that the benefits of acquiring network information to identify the optimal k seeds may be offset by seeding a few more nodes at random (without using any network information). Comparing the $k = 10$ and $k = 2$ plots in Figure 7A reinforces this conclusion: The benefit of the added information is less for larger seed sets. On the one hand, we complement the results of [1] by measuring the required number of queries to guarantee the same expected cascade sizes that is efficiently achievable using full knowledge of the network structure. Indeed, in our framework, we can make this trade-off explicit by seeding more nodes and reducing the number of queries to keep the performance fixed. On the other hand, the results of [1] are average-case over the random graph inputs and limited to cases where the cascade size is greater than $o(n)$. However, our algorithms have the guaranteed performance over arbitrary graph inputs with any cascade size.

In another related work, Manshadi et al. [44] study a model of spread where individuals contact their neighbors independently at random, and each contact leads to an adoption with some fixed probability. The contacts occur repeatedly; therefore, every cascade eventually spreads to the entire population. They characterize the time to reach a fraction of adopters as well as the contact cost (number of contacts made), in a random graph with a given degree distribution. They also propose optimal seeding strategies that only use the degree information. However, this model is not directly comparable to the influence maximization setup that we study. In our model, the realization of the influences is random and adoption spreads only through the realized edges. For us, the objective is to maximize the expected spread size and the incurred cost is in acquiring information about the influence structure (who influences whom).

6 DISCUSSION, OPEN PROBLEMS AND FUTURE DIRECTIONS

We consider the problem of choosing the k most influential nodes under the independent cascade model and using partial network information. We propose two natural ways of querying the graph structure: (i) by accessing an unordered adjacency list of the input graph — edge queries, (ii) by observing the identity of the adopter nodes when a single node is seeded — spread queries. In each case, we provide polynomial-time algorithms with almost tight approximation guarantees using a bounded number of queries to the graph structure (Theorems 10 and 11). We also provide impossibility results to lower bound the query complexity and show tightness of our guarantees (Theorems 9 and 12). Finally, we show the utility of the bounded-query framework for studying the trade-off between the cost of acquiring more network information and the benefit of increasing the spread size. Results that address the problem of seeding with partial network information are nascent and we foresee many directions for future research in this area.

Theorem 12 implies that with $o(n)$ edge queries an additive loss is inevitable in the edge query model. An interesting open problem is to provide a better result when edge queries exceed $o(n)$; either showing that the ϵn additive loss is unavoidable, even with the increased number of edge queries, or an algorithm that avoids it. Another interesting open problem is to provide tight lower bounds on the query complexity of the spread query model. An $\Omega(k)$ lower bound is easy to show — for a hard example consider a collection of k stars of size n/k each. We speculate that a lower-bound closer to k^2 is plausible but the proof would involve significant new ideas. We presented all of our results for the independent cascade model over undirected graphs with the same cascade probability (p) on every edge. In Appendix B, we discuss the extension of our results to the case of asymmetric influences (directed, weighted graphs). In Appendix C, we explain how our techniques can be

applied to other commonly posited models of diffusion, and in particular, provide the following extension to the linear threshold model of Kempe et al. [35]:

THEOREM 13. *For any arbitrary $0 < \epsilon \leq 1$, there exist a polynomial-time algorithm for influence maximization under the linear threshold model that covers $(1 - 1/e)L - \epsilon n$ nodes in expectation in $\tilde{O}_\epsilon(nk^2)$ time, using no more than $81nk^2 \log(6nk/\epsilon)/\epsilon^3 \in \tilde{O}(nk^2)$ edge queries.*

It is possible to provide tighter approximation guarantees or better query bounds by assuming that the inputs follow a known distribution. For example, Wilder et al. [60] propose an algorithm for stochastic block model inputs, and it consists of taking a random sample of T nodes and exploring their extended neighborhoods in R steps of a random walk. The outcome of the random walks is used to estimate the block sizes of each of the T nodes, and this is achieved by revealing no more than $TR = O(\log^6 n)$ nodes. The k seeds are then selected from the initial T samples, such that the k largest blocks are seeded uniformly at random.

In our queries we make use of the spread process, over which we optimize. Specifically, our queries depend on the independent cascade probability p . In practice, the spread process may not be available for querying the influence structure. For example, if a person is asked to reveal her influencers (during an edge query), she may reveal them with a probability p' that is different from the cascade probability p . Similarly, coupons (during a spread query) may have a different cascade probability p' compared to the seeds, whose locations we want to optimize. Therefore, it is interesting to know how the discrepancy between the parameter of the query method p' and the independent cascade probability p affects the influence maximization performance. One may offer new approximation guarantees that either depend explicitly on $|p - p'|$, or hold true when this difference is bounded ($|p - p'| < \epsilon$). Such results can complement the prior literature on stable and robust influence maximization (cf. [27, 31]).

Another venue for future work is to explore other ways of querying the graph structure. We can draw inspiration from the graph sampling literature (cf., e.g., [42, 49, 62]) to devise new query methods. Accordingly, one would like to obtain subsampled graphs that preserve enough network information for performing influence maximization in a satisfactory manner. We are particularly interested in queries that reveal the realization of influences over time. One can use the temporal data to complement the influence information. This is especially relevant in practice, where a slow reaction time can compromise the influence of an otherwise well-positioned node and decision-makers have preferences for earlier, rather than later, adoption [43].

A PROOFS

A.1 Proof of Lemma 3

Starting with an approximate solution satisfying $\Gamma_\delta(\Lambda'_\delta) \geq \alpha L_\delta - \beta n$ on the one hand, we have

$$\Gamma(\Lambda'_\delta) + \epsilon n \geq \Gamma_\delta(\Lambda'_\delta), \quad (5)$$

since $|\Gamma_\delta(\Lambda'_\delta) - \Gamma(\Lambda'_\delta)| \leq \epsilon n$.

On the other hand, consider Λ_δ and Λ , which are the optimum seed sets for Γ_δ and Γ , respectively. By assumption we have $\Gamma_\delta(\Lambda'_\delta) \geq \alpha \Gamma_\delta(\Lambda_\delta) - \beta n$, and since, by optimality of Λ_δ for Γ_δ , $\Gamma_\delta(\Lambda_\delta) \geq \Gamma_\delta(\Lambda)$, we get

$$\Gamma_\delta(\Lambda'_\delta) \geq \alpha \Gamma_\delta(\Lambda) - \beta n \geq \alpha \Gamma(\Lambda) - (\beta + \alpha \epsilon)n \quad (6)$$

where, in the last inequality, we have again invoked the $|\Gamma_\delta(\Lambda) - \Gamma(\Lambda)| \leq \epsilon n$ property. The proof is complete upon combining (5) and (6) to get that

$$\Gamma(\Lambda'_\delta) \geq \alpha \Gamma(\Lambda) - (\beta + (\alpha + 1)\epsilon)n.$$

A.2 Proof of Lemma 4

Fix a seed set S and let

$$\frac{(2 + \epsilon)(\delta' + \log 2)}{(2n\epsilon^2)} \leq \rho \leq 1.$$

We use a Hoeffding-Bernstein bound to claim that with probability at least $1 - e^{-\delta'}$ we have

$$|\Gamma_\rho(\mathcal{S}) - \Gamma(\mathcal{S})| \leq \epsilon n. \quad (7)$$

Let X_v be the random variable that is zero if v is not in \mathcal{V}_ρ and $\phi(v, \mathcal{S})$ otherwise. Consider their summation and note that

$$\sum_{v \in \mathcal{V}} X_v = \sum_{v \in \mathcal{V}_\rho} \phi(v, \mathcal{S}) = \rho \Gamma_\rho(\mathcal{S}).$$

Hoeffding-Bernstein inequality [56, Lemma 2.14.19] provides that

$$\begin{aligned} \mathbb{P} \left[\left| \frac{1}{n\rho} \sum_{v \in \mathcal{V}_\rho} \phi(v, \mathcal{S}) - \frac{1}{n} \Gamma(\mathcal{S}) \right| \geq \epsilon \right] &= \mathbb{P} [|\Gamma_\rho(\mathcal{S}) - \Gamma(\mathcal{S})| \geq \epsilon n] \\ &\leq 2 \exp \left(-\frac{2n\rho\epsilon^2}{2\sigma_n^2 + \epsilon\Delta_n} \right), \end{aligned} \quad (8)$$

where $\Delta_n = \max_{v \in \mathcal{V}} \phi(v, \mathcal{S}) - \min_{v \in \mathcal{V}} \phi(v, \mathcal{S}) \leq 1$ and

$$\begin{aligned} \sigma_n^2 &= \frac{1}{n} \sum_{v \in \mathcal{V}} \left(\phi(v, \mathcal{S}) - \frac{1}{n} \Gamma(\mathcal{S}) \right)^2 \\ &= \frac{1}{n} \sum_{v \in \mathcal{V}} \phi(v, \mathcal{S})^2 - \left(\frac{1}{n} \Gamma(\mathcal{S}) \right)^2 \\ &\leq \frac{1}{n} \sum_{v \in \mathcal{V}} \phi(v, \mathcal{S}) - \left(\frac{1}{n} \Gamma(\mathcal{S}) \right)^2 \\ &= \ell - \ell^2 \leq \ell \leq 1. \end{aligned}$$

In the last equality, we used the notation $\ell := (1/n) \sum_{v \in \mathcal{V}} \phi(v, \mathcal{S}) = (1/n) \Gamma(\mathcal{S})$. The bound in (8) subsequently simplifies

$$\mathbb{P} [|\Gamma_\rho(\mathcal{S}) - \Gamma(\mathcal{S})| \geq \epsilon n] \leq 2 \exp \left(-\frac{2n\rho\epsilon^2}{2 + \epsilon} \right).$$

Using $n\rho \geq (2 + \epsilon)(\delta' + \log 2)/2\epsilon^2$, we get that for all $\delta' > 0$

$$\mathbb{P} [|\Gamma_\rho(\mathcal{S}) - \Gamma(\mathcal{S})| \geq \epsilon n] \leq 2 \exp(-(\delta' + \log 2)) = e^{-\delta'}. \quad (9)$$

To complete the proof we use a union bound to claim that (7) holds for all choices of the seed set \mathcal{S} simultaneously. To claim a union bound over all $\binom{n}{k}$ choices of the seed sets \mathcal{S} , it suffices to choose $\delta' = k\delta \log n$ in (9).

A.3 Proof of Lemma 5

Consider $\Gamma_\rho^{(T)}(\mathcal{S}) = (1/\rho) \sum_{v \in \mathcal{V}_\rho} \phi^{(T)}(v, \mathcal{S})$ for a fixed $\mathcal{S} \subset \mathcal{V}$. By Chernoff bound to $\phi^{(T)}(v, \mathcal{S}) = 1/T \sum_{i=1}^T Y^{(i)}(v, \mathcal{S})$, we get that:

$$\mathbb{P} \left[\left| \phi^{(T)}(v, \mathcal{S}) - \phi(v, \mathcal{S}) \right| > \epsilon \right] \leq 2 \exp(-\epsilon^2 T/3).$$

Using $T = T_{\epsilon, \delta}^{n, k}$, by union bound over the choice of $\binom{n}{k}$ seed sets $\mathcal{S} \subset \mathcal{V}$, $\text{card}(\mathcal{S}) = k$, and n nodes $v \in \mathcal{V}$, we obtain that:

$$\mathbb{P} \left[\left| \phi^{(T)}(v, \mathcal{S}) - \phi(v, \mathcal{S}) \right| > \epsilon, \text{ for all } \mathcal{S} \text{ and } v \right] \leq 2 \exp(-\delta - \log 2) = e^{-\delta}.$$

The proof is complete upon considering the summation over $v \in \mathcal{V}_\rho$:

$$\begin{aligned} & \mathbb{P} \left[\left| \Gamma_\rho^{(T)}(\mathcal{S}) - \Gamma_\rho(\mathcal{S}) \right| \leq \epsilon n, \text{ for all } \mathcal{S} \right] = \\ & \mathbb{P} \left[\left| \sum_{v \in \mathcal{V}_\rho} \phi^{(T)}(v, \mathcal{S}) - \sum_{v \in \mathcal{V}_\rho} \phi(v, \mathcal{S}) \right| \leq \epsilon n \rho, \text{ for all } \mathcal{S} \right] \geq \\ & \mathbb{P} \left[\left| \phi^{(T)}(v, \mathcal{S}) - \phi(v, \mathcal{S}) \right| \leq \epsilon, \text{ for all } \mathcal{S} \text{ and } v \right] \geq 1 - e^{-\delta}. \end{aligned}$$

A.4 Proof of Lemma 6

Following the notation in Definition 1, let us use $\Lambda_\rho^{(T)}$ and $L_\rho^{(T)}$ to denote the maximizer of $\Gamma_\rho^{(T)}$ and its maximal value subject to the size constraint ($\text{card}(\Lambda_\rho^{(T)}) = k$). Similarly, let us denote the optimal solution to k -IM on $\Gamma_{\rho, \epsilon}^{(T)}$ and its value by $\Lambda_{\rho, \epsilon}^{(T)}$ and $L_{\rho, \epsilon}^{(T)}$, respectively. Moreover, following Definition 2, let us use $\Lambda_\rho^{\alpha, (T)}$ and $\Lambda_{\rho, \epsilon}^{\alpha, (T)}$ to denote the α -approximate solutions to k -IM on $\Gamma_\rho^{(T)}$ and $\Gamma_{\rho, \epsilon}^{(T)}$, respectively. Our goal is to show that any $\Lambda_{\rho, \epsilon}^{\alpha, (T)}$ is also $\Lambda_\rho^{\alpha(1-\epsilon), (T)}$.

It is useful to think of $\mathcal{G}_{\rho, \epsilon}^{(1)}, \dots, \mathcal{G}_{\rho, \epsilon}^{(T)}$ as subgraphs of $\mathcal{G}_\rho^{(1)}, \dots, \mathcal{G}_\rho^{(T)}$. An immediate consequence of this observation is that for any set of nodes \mathcal{S} , we have $\Gamma_\rho^{(T)}(\mathcal{S}) \geq \Gamma_{\rho, \epsilon}^{(T)}(\mathcal{S})$. We call the imaginary process whereby $\mathcal{G}_{\rho, \epsilon}^{(i)}$ is obtained after removing some nodes and edges from $\mathcal{G}_\rho^{(i)}$ an ϵ -cutting, and subsequently, we refer to $\mathcal{G}_{\rho, \epsilon}^{(i)}$ and $\mathcal{G}_\rho^{(i)}$ as the cut and uncut copies, respectively. Finally, it is also useful to define $\phi_\epsilon^{(T)}(v, \mathcal{S})$ in the exact same way as (2) but using the ϵ -cut copies $\mathcal{G}_{\rho, \epsilon}^{(1)}, \dots, \mathcal{G}_{\rho, \epsilon}^{(T)}$.

The proof follows [10, Lemma 2.4] closely. In particular, we first note that it suffices to show the existence of a set \mathcal{L} , $\text{card}(\mathcal{L}) = k$ satisfying $\Gamma_{\rho, \epsilon}^{(T)}(\mathcal{L}) \geq (1 - \epsilon)L_\rho^{(T)}$. Because if there exists such a set \mathcal{L} , then for any α -approximate solution $\Lambda_{\rho, \epsilon}^{\alpha, (T)}$ we can write (recall $\forall \mathcal{S} \Gamma_\rho^{(T)}(\mathcal{S}) \geq \Gamma_{\rho, \epsilon}^{(T)}(\mathcal{S})$):

$$\Gamma_\rho^{(T)}(\Lambda_{\rho, \epsilon}^{\alpha, (T)}) \geq \Gamma_{\rho, \epsilon}^{(T)}(\Lambda_{\rho, \epsilon}^{\alpha, (T)}) \geq \alpha \Gamma_{\rho, \epsilon}^{(T)}(\Lambda_{\rho, \epsilon}^{(T)}) \geq \alpha \Gamma_{\rho, \epsilon}^{(T)}(\mathcal{L}) \geq (1 - \epsilon) \alpha L_\rho^{(T)},$$

implying that $\Lambda_{\rho, \epsilon}^{\alpha, (T)}$ is also $\Lambda_\rho^{\alpha(1-\epsilon), (T)}$. To show the existence of such a set \mathcal{L} we use a probabilistic argument by constructing a random set \mathbf{L} , satisfying $\mathbb{E} \left\{ \Gamma_{\rho, \epsilon}^{(T)}(\mathbf{L}) \right\} \geq (1 - \epsilon)L_\rho^{(T)}$. The set \mathbf{L} is constructed as follows: Starting from $\Lambda_\rho^{(T)}$, remove ϵk nodes randomly, and replace them with ϵk

nodes chosen uniformly at random from \mathcal{V} . To see why $\mathbb{E} \left\{ \Gamma_{\rho, \epsilon}^{(T)}(\mathbf{L}) \right\} \geq (1 - \epsilon)L_{\rho}^{(T)}$, consider

$$L_{\rho}^{(T)} = \frac{1}{\rho} \sum_{v \in \mathcal{V}_{\rho}} \phi^{(T)}(v, \Lambda_{\rho}^{(T)}), \text{ and } \mathbb{E} \left\{ \Gamma_{\rho, \epsilon}^{(T)}(\mathbf{L}) \right\} = \sum_{v \in \mathcal{V}_{\rho}} \mathbb{E} \left\{ \phi_{\epsilon}^{(T)}(v, \mathbf{L}) \right\}.$$

The inequality, $\mathbb{E} \left\{ \Gamma_{\rho, \epsilon}^{(T)}(\mathbf{L}) \right\} \geq (1 - \epsilon)L_{\rho}^{(T)}$, would follow if for any node $v \in \mathcal{V}$ we have,

$$\mathbb{E} \left\{ \phi_{\epsilon}^{(T)}(v, \mathbf{L}) \right\} \geq (1 - \epsilon)\phi^{(T)}(v, \Lambda_{\rho}^{(T)}) + \epsilon \geq \phi^{(T)}(v, \Lambda_{\rho}^{(T)}).$$

It only remains to verify the truth of the former inequality, $\mathbb{E} \left\{ \phi_{\epsilon}^{(T)}(v, \mathbf{L}) \right\} \geq (1 - \epsilon)\phi^{(T)}(v, \Lambda_{\rho}^{(T)}) + \epsilon$.

First note that $\mathbb{E} \left\{ \phi_{\epsilon}^{(T)}(v, \mathbf{L}) \right\}$ represents the probability of node v being connected to one of the nodes in the random set \mathbf{L} averaged over the T copies $\mathcal{G}_{\rho, \epsilon}^{(1)}, \dots, \mathcal{G}_{\rho, \epsilon}^{(T)}$. Consider each of the T copies in our uncut sketch, $\mathcal{G}_{\rho}^{(1)}, \dots, \mathcal{G}_{\rho}^{(T)}$, and the connections between node v and the optimal set $\Lambda_{\rho}^{(T)}$ in these uncut copies. If these connections remain unchanged in the ϵ -cut copies $\mathcal{G}_{\rho, \epsilon}^{(1)}, \dots, \mathcal{G}_{\rho, \epsilon}^{(T)}$, then with probability at least $(1 - \epsilon)$ they remain unchanged after ϵk nodes in $\Lambda_{\rho}^{(T)}$ are randomly replaced. If, however, any of these connections are affected by the ϵ -cutting, then this is an indication that v belongs to a connected component of size $\tau_{\epsilon}^{n, k}$. This connected component is large enough to contain one of the ϵk random nodes of \mathbf{L} with probability at least ϵ . Indeed, the probability that none of the $\tau_{\epsilon}^{n, k}$ nodes is chosen is at most ϵ :

$$\left(1 - \frac{\tau_{\epsilon}^{n, k}}{n} \right)^{\epsilon k} = \left(1 - \frac{\log(1/\epsilon)}{\epsilon k} \right)^{\epsilon k} \leq e^{-\log(1/\epsilon)} = \epsilon.$$

A.5 Proof of Theorem A.5

Following the notation in the proof of Lemma 6 (Appendix A.4), consider any $\Lambda_{\rho, \epsilon}^{\alpha, (T)}$. Lemma 6 implies that $\Lambda_{\rho, \epsilon}^{\alpha, (T)}$ is also $\Lambda_{\rho}^{(1-\epsilon)\alpha, (T)}$, as the loss in approximation factor from limited probing is at most $(1 - \epsilon)$. Next note that Lemma 5, together with Lemma 3, implies that for $T = T_{\epsilon, \delta}^{n, k}$ with probability at least $1 - e^{-\delta}$, the value of $\Lambda_{\rho}^{(1-\epsilon)\alpha, (T)}$ for Γ_{ρ} can be lower bounded as follows: $\Gamma_{\rho}(\Lambda_{\rho}^{(1-\epsilon)\alpha, (T)}) \geq (1 - \epsilon)\alpha L_{\rho} - ((1 - \epsilon)\alpha + 1)\epsilon n$. Finally, another application of Lemma 3 with Lemma 4 yields that with at least $1 - e^{-\delta}$ probability, $\Gamma(\Lambda_{\rho}^{(1-\epsilon)\alpha, (T)}) \geq (1 - \epsilon)\alpha L - 2((1 - \epsilon)\alpha + 1)\epsilon n$. The proof is complete upon combining the preceding statements to get that, with total probability at least $1 - 2e^{-\delta}$, $\Gamma(\Lambda_{\rho, \epsilon}^{\alpha, (T)}) \geq (1 - \epsilon)\alpha L - 2((1 - \epsilon)\alpha + 1)\epsilon n$.

A.6 Proof of Theorem 8

In the first step, we bound the total number of edges used in our sketch, i.e. the T copies $\mathcal{G}_{\rho, \epsilon}^{(1)}, \dots, \mathcal{G}_{\rho, \epsilon}^{(T)}$. Let us also denote the set of all edges that appear in our sketch by \mathcal{E}_T . Fix a choice of τ nodes in one of the copies. Let \mathbf{X} be the number of edges between these τ nodes. Note that \mathbf{X} is a random variable and its distribution is fixed by PROBE (ρ, T, τ) . Using the Chernoff upper-tail and the fact that $\mathbb{E}[\mathbf{X}] \leq p \binom{\tau}{2}$, we can upper-bound \mathbf{X} as follows:

$$\begin{aligned} \mathbb{P} \left[\mathbf{X} \geq p\tau(\tau - 1)/2 + \delta' \sqrt{p\tau(\tau - 1)/2} \right] &\leq \mathbb{P} \left[\mathbf{X} \geq p\mathbb{E}[\mathbf{X}] + \delta' \sqrt{p\mathbb{E}[\mathbf{X}]} \right] \\ &\leq e^{-\delta'^2/4}. \end{aligned} \tag{10}$$

Recall from (4) that $E_{\epsilon,p}^{n,k} = p\tau_\epsilon^{n,k}(\tau_\epsilon^{n,k} - 1)/2$. Setting $\tau = \tau_\epsilon^{n,k}$ and

$$\delta' = 2\sqrt{\delta(\tau_\epsilon^{n,k} \log(n) + \log T)} \geq 2\sqrt{\delta \log\left(T \binom{n}{\tau_\epsilon^{n,k}}\right)},$$

in (10) is enough to ensure that, by union bound, with probability at least $1 - e^{-\delta}$ for any subset of size $\tau_\epsilon^{n,k}$ in all of the T copies, we have $X \leq \bar{X}$, where

$$\bar{X} := E_{\epsilon,p}^{n,k} + \sqrt{\delta(\tau_\epsilon^{n,k} \log(n) + \log T)E_{\epsilon,p}^{n,k}}. \quad (11)$$

Next note that starting from any of the $n\rho$ nodes in \mathcal{V}_ρ we never hit more than τ nodes following the limited probing procedure – see step (2-c) of the PROBE (ρ, T, τ) algorithm. Hence, the total number of components with size τ in our sketch is always less than $n\rho T$, and given (11), we can bound the total number of edges in the T copies by $n\rho T\bar{X}$. More precisely, with probability at least $1 - e^{-\delta}$, we have:

$$\text{card}(\mathcal{E}_T) \leq n\rho T\bar{X} = C_{\epsilon,\delta}^{n,k}, \quad (12)$$

where $\rho = \rho_{\epsilon,\delta}^{n,k}$, $T = T_{\epsilon,\delta}^{n,k}$, $\tau = \tau_\epsilon^{n,k}$, and $C_{\epsilon,\delta}^{n,k}$ is defined in (4).

Next note that some edges may have been queried (reported to the surveyor), but not appear in \mathcal{E}_T . This can happen for an edge e in a copy $\mathcal{G}_{\rho,\epsilon}^{(i)}$ as follows: Such an edge would have been reported by a newly probed node v to an already probed node u . Since this edge has already got its one chance of appearing in $\mathcal{G}_{\rho,\epsilon}^{(i)}$ when u was being probed, it is discarded after being reported as an edge by v – see step (2-b) of the PROBE algorithm. We can bound the number of such edges in each copy as follows. Let $A_e^{(i)}$ be the indicator variable for the event that both nodes incident to edge e are probed; let $B_e^{(i)}$ be the indicator that edge e is reported (i.e., queried) on its second chance, i.e. when the second of the two nodes incident to e is probed. Finally, let $C_e^{(i)}$ be the indicator that edge e is reported when the second of its two incident nodes is probed, conditioned on both of its incident nodes being probed (i.e., $B_e^{(i)}$ conditioned on $A_e^{(i)} = 1$). The edges e for which $B_e^{(i)} = 1$, are those which are queried but do not appear in $\mathcal{G}_{\rho,\epsilon}^{(i)}$. In (12) we bound the total number of edges belonging to \mathcal{E}_T , i.e. the edges that are queried and appear in one or more of the T copies. Our next goal is to provide a complementary bound on $\sum_i \sum_e B_e^{(i)}$, thus controlling the total number of edge queries.

We begin by noting that $B_e^{(i)} = \sum_e A_e^{(i)} C_e^{(i)}$. The indicator variables $C_e^{(i)}$, $e \in \mathcal{E}$ are i.i.d. Bernoulli variables with success probability p . Using the Chernoff upper-tail bound, conditioned on the realizations of $A_e^{(i)}$ for all $e \in \mathcal{E}$, we have:

$$\begin{aligned} & \mathbb{P} \left[\sum_e A_e^{(i)} C_e^{(i)} \geq p \sum_e A_e^{(i)} + 2n\sqrt{\delta + \log T} \right] \\ & \leq \exp \left(- \frac{4n^2 (\delta + \log T)}{2 \left(\sum_e A_e^{(i)} + (n/3)\sqrt{\delta + \log T} \right)} \right) \\ & \leq \exp(-\delta - \log T) = \frac{1}{T} e^{-\delta}, \end{aligned} \quad (13)$$

where in the last inequality we have used $\sum_e A_e^{(i)} \leq n^2$, and $\sqrt{\delta + \log T} \leq n$ for $T = T_{\epsilon, \delta}^{n, k}$ and n large enough. Union bound over $i = 1, \dots, T$ provides that with probability at least $1 - e^{-\delta}$, for all i :

$$\sum_e B_e^{(i)} = \sum_e A_e^{(i)} C_e^{(i)} \leq p \sum_e A_e^{(i)} + 2n\sqrt{\delta + \log T}. \quad (14)$$

To proceed, for any edge e , let $D_e^{(i)}$ be the indicator of the event that edge e gets at least one chance to appear in $\mathcal{G}_{\rho, \epsilon}^{(i)}$, i.e. at least one of the nodes incident to e are probed. Note that, by definition, $A_e^{(i)} \leq D_e^{(i)}$ for all i and e ; hence, replacing in (14) yields:

$$\sum_e B_e^{(i)} \leq p \sum_e D_e^{(i)} + 2n\sqrt{\delta + \log T}, \quad (15)$$

with probability at least $1 - e^{-\delta}$, for all i . In the next step, let $E_e^{(i)}$ be the indicator of the event that edge e is reported on its first chance — i.e., the first time that one of its incident nodes is probed. Note that $E_e^{(i)} = 1$, $e \in \mathcal{E}$, are those edges which are queried and appear in $\mathcal{G}_{\rho, \epsilon}^{(i)}$. Hence, from (12) we have:

$$\sum_i \sum_e E_e^{(i)} = \text{card}(\mathcal{E}_T) \leq C_{\epsilon, \delta}^{n, k}, \quad (16)$$

with probability at least $1 - e^{-\delta}$. Finally, let $F_e^{(i)}$ be the indicator of the event that edge e is reported on its first chance, conditioned on at least one of its incident nodes being probed (i.e., $E_e^{(i)}$ conditioned on $D_e^{(i)} = 1$). By definition, $F_e^{(i)}$ are i.i.d. Bernoulli variables with success probability p , and $E_e^{(i)} = D_e^{(i)} F_e^{(i)}$. Similarly to (13), using a Chernoff lower-tail bound we can guarantee that, with high probability, $\sum_e E_e^{(i)} = \sum_e D_e^{(i)} F_e^{(i)}$ is not much smaller than $p \sum_e D_e^{(i)}$. Subsequently, we can upper-bound $\sum_e B_e^{(i)}$ in (15) in terms of $\sum_e E_e^{(i)}$. These details are spelled out next.

Application of the Chernoff lower-tail bound to $\sum_e E_e^{(i)} = \sum_e D_e^{(i)} F_e^{(i)}$, yields:

$$\begin{aligned} \mathbb{P} \left[\sum_e E_e^{(i)} = \sum_e D_e^{(i)} F_e^{(i)} \leq p \sum_e D_e^{(i)} - n\sqrt{2(\delta + \log T)} \right] \\ \leq \exp \left(-\frac{n^2(\delta + \log T)}{\sum_e D_e^{(i)}} \right) \leq \exp(-\delta - \log T) = \frac{1}{T} e^{-\delta}, \end{aligned}$$

where in the second inequality we use $\sum_e D_e^{(i)} \leq n^2$. Union bound over $i = 1, \dots, T$ provides that with probability at least $1 - e^{-\delta}$, for all i :

$$p \sum_e D_e^{(i)} \leq \sum_e E_e^{(i)} + n\sqrt{2(\delta + \log T)}. \quad (17)$$

Combing (15) and (17) and taking the summation over $i = 1, \dots, T$ gives that with probability at least $1 - 2e^{-\delta}$:

$$\sum_i \sum_e B_e^{(i)} \leq \sum_i \sum_e E_e^{(i)} + (2 + \sqrt{2}) T n \sqrt{\delta + \log T}. \quad (18)$$

To complete the proof, we combine (16) and (18) to get the claimed upper-bound on the total number of edge queries:

$$q = \sum_i \sum_e B_e^{(i)} + \text{card}(\mathcal{E}_T) \leq 2C_{\epsilon, \delta}^{n, k} + (2 + \sqrt{2}) T_{\epsilon, \delta}^{n, k} n \sqrt{\delta + \log T_{\epsilon, \delta}^{n, k}},$$

with probability at least $1 - 3e^{-\delta}$.

A.7 Proof of Theorem 9

Here, similar to the entire paper, we assume that the algorithm has access to the input graph's (unordered) adjacency list, and a query (v, i) asks for the i -th neighbor of node v . In this proof we set $k = 1$ and $p = 1$. Moreover, for simplicity of presentations we assume that $3/\mu$, $1/\mu^2$, and $\mu^2 n/9$ are integers. Consider the following two graphs.

- G : This graph consists of $9/\mu^2$ cliques, each of size $\mu^2 n/9$.
- G' : This graph is constructed from G via the following random process. We select $\frac{3}{\mu}$ clusters uniformly at random. Then we select one edge from each selected cluster uniformly at random. Let $(v_1, u_1), (v_2, u_2), \dots, (v_{3/\mu}, u_{3/\mu})$ be the list of the selected edges. we remove $(v_1, u_1), (v_2, u_2), \dots, (v_{3/\mu}, u_{3/\mu})$ and replace them by $(u_1, v_2), (u_2, v_3), \dots, (u_{3/\mu-1}, v_{3/\mu}), (u_{3/\mu}, v_1)$. Note that this process connects all of the selected clusters while preserving the degree distribution (see Figure 2).

Let Alg be an arbitrary (potentially randomized) algorithm for influence maximization that queries less than $(\mu^3/27)\binom{\mu^2 n/9}{2}$ edges. Note that with $k = 1$ an optimum seed on G' spreads to $\mu n/3$ nodes. Next we show that the expected spread size from seeding the output of Alg on G' is less than $\mu^2 n/3$, which means that Alg is not a μ -approximation algorithm. This implies that there is no μ -approximation algorithm that queries less than $(\mu^3/27)\binom{\mu^2 n/9}{2} \in O_\mu(n^2)$ edges as claimed.

We use the run of Alg on G to analyze the run of Alg on G' . Note that due to symmetric construction of G we can assume that Alg seeds one of the nodes of G uniformly at random. Observe that the expected spread size of a random seed in G' is

$$\left(1 - \frac{3/\mu}{9/\mu^2}\right) \cdot \frac{\mu^2 n}{9} + \frac{3/\mu}{9/\mu^2} \cdot \frac{3}{\mu} \cdot \frac{\mu^2 n}{9} \leq \frac{2\mu^2 n}{9}$$

Moreover, note that the run of algorithm Alg on G and G' are the same unless Alg queries one of the positions (i.e., edges) that we change G to construct G' . Next we upper-bound the probability that Alg queries one of the changed edges by $\mu/3$. This implies that the expected spread size is at most

$$\left(1 - \frac{\mu}{3}\right) \cdot \frac{2\mu^2 n}{9} + \frac{\mu}{3} \cdot \frac{3}{\mu} \cdot \frac{\mu^2 n}{9} \leq \frac{\mu^2 n}{3},$$

as claimed.

Now we bound the probability that Alg queries one of the changed edges. Let A_i be the random variable that indicates the number of edges Alg queries from the i 'th clique. Recall that by assumption, Alg queries less than $(\mu^3/27)\binom{\mu^2 n/9}{2}$ edges. Hence, we have $A_i < (\mu^3/27)\binom{\mu^2 n/9}{2}$. Therefore, the probability that Alg queries a changed position in the i 'th clique is less than

$$\frac{\frac{\mu^3}{27}\binom{\mu^2 n/9}{2}}{\binom{\mu^2 n/9}{2}} = \frac{\mu^3}{27}.$$

By a union bound over all $\frac{9}{\mu^2}$ cliques we can bound the probability that Alg queries a changed position by

$$\frac{9}{\mu^2} \cdot \frac{\mu^3}{27} = \frac{\mu}{3},$$

as claimed. This completes the proof of the theorem.

A.8 Proof of Theorem 10

We first discuss the running time of SEED(ϵ'). Using a typical graph traversal algorithm (such as DFS or BFS), we can identify the connected components of all the T copies, in time that is in the order of the size of \mathcal{E}_T , i.e. $\tilde{O}(pn^2 + \sqrt{pn}^{1.5})$, by Theorem 8. To compute $\Delta(v|\mathcal{S})$, we go over the connected component of v in each of the T subsampled graphs and add up their values. These values are pre-computed for each connected component, and hence this operation takes $O(T)$ time. Recall that the value of a connected component is initially set equal to the number of initial nodes, belonging to \mathcal{V}_ρ , in that component. To add a node v to \mathcal{S} , $\mathcal{S} \leftarrow \mathcal{S} \cup \{v\}$, we reset the value of the connected component of v in each of the T copies to zero. This ensures that if we later pick another node from these components we do not double count these initial nodes that are already covered by v . This process can be done in $O(T) = \tilde{O}(k)$ rounds as well. Recall that the submodular maximization algorithm that we are using makes at most $n \log(1/\epsilon')$ queries to the function $\Gamma_{\rho, \epsilon'}^{(T)}$. Therefore the total running time of this algorithm is $\tilde{O}_{\epsilon', \epsilon, \delta}(pn^2 + \sqrt{pn}^{1.5} + nk)$ as claimed.

Next we consider the approximation guarantee of SEED(ϵ'). Fix $\epsilon' = \epsilon/7$, $\delta' = \log 5 + \log(1/\epsilon')$, $\rho' = \rho_{\epsilon', \delta'}^{n, k}$, $T' = T_{\epsilon', \delta'}^{n, k}$, and $\tau' = \tau_{\epsilon', \delta'}^{n, k}$. Running the Algorithm PROBE (ρ', T', τ'), provides access to the submodular function $\Gamma_{\rho', \epsilon'}^{(T')}$ which has the k -IM optimal solution $\Lambda_{\rho', \epsilon'}^{(T')}$. Using SEED(ϵ'), we obtain an approximate solution of k -IM for $\Gamma_{\rho', \epsilon'}^{(T')}$ in $\tilde{O}(pn^2 + \sqrt{pn}^{1.5} + nk)$ time. Call this solution $\Lambda_{\rho', \epsilon'}^{\star(T')}$. The analysis of [47, Theorem 1] implies that

$$\begin{aligned} \mathbb{E} \left[\Gamma_{\rho', \epsilon'}^{(T')}(\Lambda_{\rho', \epsilon'}^{\star(T')}) \right] &\geq (1 - 1/e - \epsilon') \Gamma_{\rho', \epsilon'}^{(T')}(\Lambda_{\rho', \epsilon'}^{(T')}) \\ &= (1 - 1/e - \epsilon') L_{\rho', \epsilon'}^{(T')}, \end{aligned}$$

where the expectation is with respect to the randomness of the SEED algorithm.

Combing the claims of Theorems 7 and 8 guarantees that with probability at least $1 - 5e^{-\delta'} = 1 - \epsilon'$, $\Gamma(\Lambda_{\rho', \epsilon'}^{(T')}) = L_{\rho', \epsilon'}^{(T')} \geq (1 - \epsilon')L - 2(2 - \epsilon')\epsilon'n$, and we probe no more than $\tilde{O}(pn^2 + \sqrt{pn}^{1.5})$ edges. Thus the expected number of nodes that are covered by the output of SEED algorithm, $\Lambda_{\rho', \epsilon'}^{\star(T')}$, can be lower bounded as follows:

$$\begin{aligned} \mathbb{E} \left[\Gamma(\Lambda_{\rho', \epsilon'}^{\star(T')}) \right] &\geq (1 - 1/e - \epsilon') \mathbb{E} \left[L_{\rho', \epsilon'}^{(T')} \right] \\ &\geq (1 - 1/e - \epsilon')(1 - \epsilon')(1 - \epsilon')L - 2(2 - \epsilon')\epsilon'n \\ &\geq (1 - 1/e - \epsilon')(1 - 2\epsilon')L - 4\epsilon'n \\ &\geq (1 - 1/e)(1 - 2\epsilon')L - (\epsilon')(1 - 2\epsilon')L - 4\epsilon'n \\ &\geq (1 - 1/e)L - 7\epsilon'n = (1 - 1/e)L - \epsilon n, \end{aligned}$$

where the first expectation is with respect to the randomness of both the SEED and the PROBE algorithms, and the second expectation is with respect to the randomness of the PROBE algorithm.

A.9 Proof of Theorem 11

Recall our notation in the SPREAD algorithm. The output of the algorithm Λ^\star is a set of k nodes that are chosen, one by one, in k steps. Let us use $\Lambda^{\star i}$ to denote the first i nodes that are selected in steps 1 to i . In step i , we choose ρ initial nodes at random to run spread queries. Let us denote the random initial nodes in step i , by u_1^i, \dots, u_ρ^i . We use A_j^i to denote the random subset of nodes that are observed to adopt when u_j^i is seeded. We reset A_j^i to \emptyset if it contains any of the $i - 1$ nodes selected in the previous steps. We consider the pool of candidates, $u \in \mathcal{V} \setminus \Lambda^{\star i-1}$, and choose the i -th seed to be the one that appears in the most subsets. To put this in mathematical notation, let

$X_{u,j}^i = \mathbb{1}\{u \in A_j^i\}$ be the indicator that u belongs to A_j^i , and set $X_u^i = \sum_{j=1}^{\rho} X_{u,j}^i$ to count the number of times that u appears in any of the subsets A_1^i, \dots, A_{ρ}^i . Subsequently, in step i , we choose

$$v^* = \operatorname{argmax}_{u \in \mathcal{V} \setminus \Lambda^{*i-1}} X_u^i,$$

and add it to Λ^* .

We analyze the steps of Algorithm 3 and show that for $\epsilon' = \epsilon/3$ and $\rho = \rho_{\epsilon}^{n,k} = 3k \log(2nk/\epsilon')/\epsilon'^3$, the output of SPREAD satisfies the desired approximation guarantee. Let us define random variable N_u^i to be the expected number of nodes that are covered by $\Lambda^{*i-1} \cup \{u\}$ but not by Λ^{*i-1} . Note that the probability that $X_{u,j}^i = 1$ is equal to N_u^i/n . Therefore, we have $\mathbb{E}[(n/\rho)X_u^i] = \mathbb{E}[N_u^i]$. Moreover, notice that choosing $v^* \in \mathcal{V} \setminus \Lambda^{*i-1}$ to maximize $\mathbb{E}[N_u^i]$ is equivalent to one step of the greedy algorithm. This is equivalent to choosing $v^* \in \mathcal{V} \setminus \Lambda^{*i-1}$ to maximize $\mathbb{E}[X_u^i]$, since $\mathbb{E}[(n/\rho)X_u^i] = \mathbb{E}[N_u^i]$.

Next note that due to submodularity, the marginal values only decrease as we add more elements. Hence, if we stop the algorithm when $\forall_u \mathbb{E}[N_u^i] < \epsilon'n/k$, in total we do not loose more than $k(\epsilon n/k) = \epsilon n$. For the sake of analysis, let us assume that the algorithm stops if $\forall_u \mathbb{E}[N_u^i] < \epsilon'n/k$: This means that the algorithm stops if it selects k seeds or $\forall_u \mathbb{E}[N_u^i] < \epsilon'n/k$ whichever comes first. Henceforth, without loss of generality, we assume that $\max_u \mathbb{E}[N_u^i] \geq \epsilon'n/k$ which means that we have $\mathbb{E}[X_u^i] \geq \epsilon'\rho/k$.

Recall that X_u^i is the sum of i.i.d. binary random variables $X_{u,j}^i$. Hence, by the Chernoff bound we have

$$\begin{aligned} \mathbb{P} \left[|X_u^i - \mathbb{E}[X_u^i]| \leq \epsilon' \mathbb{E}[X_u^i] \right] &\leq 2 \exp \left(-\frac{\epsilon'^2 \mathbb{E}[X_u^i]}{3} \right) && \text{Chernoff Bound} \\ &\leq 2 \exp \left(-\frac{\epsilon'^3 \rho}{3k} \right) && \mathbb{E}[X_u^i] \geq \epsilon'\rho/k \\ &= \frac{\epsilon'}{nk}. && \rho = \frac{81k \log \frac{6nk}{\epsilon'}}{\epsilon^3} = \frac{3k \log \frac{2nk}{\epsilon'}}{\epsilon'^3} \end{aligned}$$

Union bound over all $u \in \mathcal{V}$ and $1 \leq i \leq k$ provides that with probability at least $1 - \epsilon'$, $(1 - \epsilon')\mathbb{E}[X_u^i] \leq X_u^i \leq (1 + \epsilon')\mathbb{E}[X_u^i]$ for all u and i . This implies that the seed that our algorithm selects has marginal increase at least $\frac{1-\epsilon'}{1+\epsilon'} \geq 1 - 2\epsilon'$ times that of the greedy algorithm. Such algorithm is called $(1 - 2\epsilon')$ -approximate greedy in [25] and it is proven to return a $(1 - 1/e - 2\epsilon')$ -approximate solution [4, 25, 37]. Therefore, we can bound the expected value of the output solution of SPREAD (ρ) as follows.

$$\begin{aligned} \mathbb{E}[\Lambda^*] &\geq (1 - \epsilon')[(1 - 1/e - \epsilon')L - \epsilon'n] \\ &\geq (1 - \epsilon')[(1 - 1/e)L - 2\epsilon'n] \\ &\geq (1 - 1/e)L - 3\epsilon'n = (1 - 1/e)L - \epsilon n. \end{aligned}$$

A.10 Proof of Theorem 12

Pick an arbitrary function $f(n) \in o(n)$, and let $g(n) = \sqrt{n/f(n)}$. Note that $g(n) \in \omega(1)$. We show that an algorithm Alg that makes $f(n)$ spread queries is not μ -approximation. Consider the the example depicted in Figure 4. We have a clique of size $g(n)$ (on $g(n)$ nodes chosen uniformly at random) and $n - g(n)$ isolated nodes and we aim to seed one node. One can bound the probability

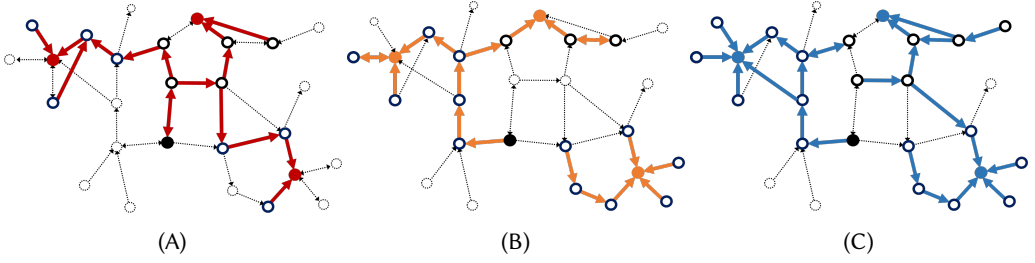


Fig. 8. Three reverse cascades are depicted in (A) red, (B) orange, and (C) blue. All cascades start from the same random initial nodes which are marked in the same color as the cascades. As the cascades diffuse in reverse, each node reveals its incoming edges at random. To score the nodes, we count the number of reachable initial nodes in each cascade and add them up. For example, the node that is marked in black scores three in the red cascade, two in the orange cascade, and one in the blue cascade. In total, it scores as high as or higher than other nodes across the three cascades. The dotted sections consist of unrealized influences in each cascade.

that Alg queries a node from the clique by

$$1 - \left(1 - \frac{g(n)}{n}\right)^{f(n)} = 1 - \left(1 - \frac{g(n)}{n}\right)^{\frac{n}{g(n)^2}} \leq 1 - \left(1 - \frac{1}{e}\right)^{\frac{1}{g(n)}}.$$

Moreover, since $g(n) \in \omega(1)$ we have $1 - \left(1 - \frac{1}{e}\right)^{\frac{1}{g(n)}} \in o(1)$. If Alg does not query a node via spread queries, it seeds one of the nodes of the clique with probability at most $\frac{g(n)}{n-f(n)} = o(1)$. Therefore, the expected number of nodes covered by Alg is at most $o(1)g(n) + 1$, which means that the approximation factor of Alg is $\frac{o(1)g(n)+1}{g(n)} = o(1)$ as claimed.

B EXTENSION TO ASYMMETRIC INFLUENCES

We presented our results for undirected graphs with a homogeneous cascade probability (p). In general, the influence graph may be directed and cascade probabilities may differ in each direction and across the edges. Consider a node v and let \mathcal{N}_v be the set of all its *incoming* neighbors (i.e., its influencers). For a directed edge $u \rightarrow v$, let p_{uv} be the probability of u influencing v . We can easily extend our results to *undirected* graphs with *heterogeneous* cascade probabilities. Note that in undirected graphs $\forall_{uv} p_{uv} = p_{vu}$. With heterogeneous cascade probabilities, the spread queries are performed as before and the adopters are observed. When performing edge queries, the probed nodes reveal each of their neighbors, with the probability associated that edge. Our bound on the edge queries would include $p_{\max} = \max_{uv} p_{uv}$ instead of p .

We can also extend our results to directed graphs (or even graphs with both directed and undirected edges). In the case of edge queries, we first modify step (2-a) of the PROBE algorithm, such that a probed node v reveals each of her *incoming* neighbors $u \in \mathcal{N}_v$, with their associated probability p_{uv} . Two directed edges that are between the same pair of nodes in opposite directions ($u \rightarrow v$ and $v \rightarrow u$) are distinguished. Therefore, as long as we do not probe a node more than once, each directed edge will get at most one chance of appearing in $\mathcal{G}_{\rho, \epsilon}^{(i)}$. Accordingly, we modify step (2-b) of the PROBE algorithm as follows: For any revealed neighbor, add the corresponding directed edge to $\mathcal{G}_{\rho, \epsilon}^{(i)}$ and proceed to probe the revealed neighbors that are not probed before. In fact, we can slightly improve our edge query upper-bound in the directed case since every directed edge that is revealed in step (2-a) of PROBE is added to $\mathcal{G}_{\rho, \epsilon}^{(i)}$ in step (2-b) of the PROBE algorithm. Subsequently,

our edge query upper-bound in the directed case consists entirely of the edges that appear in the sketch and is given by (12): with probability at least $1 - e^\delta$ no more than $C_{\epsilon,\delta}^{n,k} \in \tilde{O}_{\epsilon,\delta}(pn^2 + \sqrt{pn}^{1.5})$ edges are queried.

When limiting the probed neighborhoods in step (2-c) of the PROBE algorithm, instead of considering the size of the connected component of an initial node, we count the number of nodes that are reachable from it (i.e., the size of its realized *cone of influence*). For example in Figure 8(A), the reachable set for all of the initial nodes is empty, therefore we proceed to probe the incoming neighbors until there are no new nodes to probe. In fact, of all the initial nodes in all three cascades in Figure 8, only the leftmost initial node in Figure 8(B) has a non-empty reachable set that is a singleton.

Similarly, when scoring the candidates for seeding in steps (2) and (4-b) of the SEED algorithm, we count the number of reachable initial nodes (belonging to \mathcal{V}_ρ) rather than the number of initial nodes in the candidate’s connected component (see Figure 8). As before, if an initial node is reachable from any of the chosen seeds, then in step (4-c) of SEED we nullify its value for scoring the subsequent candidates.

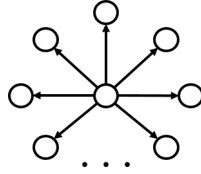


Fig. 9. With $o(n)$ spread query, one is unlikely to discover the center of the directed star network. In such a case it is impossible to guarantee a $\mu > 0$ approximation factor with fewer than $\Omega(n)$ queries.

In contrast, finding an approximately optimal seed set by performing spread queries in a directed graph is very hard. For example, consider a star graph with n leaves where all of the edges are directed away from the center toward the leaves (see Figure 9). Assume that the cascade probability on each edge is 1. In this case, if we query a leaf we only observe an isolated node and hence we need $\Omega(n)$ spread queries to find the center of the star and seed it. In a situation where running reverse spreads is a plausible way of acquiring network information (e.g., by querying the transposed graph as in [12]), our algorithm and proofs continue to hold exactly the same. In particular, we can estimate the marginal increase of a candidate node on the current seed set by counting the number of times that it has appeared in the output of the queries without any of the currently chosen seeds (i.e., the number of times that the random initial nodes are reachable from the candidate node but not from any of the currently chosen seeds). By running enough such queries and choosing the best node at every step, we can approximate greedy with enough precision as in Algorithm 3 – see also Figure 10 for a relevant illustration.

C EXTENSIONS BEYOND THE INDEPENDENT CASCADE MODEL

Here we explain how a triggering set technique that is proposed in [35] helps us devise queries in a large class of models, including the *linear threshold model*. Recall that the influence function Γ maps a seed set to a positive real number that is the (expected) number of adopters under a (randomized) model of diffusion. Kempe, Kleinberg, and Tardos [33–35] – through a conjecture that is positively resolved by Mossel and Roch [48] – identify a broad class of threshold models for which the influence function is non-negative, monotone, and submodular. Influence maximization in such models can be solved to within a $(1 - 1/e)$ approximation guarantee, following a natural greedy node selection algorithm.

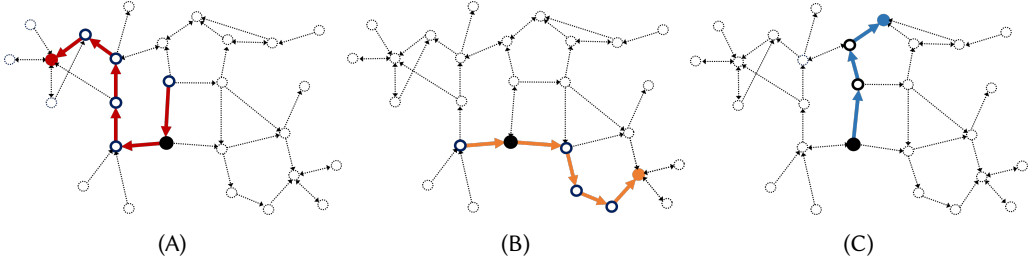


Fig. 10. Some diffusion processes can be reduced to a triggering set model; whereby, after randomly drawing a triggering set from among her neighbors, a node becomes active if any of the nodes in her triggering set are active. In such models, we can devise queries by having a probed node reveal a random realization of her triggering set, and then proceeding to probe the revealed nodes. In the case of the linear threshold model, the triggering sets are either empty or a singleton chosen randomly according to the edge weights. The reverse cascades in (A), (B) and (C) show three queries where the nodes sequentially reveal their triggering sets, starting from a random initial node. The black node appears in the output of all three “reversed cascades”, and therefore scores the highest as a seed candidate.

In a general threshold model, each node v has an activation function f_v and a threshold $\theta_v \in [0, 1]$. The activation function maps subsets of neighbors of v to a real number between zero and one. Node v becomes an adopter at time t if $f_v(\mathcal{A}_{v,t-1}) \geq \theta_v$, where $\mathcal{A}_{v,t-1} \subset \mathcal{N}_v$ is the set of all active (adopter) neighbors of node v . Approximate influence maximization with deterministic thresholds is known to be very hard (cf., e.g., [35, Section 3.2] and [52]). To avoid the intractable settings, Kempe et al. consider a randomized model where thresholds are i.i.d. uniform $[0, 1]$ variables. Mossel and Roch show that if the “local” activation functions f_v are submodular, then the “global” influence function Γ is also submodular and influence maximization can be achieved with strong approximation guarantees [48, Theorem 1.6 and Corollary 1.7]. In the special case of the *linear* threshold model, each node v is influenced by its incoming neighbours $u \in \mathcal{N}_v$ according to their edge weights b_{uv} . Node v becomes an adopter at time t if the total weight of her adopting neighbors exceeds her threshold, i.e. if $f_v(\mathcal{A}_{v,t-1}) = \sum_{u \in \mathcal{A}_{v,t-1}} b_{uv} \geq \theta_v$.

At the heart of the proofs of Kempe et al. [33, 35] lie a triggering set technique. Accordingly, each node v chooses a random subset of its incoming neighbors, which we call its triggering set and denote it by $\mathcal{T}_v \subset \mathcal{N}_v$. Node v becomes an adopter at time t if any of the nodes in \mathcal{T}_v is an adopter at time $t - 1$. The distribution according to which the triggering sets are drawn is determined by the diffusion model. However, not all diffusion processes can be reduced to a triggering set model. For those that do, their influence function is guaranteed to be submodular [35, Lemma 4.4].

For example, in the independent cascade model, the triggering set \mathcal{T}_v includes each of the neighbors $u \in \mathcal{N}_v$ with probability p_{uv} , independently at random. In the case of the linear threshold model, Kempe et al. [35] devise the following construction, assuming $\sum_{u \in \mathcal{N}_v} b_{uv} \leq 1$: The triggering set \mathcal{T}_v is comprised of a single node or no nodes at all. For $u \in \mathcal{N}_v$, the probability that $\mathcal{T}_v = \{u\}$ is equal to b_{uv} , and $\mathcal{T}_v = \emptyset$ with probability $1 - \sum_{u \in \mathcal{N}_v} b_{uv}$.

For diffusion processes that can be cast as a triggering set model, we can implement queries by having the probed nodes reveal their triggering sets, and proceeding to probe the revealed nodes. Starting from random initial nodes, each triggering set corresponds to a batch of directed edges that are incoming to the probed node. We can use the number of reachable initial nodes to implement an approximate greedy heuristic as described in Appendix B – see Figure 10. One needs to analyze the specific diffusion process and the triggering distributions to provide approximation guarantees using a bounded number of queries.

In the particular case of the linear threshold model, the proof of Theorem 11, and its adaptation to directed graphs in Appendix B, continue to hold with little change. Using k batches of ρ reversed cascades, we can provide an approximate k -IM solution for the linear threshold model over directed graphs. At each stage we choose ρ initial nodes at random and implement ρ cascades in reverse. In each reversed cascade, we start from the initial node, reveal her triggering set, and then proceeding to probe the node that is revealed in the triggering set, etc. After each batch of ρ reversed cascades, we choose the node that appears the most number of times, discarding those cascades that include the already chosen seeds. Following Theorem 11, we can provide a $(1 - 1/e)L - \epsilon n$ approximation guarantee, by running $k\rho = 81k^2 \log(\frac{6nk}{\epsilon})/\epsilon^3$ reversed cascades.

To bound the number of edge queries, recall that each triggering set in the linear threshold model consists of at most a single node. Hence, each reversed cascade corresponds to a path of length at most $n -$ see Figure 10. Therefore, we can bound the total number of queried edges by $nk\rho = 81nk^2 \log(\frac{6nk}{\epsilon})/\epsilon^3$. In Theorem 13, we state how our results translate to the case of k -IM with the linear threshold model over directed graphs.

REFERENCES

- [1] Mohammad Akbarpour, Suraj Malladi, and Amin Saberi. 2018. Just a Few Seeds More: Value of Network Information for Diffusion. Available at SSRN 3062830.
- [2] Noga Alon, Eldar Fischer, Michael Krivelevich, and Mario Szegedy. 2000. Efficient Testing of Large Graphs. *Combinatorica* 20, 4 (2000), 451–476.
- [3] Noga Alon, Eldar Fischer, Ilan Newman, and Asaf Shapira. 2009. A Combinatorial Characterization of the Testable Graph Properties: It’s All about Regularity. *SIAM J. Comput.* 39, 1 (2009), 143–167.
- [4] Ashwinkumar Badanidiyuru and Jan Vondrák. 2014. Fast Algorithms for Maximizing Submodular Functions. In *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, Portland, Oregon, USA, 1497–1514.
- [5] Eric Balkanski, Nicole Immorlica, and Yaron Singer. 2017. The Importance of Communities for Learning to Influence. In *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc., Long Beach, California, USA, 5862–5871.
- [6] Eric Balkanski, Aviad Rubinfeld, and Yaron Singer. 2016. The power of optimization from samples. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., Barcelona, Spain, 4017–4025.
- [7] Eric Balkanski, Aviad Rubinfeld, and Yaron Singer. 2017. The limitations of optimization from samples. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*. ACM, Montreal, Canada, 1016–1027.
- [8] Abhijit Banerjee, Arun G. Chandrasekhar, Esther Duflo, and Matthew O. Jackson. 2013. The Diffusion of Microfinance. *Science* 341, 6144 (2013), 1236498.
- [9] Abhijit Banerjee, Arun G Chandrasekhar, Esther Duflo, and Matthew O Jackson. 2019. Using gossips to spread information: Theory and evidence from two randomized controlled trials. *The Review of Economic Studies* 86, 6 (2019), 2453–2490.
- [10] MohammadHossein Bateni, Hossein Esfandiari, and Vahab Mirrokni. 2017. Almost Optimal Streaming Algorithms for Coverage Problems. In *Proceedings of the 29th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA ’17)*. ACM, New York, NY, USA, 13–23.
- [11] MohammadHossein Bateni, Hossein Esfandiari, and Vahab Mirrokni. 2018. Optimal Distributed Submodular Optimization via Sketching. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD ’18)*. ACM, New York, NY, USA, 1138–1147.
- [12] Christian Borgs, Michael Brautbar, Jennifer Chayes, and Brendan Lucier. 2014. Maximizing Social Influence in Nearly Optimal Time. In *Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, Portland, Oregon, USA, 946–957.
- [13] Jing Cai, Alain De Janvry, and Elisabeth Sadoulet. 2015. Social Networks and the Decision to Insure. *American Economic Journal: Applied Economics* 7, 2 (2015), 81–108.
- [14] Goylette F. Chami, Sebastian E. Ahnert, Narcis B. Kabatereine, and Edridah M. Tukahebwa. 2017. Social Network Fragmentation and Community Health. *Proceedings of the National Academy of Sciences* 114, 36 (2017), E7425–E7431.
- [15] Bernard Chazelle, Ronitt Rubinfeld, and Luca Trevisan. 2005. Approximating the Minimum Spanning Tree Weight in Sublinear Time. *SIAM Journal on computing* 34, 6 (2005), 1370–1379.
- [16] Wei Chen, Tian Lin, Zihan Tan, Mingfei Zhao, and Xuren Zhou. 2016. Robust Influence Maximization. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, San Francisco, California, USA, 795–804.

- [17] Wei Chen, Yajun Wang, and Siyu Yang. 2009. Efficient Influence Maximization in Social Networks. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09)*. ACM, Paris, France, 199–208.
- [18] Alex Chin, Dean Eckles, and Johan Ugander. 2018. Evaluating Stochastic Seeding Strategies in Networks. preprint arXiv:1809.09561.
- [19] Nicholas A. Christakis and James H. Fowler. 2010. Social Network Sensors for Early Detection of Contagious Outbreaks. *PLoS One* 5, 9 (2010), e12948.
- [20] Edith Cohen, Daniel Delling, Thomas Pajor, and Renato F. Werneck. 2014. Sketch-Based Influence Maximization and Computation: Scaling up with Guarantees. In *Proceedings of the 23rd International Conference on Conference on Information and Knowledge Management*. ACM, Shanghai, China, 629–638.
- [21] Reuven Cohen, Shlomo Havlin, and Daniel Ben-Avraham. 2003. Efficient Immunization Strategies for Computer Networks and Populations. *Physical Review Letters* 91, 24 (2003), 247901.
- [22] Pedro Domingos and Matt Richardson. 2001. Mining the Network Value of Customers. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, San Jose, California, USA, 57–66.
- [23] Hossein Esfandiari and Michael Mitzenmacher. 2018. Metric Sublinear Algorithms via Linear Sampling. In *59th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, Paris, France, 11–22.
- [24] David Godes and Dina Mayzlin. 2009. Firm-created word-of-mouth communication: Evidence from a field test. *Marketing Science* 28, 4 (2009), 721–739.
- [25] Daniel Golovin and Andreas Krause. 2011. Adaptive Submodularity: Theory and Applications in Active Learning and Stochastic Optimization. *Journal of Artificial Intelligence Research* 42 (2011), 427–486.
- [26] Amit Goyal, Francesco Bonchi, and Laks V. S. Lakshmanan. 2010. Learning Influence Probabilities in Social Networks. In *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining*. ACM, New York, New York, USA, 241–250.
- [27] Xinran He and David Kempe. 2018. Stability and Robustness in Influence Maximization. *ACM Transactions on Knowledge Discovery from Data* 12, 6, Article 66 (Aug. 2018), 34 pages.
- [28] Oliver Hinz, Bernd Skiera, Christian Barrot, and Jan U Becker. 2011. Seeding strategies for viral marketing: An empirical comparison. *Journal of Marketing* 75, 6 (2011), 55–71.
- [29] Thibaut Horel and Yaron Singer. 2015. Scalable Methods for Adaptively Seeding a Social Network. In *Proceedings of the 24th International Conference on World Wide Web*. International World Wide Web Conference, Florence, Italy, 441–451.
- [30] Piotr Indyk. 1999. Sublinear Time Algorithms for Metric Space Problems. In *Proceedings of the 31st Annual ACM Symposium on Theory of Computing*. ACM, Atlanta, Georgia, USA, 428–434.
- [31] Kyomin Jung, Wooram Heo, and Wei Chen. 2012. IRIE: Scalable and Robust Influence Maximization in Social Networks. In *Proceedings of the 12th International Conference on Data Mining (ICDM)*. IEEE, Washington, DC, USA, 918–923.
- [32] Mohammad Karimi, Mario Lucic, Hamed Hassani, and Andreas Krause. 2017. Stochastic submodular maximization: The case of coverage functions. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., Long Beach, CA, USA, 6853–6863.
- [33] David Kempe, Jon Kleinberg, and Éva Tardos. 2003. Maximizing the Spread of Influence through a Social Network. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, Washington, DC, USA, 137–146.
- [34] David Kempe, Jon Kleinberg, and Éva Tardos. 2005. Influential nodes in a diffusion model for social networks. In *Proceedings of the 32nd International Colloquium on Automata, Languages, and Programming*. Springer, Lisbon, Portugal, 1127–1138.
- [35] David Kempe, Jon Kleinberg, and Éva Tardos. 2015. Maximizing the Spread of Influence through a Social Network. *Theory of Computing* 11, 4 (2015), 105–147.
- [36] David A. Kim, Alison R. Hwang, Derek Stafford, D. Alex Hughes, A. James O'Malley, James H. Fowler, and Nicholas A. Christakis. 2015. Social Network Targeting to Maximise Population Behaviour Change: A Cluster Randomised Controlled Trial. *The Lancet* 386, 9989 (2015), 145–153.
- [37] Ravi Kumar, Benjamin Moseley, Sergei Vassilvitskii, and Andrea Vattani. 2015. Fast Greedy Algorithms in Mapreduce and Streaming. *ACM Transactions on Parallel Computing (TOPC)* 2, 3 (2015), 14.
- [38] Vineet Kumar, David Krackhardt, and Scott Feld. 2018. Network Interventions Based on Inversity: Leveraging the Friendship Paradox in Unknown Network Structures. (2018). Working Paper, Yale University.
- [39] Vineet Kumar and K Sudhir. 2019. Can friends seed more buzz and adoption? Cowles Foundation Discussion Paper.
- [40] Silvio Lattanzi and Yaron Singer. 2015. The Power of Random Neighbors in Social Networks. In *Proceedings of the 8th ACM International Conference on Web Search and Data Mining (WSDM '15)*. ACM, Shanghai, China, 77–86.
- [41] Siyu Lei, Silviu Maniu, Luyi Mo, Reynold Cheng, and Pierre Senellart. 2015. Online Influence Maximization. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, Sydney, NSW, Australia, 645–654.

- [42] Jure Leskovec and Christos Faloutsos. 2006. Sampling from Large Graphs. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, Philadelphia, PA, USA, 631–636.
- [43] Barak Libai, Eitan Muller, and Renana Peres. 2013. Decomposing the value of word-of-mouth seeding programs: Acceleration versus expansion. *Journal of Marketing Research* 50, 2 (2013), 161–176.
- [44] Vahideh Manshadi, Sidhant Misra, and Scott Rodilitz. 2019. Diffusion in Random Networks: Impact of Degree Distribution. *Operations Research*, forthcoming.
- [45] Shodai Mihara, Sho Tsugawa, and Hiroyuki Ohsaki. 2015. Influence Maximization Problem for Unknown Social Networks. In *Proceedings of the International Conference on Advances in Social Networks Analysis and Mining*. IEE/ACM, Paris, France, 1539–1546.
- [46] Shodai Mihara, Sho Tsugawa, and Hiroyuki Ohsaki. 2017. On the Effectiveness of Random Jumps in an Influence Maximization Algorithm for Unknown Graphs. In *International Conference on Information Networking (ICOIN)*. IEEE, Da Nang, Vietnam, 395–400.
- [47] Baharan Mirzasoileiman, Ashwinkumar Badanidiyuru, Amin Karbasi, Jan Vondrák, and Andreas Krause. 2015. Lazier Than Lazy Greedy. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence*. AAAI, Austin, Texas, USA, 1812–1818.
- [48] Elchanan Mossel and Sebastian Roch. 2010. Submodularity of Influence in Social Networks: From Local to Global. *SIAM J. Comput.* 39, 6 (2010), 2176–2188.
- [49] Peter Orbanz. 2017. Subsampling Large Graphs and Invariance in Networks. preprint arXiv:1710.04217.
- [50] Elizabeth Levy Paluck, Hana Shepherd, and Peter M. Aronow. 2016. Changing Climates of Conflict: A Social Network Experiment in 56 schools. *Proceedings of the National Academy of Sciences* 113, 3 (2016), 566–571.
- [51] Gal Sadeh, Edith Cohen, and Haim Kaplan. 2019. Sample Complexity Bounds for Influence Maximization. *Innovations in Theoretical Computer Science (ITCS)* accepted paper, arXiv:1907.13301.
- [52] Grant Schoenebeck and Biaoshuai Tao. 2019. Beyond Worst-case (In)Approximability of Nonsubmodular Influence Maximization. *ACM Trans. Comput. Theory* 11, 3, Article 12 (April 2019), 56 pages.
- [53] Lior Seeman and Yaron Singer. 2013. Adaptive Seeding in Social Networks. In *Proceedings of the 54th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, Berkeley, California, USA, 459–468.
- [54] Sebastian Stein, Soheil Eshghi, Setareh Maghsudi, Leandros Tassioulas, Rachel K. E. Bellamy, and Nicholas R. Jennings. 2017. Heuristic Algorithms for Influence Maximization in Partially Observable Social Networks. In *Proceedings of the 3rd International Workshop on Social Influence Analysis*. CEUR Workshop Proceedings, Melbourne, Australia, 20–32.
- [55] Amanda L. Traud, Peter J. Mucha, and Mason A. Porter. 2012. Social Structure of Facebook Networks. *Physica A: Statistical Mechanics and its Applications* 391, 16 (2012), 4165–4180.
- [56] Aad W. van der Vaart and Jon A. Wellner. 1996. *Weak Convergence and Empirical Processes: With Applications to Statistics*. Springer-Verlag, New York, NY, USA.
- [57] Chi Wang, Wei Chen, and Yajun Wang. 2012. Scalable Influence Maximization for Independent Cascade Model in Large-Scale Social Networks. *Data Mining and Knowledge Discovery* 25, 3 (2012), 545–576.
- [58] Zheng Wen, Branislav Kveton, Michal Valko, and Sharan Vaswani. 2017. Online Influence Maximization under Independent Cascade Model with Semi-Bandit Feedback. In *Advances in Neural Information Processing Systems 30*. Curran Associates, Inc., Long Beach, California, USA, 3022–3032.
- [59] Bryan Wilder, Nicole Immorlica, Eric Rice, and Milind Tambe. 2017. Influence Maximization with an Unknown Network by Exploiting Community Structure. In *Proceedings of the 3rd International Workshop on Social Influence Analysis*. CEUR Workshop Proceedings, Melbourne, Australia, 2–7.
- [60] Bryan Wilder, Nicole Immorlica, Eric Rice, and Milind Tambe. 2018. Maximizing Influence in an Unknown Social Network. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*. AAAI, New Orleans, Louisiana, USA, 4743–4750.
- [61] Bryan Wilder, Amulya Yadav, Nicole Immorlica, Eric Rice, and Milind Tambe. 2017. Uncharted but not Uninfluenced: Influence Maximization with an Uncertain Network. In *Proceedings of the 16th Conference on Autonomous Agents and Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, São Paulo, Brazil, 1305–1313.
- [62] L.-C. Zhang and M. Patone. 2017. Graph sampling. *METRON* 75, 3 (01 Dec 2017), 277–299.
- [63] Honglei Zhuang, Yihan Sun, Jie Tang, Jialin Zhang, and Xiaoming Sun. 2013. Influence Maximization in Dynamic Social Networks. In *Proceedings of the 13th International Conference on Data Mining (ICDM)*. IEEE, Dallas, Texas, USA, 1313–1318.